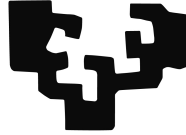


University of the Basque Country (UPV/EHU)

eman ta zabal zazu



Universidad
del País Vasco

Euskal Herriko
Unibertsitatea

Department of Computation Sciences and Artificial Intelligence

Documentation

Contributions to time series analysis, modelling and forecasting to increase reliability in industrial environments

Meritxell Gómez Omella

- 1. Supervisor* **Basilio Sierra**
Department of Computation Sciences and Artificial Intelligence
University of the Basque Country (UPV/EHU)
- 2. Supervisor* **Susana Ferreiro**
Department of Intelligent Information Systems
Tekniker

May 31, 2023

Contributions to time series analysis,
modelling and forecasting to increase
reliability in industrial environments

Meritxell Gómez Omella

May 31, 2023

Meritxell Gómez Omella

Contributions to time series analysis, modelling and forecasting to increase reliability in industrial environments

Documentation, May 31, 2023

Supervisors: Basilio Sierra and Susana Ferreiro

University of the Basque Country (UPV/EHU)

Department of Computation Sciences and Artificial Intelligence

Street address

Postal Code and CITY

Contents

I	Fundamentals	1
1	Background	3
1.1	Introduction and Motivation	4
1.2	Hypothesis and Objectives	10
1.3	Research Environment	13
1.3.1	Tekniker	13
1.3.2	UPV/EHU	14
1.3.3	ENSAM	14
1.4	Research Projects	15
1.5	Contributions	19
1.6	Thesis structure	22
2	Contributions to Time Series Life Cycle	25
2.1	Time Series Basics	27
2.2	Data Quality in Time Series	30
2.2.1	The dqts R package	35
2.3	Time Series Modelling	38
2.3.1	Tradicional Forecasting Methods	39
2.3.2	Machine Learning in Time Series	46
2.3.3	Time Series Cross-Validation	57
2.3.4	Time Series Feature Extraction	60
2.3.5	Forecasting Strategies	66
2.4	Error Evaluation	72
2.4.1	Regression Performance	73
2.4.2	Classification Performance	77

3	Industrial Applications	81
3.1	Introduction	82
3.2	Poultry Chain Control Management	86
3.2.1	Context and Motivation	86
3.2.2	Use Case Presentation	87
3.2.3	Proposal Solution	88
3.2.4	Conclusions and Outcomes	97
3.3	Energy Demand Forecasting	99
3.3.1	Context and Motivation	99
3.3.2	Use Case Presentation	100
3.3.3	Proposal Solution	101
3.3.4	Conclusions and Outcomes	108
3.4	Porosity Detection in Additive Manufacturing	110
3.4.1	Context and Motivation	110
3.4.2	Use Case Presentation	111
3.4.3	Proposal Solution	114
3.4.4	Conclusions and outcomes	123
3.5	Manufacturing Production Line Diagnosis	124
3.5.1	Context and Motivation	124
3.5.2	Use Case Presentation	126
3.5.3	Proposal Solution	127
3.5.4	Conclusions and Outcomes	134
4	Closing Remarks	137
4.1	Conclusions	138
4.2	Future Work	144
	Bibliography	147
	List of Figures	161
	List of Tables	165

II	The research	167
5	An IoT Platform towards the Enhancement of Poultry Production Chains	169
6	k-Nearest Patterns for Electrical Demand Forecasting in Residential and Small Commercial Buildings	191
7	On the Evaluation, Management and Improvement of Data Quality in Streaming Time Series	225
8	Optimizing Porosity Detection wire Laser Metal Deposition Processes using Data-driven Classification Techniques.	245
9	Other publications	271
A	Implementation of dqts R package	321
B	Implementation of KNPTS R package	331
	Declaration	337

Por los que se fueron y no volverán.
Por los que estuvieron y están.

Acknowledgement

Antes de ponerme emotiva, quiero agradecer a Tekniker la oportunidad brindada para el desarrollo de este trabajo de investigación. Gracias por poner a mi alcance todos los recursos para mi aprendizaje. En particular, a la unidad de SII por esforzarse en hacerme un poco más informática. A Aitor, por haber confiado en mí en tantas ocasiones, y a Susana, por haber aceptado la responsabilidad de la codirección de este trabajo. También a Basilio, por sus alentadoras aportaciones como codirector de la tesis desde la UPV/EHU. And to the PIMM Lab team at ENSAM, especially to Paco for your warm welcoming within your group in Paris.

Un especial agradecimiento a mis directores en la sombra. A Kerman, que aunque eres demasiado humilde para reconocerlo, eres el referente que ha marcado mis pasos y siempre aprendo algo nuevo de ti. A Eider, ha sido recomfortante compartir ideas con otra cabeza matemática pero más recomfortantes han sido tus saquitos calientes. A Iker, por animarme a empezar este camino y confiar en mis capacidades desde el inicio. A Ana, por darme la fórmula para transformar las dificultades en oportunidades. Y a Cristina por ayudarme a poner la guinda del pastel. Sin vosotros, este trabajo no sería hoy una realidad, así que, ieskerrik asko! Y... en otro orden de cosas, gracias a Ruben por ser el presidente de la esquina del hate.

Gracias a mi familia por la educación recibida, los ánimos y la confianza. Por convertirme en una mujer independiente y segura que persigue sus metas con convicción. Gracias mama, papa y tato, por estar siempre a mi lado. Os quiero mucho. También a mi abuela Carmela, por ser la mujer más fuerte que conozco y escucharme

todos los días, a pesar de no entender por qué quiero ser doctora ahora si yo siempre quise ser matemática.

A mi gran amiga en Euskadi, Antía por protagonizar los mejores recuerdos que me llevo del norte y por ser ese apoyo que tantas veces he necesitado, ¡gracias! Eres irreplicable, y menos mal ;)

Muchísimas gracias Mikel, por demostrarme que se puede, que existe una forma de hacer que las cosas pasen. Y sobre todo, por darme tu cariño y tu amor desinteresados que me han curado.

A Maria Bielsa por contagiarme con tu energía incansable de descubrir y disfrutar y esa esencia preciosa que me hace mucho bien. A Maria Villacorta perquè ja no sé quant de temps fa que m'aguantes i això es tot un mèrit. Gràcies per portar-me sempre xocolata quan em fa falta. A todos mis chicos de Tarragona (y Madrid), me quedaría sin páginas para nombraros. Vosotros sabéis quién sois. I a vosaltres, les xurris, que ens hem vist créixer. Cada uno de vosotros y vosotras sois increíbles y os quiero muchísimo. A mis amigas en la distancia Kitty y Amanda, con las que siempre encontramos un momento para dejarlo todo pausado y compartir. ¡Qué sanador es sentirnos cerca a pesar de la distancia! Un agradecimiento enorme a mi familia de la Erribera, porque no existe un lugar mejor para pasar un confinamiento que nuestro barrio y no existe mejor compañía que vosotras, mis vecinas.

En general, a todas las personas que de una forma u otra me habéis acompañado hasta aquí. A los que pasasteis rápido y a los que dejasteis huella. A los que llegasteis para quedaros y a los que tuvisteis que marchar. A los que nos vamos a reencontrar y a los que jamás volverán. A todos vosotros y a todas vosotras que seguro que me enseñasteis algo, muchas gracias de corazón.

Abstract

The integration of the Internet of Things in the industrial sector is considered a prerequisite for achieving intelligence in a company. To obtain this, AI systems with analytical and learning capabilities are required for the optimisation of industrial processes. This research work focuses on improving current approaches or proposing new ones to increase the reliability of AI solutions based on time series data and introduce them effectively in the industrial sector. This intervention is carried out in three different phases of the time series data life cycle by increasing data quality, model quality and error quality. A standardised definition of quality metrics for evaluation is proposed and these functionalities are collected in the R package `dqts` together with quality enhancement functions. Furthermore, an exploration is made of the steps to follow in time series modelling from feature extraction and transformations, a correct cross-validation, the choice of the most appropriate prediction strategy and the application of the most efficient prediction model. The KNPTS method based on the search for patterns in the historical time series is presented as a convenient R package for estimating future data. Finally, the use of elastic time series similarity measures is proposed as an alternative to quantify the performance of a regression model and the importance of using appropriate classification metrics in unbalanced class problems is emphasised. All these contributions have been validated in four industrial use cases with different characteristics from four different research fields: product quality in the agri-food sector, electricity consumption forecasting in households, porosity detection in additive manufacturing and machine diagnostics.

Resumen

La integración del Internet of Things en el sector industrial se considera un prerequisite para alcanzar la inteligencia en una compañía. Para conseguirlo, se precisan sistemas de IA con capacidades analíticas y de aprendizaje para la optimización de los procesos industriales. Este trabajo de investigación se centra en mejorar los planteamientos actuales o proponer otros nuevos para aumentar la fiabilidad de las soluciones de IA basadas en datos de series temporales e introducir las eficazmente en el sector industrial. Esta intervención se realiza en tres fases diferentes del ciclo de vida de los datos de series temporales mediante el aumento de la calidad de los datos, la calidad de los modelos y la calidad de los errores. Se propone una definición estandarizada de las métricas de calidad para su evaluación y se recogen esas funcionalidades en el paquete `dqts` de R junto con funciones de incremento de la calidad. Además, se hace una exploración de los pasos a seguir en el modelado de las series temporales desde la extracción de las características y sus transformaciones, una correcta validación cruzada, la elección de la estrategia de predicción más adecuada y la aplicación del modelo de predicción más eficiente. El método KNPTS basado en la búsqueda de patrones en el histórico de la serie temporal se presenta como un paquete de R de uso cómodo para la estimación de datos futuros. Por último, se propone el uso de medidas elásticas de similitud de series temporales como alternativa para cuantificar el rendimiento de un modelo de regresión y se enfatiza la importancia del uso de las métricas de clasificación adecuadas en problemas de clases desbalanceadas. Todas estas contribuciones se han validado en cuatro casos de uso industriales con características diferentes de cuatro campos de investigación distintos: calidad de producto en el sector agroalimentario, previsión de consumo eléctrico en viviendas, detección de porosidad en fabricación aditiva y diagnóstico de máquinas.

List of Abbreviations

- AI** Artificial Intelligence
- AM** Additive Manufacturing
- ARIMA** Autoregressive Integrated Moving Average
- CAD** Computer-Aided Design
- CI** Confidence Intervals
- CT** Computed Tomography
- DL** Deep Learning
- DQ** Data Quality
- DT** Decision Tree
- DTW** Dynamic Time Warping
- ED** Edit Distance
- EDR** Edit Distance for Real Sequences
- GBM** Gradient Boosting Machines
- IIoT** Industrial Internet of Things
- IoT** Internet of Things
- KNN** k-Nearest Neighbours
- KNPTS** k-Nearest Patterns in Time Series
- KPI** Key Performance Indicator

LCSS Longest Common Subsequence

LR Linear Regression

LSTM Long Short-Term Memory

MAE Mean Absolut Error

MAPE Mean Absolut Percentage Error

MIMO Multi-Input Multi-Output

ML Machine Learning

MSE Mean Squared Error

NDT Non-Destructive Testing

PdM Predictive Maintenance

PLC Programmable Logic Controllers

RF Random Forest

RMSE Root Mean Squared Error

RNN Recurrent Neural Networks

RSAIT Robotics and Autonomous Systems Group

SARIMA Seasonal Autoregressive Integrated Moving Average

SBD Shape-Based Distance

SMOTE Synthetic Minority Over-sampling Technique

SVM Support Vector Machines

SVR Support Vector Regression

TDA Topological Data Analysis

TWED Time Warp Edit Distance

VAR Vector Autoregression

VARIMA Vector Autoregression Integrated Moving Average

wLMD wire Laser Metal Deposition

WN White Noise

XGBoost Extreme Gradient Boosting

Part I

Fundamentals

Background

1

” *It matters little who first arrives at an idea, rather what is significant is how far that idea can go.*

— **Sophie Germain**

(Mathematician, Physicist, and
Philosopher)

Sophie Germain discovered her passion for mathematics hidden in her father’s library. Later in her life, she had to study unofficially by asking for notes from other students because she was denied access to university. She was a woman. Despite this, she managed to do important studies in mathematical analysis and number theory and made important contributions to the proof of Fermat’s last theorem, which took more than 300 years to be finally proved. She also did research in elasticity theory and won an extraordinary prize from the Paris Academy of Sciences. Germain, like many other prominent women in the history of science, had to use a male pseudonym in order to receive peer review and make her work public. Nevertheless, she went down in history with humility, defending knowledge above the author’s prominence, fame, or recognition.

Thanks to the perseverance of female mathematicians like Sophie, and the struggle of so many other women throughout history, we can make our contributions visible today and proudly sign them with our true female name. We owe the progress and advancement in science and specifically in mathematics to them, as well as being our motivation to continue researching the paths they opened up.

1.1 Introduction and Motivation

Artificial Intelligence (AI) refers to the design of computer systems that simulate human intelligence through the development of algorithms and statistical models (1). Those methods, based on mathematical theory, perform tasks such as visual perception, speech recognition, decision-making, and language translation (2; 3; 4). This revolutionary breakthrough is seen as a key technology in the digitisation of industry for the economic and social transformation of recent years.

Digitisation of Industry is defined as the integration of digital technology into the production processes and operations of industries. This process involves the use of data, digital devices, software, and algorithms to optimise production, improve efficiency, and enhance the quality of products and services (5). The goal of digitisation is to increase competitiveness, reduce costs, and improve customer experiences and has led to an exponential growth in the amount of data stored. According to that, increasingly sophisticated strategies are required to extract the maximum knowledge to truly unlock its potential (6). In the digital age, vast amounts of data are generated and collected from various sources in industries, including customers, operations, and sensors. Sensors are devices that can detect and measure physical and environmental conditions and play a key role in the definition and evolution of the Internet of Things (IoT) concept.

The IoT focuses on connecting devices through the internet and AI on providing those devices with intelligence acquired from data (7; 8). This revolutionary combination is the basis of the Industry 4.0 work environment, that combines advanced production techniques with intelligent technologies. The integration of IoT in the industrial sector is often referred to as the Industrial Internet of Things (IIoT) (9; 10). IIoT is considered a prerequisite for a company to be smart. For

example, a company that integrates IIoT into smart manufacturing indicates that it has the potential to fundamentally change the way products are designed, manufactured, supplied, used, remanufactured and ultimately retired (11; 12). To do so, the information system and production management software must be coupled and/or enriched with deep analytical capabilities (13) and learning capabilities (14).

Data collected from sensors are generally stored in databases with a time series structure, i.e. values are associated with a unique timestamp in order to sort the data over time. This type of data structure is widely used in different industrial environments. For example, time series data from sensors in a manufacturing facility can be used to track the performance of equipment over time, allowing engineers to identify patterns and trends, and optimise maintenance schedules. In the energy sector, time series data from sensors can be used to monitor energy consumption and identify opportunities for efficiency. Overall, time series data from sensors is an important aspect of the digitisation of industry, providing valuable information for decision-making and improvement (15).

AI is a very powerful tool that can support classical approaches to solve physical, logistical, or economic problems, among others, or even replace them. AI solutions are present in multiple daily applications involving time series and have become an important target of leading technology companies and research centres. Data-based techniques such as Machine Learning (ML) are emerging and accelerating that technological revolution. However, some services such as machine diagnosis or robot-human interaction are still in the research stage and growing research is focused on improving current approaches or proposing new ones to increase the reliability of solutions and introduce them effectively in the industrial sector (16). Therefore, resources in research and development of intelligent data-driven solutions for industry are focused on the pos-

sibility of improving confidence in these novel approaches. The quality requirements for the commercialisation of these solutions are high, so it is important to focus efforts on improving the quality of the information available, the development done and the results obtained.

In the international framework, the European Commission promotes AI research as one of the key strategies for advancing global challenges and industrial competitiveness. Horizon Europe is a multi-annual program for research and innovation that sets the priorities to support them ¹. AI is mentioned as a pillar to achieve more efficient and sustainable industries, advances in personalised medicine or intelligent energy systems. As a complement to these actions, the Digital Europe program covers AI with special emphasis on the digital transformation of the economy and society, developing and strengthening its capabilities.

The Basque Country has a research and innovation strategy for specialisation intelligence (PCTI 2030) ² based on the PCTI 2020 and RIS3 strategy ³, an acronym created by the European Commission that means: “Research and Innovation Strategy for Smart Specialisation“. It aims to focus the tasks of generating and exploiting knowledge on priority sectors for Basque society as can be seen in Figure 1.1: Energy, Advanced Manufacturing and Bioscience and Health based on a triple transition (technological-digital, energy-climatic and social). Its mission is to improve well-being, sustainable economic growth and employment through a research and innovation policy based on smart specialisation and improving the efficiency of the Basque Science, Technology and Innovation System.

¹https://ec.europa.eu/info/files/orientations-towards-first-strategic-plan-horizon-europe_en

²<https://www.euskadi.eus/pcti-2030/web01-a2pcti30/es/>

³<https://www.euskadi.eus/informacion/ris3/web01-a2kulind/es/>

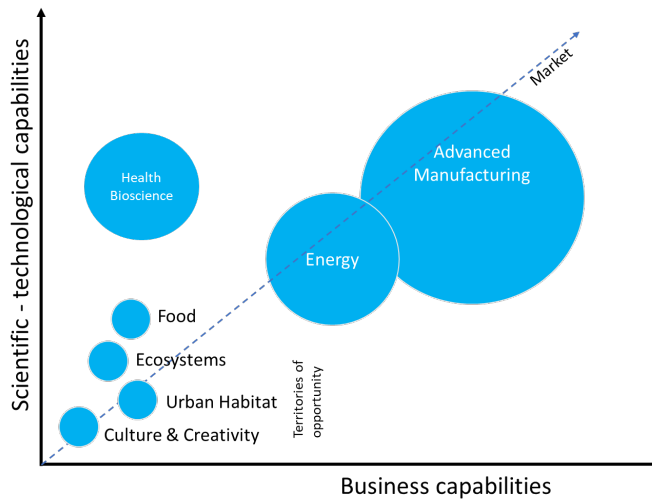


Fig. 1.1.: Smart specialisation areas in Euskadi based on the interaction of business capabilities, scientific-technological capabilities and market opportunities.

As it is mentioned, the first step to providing intelligence to machines and systems is the installation of IoT sensors that contribute to the storage of large amounts of data ordered over time. Databases store time series as structured data often at regular time intervals. The use of time series data is analogous to that of other data sets, but every stage from acquisition to deployment requires adaptations and special considerations due to their nature. This work focuses on the analysis of time series in industrial environments, with special emphasis on the differences in their treatment compared to non-ordered data sets, for the development of smarter and sustainable solutions.

For each of the problems to be solved with data-driven models, the first task after understanding the business needs is to define and collect the data. Once the information is available, the work of the data scientists and engineers goes from data acquisition to the implementation and deployment of the solution. The phases of this process can be seen in the wheel of Figure 1.2 are explained below.

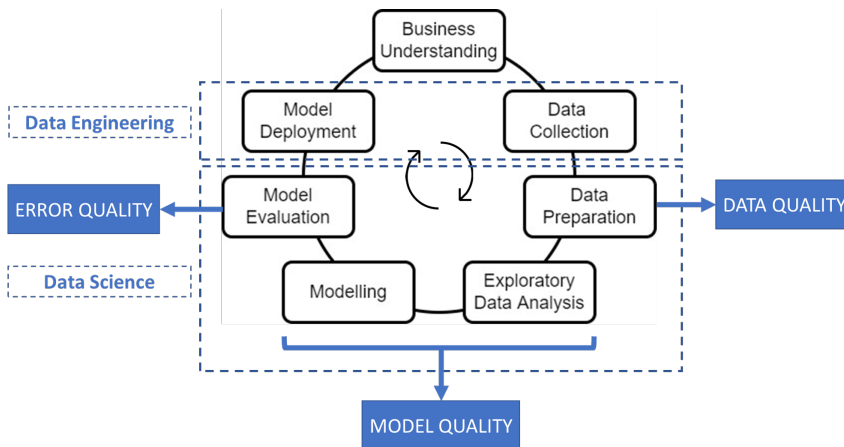


Fig. 1.2.: Time series data life cycle and the three phases explored for improving the quality and reliability of data-driven developments.

As it can be seen, once the data is obtained, the task of preparation begins, which consists of matching the data to the tools available to ensure that it can be worked with. The quality of the data collected and stored in the databases is one of the most important aspects to be controlled in this phase, prior to data analysis. Poor Data Quality (DQ) has a negative effect on AI activities, such as anomaly detection, diagnosis and/or forecasting (17). Often, the tasks of data inspection and cleaning, along with the subsequent exploratory analysis, occupy more than 80% of the data analysts' time. For that reason, facilitating data preparation and ensuring its high quality is considered a priority task in the area of data analytics.

In the next step, before generating the model, some exploratory analysis tasks are carried out that consist of choosing the appropriate information and the possible generation of extra data that can support the available data. Those tasks are known as feature engineering or feature extraction and feature selection (18). In time series, the variables that are used as inputs for the prediction often refer to date- and time-related features. The temporal relationships between the available values and the cyclical characteristics of the

features extracted from the time are the two main considerations in that step.

Once the variables have been chosen, there are several techniques to generate models that range from the classic statistical approaches (19) to the most sophisticated AI techniques and tools (20). However, the application of the latter to time series data is not direct and requires a precise treatment to avoid overfitting the models and obtaining unrepresentative results.

To decide the algorithm to be used and quantify the errors of the predictions that the final model is providing, there are metrics and performance measurements that compare the real values with the estimated ones. Some of them are more suitable than others depending on the characteristics of the problem and, in particular, not all of them are suitable for time series forecasting.

The deployment of the generated solution is the last phase and consists of the integration of the model into a system to obtain predictions, often automatically and periodically. The process represented in Figure 1.2 is cyclical because, once the model is developed and deployed, it is necessary to analyse its results and report the errors provided. In this way, the response from the solution to the initial problem can be quantified. Therefore, continuous learning and improvement of the generated models is possible, for instance from the collection of new data.

Although other steps have been studied and mentioned, this work is focused on the four phases of the data lifecycle corresponding to the data scientist's tasks. These four phases are grouped into three key blocks to improve the reliability of solutions based on time series data: the evaluation and improvement of the quality of the data received, the features and algorithms used to forecast future values and the selection of error measures used to quantify the accuracy of the models.

1.2 Hypothesis and Objectives

The main objective of this work is to increase the reliability of data-driven AI solutions for today's industry. In general, due to the characteristics of this data, this confidence is considered to increase with a proportional increase in the quality of the end-to-end proposals, i.e. the quality of the systems that are installed in industrial processes.

The three phases mentioned above are chosen to make this increase in quality, as they correspond to the tasks of data scientists in time series analysis, modelling and forecasting. In the development cycle of data-driven solutions, DQ is part of the first task of preparation and cleaning. Next, model quality is ensured by good feature extraction and algorithm choice. Finally, the accuracy measure chosen contributes to the quality of the errors provided by the model.

The hypothesis of this work is that improving the quality in these three phases will help to increase the reliability of data-driven solutions incorporating AI technologies in industrial environments. These achievements contribute to the reduction of time and resources in the analysis and to increase the explainability of the developments and results. Finally, this improvement in confidence has a positive impact on certain sectors apparently reluctant to AI relying on these technologies for the optimisation of their processes and support in decision making.

- A clear definition of the methodology and a standard usage is essential to increase the quality of time series data. For this purpose, an R package is considered appropriate to be added to a modular application. This provides speed in the calculation of quality and allows its visualization. In addition, the package would contain the implementation of useful solutions to increase quality through appropriate modifications.

- Relying on improved models through proper feature extraction, proper time series cross-validation and model selection that considers the temporality of the data helps the deployment to ensure less degradation over time.
- The comparison of different error measures, adding the elastic distances between time series, ensures a choice of the method that provides the most realistic results.

In order to validate the above hypotheses, four real case studies are chosen from different industrial fields:

- The first case focuses on the agri-food sector where a decision system had to be developed to support the production of poultry meat in order to increase the final quality of the product. This case is chosen for two main reasons: the first is that the quality of data received by farm and animal transport operators is often not adequate. Collaboration is more costly than in other areas and missing data are often abundant in procurement. This is therefore a clear case where increased DQ is needed; the second reason is that the confidence this sector has in the computing world is particularly low. The aim is to increase the quality of the models by proposing simple solutions with a high degree of applicability.
- The second case study of this work focuses on electricity consumption forecasting. The aim is to develop a system for estimating the electricity demand of buildings in order to adapt the production curves to the end use and thus optimise energy production. The quality of the data received by the sensors in these installations tends to have connection losses, causing stoppages in the acquisition and collection of high consumption peaks. In addition, it is a suitable case for testing models based on patterns due to the seasonality of these series. Finally, validating elastic error measures on these data is an innovative

challenge as static metrics predominate in the literature. For these three reasons, it is an interesting case for the validation work.

- The next case was oriented towards additive manufacturing with the aim of developing a model for the detection and estimation of the porosity of parts manufactured by wire metal deposition. This work was chosen for several reasons: firstly, in relation to the quality of the data, the cost of having labelled data in this industry leads to a lack of repeatability in the samples and, therefore, a lack of data with variability; in addition, this case is chosen because of the innovation of the data-driven approach in this industry and the challenge in the modelling phase to achieve accurate pore estimation. It is an ideal opportunity to implement balancing and classification assessment techniques on unbalanced assemblies as the number of pores in the parts represents a very small proportion of the available data.
- In order to validate the contributions made throughout this research work, it is considered appropriate to have a use case in which the data comes from a controlled scenario. Therefore, the last case consists of the diagnosis of experiments carried out in a test bed provided by the organisation of the PHME 2021 Congress. It is decided to participate because it is assumed that the quality of the data provided is adequate to arrive at an optimal solution and it is also assumed that such a solution exists. That is, the search for a fault classification model is possible using the data provided. Furthermore, the proposed evaluation system took into account the imbalance of the data and was an opportunity to validate that time series approaches have an important role in the industry.

1.3 Research Environment

During the development of this industrial thesis, I have been part of the Tekniker Intelligent Information Systems team (Eibar, Spain) that has worked in collaboration with the University of the Basque Country on different European and national projects. In addition, a doctoral stay was carried out at the Process and Engineering in Mechanics and Materials laboratory of Arts et Métiers ParisTech (Paris, France).

1.3.1 Tekniker

Tekniker is a technology research centre specialising in Additive Manufacturing technologies, Surface Engineering and ICTs, located in Eibar. Its main mission is to provide knowledge to society through I+D+i, contributing in a sustainable way to the competitiveness of business. Tekniker groups its capacities and technologies and directs them to the market to apply them to different key industrial sectors such as aeronautics, energy and infrastructure. Nevertheless, due to the history of the Basque industry, Tekniker has focused part of its dedication on the machine tool and manufacturing sector, working on the design and simulation of machines, manufacturing and optimisation of processes through monitoring the state of health of the machine.

Tekniker is organised into 12 units that work horizontally to meet the challenges presented by the industries. In particular, the Intelligent Information Systems (SII) unit is a multidisciplinary team made up of computer scientists, engineers, physicists and mathematicians working mainly for reliability and maintenance optimisation of systems. The SII Unit is focused on the integration and management

of existing data and information that enables prediction, optimisation and intelligent decision support related to different processes in industrial sectors. A member of the analytical team of the SII unit has co-led this research work, which has been carried out mainly at Tekniker's facilities.

1.3.2 UPV/EHU

The necessary knowledge for the development of data-based modelling methods was acquired through the UPV/EHU's master's degree in Modelling and Mathematical Research, Statistics and Computing. After that, during the development of this thesis, we had the tools and support of the Computer Science and Artificial Intelligence department of the Faculty of Informatics.

The direction of this work has been led by a member of the Robotics and Autonomous Systems Group (RSAIT), specialists in ML and data analysis techniques and applications, autonomous robots, robot-human interaction and computer vision.

1.3.3 ENSAM

Part of the research included in this thesis was developed in collaboration with the Processes and Engineering in Mechanics and Materials (PIMM) laboratory from Arts et Métiers ParisTech (ENSAM). During a fruitful doctoral stay from September to December 2021 in Paris, new topological techniques for modelling time series in industrial environments were learned.

The PIMM brings together a wide range of specialists ranging from the mechanics of materials and structures to metallurgy and polymer

chemistry, from forming and assembly processes to advanced methods of numerical simulation. The work focuses in particular on the consequences of the processes on the properties of use, through the defects and modifications of the generated microstructures. The activities developed in structural dynamics and in control and monitoring of systems, beyond their own justification, allow making numerous contributions to the understanding and simulation of processes.

1.4 Research Projects

The development of this thesis has been partially financed by research projects whose objectives were aligned with the main purposes of this work.

- The IoF2020 (Internet of Food and Farms 2020) ⁴ project was dedicated to accelerating the adoption of IoT for securing sufficient, safe and healthy food and to strengthen the competitiveness of farming and food chains in Europe. The IoF2020 consortium of 73 partners, led by Wageningen UR and other core partners of previous key projects such as FIWARE and IoT, leveraged the ecosystem and architecture that was established in those projects. The heart of the project was formed by 19 use cases grouped in 5 trials with end users from the Arable, Dairy, Fruits, Vegetables and Meat verticals and IoT integrator that demonstrated the business case of innovative IoT solutions for a large number of application areas. Tekniker worked in the Poultry Chain Management use case, whose aim is to increase the poultry production efficiency and comfort of the animals by designing prediction algorithms based on data extracted from the chain of raising, loading and transporting.

⁴<https://www.iof2020.eu/>

- The 3KIA (Propuesta Integral y Transversal para el Diseño e Implantación de Sistemas Confiables basados en Inteligencia Artificial) project was precisely and fundamentally oriented towards useful and practical AI, building technological bridges that face the challenges related to health, the environment, energy, transportation, security or industry. Its main contribution lies in addressing, jointly and comprehensively, each of the design factors that will enable the massive adoption of AI technologies in production sectors. Technical contributions that promote a practical, useful and responsible AI were found, with the objective of integrating methodologies that remove the barriers associated with the implementation of AI. 3KIA is not aimed at solving a specific problem or challenge in a particular RIS3 sector but rather focuses on the AI technologies themselves.
- RESPOND (integrated demand REsponse Solution towards energy POSitive Neighbourhoods) ⁵ was a Horizon 2020 project aiming to bring Demand Response (DR) solutions to neighbourhoods across Europe. In the last years, the need for energy savings has increased. However, the Demand Responses initiatives have been available only for major energy users like the industry. Some of the objectives of RESPOND included managing to influence the daily life habits of users to change their consumption, through these changes reduce peaks in demand for high energy and adjust changes to electricity production to maximise the exploitation of energy coming from renewable sources. One of the tasks in which Tekniker was involved, consisted in forecasting the electricity consumption of the users at the unit building level as accurately as possible. Later, using the results, tips or suggestions were obtained to provide users to modify their consumption.

⁵<https://cordis.europa.eu/project/id/768619>

- The REACT (Renewable Energy for self-sustAinable island CommuniTies) ⁶ project delivers a scalable and adaptable cloud-based ICT platform for renewable energy sources (RES) and storage-enabled infrastructures, supporting a holistic cooperative energy management strategy at the community level. REACT combines an optimal control of community-owned energy assets, both conventional and renewable, with cooperative DR actions, both explicit and implicit, to maximally exploit the flexibility of energy demand. To achieve that objective, Tekniker developed data-driven models using AI time series algorithms to predict short-term electricity demand. Furthermore, REACT delivers effective business models making a synergy of the grid- and community-centric approaches for sustainable RES solutions, increased renewable energy exploitation, integrated and digitalised smart grids, and DR programs. REACT solution was validated in 3 demo islands, while replication plans were made for 5 other islands across the EU.
- REDAMP (Real-time monitoring of dED Additive Manufacturing Process for zero defect manufacturing) ⁷ is a 2 years program aiming at adapting advanced on-line monitoring and Non-Destructive Testing (NDT) techniques for early defects detection, associated to AI techniques. Furthermore, it is an educational and open platform for the dissemination of Directed Energy Deposition (DED) processes and performance demonstrations. Tekniker uses proven AI methods to analyse the correlation between monitoring data and built part quality by NDT technologies (Xray, ultrasonic inspection). The system developed allows early defect detection allowing its immediate repair, avoiding material waste and providing a pathway to

⁶<https://react2020.eu/>

⁷<https://www.eitmanufacturing.eu/news-events/activities/real-time-monitoring-of-ded-additive-manufacturing-process-for-zero-defect-manufacturing-2/>

certification of large-scale metal AM technologies that use an arc welding process (WAAM) or a laser with powder (LMD).

- The AI-PROFICIENT (Artificial Intelligence for improved PROduction effICIENCY, quality and maiNTenance) ⁸ project paves the way for the integration of advanced AI technologies to the manufacturing domain through an evolution from hierarchical and reactive decision making to self-learning and proactive control strategies. The proposed approach is underpinned by predictive and prescriptive AI analytics at both component and system levels, by cross-fertilising edge and platform AI, while leveraging human knowledge and feedback for reinforcement learning (human-in-the-loop). AI-PROFICIENT aspires to bring advanced AI technologies to the manufacturing and process industry while improving production planning and execution, and facilitating the collaboration between humans and machines. Taking full advantage of AI capabilities and human knowledge, AI-PROFICIENT will develop proactive control strategies to improve the manufacturing process over three main vectors: production efficiency, quality and maintenance. AI-PROFICIENT will increase the positive impact of AI technology in the manufacturing process at two pilot sites, under different scenarios of significant economic value.
- The PRENERGET project aims to create a software service for automatic periodic forecasts, evaluation of prediction errors and retraining of the models in production if necessary. Although the design consists of a modular tool that accepts any time series data, the focus of the project is on building electricity demand data.

⁸<https://ai-proficient.eu/>

1.5 Contributions

Throughout this work, a reflection is made on the procedure of analysing, modelling and forecasting time series data. The contributions are organised into three groups corresponding to the three phases of the analysis cycle mentioned above.

1. This research work highlights the negative impact caused by low DQ on the conclusions drawn in the analysis of data and the generation of forecasting models, both at the theoretical and applied level. It addresses the need to unify the concepts that have been emerging in recent years and defines the dimensions in which they should be measured. For time series, and specifically for data received by sensors, the need for specific metrics to quantify the quality of the data sets being received is detected. For this reason, a quality module is designed that can be integrated into industrial systems for real-time integration. The module consists of an R package containing a set of metrics that provides information on the quality of the process data in different dimensions of interest, quantified using indicators between 0 and 1. Correction functions are also programmed to manipulate the database values and prevent future analyses from being influenced by errors in the capture. All this development is available on GitHub, i.e. the R package is openly available to the entire community⁹. This facilitates the implementation of quality indicators in data analysis studies carried out with this software. All these advances in time series DQ were published in 2022 in the Q1 quartile IEEE Access journal under the title *On the Evaluation, Management and Improvement of Data Quality in Streaming Time Series* (21).

⁹<https://github.com/MeritxellGomez/dqts-R-package>

2. The second phase of this work pursues the hypothesis that statistical modelling or fitting of AI models in time series requires a different treatment from that of data without temporal order. This work visualises this need by comparatively demonstrating the changes in the efficiency and effectiveness of the models after considerations related to the expected behaviour of the time series.

In the paper entitled *An IoT Platform towards the Enhancement of Poultry Production Chains* published in 2020 in the Q1 quartile *Sensors* journal, a proposal was made to extract time series features from different sources and then combine them in a decision tree algorithm (22). This helped to facilitate decision-making in an unpromising scenario due to the lack of records and highlighted the importance of extracting and selecting appropriate features.

In the feature extraction phase, the importance of transforming time variables with sine functions is demonstrated to allow distance-based algorithms to use these inputs realistically. Furthermore, as a contribution to time series modelling, an algorithm has been proposed for forecasting future time series data based on the patterns found in the historical data. This method is called *k-Nearest Patterns in Time Series (KNPTS)* and has been developed in R. Its use is open to the whole community as the functions were collected in a package that was published in Github ¹⁰. Both, the proposed cyclic transformations and the use and effectiveness of the KNPTS method were presented at the SGAI 2021 congress where it was awarded as the best application paper by a student and was published in the proceedings as *Short-term Forecasting Methodology for Energy Demand in Residential Buildings and the Impact of the COVID-19 Pandemic on Forecasts* (23). Later, the work was extended and further

¹⁰<https://github.com/MeritxellGomez/knpts-R-package>

studied for publication in the Q1 quartile journal *Energy and Buildings* under the title *k-Nearest patterns for electrical demand forecasting in residential and small commercial buildings* (24).

Another contribution to the extraction of time series characteristics is Topological Data Analysis (TDA). The application of topological methods for time series modelling contributes to the assertion that the occurrence of pores in additive manufactured parts is a local event, which cannot be anticipated. Therefore, the detection of such faults using supervised classification techniques can be done in a time-independent manner, which demonstrates that in some specific industrial cases, the use of time series is limited to the nature of the data and events.

Finally, in multivariate time series classification, it is proposed to use observation-by-observation classification algorithms that provide a probability of belonging to each class. With the support of a Kalman filter-type algorithm, the classes are propagated to obtain a predominant class in each experiment. This innovative contribution was published in the proceedings of the PHME 2021 conference under the title *Divide, Propagate and Conquer: Splitting a Complex Diagnosis Problem for Early Detection of Faults in a Manufacturing Production Line*, after winning the Data Challenge of the same year (25).

3. The third phase of this work is focused on the evaluation of the models trained, i.e. measuring the quality of the results provided by the supervised models. Elastic time series similarity measures such as Dynamic Time Warping (DTW) and Edit Distance for Real Sequences (EDR) are proposed as a more suitable alternative to measuring the accuracy of regression forecasts in some cases. Their use is encouraged in problems where the frequency of acquisition is high, the data history is large and there is a periodicity in the behaviour of the output

(23; 24). In classification problems, the use of different precision metrics in cases of unbalanced data is emphasised and the effect of each of them is studied with this objective. In classification problems, it highlights the use of different accuracy metrics in cases of unbalanced data and compares some decisions that would be made using Accuracy and Recall in different scenarios. In unbalanced data, the use of metrics that penalise this inequality is emphasised (25).

1.6 Thesis structure

After a general introduction and contextualisation that is done in this chapter, the rest of the thesis is structured as follows.

Chapter 2 presents the main contributions done in time series analysis in industrial environments. This chapter aims to collect proposals for improving the reliability of intelligent solutions based on time series data. Initially, definitions and basic concepts of time series are introduced in Section 2.1. Then, the objective is addressed with the three-pronged approach of improving the quality of data, models and errors. First, a proposal for measuring DQ in time series data can be found in Section 2.2. In addition, the functionalities of the R package `dqts` implemented for the treatment of time series DQ are presented. Section 2.3 presents the procedures to follow for modelling time series using AI algorithms that allow both continuous forecastings in regression and classification problems. Finally, in section 2.4 a reflection is made on how to assess the performance of models based on time series data, both in the case of regression and classification.

Chapter 3 is focused on industrial applications of time series analysis. The contributions proposed in the previous chapter are applied in four real areas of national and international research trends: poultry

chain control management, energy demand forecasting, porosity detection in additive manufacturing and manufacturing production line diagnosis.

This work is concluded in Section 4 and new lines of research that arise after the end of this project are proposed.

During the research process, scientific contributions have been made concerning the issues addressed in this study. The main ones in chronological order of publication are listed below.

The article titled *An IoT Platform towards the Enhancement of Poultry Production Chains* (22) in Section 5 explains a support decision system based on data collected across different phases of the poultry production chain. Through the extraction of Key Performance Indicator (KPI), a decision tree model was trained that allowed the classification of the quality of poultry meat based on the historical information collected.

Section 6 presents the research developed in a European project focused on demand response. The article entitled *k-Nearest Patterns for Electrical Demand Forecasting in Residential and Small Commercial Buildings* presents a comparison of the effectiveness of predicting future electricity consumption values with two different algorithms.

The contributions made to the quality of time series data are detailed in the article *On the Evaluation, Management and Improvement of Data Quality in Streaming Time Series* (21) found in Section 7. In that work, the metrics needed to quantify the quality of data in time series are explained. It was implemented in R and a library open to the community was published. In addition, the importance of controlling and rectifying the quality of the data is highlighted and the use of this tool in the industry is proposed.

The last impact contribution in this work is titled *Topological Data Analysis and Supervised Classification for Porosity Prediction in wire*

Laser Metal Deposition Processes. Section 8 contains the work carried out during the international stay in Paris. The article details how to detect the internal pores that appear in metal parts manufactured using a wire laser metal deposition process. Using historical data from additive manufacturing processes, a model capable of detecting pores with high precision was fitted, thus being able to replace classic tomography with a predictive model based on data.

Finally, in Section 9 the proceeding papers of congresses in which it has participated during this research and the awards that it has received are added.

Contributions to Time Series Life Cycle

” *Escriure bé costa. Per escriure bé entenc dir amb la màxima simplicitat les coses essencials*

— **Mercè Rodoreda**
(Catalan novelist and poetess)

The analysis of experimental data observed over time leads to new and unique problems in statistical modelling and represents a particular field of study in the AI paradigm. In this subject, the acquired values require methods derived from this temporal characteristic instead of being treated as independent observations.

In industry, the life cycle of a time series starts with the installation of sensors that capture the data necessary for the study to be carried out and ends with the implementation of the developed solution. This process can be long and complex and for a correct drawing of conclusions, the temporal characteristics of the data should be properly managed. As it is mentioned in Section 1.1, there are essential steps in the analysis of time series that correspond to data scientist tasks that are considered essential for their correct interpretation.

As the Catalan writer Mercè Rodoreda said, "Good writing is difficult and writing essential things with the utmost simplicity is a hard task". Considering this, with the aim of synthesising the concepts and facilitating understanding, the contributions in this Chapter are

structured around three key parts of the time series life cycle to improve the reliability of AI systems involving them: data quality, modelling and, evaluation.

Section 2.1 covers the primary concepts of time series data. These definitions are essential for establishing the notation that is used from this Section onwards and for understanding the essence of time series data sets.

Section 2.2 focuses on time series DQ and provides a comprehensive definition of metrics classified into five dimensions. These metrics aid in quantifying the DQ for processing and subsequently improving the quality using corrective functions. Additionally, Subsection 2.2.1 elaborates on the R `dqts` package, which encompasses all of these functionalities.

Section 2.3 is dedicated to improving the reliability of data-driven models based on time series data. It commences in Subsection 2.3.1 with an introduction to classical models and in Section 2.3.2 to ML models, with a detailed explanation of the KNPTS algorithm and the `knpts` R package presentation. Subsequently, Subsection 2.3.3 emphasises the use of cross-validation, Subsection 2.3.4 focuses on creating appropriate features and their proper handling, and finally, Subsection 2.3.5 discusses various forecasting strategies and their impact on model development.

Finally, in Section 2.4 there is a reflection on the various error measures available to evaluate prediction results. On one hand, in Subsection 2.4.1, elastic similarity measures for time series are introduced to quantify errors in regressions. On the other hand, in Subsection 2.4.2, the impact of different techniques for measuring classification accuracy in the case of imbalanced class data is explained.

2.1 Time Series Basics

Time series is a set of ordered data expressed by $\{Y(t)\}$ for $t = 1, \dots, T$, where Y represents the measurement variable and the index t is the time that is measured from instant 1 to instant T . In some cases, Y_t is used instead of $Y(t)$.

The primary classification of time series pertains to the number of variables measured over time. Univariate time series pertains to a data set with a single time-dependent variable, while multivariate time series pertains to a data set with more than one time-dependent variable. The analysis of both univariate and multivariate time series is a critical tool in various fields, such as finance, economics, engineering, and environmental science. While some definitions, such as the correlation between variables, are specific to multivariate time series, numerous concepts are elucidated using univariate time series as they can be effortlessly extended to multivariate time series by applying them to each measured variable.

A univariate time series can be decomposed into various components (26).

- **Trend:** The long-term direction of the data. It reflects the overall pattern or tendency of the data to increase or decrease over time. The trend component at time t can be represented by $T(t)$.
- **Seasonality:** The regular and repeating pattern of fluctuations in the data that occur over fixed intervals over time, such as daily, weekly or monthly. This can be due to factors like weather, holidays, or sales cycles. $S(t)$ represents the seasonality component at time t .
- **Cyclical:** The irregular pattern of fluctuations that occur at non-fixed intervals over a period of time longer than the seasonal

pattern. The notation to represent the cyclical component at time t is $C(t)$.

- **Residual or Error:** The random fluctuations or noise that is left over after the other three components have been accounted for. These can be caused by unforeseen events or measurement errors. The residual component or error term at time t is represented by $e(t)$.

A stationary time series is a time series where the statistical properties, such as the mean, variance, and autocorrelation, do not change over time. If a time series is not stationary, it can be transformed into a stationary time series by applying techniques such as differencing or logarithmic transformation. In particular White Noise (WN) is a type of random signal that has equal intensity at all frequencies, meaning it has no correlation or structure in the time series data. In other words, the observations are generated independently and identically from a distribution with a constant mean and variance.

Understanding the different components of time series data is important for analysing and forecasting future trends. Different statistical techniques and models can be used to identify and decompose a given time series into its components, allowing for more accurate predictions and insights into the underlying patterns of the data.

The most common method for time series decomposition is the additive model, which assumes that the time series can be expressed as the sum of its four individual components. Then, let $Y(t)$ be the value of the time series at time t , the additive decomposition is expressed in Equation 2.1.

$$Y(t) = T(t) + S(t) + C(t) + e(t) \quad (2.1)$$

Similarly, the multiplicative decomposition model assumes that the time series can be expressed as a product of its components, as shown in Equation 2.2.

$$Y(t) = T(t) \cdot S(t) \cdot C(t) \cdot e(t) \quad (2.2)$$

Figure 2.1 shows the additive decomposition of a time series of monthly gas consumption data for Australia. From top to bottom: the original time series $Y(t)$, its trend component $T(t)$, the seasonal component $S(t)$ and the residual term $e(t)$. This residual component should be stationary or as stationary as possible given the characteristics of the time series. In the trend graph, we clearly see an incremental component that reveals the increasing behaviour of the data. In the seasonal chart, the same pattern of length 12 is repeated over time in the data, so it is understood that the gas consumption series has an annual seasonality. This means that the data received from year to year share a similar shape. In the graph below, the residual component, also known as the error term or random term, is the result of subtracting trend and seasonality from the original series by the operation of Equation 2.3.

$$e(t) = Y(t) - T(t) - S(t) \quad (2.3)$$

In the case of the multiplicative decomposition shown in Figure 2.2, the error or residual component has a more stationary behaviour than in the additive case, and it is obtained by Equation 2.4.

$$e(t) = \frac{Y(t)}{T(t) \cdot S(t)} \quad (2.4)$$

In essence, a time series is an ordered collection of data gathered at regular time intervals that can be decomposed into the four men-

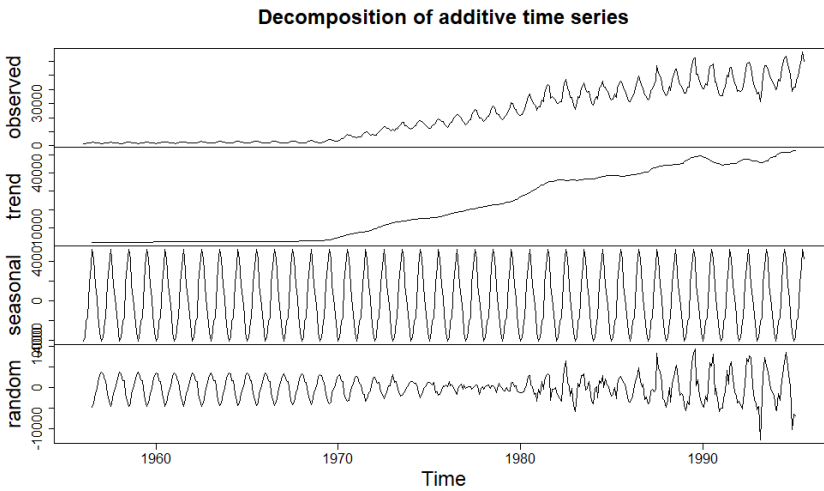


Fig. 2.1.: Additive decomposition of monthly gas consumption time series for Australia.

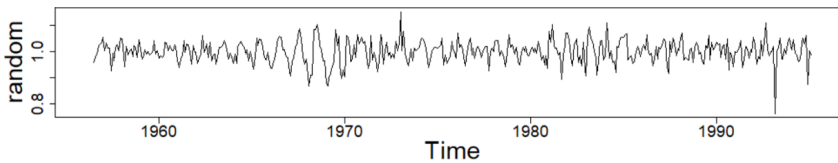


Fig. 2.2.: Multiplicative decomposition of monthly gas consumption time series for Australia

tioned components of trend, seasonality, cyclical, and residual or error.

2.2 Data Quality in Time Series

Data is a powerful source of information, yet despite its pervasive presence in all industrial domains today, the correct use of data for optimal exploitation remains a challenge for data scientists to develop intelligent data-driven solutions. Part of the difficulty of integrating AI solutions into industrial systems lies in the quality of the data

collected. The goal of DQ is widely pursued due to the increase in the amount of data that companies store in their databases. With the popularisation of data-driven AI solutions, organisations are aware of the importance of measuring their quality to identify errors and face losses. The focus is set, thus, on reporting and improving DQ to improve the reliability of industrial databases.

Understanding and measuring DQ with the right tools is necessary to improve outcomes and increase confidence in data-driven decisions (27). The development of such tools to measure the quality of time series data requires a clear definition of the metrics that provide values for DQ quantification. However, inadequate definitions of DQ are prevalent throughout research and contributions in various fields pertaining to DQ. Whereas some sources attempt to address the wide variety of possible cases, others provide definitions for specific fields and rarely the proposed approach can be used in other areas. In short, there is a need for a concise definition that is both sufficiently concrete to deal with the particularities of time series data and sufficiently broad to be applied to all areas of research involving time series, without the influence of the application or solution to be sought with the data set.

One of the most common approaches in the search for a method of measuring DQ is the use of metrics that quantify different areas of quality. In other words, the aspects to be assessed are organised into dimensions and each dimension, is composed of a set of metrics whose values indicate the quality of the data. Regarding those, at the beginning of the 20th century, dimensions such as *Accuracy*, *Timeliness*, *Interpretability* and *Accessibility* were introduced, which are the basis of future definitions to measure DQ (28; 29). Nowadays, the use and definitions of the dimensions for measuring DQ have evolved according to industry requirements and the characteristics and quantities of data available (30).

For the correct definition of the metrics, maximum domain knowledge and a complete understanding of the issues involved are required. Then, the need to define complete methodologies for the study of DQ arose, including the need to assess and improve DQ and the possibility to monitor these methodologies (31; 32).

To fulfil this requirement, a new approach is proposed, which is motivated by the recognition of the need to monitor quality metrics in time series. This approach ensures a high DQ over time through the capability of correcting quality issues (33). In this case, DQ is defined as a combination of the quality scores obtained on 11 different metrics taking values from 0 to 1, where 0 represents the lowest quality and 1 is the highest quality.

Those metrics are organised into 5 dimensions which cover the needs identified in the literature in measuring the quality of time series data: *Conformity*, *Uniqueness*, *Timeliness*, *Accuracy* and *Completeness*. These dimensions and metrics are described below.

- *Conformity* dimension measures the amount of data correctly stored according to the database standards. Two metrics are used to quantify quality in this regard: *Names* and *Formats*. The *Names* metric measures the proportion of variables labelled with the correct name. In this sense, according to the data standard, it is expected to receive a variable with an identifying label, data that maintain that label over time are considered high-quality data. As for *Formats*, the data received may be of different typologies. For instance, variables can be measured using numerical values or categorical labels can be received for certain information. In this context, the *Formats* metric calculates the proportion of data that retains the typology corresponding to the standard. Thus, a high score on this metric indicates that there has been no change over time in the type of data being received.

- *Uniqueness* dimension is widely used to control repetitions in data. In the case of multivariate time series, repeated measurements are generally assumed in most of the variables recorded. However, repetitions in time measurements are not allowed, and, thus, different data cannot be stored in the database at the same time instants. Therefore, *Time Uniqueness* metric is defined, which measures the proportion of unique timestamps throughout the data set.
- *Timeliness* dimension defines the temporal possibilities provided by the data in terms of their frequency of acquisition. In this case, there is a metric with the same name, *Timeliness*, which quantifies the time delays by measuring the proportion of registrations that are received without exceeding the expected waiting time.
- *Accuracy* dimension is the degree of reproducibility of measured values that may or may not be close to real-world values. Four metrics are defined in this dimension: *Range*, *Consistency*, *Typicality*, and *Moderation*.

In general terms, *Range* is a metric that measures the proportion of values that lie between the allowed values for each of the variables collected. It is therefore necessary to set a maximum and minimum value in advance to decide whether the information lies within these ranges.

Variables that follow a Gaussian distribution have values that are concentrated around their mean with a certain degree of dispersion. This property enables the calculation of Confidence Intervals (CI), which are boundaries established to ensure that a certain percentage of the values fall within them. *Consistency*, *Typicality*, and *Moderation* metrics are the proportion of values that anomalously lie outside the intervals with 80%, 95% and 99% confidence levels. An example of a univariate time

series with a Gaussian distribution and three CI can be seen in Figure 2.3.

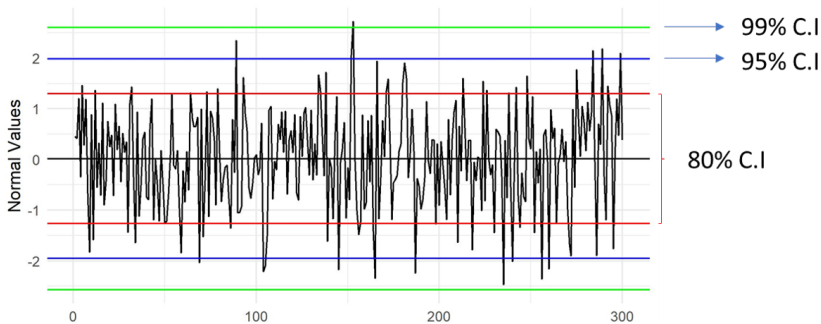


Fig. 2.3.: Example of simulated values following a Gaussian distribution and the three boundaries of the CI with confidence levels of 80%, 95% and 99%

- *Completeness* dimension refers to the degree of present elements in the data set and it is one of the most important points of DQ. The first metric, named *Completeness*, indicates the proportion of elements received in the data set. In other words, it is a complementary indicator to the overall missing value ratio. Due to the importance of knowing if the information on a particular variable has stopped being received at some point in time, *Completeness by Variables* metric is defined. This indicator is the proportion of variables present in the data set with respect to the number of variables expected to be received. Similarly, the *Completeness by Observations* metric is also defined, which measures the proportion of records received against those expected to be received.

To sum up, 11 different metrics from 5 dimensions have been defined to measure DQ as can be seen in Table 2.1. The overall indicator of the data set quality is obtained by a weighted combination of these metrics. This technique allows giving more importance to specific metrics by assigning them a higher coefficient.

Tab. 2.1.: Summary of problems identified using each of the 11 proposed DQ metrics grouped in 5 dimensions.

Dimension	Metric	Problem identification
Conformity	Names	Wrong variable names
	Format	Different data formats
Uniqueness	Time Uniqueness	Repeated timestamps
Timeliness	Timeliness	Excessive waiting times between observations
Accuracy	Range	Values out of range
	Consistency	Values out of the 80% CI
	Typicality	Values out of the 95% CI
	Moderation	Values out of the 99% CI
Completeness	Completeness	There are missing values
	Completeness by Observations	Some observations are lost
	Completeness by Variables	Some variables are lost

2.2.1 The `dqts` R package

The development of the DQ metrics defined in the previous section was carried out with the statistical software R. The `dqts` R package was developed to address the need for assisting data analysis in handling low-quality data with a robust procedure. The `dqts` R package is available in GitHub for its use by the entire R community¹ and its implementation and use is explained in more detail in Appendix A. In summary, it contains four main functions:

- The `DQ` function provides the value of each quality metric and the total quality indicator obtained from the weighted combination of all metrics. It calculates DQ metric values in two alternative ways: in the whole time series or by moving windows.
- The `plotDQ` function allows the visualisation of the quality metric scores and receives as input the output of `DQ` function. There are two types of graphs depending on whether the `DQ` function has been run on the complete data set or per window. In the first case, the function shows a bar chart with a single

¹<https://github.com/MeritxellGomez/dqts-R-package>

score value for each metric, and in the second case, it displays a line chart to see the time evolution of the metric values.

- The `deepDQ` function provides details about failures in the indicated metric. There are two ways to execute the function: one to provide the value of quality metrics for each variable, and another to determine the position of the values that cause low quality in a specific metric.
- The `handleDQ` function estimates solutions to faults and returns a data set with the necessary changes for that metric to achieve the highest quality score. For each metric, there are different correction functions available. These functions range from removing low-quality values to using various techniques for imputing anomalous values with other values that improve data quality. The correction techniques are explained in Appendix A.

The `dqts` package can be integrated into applications with an interactive user interface, that allows the selection of different actions related to the execution of those four main functions. In this way, the user is always aware of the evolution of the DQ index. Additionally, the reasons behind the decline of the index are displayed, and the user is allowed to choose the methods for resolving poor quality. Real-time checks are performed to observe the influence of their choices on the final quality of the data. An example implementation is shown in Figure 2.4, where the different metrics to be treated are displayed on the left along with a drop-down tab to select the resolution methods. In the centre of the screen, the status of the data after each of the decisions made is displayed.

Among the advantages of the `dqts` package, the calculation of the quality of time series data is standardised to provide a numerical value that allows the comparison of quality between different sets. Furthermore, quantifying quality enables data users to discern when data meets or not the minimum quality requirements. In addition,



Fig. 2.4.: Example of integration of the dqts package in an industrial system for the evaluation and correction of manufacturing data.

its deployment as a package allows for integration into systems that facilitate the use of corrective functions to increase DQ.

Figure 2.5 shows an example of low-quality data with completeness and temporality failures, and what a prediction using a time series model (Autoregressive Integrated Moving Average (ARIMA)) that learns the behaviour of the data from this set would look like. As can be seen, the future forecast cannot achieve adequate accuracy because the quality of the input data was low.

However, Figure 2.6 shows how the same algorithm manages to estimate the future data more accurately after the use of the time series quality correction functions. In this case, the out-of-range value and missing values have been estimated using the KNPTS. After the changes, the algorithm learns the behaviour of higher-quality data, enabling it to make more accurate predictions that align better with the result that would have been obtained with error-free data.

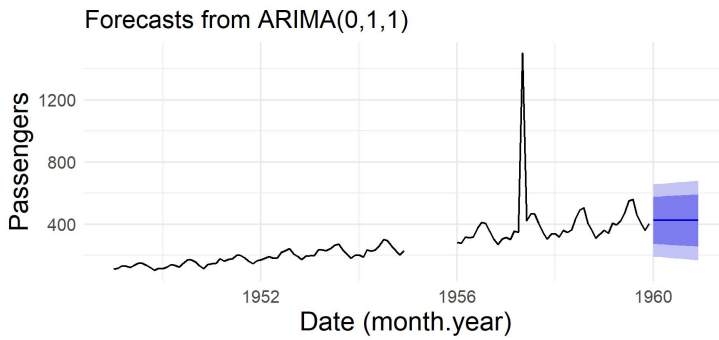


Fig. 2.5.: One year ahead forecasting training ARIMA model with 0.95 in Completeness and 0.99 in Range

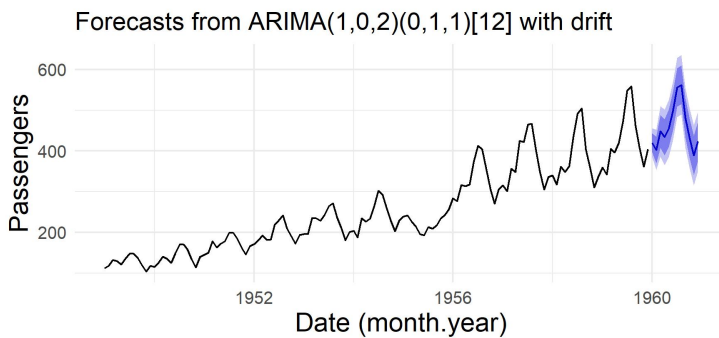


Fig. 2.6.: One day ahead forecasting using fixed data with the highest quality.

2.3 Time Series Modelling

Time series analysis is an essential method in emerging research fields such as food health, energy demand forecasting or smart manufacturing (19). As discussed above, recent monitoring of different areas of industry leads to the capture of sensor data and these time-ordered data require specific time series techniques for their analysis. In many cases, the application of classical analytical methods without taking into account the time structure can lead to erroneous decisions that negatively influence the reliability of the models. In general,

time series data cannot be treated as independent and identically distributed data as in other types of analysis.

Time series modelling is a process to estimate a mathematical expression for the calculation of one or more collected variables in order to make forecasts. Once such a model is found, it can be used for forecasting future time series data and thus use the estimated values for different purposes in the industry. Time series models typically involve the use of time lags, seasonality, and trend components, which are estimated from historical data and used to make predictions for future time periods. There are different approaches to modelling time series. On the one hand, classical techniques for time series regression are based on statistical models and, on the other hand, the most up-to-date ML models. Statistical analysis often assumes linear relationships between input and output variables or normal sampling distributions, but these assumptions are not always fulfilled, especially in industrial environments. The classical regression models that assume the linearity of time series data are ARIMA models and their derivatives. As for ML models, they can deal in a general way with non-linearity. Examples of these ML models are unsupervised clustering algorithms or k-Nearest Neighbours (KNN) and Support Vector Machines (SVM) as supervised algorithms. In addition, ML methods learn from past experiences based on the actual data collected and contemplate more complex scenarios that fit reality more accurately in most cases. The following Sections describe both approaches in detail.

2.3.1 Tradicional Forecasting Methods

The most common time series regression models are the ARIMA (AutoRegressive Integrated Moving Average) model and its variants (34), which combine both autoregression and moving average components to model time series data.

Autoregression (AR) models use past values of the time series $Y(t-1)$, $Y(t-2)$, ... as input to predict future values $Y(t)$. The notation $AR(p)$ indicates an autoregressive model of order p . and it is expressed by Equation 2.5

$$Y(t) = \phi_1 Y(t-1) + \dots + \phi_p Y(t-p) + e(t) \quad (2.5)$$

where $e(t) \sim WN(0, \sigma^2)$ and ϕ_i constants for $i = 1, \dots, p$.

As for Moving Average (MA) models use the error term (residuals) from a time series forecasting model $e(t-1)$, $e(t-2)$, ... to predict future values $Y(t)$. The notation $MA(q)$ is used to a moving average model of order q as in Equation 2.6.

$$Y(t) = e(t) + \theta_1 e(t-1) + \dots + \theta_q e(t-q) \quad (2.6)$$

with $e(t) \sim WN(0, \sigma^2)$ and θ_j constants for $j = 1, \dots, m$.

Then, the ARIMA model consists of three components:

1. The Autoregression (AR) model, uses the dependent relationship between an observation and some number of lagged observations to make predictions.
2. The Integration (I) model, which takes differences between observations to make the time series stationary.
3. The Moving Average (MA) model, uses the dependency between an observation and a residual error from a moving average model and applies it to lagged observations.

ARIMA models are typically denoted as $ARIMA(p, d, q)$, where p is the order of the autoregressive part, d is the degree of differencing (the number of times the data have had past values subtracted from them), and q is the order of the moving average part. The mathematical expression for ARIMA models is shown in Equation 2.7.

$$Y(t) = \phi_1 Y(t-1) + \dots + \phi_p Y(t-p) + \theta_1 e(t-1) + \dots + \theta_q e(t-q) + e(t) \quad (2.7)$$

Seasonal Autoregressive Integrated Moving Average (SARIMA) models are similar to ARIMA models but can capture both the autocorrelation and seasonality of a time series by Equation 2.8.

$$Y(t) = \phi_1 Y(t-1) + \dots + \phi_p Y(t-p) + \theta_1 e(t-1) + \dots + \theta_q e(t-q) + \varphi_1 Y(t-s) + \dots + \varphi_n Y(t-sn) + \vartheta_1 e(t-s) + \dots + \vartheta_m e(t-sm) + e(t) \quad (2.8)$$

where $\varphi_1, \dots, \varphi_n$ are the seasonal AR parameters, which represent the effect of the past seasonal values of the time series on its current value; $\vartheta_1, \dots, \vartheta_m$, are the seasonal MA parameters, which represent the effect of the past seasonal error terms on the current value of the time series; and s is the number of time periods in a season.

There are several advantages of using ARIMA models for time series analysis. The ARIMA model is very flexible and can be applied to a wide range of time series data, including data with trend and seasonality. They are also relatively robust to non-stationary data and simple to implement and interpret. That simplicity and the non-precision of lengthy parameter estimation processing also make them very computationally efficient models and can be very accurate on data that are not strongly influenced by external agents. Finally, ARIMA models can handle missing values and can interpolate them, which is useful when the time series data is not complete. However, ARIMA models make strong assumptions about the underlying data, such as stationarity and linearity, which may not always hold in practice (35). They have other limitations, such as difficulty to handle exogenous variables and complex data patterns, or a limited ability

to handle non-stationary and non-normal data. In addition, autoregressive models require a more frequent release of forecasts due to the accumulation of errors in long future forecasts. The following lines provide an example that illustrates this phenomenon.

Daily data on electricity demand in Spain are used to adjust a SARIMA model because a weekly periodicity is detected ². The data started to be recorded on 1 January 2004 and up to 31st December 2012. It is decided to use the data prior to 1st January 2011 for training and to use the consecutive data to illustrate the effect of the accumulation of forecast error.

In Figure 2.7, the fitted SARIMA model is used to launch a single prediction for a time window of one week, i.e. the 7 values after the series are predicted by using the data up to 31 December 2010. The forecast shown in red in the graph is in line with the actual values and maintains its trend. In addition, the grey shading indicates the 95% CI that has been obtained for this forecast.

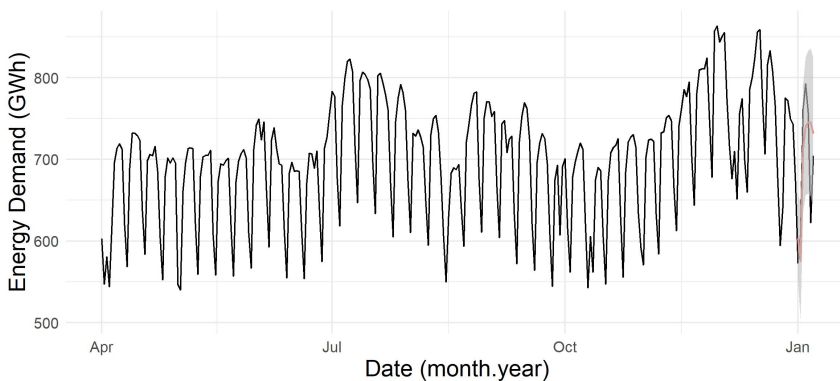


Fig. 2.7.: One-week-ahead forecasting in one execution using SARIMA model for energy demand.

Autoregressive models are not suitable to estimate the future value of a quantity of data greater than the periodicity of the time series. For

²<https://rpubs.com/Peque/energy-demand-forecast>

instance, the ability to extrapolate is lost when a ten weeks ahead forecast is requested, due to the weekly periodicity of the data. As can be seen in Figure 2.8, this type of algorithm assumes that the estimate made for the first 7 days is valid for the rest of the weeks and makes an approximate copy of that pattern. In addition, the grey shading spreads over time, leading to an accumulation of errors and a loss of prediction accuracy.

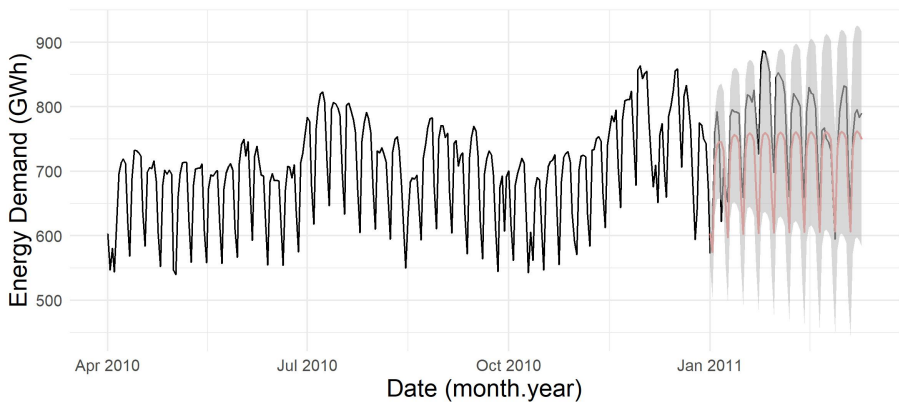


Fig. 2.8.: Ten-weeks-ahead forecasting in one execution in weekly data using SARIMA model for energy demand.

The solution to prevent such phenomena is the periodic release of the forecast model. In this approach, each model run predicts a portion of the future data and uses all the available prior actual values. If only one of the future values is predicted in each iteration, none of the estimated values is used as input. The forecasts made with this methodology are shown in Figure 2.9, in which the graph shows the forecasts for the next 10 weeks with significantly higher accuracy than in the previous case. In addition, this type of approach avoids error propagation, which remains approximately constant. The prediction window and the deployment mode of the model may influence the decision of the model to be used, despite their accuracy results.

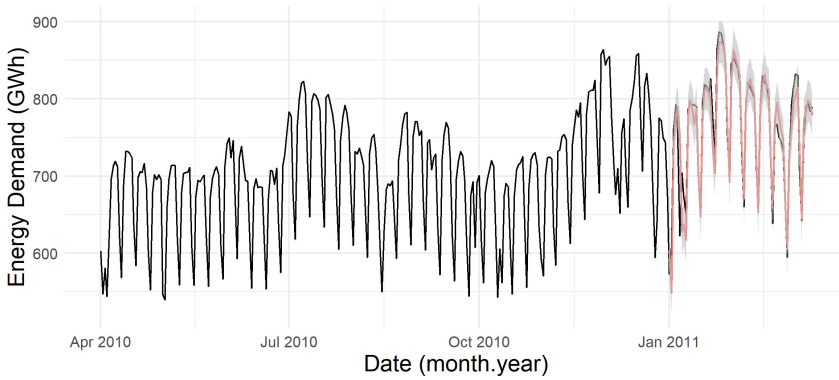


Fig. 2.9.: Ten weeks ahead forecasting in seventy executions in weekly data using SARIMA model.

ARIMA models are widely used in various industries for time series analysis and forecasting, such as: finance, to forecast stock prices, currency exchange rates, and interest rates; economics, to analyse and forecast economic time series data such as inflation and unemployment; or marketing to analyse and forecast time series data such as sales, customer behaviour, and market trends. ARIMA models can be also used in the energy domain to forecast energy consumption, as seen in the example below. These models are able to identify trends in energy usage and optimise the operation of power plants and other energy infrastructure. However, such techniques are efficient when data have an identifiable trend and seasonality and do not depend on other external factors that are not included in the model. This is the main reason why the use of ML models on time series data has become widespread in a variety of complex environments.

However, the study of ARIMA models and their extensions continues in order to adapt them to industrial environments. This is the case of the ARIMAX (AutoRegressive Integrated Moving Average with exogenous inputs) model, which allows the inclusion of additional exogenous variables, which may help improve the model's predictive accuracy (36). Also, Vector Autoregression (VAR) models

consider multiple time series as input and are useful for modelling relationships between multiple time series and Vector Autoregression Integrated Moving Average (VARIMA) models consider multiple time series as input and have both autoregression and moving average components to model time series data (37). Finally, State-Space models (SSM) are a class of time series forecasting models that are based on the state-space representation of a system, which is a mathematical model that describes the internal state of a system and how it changes over time (38). It is commonly used in a variety of fields, including engineering, physics, economics, and finance, to model complex systems that are not easily described by a simple mathematical equation. A state-space model consists of two equations: the state equation and the observation equation. The state equation describes how the state of the system evolves over time, while the observation equation describes how observations of the system are related to the underlying state.

In combination with state-space models, the Kalman filter is introduced as a predictor for time series. The Kalman filter is a powerful mathematical algorithm for estimating the underlying state of a time series in the presence of noise (39). It is widely used in control systems, signal processing, and risk management in finances. In the context of univariate time series analysis, the Kalman filter can be used to estimate the state of a time series given noisy observations. The state vector in the Kalman filter for univariate time series analysis includes the true value of the time series at each point in time, as well as the error terms for the AR and MA components of the model. The observation vector includes the noisy measurements of the time series. The Kalman filter operates by combining information from the previous estimate of the state, the new observation, and the model equations and recursively updating its estimate of the state based on the available observations and the model equations.

The Kalman filter can be used to generate both filtered and smoothed estimates of the state of the time series. The filtered estimate is the estimate of the state at each point in time based on all the available observations up to that point. The smoothed estimate is the estimate of the state at each point in time based on all the available observations in the entire time series. In short, this algorithm calculates the different probabilities of the system state and then superimposes them on the different measurements taking into account their added noise component.

2.3.2 Machine Learning in Time Series

Forecasting future values in time series can be reduced to a supervised learning problem by creating features. Those values used as inputs to the model can be either the past values of the time series or other external explanatory variables. Time series can be used in both regression and classification problems, depending on the objectives of the task. The output of the model varies depending on the type of problem being addressed: in the case of a regression problem, the model's output is the estimation of future values of the time series, and in the case of a classification problem, the output can be the class to which the time series belongs or the next future class in a time series.

Both strategies can be applied to time series data as long as an appropriate cross-validation (CV) process is considered, the extraction of the features mentioned above is done in a successful way and cyclical transformations are taken into account when necessary. Also important is the strategy to avoid error propagation and to ensure a realistic deployment. All these considerations are explained in detail below.

Regression

ML regression is the subset of ML algorithms that are used to model the relationship between variables in order to make predictions or forecast numerical values. Thus, those algorithms can be used more concretely in time series if appropriate decisions are made in the procedure. In addition to the patterns in the variable of interest, it may have relationships with other features that are not identified with classical time series regression models. Continuing with the previous notation, Y_t is the value of the time series at time t , and the n features used as inputs in the model at time t are denoted by X_t^1, \dots, X_t^n .

The most common ML-based regression algorithms are the following:

- Linear Regression (LR) models the linear relationship between a dependent variable (output or target) and one or more independent variables (inputs or features) as shown Equation 2.9 (40). LR finds the best-fit line that minimises the error between the predicted (\hat{Y}_t) and actual values (Y_t).

$$Y_t = \beta_0 + \beta_1 X_t^1 + \dots + \beta_n X_t^n \quad (2.9)$$

LR makes certain assumptions, such as linearity, independence of errors, homoscedasticity (constant variance of errors), and normality of errors. In this sense, the violation of these assumptions may affect the accuracy and reliability of the LR model.

Another characteristic of LR models is that these models are highly interpretable, in which the coefficients $\beta_0, \beta_1, \dots, \beta_n$ represent the direction and magnitude of the relationship between the features and the target variable. However, it is important to scale the features before applying LR to prevent the features with larger values from dominating the regression equation.

- The SVM algorithms are particularly well-suited for tasks where the data is not linearly separable and requires non-linear decision boundaries or hyperplanes (41). When this technique is used for regression tasks where the output is numerical it is often referred to as Support Vector Regression (SVR).

SVR finds the hyperplane that best fits the data with a certain tolerance while minimising the error between the predicted values and the actual values of the target variable. Furthermore, it incorporates a regularisation term C to control the complexity of the model and prevent overfitting. A smaller value of C results in a wider tolerance band, allowing for more errors, while a larger value of C results in a narrower tolerance band, allowing for fewer errors. It is also possible to handle non-linear data using a technique called the "kernel trick". A kernel is a mathematical function that maps the data points into a higher-dimensional space where non-linear patterns can be captured in the data. SVRs are generally efficient for small to medium-sized data sets. The training time can increase significantly with large data sets and the computational complexity can be further increased with the use of kernel functions. In Figure 2.10, a classic two-dimensional example can be seen illustrating the idea behind the kernel trick. In this case, the tolerance bands with an ξ width only exclude one of the training set values.

The interpretability of SVR models can be limited compared to LR, as the decision boundary is often non-linear and can be complex in higher-dimensional feature spaces. However, feature importance measures and model visualization techniques can still provide insights into the importance of different features in the model.

- The KNN is a type of instance-based or lazy learning algorithm that makes predictions based on the values of the k most

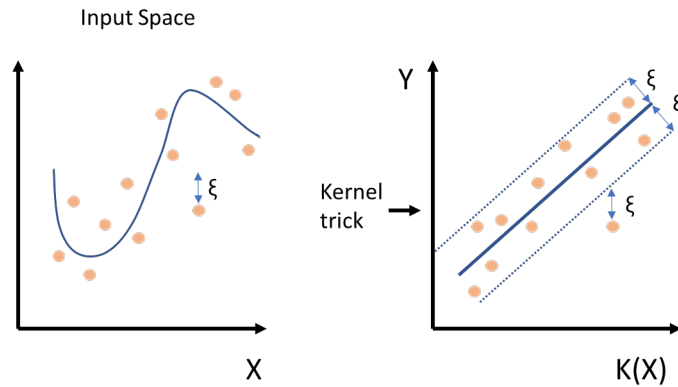


Fig. 2.10.: Two-dimensional example showcasing the effect of the kernel trick in the input space of the SVM model.

similar records in the training data. No explicit model or parameters are learned during the training phase. When making predictions for new, unseen data points, KNN regression finds the k -nearest neighbours to the query data point based on a distance metric in the feature space. Once the k closest observations are identified, KNN regression takes a combination of the output values of these neighbours as the predicted value for the new data point (42).

The main disadvantage of KNN regression is that it can be computationally expensive, especially when dealing with large data sets, as it requires calculating distances between the query data point and all training samples. Nevertheless, KNN is a widely used algorithm in regression, especially when the data has localised patterns or when interpretability is important.

- Decision Tree (DT) regressor is a type of regression algorithm that uses tree structures to model the relationship between variables (43). It is a non-parametric, supervised learning algorithm that can model both linear and non-linear relationships in the data. DT starts with a root node, which represents the

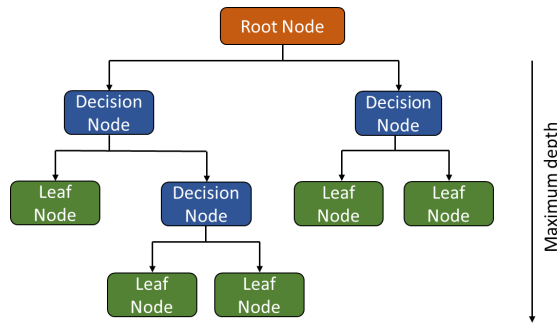


Fig. 2.11.: Graphic representation of the operation of a DT with a maximum depth of 3.

entire dataset and then it splits the data into branches based on the values of a chosen feature and a chosen splitting criterion such as gini impurity. The root node then branches out into decision nodes, each representing a possible outcome of the chosen attribute. This process, represented in Figure 2.11, is repeated until a stopping criterion is met and a leaf node is reached. The predicted value for a new data point is found by navigating down the tree from the root node to a leaf node and it is the weighted average of the training samples in the corresponding leaf node. The maximum depth of a DT refers to the longest path from the root node to the leaf node. It represents the number of decision nodes and attribute evaluations required to reach a prediction. Controlling the maximum depth of a tree can help avoid overfitting, where the model becomes too specific to the training data and performs poorly on new data.

DTs can handle missing values, which makes them robust in that sense and they are highly interpretable, as the decisions made at each splitting node can be easily visualised and understood. However, DTs are prone to overfitting if the correct hyperparameters are not decided upon.

- The Random Forest (RF) regressor is an ensemble learning algorithm that combines multiple DT to improve prediction accuracy and robustness (44).

RFs are less prone to overfitting compared to single DTs because the averaging of predictions from multiple trees helps to reduce noise and variance in the predictions. However, overfitting can still occur if the trees in the forest are too deep or too numerous. Proper tuning of hyperparameters, such as the maximum depth of trees, the number of trees in the forest, and minimum samples in leaf nodes, can help mitigate overfitting. RF may not always be as interpretable as a single DT but it can provide estimates of feature importance based on the decrease in impurity, which can help identify the most important features for making accurate predictions.

- Gradient Boosting Machines (GBM) are an ensemble learning technique that combines weak learners (e.g., DT) to create a strong learner. Gradient Boosting Regression iteratively adds DTs to the model, with each tree correcting the errors of the previous ones, and results in an ensemble model with improved prediction accuracy.

GBM use a technique called gradient descent to update the model parameters in each iteration. It calculates the gradient of the loss function with respect to the predicted values, and then the model parameters are updated in the direction of the negative gradient to minimise the loss (45).

This complex algorithm has several hyperparameters that need to be tuned to optimise the model's performance and also provides options for regularisation techniques to prevent overfitting. As with the RF model, GBM is typically not as interpretable as linear regression models, as they are an ensemble of multiple weak learners. To understand how the model is

built, the feature importance can be calculated based on the contribution of each feature in the decision trees.

Extreme Gradient Boosting (XGBoost) is an optimized and efficient implementation of GBM that incorporates several enhancements and additional features. It introduces regularization to avoid overfitting, uses tree pruning techniques to improve generalization, handles missing values in the data, supports parallelism in processing, and provides a measure of feature importance. These improvements make XGBoost faster and more accurate compared to traditional implementations of Gradient Boosting Machines.

The algorithm choice depends on the data characteristics, problem requirements, and desired performance metrics. It is important to evaluate and compare different algorithms to select the one that best suits the specific problem and data at hand and to properly validate and fine-tune the chosen model for optimal performance.

In data with identifiable cyclical behaviour where the input-output relation cannot be detected with classical ML techniques, an efficient alternative is the KNPTS algorithm. This method proposes the use of moving windows to find similar patterns in the historical time series. KNPTS, therefore, bases future estimates on the similarity to past events.

The KNPTS method can be considered as a variant of the KNN where the nearest neighbours are defined as the most similar subsets of data in the time series. Given the window size $W \in \mathbb{N}$, the last values of the output variable (Y_{T-W}, \dots, Y_T) are used as a reference pattern in the training process. This subset of data is compared with all the other subsets of length W in the series, that is, $Y^j = (Y_j, \dots, Y_{j+W})$, for all $j = 1, \dots, T - W - H$. Similarity can be measured with different strategies such as Euclidean distance or DTW. Therefore,

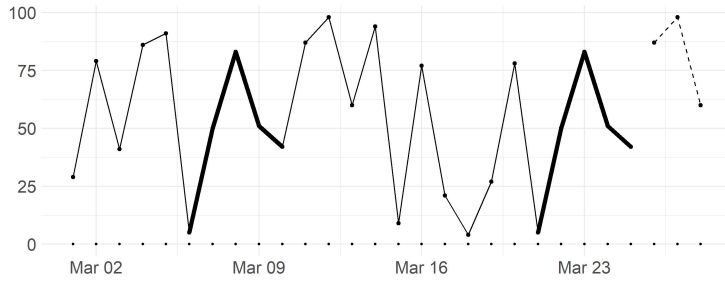


Fig. 2.12.: Nearest-neighbour three-step-ahead forecasts. A window of length five is selected and one neighbour is used to estimate the next three values.

the K nearest neighbour patterns are the K subsets with the lowest distance value (46).

An example of KNPTS for multiple-step forecasting is shown in Figure 2.12. It represents a simple case with $W = 5$, that is, 5 previous values are used as reference patterns to find the most similar data subset in the time series. In this case, the number of neighbours is fixed as $K = 1$, so the most similar pattern is used to forecast three future values of the time series, so $H = 3$.

The estimation of future values Y_{T+1}, \dots, Y_{T+H} is calculated by the weighted average of the next values of each nearest neighbour $\lambda = 1, \dots, K$, as shown in Equation 2.10 where ω_λ is the weight of the h -th point following the λ -th neighbour $Y_{\lambda+W+h}^\lambda$.

$$Y_{T+h} = \frac{\sum_{\lambda=1}^K \omega_\lambda Y_{\lambda+W+h}^\lambda}{\sum_{\lambda=1}^K \omega_\lambda} \quad (2.10)$$

This argument is applied to all $h = 1, \dots, H$ horizon values to be forecast. The rationale behind this method is that, in two different periods of time with similar values, the following value should be similar as well.

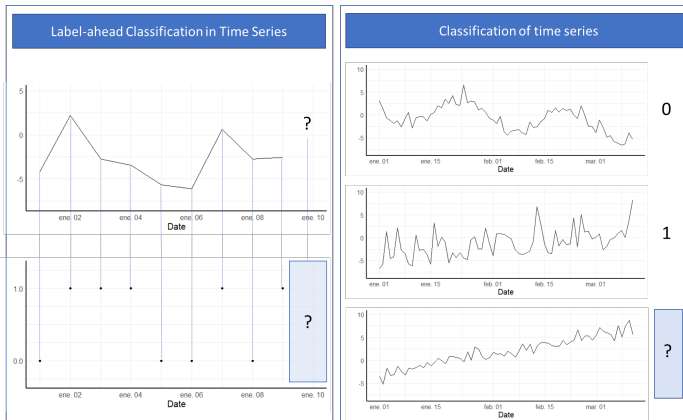


Fig. 2.13.: Two different approaches of time series classification problems.

The KNPTS method is implemented in an R package that is available for installation from Github ³ and it is explained in more detail in Appendix B.

Classification

Time series classification refers to the task of assigning labels or categories to time series data based on their patterns or features. Time series classification problems can be divided into two main types, which are summarised in Figure 2.13: label-ahead classification and classification of the entire time series. The picture on the left shows the problem of forecasting the label of the unknown future value of the time series based on the labels of previous values. The example includes a univariate time series that takes two possible classes, 0 or 1. Therefore, it is a binary classification of the next value in the time series. In addition to using information from the previous labels, it is possible to add information as inputs from other types of features. In general, ML models used in regression tasks are similarly applicable in this type of classification. However, in classification tasks, the output is not a continuous value but rather a categorical value that can take two or more classes.

³<https://github.com/MeritxellGomez/knpts-R-package>

The problems associated with labelling entire time series using information from other previously labelled series are illustrated in the right of Figure 2.13. To address this task, two approaches are proposed, which are selected based on the problem objectives and the available data characteristics. On the one hand, the characterisation of the entire time series by extracting a finite number of features to which a class is assigned. On the other hand, a point-by-point classification of all observations in the time series followed by a global assessment of the labelling based on the obtained classifications (47). The selection of either approach is made according to problem requirements.

The first approach involves extracting a fixed number of features from each time series, which are then used as inputs to the classification model. This reduces each time series to a feature vector and the time series data set to a multivariate data set that represents its behaviour, with as many observations as time series available for training. Therefore, feature extraction and selection are critical in this process, as the likelihood of capturing the unique behaviour that characterises the series decreases for long-time series or those with a high acquisition frequency. This approach is applicable to both univariate and multivariate time series, with the only difference being the number of features created.

Alternatively, in the second approach, each observation in the time series can be individually classified by using the appropriate features as inputs for the classification model. The resulting classification is then used to assign labels to each observation in the time series. In order to obtain a final label for the entire time series, a criterion must be established. One possibility is to use the mode of the obtained classes, i.e., the most frequent class. Another option is to use a propagation algorithm based on the Kalman filter operation, which assumes an equal probability of the time series belonging to each class. For instance, for binary classification problems, the initial assumption is

that the time series has a 50/50 probability of belonging to either class 0 or class 1. After obtaining the labels for each observation in the time series using an ML model that provides the probability of belonging to each class, the resulting vector of observations is used to iteratively update the probability of the time series being labelled as class 0 or class 1. The time series is run through this iterative process, and it is assumed that after convergence, the class with the dominant probability is assigned as the label for the entire time series.

Both classification problems with time series data are very common in industrial environments. To give an example, they are the basis of many machine diagnosis problems. In this context, time series are typically signals collected during an industrial process, and it is often necessary to label the processes according to their quality in order to determine the status of the machine. For instance, the goal may be to identify in real-time when the process is going to exhibit anomalous behaviour, while in the other case, the goal may be to classify completed experiments according to a quality indicator that can take on different values. Furthermore, time series classification has applications in numerous fields, including finance, healthcare, and speech recognition. For instance, in healthcare, time series classification can be used to predict the onset of a disease or identify patients at risk of adverse events.

An imbalance of the available data is a commonly encountered issue when applying data-driven techniques for time series classification. Data imbalance refers to a situation in which the number of instances in different classes of a data set is not evenly distributed. This means that some classes have significantly fewer instances than others, making it difficult for ML models to accurately predict the underrepresented classes. There are several techniques that can be used to deal with data imbalance depending on the specific characteristics of the data set and the problem at hand. The most commonly used approach is re-sampling although one can also choose to modify the

cost matrix of a classification algorithm to give higher costs to misclassifications of the minority class or training a classifier to identify instances of the minority class as anomalies or outliers in the data.

Re-sampling involves either oversampling the minority class or undersampling the majority class to balance the data set. Oversampling techniques include randomly replicating instances of the minority class or generating synthetic examples using algorithms such as Synthetic Minority Over-sampling Technique (SMOTE) (48). Undersampling techniques include randomly removing instances from the majority class or selecting instances using a variety of methods such as Tomek Links (49) or Edited Nearest Neighbours (50). Another approach to data balancing is the combination of undersampling and oversampling techniques. For example, the combination of SMOTE and Tomek links is known as SMOTE-Tomek or SMOTE-TL (51).

In both regression and classification problems, the reliability of ML models can be increased by the correct treatment of some of the tasks performed during fitting. On the one hand, the choice of model parameters is carried out by ensuring their generality and preventing overfitting through appropriate cross-validation techniques. On the other hand, those algorithms are trained using the information provided by the correct features. In addition, the forecasting strategy is key when evaluating the effectiveness of the methods, as the performance characteristics are considered in order to avoid unrealistic error accumulations.

2.3.3 Time Series Cross-Validation

Regardless of the data-driven model and whether the objective is regression or classification, it is advisable to reserve a certain amount of data for testing and to use a higher percentage of the data for the model fitting phase, known as training. In the training phase, the

aim is to find an expression whose combination of parameters fits the data as accurately as possible without losing the generality to fit future data as well. For that purpose, cross-validation techniques which have some particularities in the case of time series data are used.

Cross-validation is a technique used to train and evaluate a data-driven model by dividing the data set into multiple subsets (folds) and training the model on one subset while testing it on the remaining subsets (52). This technique is used to help ensure that the model is able to generalise well to unseen data and to identify any issues with the model, such as overfitting. By evaluating the model on different subsets of the data, the accuracy is more representative of the whole data set. This approach is useful to identify model issues and improve the model's performance.

There are several different types of cross-validation techniques, including k -fold cross-validation, where the data set is divided into k subsets and the model is trained and tested k times, each time using a different subset as the test set. This procedure is done to prevent overfitting and evaluate the model performance in a more robust way than simple train-test partition (53). An example using $k = 5$ is shown in Figure 2.14. A combination of the parameters obtained in each of the five iterations is used to fit the final model and validate it on the test set. Another popular method is the leave-one-out cross-validation method. The procedure is equivalent to the k -fold cross-validation technique but uses the same number of folds as values in the training set. Thus, each fold has a single value to test, which is predicted using the information contained in the remainder.

These methods cannot be used directly in time series. Instead, time series cross-validation involves splitting the data set into multiple folds or subsets considering that each fold represents a different time segment. Therefore, in order to be more representative of reality,

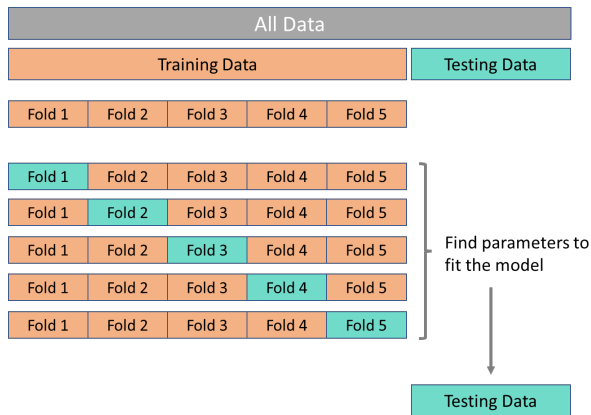


Fig. 2.14.: 5-fold cross-validation.

the fold chosen for validation in each iteration cannot contain past data. There are two ways to approach this validation technique: using either sliding windows or expanding windows, as shown in Figure 2.15 (54). On the left, a certain duration is chosen which fixes the size of the training data window for each iteration. Validation is performed at each step with subsequent data, also with a fixed duration that is maintained at each iteration. It is common to keep the proportion of the train-test partition done in the original time series. The picture on the right shows cross-validation in time series using an expanding window. In this case, the start of the training sets in each iteration is fixed at the start of the original time series. The size of the training window expands, adding future data to the training set and the proportion of train and set data grows in each iteration. In both cases, the size of the windows can be modified to avoid overlaps between data from different iterations. An interesting option is to force the training sets in each iteration to contain exactly the training and test data from the previous iteration. This gives rise to overlapping and non-overlapping versions of each of the methods explained.

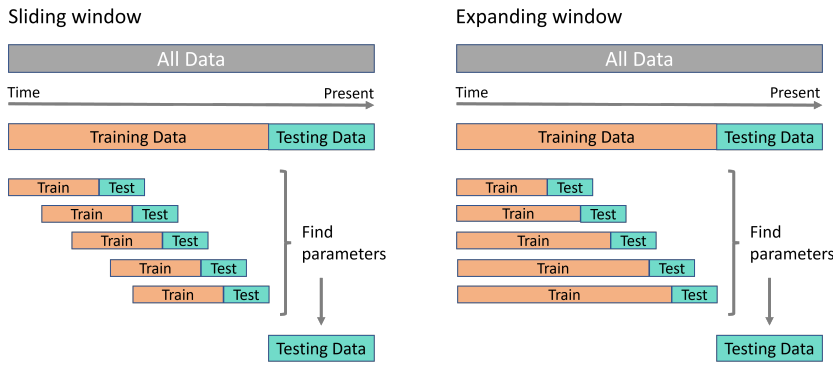


Fig. 2.15.: Time series cross-validation using sliding windows and expanding windows.

2.3.4 Time Series Feature Extraction

For the use of both supervised and unsupervised learning models, explanatory variables are necessary for the estimation of the variable to be predicted or to find relationships in data. These are known in ML as features and there are multiple techniques for extracting and selecting the most appropriate ones according to the problem to be tackled.

Specifically, time series feature extraction is the process of identifying and creating relevant features from time series data that can be used as input to a ML model (55). Some specific methods of feature extraction in time series include features based on signal data, date- and time-related features, lag features or topological features. The choice of a time series feature extraction method depends on the type of data and the problem to be solved. This is a crucial step in time series analysis and should be performed to ensure that the most relevant features are selected.

Features based on signal data

These features are calculated from the raw time series data and include statistical features, time and frequency domain features,

correlation features and wavelet-based features. Some of the most used features extracted from time series are listed below.

- Basic statistics such as the *mean* value of the time series, *minimum* and *maximum*, that is the lower and the higher values; or *standard deviation* to measure the variability,
- *Skewness*, a measure of the asymmetry of the data distribution or *kurtosis*, a measure of the "peakedness" of the data distribution (56).
- *Entropy*, a measure of the complexity of the time series based on its regularity (57).
- Correlation values as *autocorrelation*, a measure of the correlation between the time series and a lagged version of itself.
- Features from some transformations such as the *coefficients of the Fourier series* representation of the time series or the *coefficients of the wavelet decomposition* of the time series (58).

Date- and Time-related features

The values extracted from the timestamp and the calendar such as the day of the week, the hour of the day, the season or the public holidays, among others; can be used as inputs of ML models.

Some of those features extracted have a cyclical behaviour and therefore, have to be transformed in an appropriate way in order to be used in some of the ML algorithms that calculate distances between observations. This process is executed to prevent the identification of non-existent patterns due to the fact that values that are actually close to each other can be considered distant and vice versa. The most common transformation v consists of using the sine and cosine functions together with the period P of the data⁴. Once the periodicity is found, Equation 2.11 is applied to each value Y_t , $t = 1, \dots, T$.

⁴<https://www.avanwyk.com/encoding-cyclical-features-for-deep-learning/>

Tab. 2.2.: Values of the *hour* variable to be used as input to the KNN algorithm.

t	hour	Y	Distance
1	1	$Y(1)$	0
2	2	$Y(2)$	1
3	3	$Y(3)$	2
...
22	22	$Y(22)$	21
23	23	$Y(23)$	22
24	24	$Y(24)$	23
25	1	$\hat{Y}(25)$	

$$v : [0, P] \longrightarrow [-1, 1]$$

$$Y_t \rightarrow v(Y_t) = \left(\sin \left(\frac{2\pi Y_t}{P} \right), \cos \left(\frac{2\pi Y_t}{P} \right) \right) \quad (2.11)$$

Two variables are obtained from each of the transformations made (one from the sine and the other from the cosine). The distance between consecutive bivariate data obtained after this transformation is constant.

This procedure is illustrated with a simple example using the KNN method and the cyclical input variable *hour*. There are only 24 hourly values available for a single day and the periodicity of the *hour* is $P = 24$. The output variable is labelled $Y(t)$, and then the values from $t = 1$ to $t = 24$ are used to estimate the value of $Y(25)$. The number of neighbours is fixed as $k = 3$. The variable *hour* takes value $H = 1$ when $t = 25$, so its three nearest neighbours are the values $hour = 1$, $hour = 2$ and $hour = 3$; as can be seen in Table 2.2.

Tab. 2.3.: Values of the sine and the cosine of the *hour* variable to be used as inputs to the KNN algorithm.

t	<i>sinhour</i>	<i>coshour</i>	Y	Distance
1	0.2588	0.9659	Y(1)	0
2	0.5	0.866	Y(2)	0.2611
3	0.7071	0.7071	Y(3)	0.5176
...
22	-0.5	0.866	Y(22)	0.7654
23	-0.2588	0.9659	Y(23)	0.5176
24	0	1	Y(24)	0.2611
25	0.2588	0.9659	$\hat{Y}(25)$	

The combination of $Y(1)$, $Y(2)$ and $Y(3)$ is used to estimate $Y(25)$ using the average depicted in Equation 2.12.

$$\hat{Y}(25) = \frac{Y(1) + Y(2) + Y(3)}{3} \quad (2.12)$$

However, a correct understanding of the time variable would take as a nearby value the midnight of the previous day, so that the cyclical behaviour is not represented in this approach. Therefore, the *hour* variable is transformed as in Equation 2.11 and the data in Table 4 are obtained. The right column shows the Euclidean distance between the observation at $t = 25$ and each of the corresponding data from $t = 1$ to $t = 24$. The minimum values correspond to $t = 1$, $t = 2$ and $t = 24$.

Therefore, after trigonometric transformation, the values chosen to estimate the value of $Y(35)$ are $Y(1)$, $Y(2)$ and $Y(24)$, as it is shown in Equation 2.13.

$$\hat{Y}(25) = \frac{Y(1) + Y(2) + Y(24)}{3} \quad (2.13)$$

Lag features

A lag is the time interval between the current time period and the

past time period used to make the prediction. For example, if the value of a variable from two periods ago is used to predict its value in the current period, a lag of 2 is introduced as input in the model. The most common notations, in this case, are the use of $Y(t - d)$ or Y_{t-d} for the lag of d periods in time series $Y(t)$ or Y_t . Lags are important in time series forecasting because help to capture trends and seasonality in the data in a similar way as ARIMA models do. These features are calculated by shifting the time series data by a certain number of time steps and including the lags of the output variable and those of the input variables.

There are different ways of introducing lags into time series forecasting models. The first has been discussed above and involves fitting models of the ARIMA family. Another way is the manual creation of the lags of the available variables to be used as inputs in classical ML algorithms. Finally, the extension of the use of this type of feature for training ML and Deep Learning (DL) models has given rise to algorithms such as Recurrent Neural Networks (RNN) or, particularly, Long Short-Term Memory (LSTM), that use the past values of the time series to estimate future values (59).

In time series forecasting using lags, error propagation can occur when errors made in predicting past values of a variable are carried forward and affect the accuracy of future predictions. This effect can be magnified as the number of lags used in the model increases. As mentioned above for ARIMA models, one way to mitigate this phenomenon is the periodic execution of the forecasting model.

Topological features

The topological features of a time series are mathematical structures that can be used to describe the shape, structure, and complexity of the data over time. These features are used in the field of TDA to extract information about the underlying geometric and topological properties of the data (60).

Some of the topological features that can be extracted from a time series include persistence diagrams (61), which captures the evolution of topological features, such as connected components, loops, or voids, as time progresses; Betti numbers (62), which counts the number of connected components, loops, voids, and higher-dimensional structures in a time series; or mapper graphs (63; 64) which captures the topological relationships between different regions or clusters in data, providing insights into the global structure.

The extraction of persistence diagrams from time series and the creation of persistence images from them are explained in detail in the following lines. A persistence diagram is a graphical representation of the evolution of topological features over time. It provides a way to visualise the lifespan of topological features, such as connected components or holes, as they appear and disappear in data. To construct a persistence diagram, TDA algorithms first transform data into a simplicial complex, a mathematical structure that encodes the topological features of data. This simplicial complex is then analysed using persistent homology, a mathematical tool that identifies the connected components, holes, and higher-dimensional voids that persist across different scales in the data.

The output of the persistent homology algorithm is a set of points in a two-dimensional plane, where each point corresponds to a topological feature and its position reflects its birth and death times. Specifically, each point has two coordinates: the birth time, which represents the scale at which the feature first appears in data, and the death time, which represents the scale at which the feature disappears or merges with another feature.

The persistence diagram can be visualised as a scatter plot, where each point represents a topological feature and is coloured by its dimensionality.

First, let $\{Y_t, t = 1, \dots, T\}$ a univariate time series (Figure 2.16a), there are different topological approaches to obtain a point cloud (65). The number of points P in the point cloud varies according to the method determined.

Lover-star filtration or sublevel set filtration is a stable way of describing local minimums and maximums in a time series (66). Therefore, P min-max pairs $(b^1, d^1), \dots, (b^P, d^P)$ have been constructed where b refers to the minimum or birth and d refers to the maximum or death. This set of points represented in the plane defines the persistence diagram (Figure 2.16b). As d values are always greater than or equal to their b pairs, the points $(b^1, d^1), \dots, (b^P, d^P)$ are grouped in the upper part of the diagonal $b = d$ and they can be represented in the life diagram $(b^1, d^1 - b^1), \dots, (b^P, \dots, d^P - b^P)$ (Figure 2.16c). Distances are possible between those clouds of points but applying ML methods is still challenging. An alternative way is to associate to each point from the lifetime diagram a bivariate normal distribution, weighted and then, integrated into different patches on a square domain covering the support of the regularised lifetime diagram, leading to the so-called persistence images (Figure 2.16d) (67).

2.3.5 Forecasting Strategies

The strategies used for time series forecasting can be categorised according to various criteria such as the size of the prediction horizon or the number of estimated values provided in each iteration.

According to the forecast horizon, strategies can be classified as short-term or long-term. Short-term forecasting, also known as near-term or immediate forecasting, typically involves predicting the values of a time series variable in the immediate future, usually within a few days, weeks, or months. Short-term forecasting is useful for operational decision-making, such as inventory management,

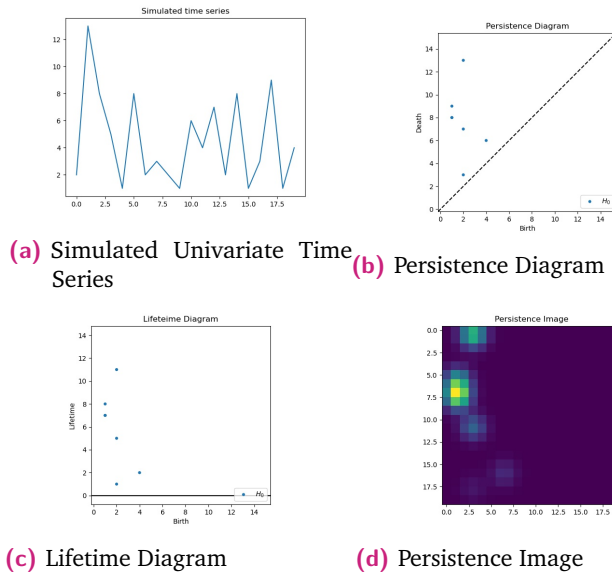


Fig. 2.16.: Process of creating persistence images of a univariate time series using TDA

demand forecasting, and resource allocation. It uses simpler models that are updated more frequently. Long-term forecasting, on the other hand, involves predicting the values of a time series variable in the distant future, usually beyond a year or more. Long-term forecasting is useful for strategic decision-making, such as business planning, investment analysis, and policy formulation. It may require different approaches, such as econometric models, simulation models, and ML techniques that are specifically designed for handling long-term trends and patterns.

Another alternative classification is based on the number of steps ahead being predicted in each model execution: one-step ahead forecasting and multi-step ahead forecasting. A general problem is assumed in which (F^1, \dots, F^n) are used as the vector of extracted features in addition to the previous M values of the time series Y_{T-M}, \dots, Y_T for the prediction of future values Y_{T+1}, \dots, Y_{T+H} . Using

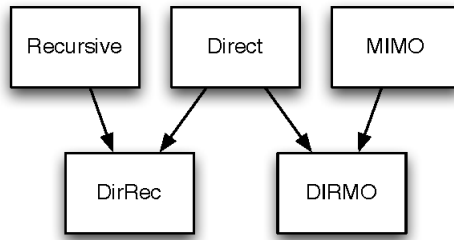


Fig. 2.17.: The five different forecasting strategies and their relationships.

this notation, F_t^i is the value of feature i at time t , where $i = 1, \dots, n$ and $t = 1, \dots, T$.

One-step ahead forecasting, also known as single-step forecasting, involves predicting the value of a time series variable typically the next time period in the sequence. Then, $H = 1$ and $\{F_{T+1}^1, \dots, F_{T+1}^n, Y_{T-M}, \dots, Y_T\}$ are the values used to estimate the value of Y_{T+1} . One-step-ahead forecasting is often used when the goal is to make immediate short-term predictions or when the underlying patterns in the time series change frequently, making longer-term predictions less reliable.

Multi-step ahead forecasting is the approach that involves predicting the values of a time series variable multiple steps ahead; that is, beyond the next time period. Therefore, it consists of estimating the next H values $\{Y_{T+1}, \dots, Y_{T+H}\}$ using the previous values $\{Y_{T-M}, \dots, Y_T\}$ and the features $\{F_{T+H}^1, \dots, F_{T+H}^n\}$. For example, if the time series is measured in months, multi-step ahead forecasting may involve predicting the values for the next 3, 6, or 12 months. Multi-step ahead forecasting is used when the goal is to make medium- to long-term predictions for planning and strategic decision-making, or when the underlying patterns in the time series change less frequently and longer-term predictions are more reliable. In order to predict more than one value $H > 1$ in time series using ML algorithms, five strategies can be chosen (68).

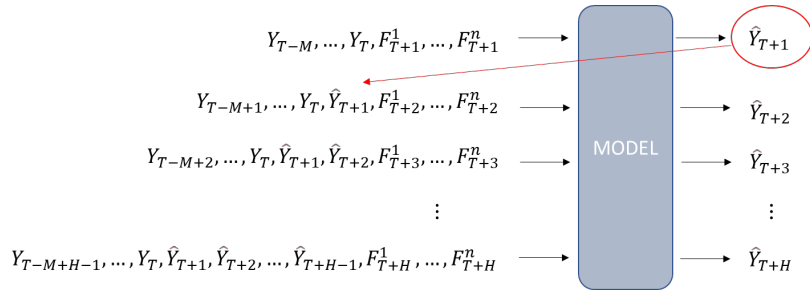


Fig. 2.18.: Outline of the *Recursive* strategy for forecasting the next H values of a time series using M lags and n features.

- *Recursive Strategy.* The *Recursive* strategy is also called *Iterative Strategy*. This strategy trains a single model to forecast only one future value. Then, it is recursively used to forecast all the H future values as can be seen in Figure 2.18. The first value Y_{T+1} is estimated by applying the model to historical data $\{Y_{T-M}, \dots, Y_T\}$. Then, this just obtained value is used as a part of the input variables for forecasting the second value. This process is repeated recursively to estimate the H future values. An important consideration of the *Recursive* approach is the use of estimated values as inputs in the following steps, which causes the model to be prone to the accumulation of intermediate errors. Depending on the noise present in the time series and the forecasting horizon, the *Recursive* strategy may suffer from low performance in multi-step ahead forecasting tasks. Indeed, this is especially true if the forecasting horizon H exceeds the number of past values used to train, as at some point all the inputs are forecasted values instead of actual observations.
- *Direct Strategy.* This strategy is also called *Independent Strategy* because each future value is estimated independently from others as it is shown in Figure 2.19. In this strategy, H models are trained independently, one for each point to be predicted, using as inputs the values of $\{Y_{T-M}, \dots, Y_T, F_{T+1}^1, \dots, F_{T+1}^n\}$. Then,

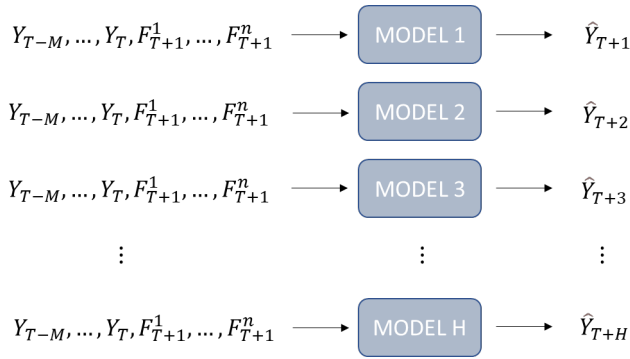


Fig. 2.19.: Outline of the *Direct* strategy for forecasting the next H values of a time series using M lags and n features.

model 1 is trained to predict the value \hat{Y}_{T+1} , model 2 is trained to predict the value \hat{Y}_{T+2} , and so on, until training model H to predict the value of \hat{Y}_{T+H} .

In that case, models are immune to error accumulation because they do not use prior estimates as inputs. However, training H independent models can be a computationally more costly task, in addition to not considering complex dependencies between variables. A multi-step ahead forecast is done by concatenating the H predictions obtained from each model.

- *DirRec Strategy*. This strategy, as its name suggests, combines the *Direct* and *Recursive* strategies. It learns H different models for each value to be estimated and, at each step, the just forecast value is added to the input set of the previous step.
- The *Multi-Input Multi-Output (MIMO)* executes one prediction after learning a single multiple-output model. This strategy is generally used for the prediction of several variables of interest at the same time. However, in time series, the target vector can be considered of length equal to the number of points to be predicted H . The feature vector used as input contains the M previous values and values from n features that can

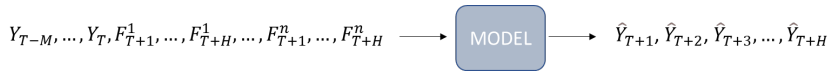


Fig. 2.20.: Outline of the *MIMO* strategy for forecasting the next H values of a time series using M lags and n features.

be extracted from H different timestamps. Then, forecasts are returned in one execution by a multiple-output model, as shown in Figure 2.20.

- The *DIRMO Strategy* is a combination of the *Direct* and *MIMO* strategies. It forecasts the H future values in $b = \frac{H}{s}$ blocks, each with an output of size s . Data from each block is forecasted using a *MIMO* approach. Then, $b = \frac{H}{s}$ models are trained in the *DIRMO* strategy. When $s = 1$, the number of models to be trained is equal to H , corresponding to the *Direct* strategy. On the other hand, when $s = H$, there is only one model to be trained and this approach corresponds to the *MIMO* strategy. Calibrating the parameter s enhances the flexibility of the *MIMO* strategy by adjusting the dimensionality of the outputs, striking a beneficial balance between maintaining a higher level of stochastic dependency among future values and increasing the predictor's versatility.

Although generating forecasts for multiple steps ahead is a crucial problem in modelling, it has received limited investigation (69). The selection of an inappropriate strategy can result in the accumulation of errors arising from the use of lagged variables. Moreover, the model's performance may appear unrealistic if it is evaluated in an environment that differs from that of its subsequent deployment. This issue is particularly significant for long-term forecasting, where the choice of strategy can considerably bias the error calculation. Therefore, special attention is given to this problem.

2.4 Error Evaluation

A key issue in the development of AI or ML-based tools to solve industrial problems is the validation of the proposed solution. This process consists of deciding whether a model is good enough and can be considered optimal in the developed scenario. The wide range of methods proposed in the literature shows that this is an open question of great current interest and there is no standard for quantifying model validity (70).

The procedure for deciding the best prediction model (regression or classification) is called evaluation, i.e. the quantification of the accuracy of the chosen method. There are different techniques to carry out this task, some of them are mentioned below. However, this chapter is not a review of existing methods to evaluate ML models in the literature, but rather a reflection on the impact that the choice of one method over another can have on deployment and exploitation.

Evaluating a model typically consists of comparing the values estimated by the model against the actual values collected. The concept of comparison in Mathematics is linked to metric functions. A metric is a distance function defined in a metric space that satisfies the following conditions.

1. $d(x, y) = 0 \Leftrightarrow x = y$
2. $d(x, y) = d(y, x)$
3. $d(x, z) \leq d(x, y) + d(y, z)$ (Triangle inequality)

To measure the similarity between two elements, different metrics or distance functions can be calculated. In the case of ML models, the vector of actual values and the vector of estimated values provided

by the model are compared. The closer the two vectors are, the more similar they are.

In time series, other similarity measures can be defined that do not necessarily satisfy the triangular inequality, i.e. they are not distance metrics (71). In what follows, the performance measure or the error measure is used to refer to evaluation measures of models in order to cover them all in a general way, both metric and non-metric.

The error measures used in regression problems are different from those proposed in classification problems. However, in both cases, it is important to choose well the measure to use, to know how to interpret it correctly and to act accordingly with the result obtained. There is a relevant risk of focusing on the numerical value of the accuracy obtained by an error measure rather than on the business result (72). The main idea is that one measure may give an objectively better numerical result than another but the prediction obtained by that model does not correctly satisfy the requirements of the problem and vice versa. This may be due to the context, the number of estimated values, the frequency of the time series data acquisition, or other reasons that are not reflected in a measure calculation.

2.4.1 Regression Performance

The output in regression problems is one or more numerical variables that take values in an interval and are therefore continuous in that interval. The comparison between prediction and reality is done using a function that takes as inputs both the actual and the estimated values and provides as output one or more values that quantify the difference between them.

Error measures for evaluating the regression model and validating the solution can be classified into two groups: static measures and elastic measures.

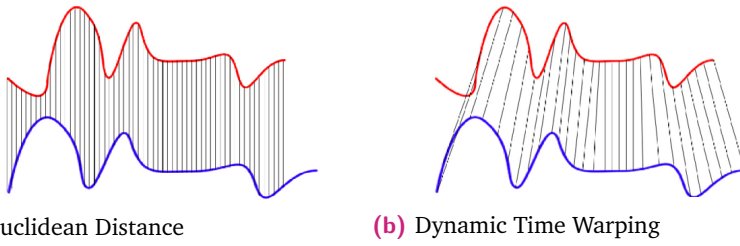


Fig. 2.21.: Comparison of two different ways of mapping two time series

Static measurements compare the vector of observations and the vector of predictions on a point-by-point basis. The most commonly used regression error measure is the Mean Squared Error (MSE), which measures the average of the squared differences between the predicted and actual values, shown in Equation 2.14 (73).

$$MSE(Y, \hat{Y}) = \frac{1}{H} \sum_{t=T+1}^{T+H} (Y_t - \hat{Y}_t)^2 \quad (2.14)$$

In the case of time series data, using static measures involves comparing actual and estimated values by aligning both vectors over time. The alignment of the values to be compared with a static measure based on the Euclidean distance is shown in Figure 2.21a.

Other regression metrics similarly based on the Euclidean distance between predicted and actual values are: the Mean Absolute Error (MAE), the Mean Absolute Percentage Error (MAPE) or the Root Mean Squared Error (RMSE). Static measures are widely used in industrial application areas (74).

$$MAE(Y, \hat{Y}) = \frac{1}{H} \sum_{t=T+1}^{T+H} |(Y_t - \hat{Y}_t)| \quad (2.15)$$

$$MAPE(Y, \hat{Y}) = \frac{100}{H} \sum_{t=T+1}^{T+H} \frac{(Y_t - \hat{Y}_t)}{Y_t} \quad (2.16)$$

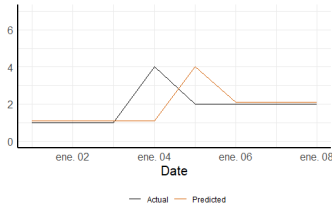
$$RMSE(Y, \hat{Y}) = \sqrt{\frac{1}{H} \sum_{t=T+1}^{T+H} (Y_t - \hat{Y}_t)^2} \quad (2.17)$$

On the other hand, elastic similarity measures for time series can be used to measure forecasting error in an ML model (75). These measures allow for many-to-one point comparison in time series. Therefore, the actual vector and estimated vector may be similar but have differences in length, scaling, and time shift (76). The most popular elastic similarity measure is the DTW, which measures the distance between two time series by finding the optimal alignment between them, allowing for warping in time and scaling, as can be seen in Figure 2.21b. DTW considers all possible alignments and chooses the one with the minimum distance.

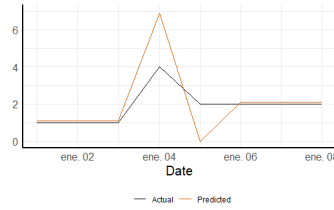
Another example of elastic measure is the EDR that measures the distance between two time series by allowing insertion, deletion, and substitution of points in both series. The aim is to quantify the number of these three operations necessary to obtain the real series from the estimate provided by the model. The Edit Distance (ED) was initially used to calculate the similarity between two strings. In order to define matching between numerical series, different adaptations are proposed (77). Given a positive threshold ϵ , the distance between two points is reduced to 0 or 1 using Equation 2.18. The lower the number of operations required, i.e. the lower the EDR value, the more similar the time series of actual and predicted values are.

$$d_\epsilon(\hat{y}_t, y_t) = \begin{cases} 0, & d(\hat{y}_t, y_t) \leq \epsilon \\ 1, & d(\hat{y}_t, y_t) > \epsilon \end{cases} \quad (2.18)$$

There are other time series similarities measures such as the Longest Common Subsequence (LCSS), the Shape-Based Distance (SBD) or



(a) Forecasting result of the time series using model 1 which obtains MSE = 1.56 and DTW = 1.3.



(b) Forecasting result of the time series using model 2 which obtains MSE = 1.56 and DTW = 10.9.

Fig. 2.22.: comparison of two time series forecasts with the same value of MSE but different DTW.

the Time Warp Edit Distance (TWED), a combination of the time warp and the edit distance measures.

Simple results simulated from two different forecasting models are shown in Figure 2.22 in order to illustrate the effect of measuring the accuracy of models with static or elastic approaches.

When using static metrics for the decision of the best prediction model, both cases would provide the same value for the MSE or any other metric based on the Euclidean distance measured point-by-point. It can be observed that both predictions provide an estimated series with different behaviours. In the case of model 1, there is a shift in the prediction compared to the real values of the time series. On the other hand, model 2 provides estimated values that are very different from reality, which does not adjust to the real series even correcting for the displacement. However, the value of the MSE does not report that difference, and it is necessary to employ other elastic measures such as DTW. This difference is reflected in the value of the DTW measure, which obtains a value of 1.3 in Model 1 and a value of 10.9 in Model 2. If this measure of time series similarity is used as a quantifier of the accuracy of the forecasting model, model 1 is chosen because it has an error approximately eight times smaller than model 2.

In this way, the elastic error measures do not penalise in the case that the forecast is shifted in time. This is particularly useful in real industry problems where the frequency of data acquisition is high and such cases are more likely to occur. The use of these measures for quantifying the accuracy of time series regression models is not widely available in the literature. However, in some problems where their use is being introduced, it has proved to have many advantages for the correct interpretation of the developments.

2.4.2 Classification Performance

A ML classification problem involves assigning a label to an input data set. That is, the output variable of the model is categorical and the way to measure the performance of the approximation is the success or failure of the class assignment.

Time series classification consists of labelling chronological data sets. The simplest problem is binary classification, when the output variable can only take two values, commonly the absence or presence of an event, represented by 0 or 1. Once the predictive model provides an estimated labelling of the validation observations, it is possible to construct the confusion matrix, which is a comparison of the predicted and actual values, as shown in Table 2.4. Each row of the matrix contains the information on the actual observations and each column represents the prediction made. The confusion matrix is used in industrial environments to represent the classification result and, from it, the performance indicators are extracted.

Tab. 2.4.: Confusion Matrix in a Binary Classification

		Predicted	
		0	1
Actual	0	True Negative (TN)	False Positive (FP)
	1	False Negative (FN)	True Positive (TP)

The rest of the measures to evaluate classification processes are explained (78) from the confusion matrices. To emphasise the importance of choosing the appropriate metrics for each problem, the best-known measures are defined and the results they provide in different cases are compared below.

- The *Accuracy* is a general measure of the correctness of the model in both classes and it is calculated by using Equation 2.19.

$$Accuracy = \frac{TN + TP}{TN + FP + FN + TP} \quad (2.19)$$

It can be extended to cases of multiclass classification by calculating the proportion of labels that have been assigned correctly.

- The *Precision* is the proportion of TP in the observations classified as positives and the *Recall* is the proportion of TP in the set of observations that should be classified as positives because they really are. Therefore, the expressions used to calculate them are Equation 2.20 and Equation 2.21, respectively.

$$Precision = \frac{TP}{TP + FP} \quad (2.20)$$

$$Recall = \frac{TP}{TP + FN} \quad (2.21)$$

- The *F1 score* is the harmonic mean of precision and recall. By the definition in Equation 2.22, the F1 score is high when both precision and recall are high and, analogously, the F1 score is low if both are low.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (2.22)$$

Tab. 2.5.: Two different binary classification results with the same accuracy.

		Predicted	
		0	1
Actual	0	100	0
	1	9	1

		Predicted	
		0	1
Actual	0	91	9
	1	0	10

Accuracy is a widely used metric to quickly and generally know the performance of the prediction. However, this measure assigns the same importance to each of the classes, which can cause confusing values of the metric when the data sets are unbalanced.

The example of confusion matrices in Table 2.5 shows two different binary classifications. In both cases, the accuracy value is 0.9182 as 101 out of 110 samples are correctly classified. However, it is clear that the classification of elements with label 1 is not equally correct in both cases. In the table on the left, only 1 of the 10 samples is correctly classified and in the table on the right, all of them are correctly classified. Therefore, an accuracy-based criterion could choose either of the two results as valid since it would evaluate that metric with the same value. In cases where the minority class is 1, it is advisable to compare the recall which in this case goes from 0.1 on the left to 1 on the right. A recall-based criterion chooses the prediction model whose results are summarised in the confusion matrix on the right in the Table.

Time series classification techniques have different applications in industrial environments that include fault detection in manufacturing processes or pattern recognition to monitor patients' health conditions, among others (79). The use of metrics other than accuracy is recommended in all classification problems, as in industry most of the data sets to be labelled are unbalanced due to the small amount of data from failed experiments in industrial processes.

Industrial Applications

3

” *With applications involving data, so often the data refuses to cooperate.*

— **Nan M. Laird**
(Mathematician and Statistician)

Nan’s scientific path was not clearly defined from the start. Due to her religious upbringing and the fact that at her school women were not encouraged to become prestigious scientists, she abandoned mathematics very early to study French. Later, she began to study computer science but statistics soon fascinated her due to a course where she learned that math formulas can be used to make mundane life decisions, like whether or not to take your umbrella to work.

After graduating in statistics, Nan worked on Kalman filtering for the Apollo Man to the Moon Program at MIT Labs. She joined the faculty of Harvard School of Public Health upon receiving her PhD, and remains there as a research professor, after her retirement in 2015. She has made fundamental contributions to statistical methods for longitudinal data analysis, missing data and meta-analysis and she has received many awards for them.

Although she was always interested in looking for a general framework, she admitted her interest in developing methods applicable to problems with specific conditions and concluded: "With applications involving data, so often the data refuses to cooperate. The answers you get can depend more on the characteristics of the data and not the statistical method" (80). Thus, in order to increase reliability in

AI-based systems for industrial process optimisation, decisions are made considering that the characteristics of the problem hold the keys to a correct solution proposal.

3.1 Introduction

The use of AI has been rapidly increasing across various fields over the last decades, transforming the way society lives and works. Indeed, the positive trend in the commitment to ML and DL techniques is reflected in the industry. Recently, research has been extended to new domains of Industry 4.0 such as Food Safety, Energy Efficiency, Zero Defect Manufacturing and Predictive Maintenance. The results obtained in those areas thanks to data-based solutions are very promising, especially those using ML techniques. However, despite being important advances for the scientific community, they continue to lack interest for industrial practitioners and their actual application in the industry remains a challenge, and indeed there are concerns about the reliability of AI solutions. This is mainly due to the lack of confidence of more classical industries towards innovative AI techniques and the limited quality that data-driven solutions are able to achieve in terms of accuracy, computational efficiency and development costs.

The most explored ML techniques in industrial problems are based on supervised learning (81) and are used to solve both prediction and classification problems. Some of the factors that contribute to the low reliability of this type of approach are low DQ; the lack of transparency and robustness of the models and their bias; and the difficulty in interpreting solutions and their accuracy.

The main problems in industry related to DQ are due to incompleteness in collection causing biases, noise and inconsistency. In addition,

the correct application of supervised learning methods requires labels in the data that are often invalid, due to both class mismatches and labelling failures caused by the systems in charge. Regarding labelling tasks, data imbalance is a classic challenge encountered in industrial environments, that occurs when the distribution of different classes or categories in the training data is not equal. This can result in biased or inaccurate predictions, as the model may be biased towards the majority class or under-represented classes may have insufficient data for learning.

On the other hand, correct extraction and detection of data characteristics are essential to provide the model with the necessary information without causing overfitting. That helps to maintain robustness in the deployment of the solution. Furthermore, the adaptation of the models to the time series data in both training and fitting is crucial to maintain accuracy in the validation and to achieve a less biased development. Finally, the transparency of the solutions is based on the interpretability and explainability of the developments made. Many AI models, such as DL models, are often considered "black boxes" as their decision-making processes may not be easily interpretable by humans. A lack of model interpretability and explainability can result in low trust and confidence in model forecasts, especially in industries where accountability is crucial. Improving the reliability of AI models in the industry requires addressing these challenges rigorously.

The most common challenge related to measuring errors in AI solutions is choosing the right evaluation metric. Different approaches may be suitable for different types of tasks or applications, and the choice of metrics can significantly impact the error measurements. Especially in the case of time series obtained from real industrial processes, it is important to consider alternatives to the classical techniques used to evaluate the performance of AI models. In this regard, it is important to consider the sample size and representa-

tiveness and the uncertain or ambiguous situations where the correct prediction or decision is not well-defined. That refers to the comparison of the results obtained from forecasting and classification errors against other methods that can be used as benchmarks. This allows quantifying the improvement of the metrics in proportion to other results and increases the reliability of the error measurements. In summary, it is important to carefully select appropriate evaluation metrics, address uncertainty and ambiguity, consider sample size and representativeness, and create benchmark data sets when possible.

Robust testing, validation, and verification of AI models, as well as ongoing monitoring and feedback loops, can help improve the reliability of AI solutions. There is also a concern about the ethics of AI solutions, especially those involving human-machine interaction where the focus is on operator safety. However, that part of AI reliability is beyond the scope of this research and is not discussed in depth. Therefore, addressing the low reliability of AI solutions requires efforts to improve DQ, reduce bias, increase transparency and enhance robustness.

The rest of the chapter is organised into four sections for the solution of four industrial cases in which supervised learning techniques have been applied to improve reliability in AI developments involving time series.

- In Section 3.2, a case of a search for rules relating data collected during the poultry production chain to meat quality is presented.
- In Section 3.3, a case of regression to forecast electrical consumption data is explained.
- In Section 3.4, a case of classification of porosity in an additive manufacturing process is developed

Tab. 3.1.: Classification of use cases in industrial applications based on data quality and number of samples.

	Small n° of samples	Large n° of samples
Low DQ	Poultry Chain Control Management	Energy Demand Forecasting
High DQ	Porosity Detection in Additive Manufacturing	Manufacturing Production Line Diagnosis

- In Section 3.5, a diagnosis problem in a manufacturing line that continuously tests fuses and suffers from several failures is presented.

Regarding the available data for each of these cases, they have been chosen to cover the four combinations between low and high-quality data and cases where there are sufficient repetitions or samples and cases where there are not. This classification is shown in Table 3.1.

In the modelling part, there is a regression problem related to forecasting electricity demand and three classification problems. The first one is the poultry chain case, which is a classification problem for designing a decision-making system. It employs a signal-based feature extraction strategy to model a tree that provides interpretability. In the case of porosity in additive manufacturing, two different classification strategies are compared to classify the dataset point by point using more complex algorithms. Lastly, the experiment diagnosis case involves classifying complete time series in the shortest possible time.

Finally, in the error measurement, the impact of elastic measures is compared in the regression case. For classification, there are three different cases. The first case involves a small number of repetitions in the poultry chain. The second case deals with sufficient samples with a controllable imbalance in healthy experiments. Lastly, there is an extreme case of imbalance in the majority class, which could have a devastating effect if not appropriately addressed.

In summary, the reason for choosing these cases is twofold: on the one hand, this selection brings variability as each one presents a different problem in the three areas to be improved and on the other hand, the cases belong to different fields of the current industry for which research is invested and innovative solutions are required in recent years.

3.2 Poultry Chain Control Management

3.2.1 Context and Motivation

The sustainability of the world's demographic, social and economic systems are issues that arise along with the growth of the global population. One of the main challenges that governments are focusing on in recent times is related to agriculture. The sustainable optimisation of plant cultivation and animal breeding both for direct consumption and for other products is being pursued. In this context, meat production is one of the most important agricultural sectors in Europe.

AI technologies are also being introduced in the agri-food sector due to the growing trend of digitalization. In particular, the data collected hides patterns that help decision-making for the optimisation of poultry-rearing processes. However, the task of coordination with the agricultural sector is particularly challenging as it is historically so far removed from computing areas. As a result, the quality of the data reaching the analysts is often low, incomplete and late, as it requires the collaboration of farm and slaughterhouse operators. In addition, the reliability of results in this sector is more demanding due to the low confidence in the farming industry. To break down this barrier, communication between operators and analysts

is considered essential, with the latter providing intuitive explanations and inexpensive implementations. The reason for choosing this case, therefore, is composed of two challenges: the development of explanatory predictive models to make simple decisions in poultry farms, loading and transport, and obtaining reliable results when the quality of the available data is low.

3.2.2 Use Case Presentation

In accordance with the tasks set out in the H2020 Internet of Food & Farm 2020 (IoF2020) project, the objective was to develop an information system for operators involved in the poultry chain to enable them to make decisions for the improvement of the final meat quality. To this end, the different stages of the process were monitored to acquire data and train ML models. The objective of the supervised learning task was to provide decision rules for the optimisation of the process in order to increase meat quality. Fitting transparent and easily explainable models contributes to increasing confidence in data-driven solutions applied in the agri-food industry.

The research project involved 4 different poultry farms that provided data from the production chains at four different phases: breeding in the farm, load to trucks, transport and reception in the slaughterhouse. In the first three phases, time series data are collected from different sources and in the last phase, multivariate data are available for each of the breeding.

- Breeding comprises the approximately 7 weeks from animals arrive at the farm until they are considered to be of the right size. Two-time series data (temperature and humidity) are collected every 30 minutes from 6 environmental sensors installed at different points.

- Loading is the procedure in which the operators load the chickens into the truck and it is monitored by electronic wristbands. Arm sway acceleration in three axes is acquired during the whole loading phase with a frequency of 1 ms. Therefore three signals with a time series structure are collected during the loading phase.
- Transport comprises the truck journey from the farm to the slaughterhouse and has a different duration depending on the location of the farm. To monitor animal transportation, trucks are equipped with sensors that measure temperature, humidity and acceleration in three axes. Those five-time signals are the data available for the analysis of the poultry transport phase to the slaughterhouse.
- Finally, approximately 15 indicators are collected at the slaughterhouse from a random sample of 200 chickens from each truck. Included in the set of KPIs are some to assess the physical condition of the animals on arrival and others related to the performance of the processing and packaging tasks of the chickens. One of the indicators is a categorical variable to measure meat quality, that takes two values: A if the quality of the final meat is considered to be of high quality and B otherwise. The remaining indicators are numerical. The number of samples from each farm varies according to the farm's production. Initially, information was available for 12 to 16 poultry flocks that completed the chain.

3.2.3 Proposal Solution

Initially, the R `dqts` package is used for the evaluation of DQ in the time series available for the analysis of the problem.

- To calculate DQ metrics of the time series received on the farm, the maximum temperature is set to 33 °C and the minimum to 20 °C. Those values are agreed with operators who decide that values outside that range can be considered erroneous sensor measurements. On the other hand, humidity is measured in percentages and an anomalous value is understood to be any outside the range [0, 100]. The execution of the DQ metrics returns failures in *Range* and *Timeliness*, with respective values of 0.97 and 0.99. In *Range* it is detected an average of 19 outliers for the temperatures of the 6 sensors on the farm. As the number of out-of-range values is considered small in proportion to the length of the time series and they occur at non-consecutive points, it is decided to correct those values by replacing them with a linear interpolation between the previous and next values. In addition, the sensors lose connection on a few occasions causing a decrease in the *Timeliness* metric score. Those stops lead on average to a loss of 8 observations, which are also imputed by linear interpolation. Figure 3.1 shows the imputed values in the temperature series received by sensor 2 in one of the farms using the `handleDQ` function of the `dqts` package. In total 13 out-of-range values were found, above 33°C or below 20°C. The horizontal bands on the graph limit these bands and the corrected values are shown in green on the graph. Missing values were imputed in red on the graph after identifying five waiting times longer than 30 minutes between consecutive observations.

The remaining DQ metrics reach the maximum score so that after those two corrections, the temperature and humidity time series in the rearing data are of the highest possible quality.

- None of the three signals recorded during loading showed failures in the quality metrics. In this case, the *Range* was not evaluated as the detection of abrupt movements was the main

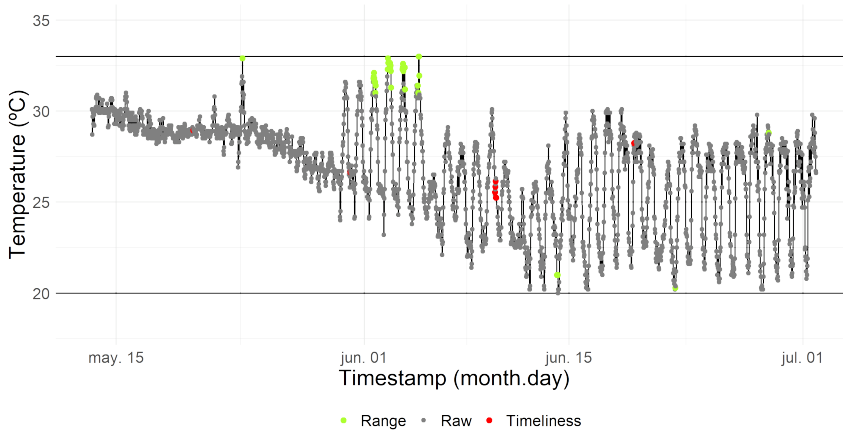


Fig. 3.1.: Temperature time series of sensor 2 after DQ correction with the `dqts` package of R

objective of this phase. Each of the loads corresponds to a data set of 8700 to 9500 values captured every second, i.e. with a duration of approximately 2.5 hours.

- In the case of acceleration in the transport phase, it makes no sense to set strict ranges since it is precisely the abrupt acceleration values that correspond to peaks in the signals that are to be detected. DQ achieves the highest value in all cases so, no technique for quality management is required.
- The data sets sent from the slaughterhouse was generated by the operators by means of a simple questionnaire. In this way, the information collection procedure is standardised and the quality of these data is high. However, there are four features related to the neck and knuckles of poultry for which no information is available. In that case, the quality correction technique is elimination as there are no values for these variables to be imputed.

For the generation of rules to make decisions in the production chain, the strategy of extracting features from each of the available time

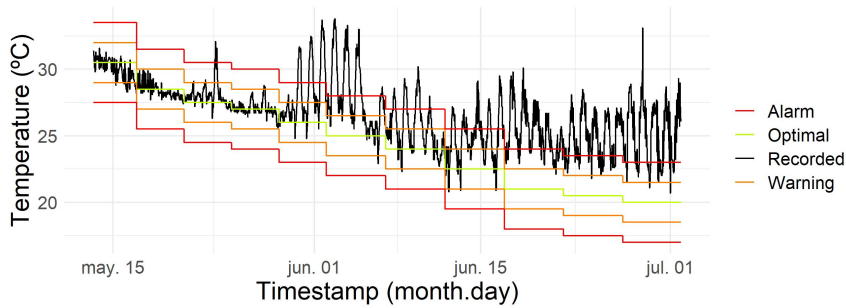


Fig. 3.2.: Temperature and comfort bands captured in one of the farms.

series is chosen. A feature extraction task is carried out to obtain KPIs that characterise the signals and can be used as inputs in the ML models. This process is customised at each stage of breeding, loading and transport. In the last phase, operators already provided values for different numerical quality indicators when animals arrive at the slaughterhouse.

- The comfort requirements of the farm vary with the growth of the animals and five KPIs are defined from this premise for each signal during breeding. First *Temperature Warning* and *Temperature Alarm* measure the percentage of values that deviate abnormally from the expected optimum temperature. An example of the temperature captured by one of the sensors during rearing on one of the farms is shown in Figure 3.2. The optimum temperature at which the farm should be is shown in green. That temperature varies depending on the day of rearing, as the animals need different comfort conditions depending on their maturity. The orange bands limit the values for the calculation of the *Warning* indicator. Values outside the red limits will be counted in the *Alarm* indicator. As can be seen, maintaining the optimal temperature inside the farm is a difficult task, especially in the hottest months of the summer.

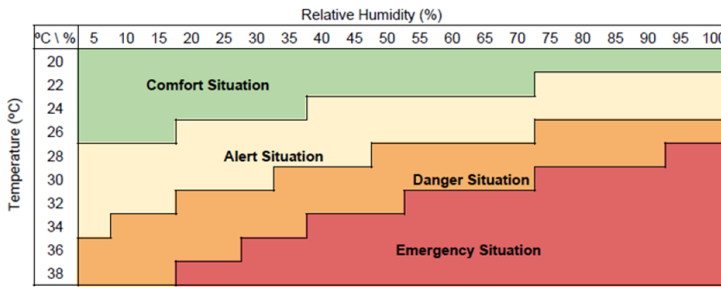


Fig. 3.3.: HSI boundaries used to evaluate the risk of heat stress in poultry.

Furthermore *Alert Situation*, *Danger Situation* and *Emergency Situation* indicators are percentages of different stress levels calculated from a combination of farm temperature and humidity. It is known that the effect of humidity on the quality of poultry meat depends on the temperature at that time. Those three indicators are based on the data in the table of Figure 3.3 of the Heat Stress Index (HSI) taken from studies by experts in the field (82).

One of the answers provided by the decision system when introducing these five inputs is the impact of the anomalous values of temperature during rearing. Based on the results of the model, it is evaluated whether these indicators could be defined more loosely.

- Three KPIs are defined in the loading phase from the acceleration calculated from its three components. *Saturation Rate* is the average of the amount of acceleration data that exceeds the sensor range per minute. Furthermore, *Mean Accumulation* and *Standard Deviation* are based on the amount of acceleration every minute, calculated from the maximum value of the acceleration module every second.

- Four environmental KPIs are defined based on expert knowledge in the transport phase: *Low Temperature* (below 18°C), *High Temperature* (aver 31°C), *Low Relative Humidity* (below 60%) and *High Relative Humidity* (over 80%). In addition, from the module of the acceleration of the truck, a percentage of abrupt movements can be calculated and is used as a fifth indicator. This indicator is calculated using a peak detection algorithm applied to the time series of the acceleration signal called The Smoothed Z-score Peak Detection Algorithm. This algorithm has three input parameters. The first one indicates the size of the moving window, i.e. it determines the number of values used for smoothing. In this case, it is set to 30 to use the previous half hour, as the data are acquired on a minute-by-minute basis. The threshold parameter, set to 3, indicates the number of standard deviations from the mean to construct the bands. Finally, the influence can be set to 0 or 1 depending on whether an update of the threshold throughout the calculation is allowed. In that case, the threshold remains constant, so the influence is 0. The result of the algorithm applied to one of the sensors on a truck can be seen in Figure 3.4. The KPI extracted from this algorithm is named *Abrupt Movements* and it represents the percentage of out-of-band values with respect to the total number of observations collected.

After the feature extraction process, each of the poultry rearings corresponds to multivariate data with values from each of the stages of the chain. Specifically, 27 indicators are defined from each poultry chain as can be seen in Figure 3.5. KPIs in grey are used as inputs and the binary indicator of final meat quality shown in blue is used as the output to be estimated. The observations are considered independent, so that after this pre-processing no special time series treatment is required.

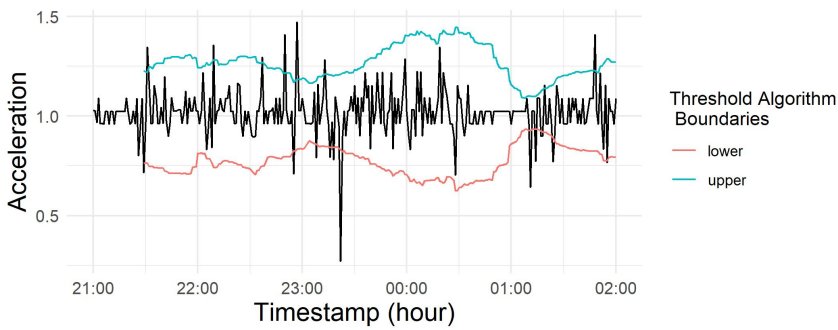


Fig. 3.4.: Boundaries for detecting anomalous peaks in the truck acceleration module

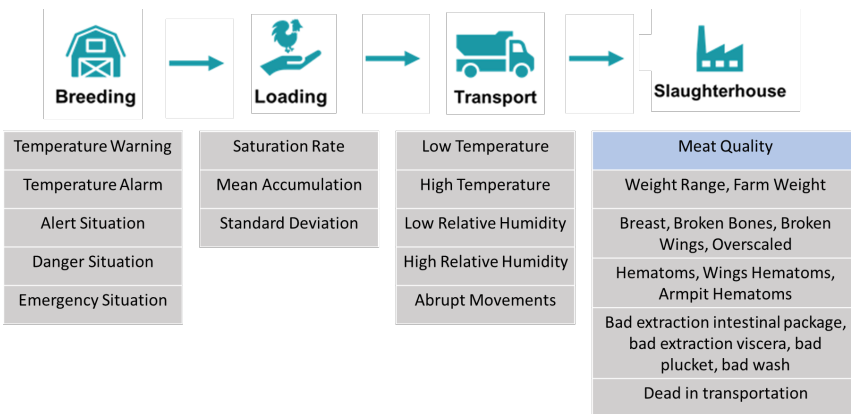


Fig. 3.5.: Summary of features extracted from the time series at different poultry chain steps.

Initially, between 12 and 16 data sets of breedings are available for each of the farms. This information is considered inadequate for the training of a supervised learning model due to the small number of observations available to identify patterns of behaviour. Therefore, to increase the reliability of the classification model, the observations are replicated five times by adding noise to create a more variable data set that helps generalisation.

As the objective was to create rules that allow decisions to be made at different stages of the chain, a DT algorithm called CART was chosen. That is a supervised learning algorithm that uses a branching structure to partition the data and obtain relationships between the input variables and the output variable, in that case, the final quality of the meat.

All development is implemented in R and for the adjustment of the parameters of the DT of the *rpart* package, a partition of 70% of the data is made for training and 30% for testing using a 5-fold cross-validation. The Gini impurity is used as the splitting criterion, and the other control parameters specified are the minimum number of observations required to split a node (*minsplit* = 8), the minimum number of observations required in a terminal node (*minbucket* = 5), and the maximum depth of the tree (*maxdepth* = 3).

With the information from the best tree obtained after the fitting process, the final model is built using all the data. Figure 3.6 shows an example of an initial DT for one of the farms. The tree starts at the top with 100% of the data in the training set, where 75% belongs to class "A" and 25% to class "B". The colour of the nodes is related to the label assigned to the data of that group. Two partitions are performed by taking the *Abrupt Movements* and *Low Temperature* variables and classification into three groups is reached in the final ramification. On the left, the first group labelled "A" contains 72% of the data with no representation of class "B". The second group labelled "B" contains 9% of the total input data and 71% of that group

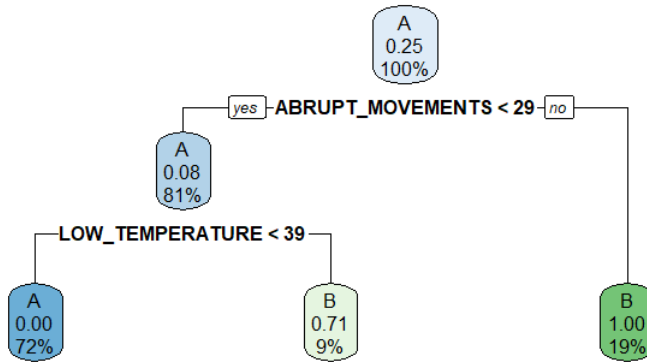


Fig. 3.6.: Initial DT for one of the farms simulating data adding noise to replications.

actually belongs to class "B", i.e. the predictions in this rectangle have an accuracy of 71%. Finally in the last group labelled "B" there are 19% of the data and no values classified as "A". The result can be interpreted as: "if the percentage of *Abrupt Movements* is less than 29% and the percentage of *Low Temperature* is less than 39%, the final meat quality is A; and otherwise, the final meat quality is B".

In this case, the positive class corresponds to meat quality "A". The importance of the decision support system based on algorithms predicting the quality of the poultry meat is focused on identifying animals with quality "B" that can be removed from the flock. The objective of the problem is to identify all "B" values and not allow any batch of "B" quality poultry to be considered "A" quality. The opposite risk of losing any batch of "A" quality by being labelled as "B" is considered acceptable. Then the objective is to reduce the *False Positive* values as much as possible. That is, the classification model has to provide the highest *Precision* value. The confusion matrix obtained in the test set with the above tree is shown in Table 3.2 with a *Precision* score of 1.

Tab. 3.2.: Result of the classification on the test set.

		Predicted	
		B	A
Actual	B	3	0
	A	1	22

The simulated observations are replaced by new records as data from other poultry chains become available. In this way, each time new data is received, the training set is modified to provide a DT with increasingly fine-tuned rules. The more poultry chain data provided, the higher the reliability of the generated rules.

The exploitation of those indicators and the prediction algorithm is done through a Shiny application that allows the visualisation of the collected chain data and the DT built from this information. The complete decision support system (DSS) is named PUMA (PoUltry Management Advisor), an Artificial Intelligent system inspired by the inherent requirements of the improvement of the whole poultry production chain. A screenshot of the final step for one of the farms participating in the project is shown in Figure 3.7. The tool allows the selection of the variables involved in the DT. This allows decisions to be made at different stages of the chain, avoiding the addition of indicators belonging to past stages. As can be seen, the model has been enriched with more data and the complexity of the rules reflects the variability in the set.

3.2.4 Conclusions and Outcomes

The PUMA decision support system and the methodology used for its development provide easy handling of the classification models and easy interpretation of the results. By improving the data set employed in the AI-based system, its reliability is increased by providing a more comprehensive and representative sample of the real-world

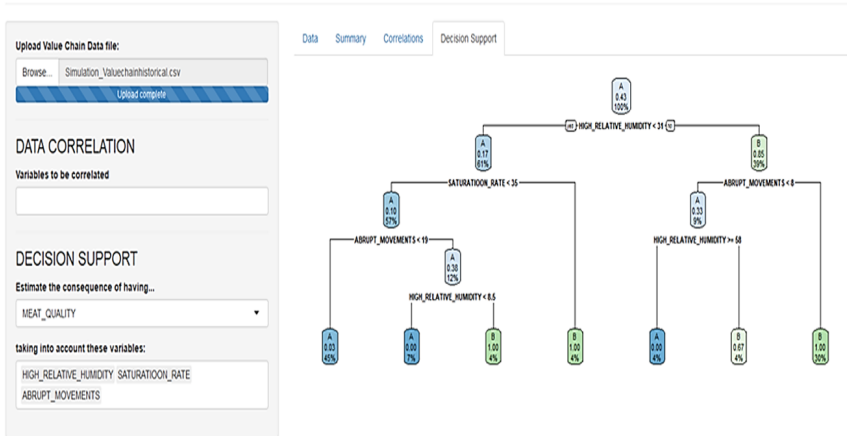


Fig. 3.7.: Capture of the DT for quality improvement in the poultry production chain provided by the PUMA tool.

scenario. Furthermore, the use of a classification tree provides significant advantages in terms of explainability. Unlike other complex ML models, classification trees are inherently interpretable and transparent in their decision-making process. This is particularly valuable in applications where comprehending and justifying the decisions made by the system are necessary.

The experimentation and results of this research work were published in the Q1 quartile journal *Sensors*. The article entitled *An IoT Platform towards the Enhancement of Poultry Production Chains* is attached in Section 5 (22).

3.3 Energy Demand Forecasting

3.3.1 Context and Motivation

Demand Response (DR) activities including load shifting or peak shaving have a huge potential to match energy demand with the energy supply side, thus avoiding these undesirable peaks (83). The first stage in the design of such systems is the accurate forecasting of electricity demand in order to avoid unnecessary investments. Short-term forecasting is used to adapt electricity production and optimise costs based on hour-ahead, day-ahead and week-ahead studies.

Forecasting energy demand at the household, building or even community level is a difficult challenge due to the number of parameters involved and their variability (84). This problem is usually approached from a univariate time series analysis point of view since consumption data are collected over time and a forecast for a future time is requested. The data-driven approaches applied are diverse and range from classical statistical time series algorithms such as ARIMA (85) to ML such as Support Vector Machines (86; 87) or DL algorithms such as Neural Networks (88; 89; 90).

Different types of data are often used as inputs for such models, depending on their availability. The most relevant are weather variables such as temperature and humidity (91; 92), building occupancy information (93; 94) or calendar variables such as day of the week or type of day (95; 96).

Quality issues in energy consumption data acquisition are primarily caused by sensor disconnections, resulting in short or prolonged periods of missing data and negatively impacting the collected information. When the sensors reconnect, there is often a consumption peak due to the accumulation of values that were not saved during the disconnection. Furthermore, in the case of electricity demand

forecasting, there is a significant cyclical component, with historical patterns being more repetitive than in other industrial forecasting problems. This cyclicity makes electricity demand forecasting an interesting case study.

3.3.2 Use Case Presentation

One of the main tasks of the RESPOND project mentioned in Section 1.4, was forecasting hourly electricity consumption for the subsequent day to optimise the energy production based on demand. The project used data from various single-family houses located in Madrid (Spain), Aran Islands (Ireland), and Aarhus (Denmark), where sensors were installed in each house to capture hourly electricity consumption data in kWh, with a univariate time series structure. The data was available from autumn 2018 to July 2020, the period during which this study was conducted. For the study, 24 dwellings in Madrid were selected to test different algorithms based on historical electricity consumption data.

In the REACT project, the activity was focused on island communities, so data from buildings located on the island of Lanzarote (Spain) have been used for the experimentation. Data on hourly electric consumption have been provided by the Spanish utility Fenie Energia2. The buildings have a label that identifies them as dwellings or small commercial, but it is not possible to know the area, population, number of rooms, opening hours, the lifestyle of the users, etc. The date on which the data collection begins varies for each data set, although the majority are from the second half of 2018. The data export date was 2020/03/15 so all time series end on that date. Hourly electrical consumption data (Wh) are collected for each of the buildings. Therefore, 364 data sets with records of very different duration from 1 week (168 records) to more than 2 years (18253

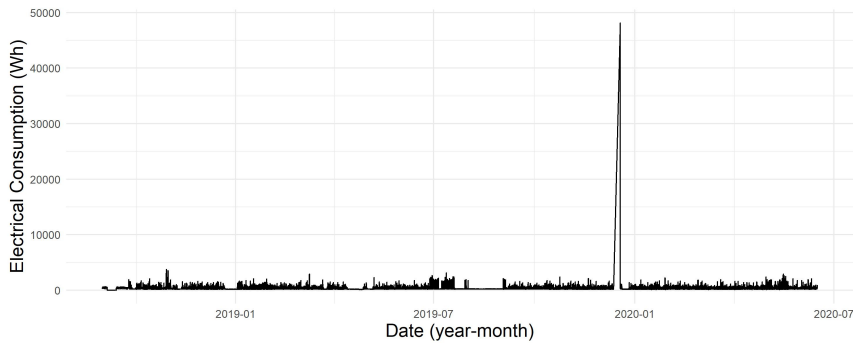


Fig. 3.8.: Outlier produced after the temporary disconnection of the sensor in one of the houses in Madrid.

records) are available. The main objective, in this case, remains the same: forecasting future electric demand for the upcoming days.

3.3.3 Proposal Solution

The main quality issue detected in the data was the loss of data due to signal connection loss with the sensors, which is a common issue in industrial problems. When the connection was reestablished, a peak was detected due to the accumulation of data during the shutdown period. This phenomenon can be observed in Figure 3.8.

Using the `dqts` time series DQ package explained in Section 2.2, it is instantaneous to know the reliability of the available data, prior to analysis. This example gives a *Timeliness* value of 0.9865 and a *Range* value of 0.9999. Using the `deepDQ` inspection function, Table 3.3 is obtained, so it is known that in this case there have been four signal disruptions of varying durations. In total, 153 values have not been received. In addition, 3 out-of-range values are detected in the positions listed in Table 3.4, which correspond to three of the sensor reconnection timestamps. The value for the loss on November

Tab. 3.3.: Output of deepDQ function executed in *Timeliness* metric

Loss Start	Loss Finish	Waiting Time	Missing Amount
2019-06-20 14:00:00	2019-06-20 17:00:00	3 hours	2
2019-07-05 07:00:00	2019-07-05 14:00:00	7 hours	6
2019-11-07 18:00:00	2019-11-07 20:00:00	2 hours	1
2019-12-12 14:00:00	2109-12-18 15:00:00	145 hours	144

Tab. 3.4.: Output of deepDQ function executed in *Range* metric

Variable	Date
Energy	2019-06-20 17:00:00
Energy	2019-07-05 14:00:00
Energy	2019-12-18 15:00:00

7, 2019, was not detected as an outlier because the accumulation produced in two observations was not considered sufficiently high.

The `handleDQ` function is used to generate and impute missing values, as well as correct out-of-range values using the KNPTS method. Figure 3.9 shows an example of the correction performed on July 5, 2019. After this correction process, the electricity consumption time series data achieves the highest quality score.

After obtaining the desired DQ, several models are compared to evaluate their forecasting accuracy. The relationship of the data to past hourly consumption records suggests the use of ARIMA-derived models. The SARIMA model is also selected because it is suitable for data with daily periodicity. The additive decomposition

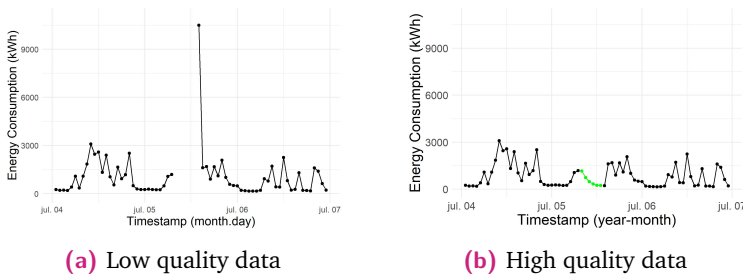


Fig. 3.9.: Example of the process of increasing DQ with the *dqts* package

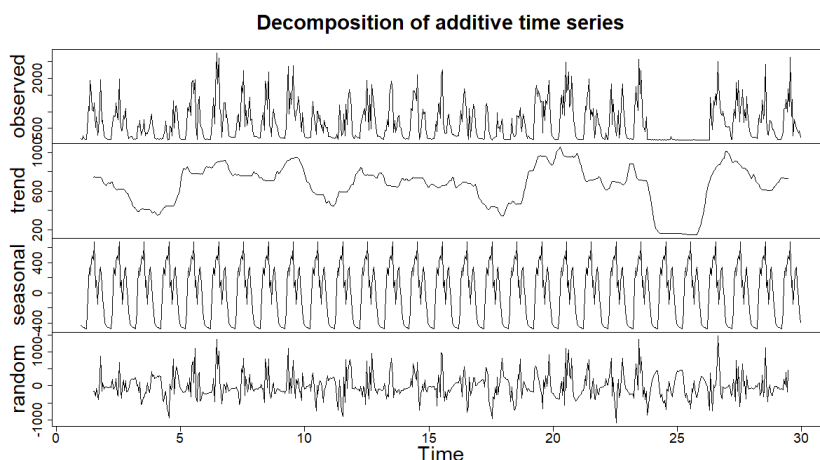


Fig. 3.10.: Additive decomposition of one month's data from hourly electricity consumption series

in Figure 3.10 corresponds to one month's data from one of the electricity consumption series. The third graph shows the clearly daily seasonality hidden in this type of data. The problem with time series models that do not admit other external features as inputs are that some behaviours are not easy to capture. On weekends or holidays, electricity consumption may vary in relation to the rest of the days. Another particularity occurs on holiday days when no consumption takes place in the house, as can be seen between the 24th and 26th of this example.

The parameters of the SARIMA model can vary considerably depending on the training data selected. For example, using the values shown above, the best model would be a SARIMA(4,0,0)(2,0,0,0)[24] using only autoregressive variables, which would cause early degradation of the model due to error accumulation. The prediction for the next 10 days is shown in Figure 3.11.

However, a SARIMA(1,0,1)(2,1,0)[24] model is fitted if the training set has no notable behaviours such as those observed during the holidays. For instance, if the training set comprises data from the

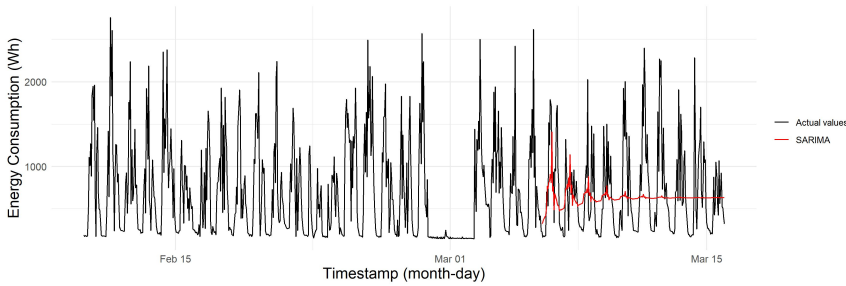


Fig. 3.11.: Hourly forecast of electricity consumption data for ten days using SARIMA(2,0,0)(2,0,0) [24].

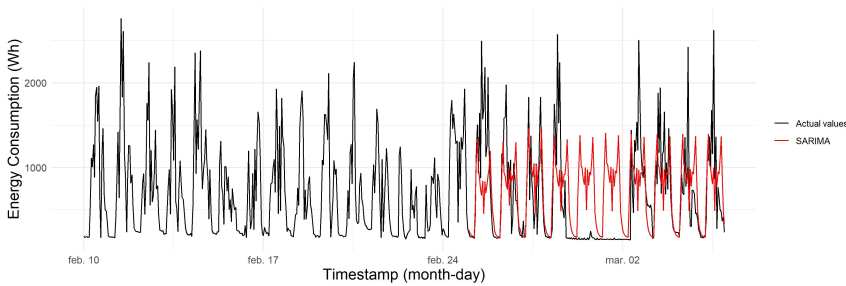


Fig. 3.12.: Hourly forecast of electricity consumption data for ten days using SARIMA(1,0,1)(2,1,0) [24].

10th of January and data up to the 25th of February, and a forecast for 10 days ahead is conducted, the outcomes presented in Figure 3.12 are achieved. In this instance, the error accumulation was less apparent, but the model failed to capture variations in the time series patterns as it was not trained on the available data.

If the forecast horizon is longer than the seasonal pattern of the data, the model tends to replicate comparable patterns each time a period is finished. This phenomenon was addressed in Section 2.3.1, which concluded that a different model training strategy should be considered to achieve a good fit. In this scenario, long-term forecasts are required, and there is no plan to retrain the models on a daily basis. For that reason, SARIMA models may not be a suitable choice for addressing such problems.

In addition to SARIMA models, three ML algorithms commonly used in the literature for this type of problem are compared: LR, SVM, and KNN. Date and time features, including the day of the week, day of the month, the month of the year, and the hour of the day as numerical features, the season of the year as a categorical variable, and a binary variable indicating if the day is a working day or not, are used as input variables for the ML models. The hour and month variables are transformed using the trigonometric transformation of Equation 2.11 to preserve their cyclic behaviour in the distance-based algorithms.

In this case, sliding windows cross-validation is used, as it is deemed more appropriate than the expanding window method, given the considerable historical data available. The training windows are set to last for one month, while the testing windows last for ten days, which corresponds to the prediction horizon size.

Accuracy was measured in this case using RMSE only and the algorithm that provided the lowest error was KNN. An example of future data forecasts using the KNN with $k = 9$ neighbours is shown in Figure 3.13. With this technique, the algorithm is capable of estimating future values by taking an arithmetic mean of the 9 most similar values in terms of calendar data. This similarity is calculated based on the Euclidean distances between each future point and all available data in the historical record. As can be seen, some of the consumption peaks cannot be accurately captured. This fact suggests that more historical data may be necessary, or that there is a random component that the algorithm is unable to capture with the available information.

To continue research along these lines, emphasis was placed on the data from the 10 houses in Madrid. After repeating the data cleaning procedure to improve DQ, it was observed that electricity consumption data for private residences tend to repeat patterns over time. The behaviour of users has a cyclical component that is

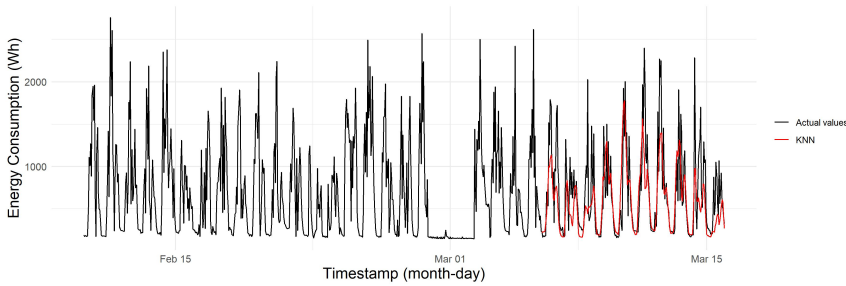


Fig. 3.13.: Hourly forecast of electricity consumption data for ten days using the KNN with date and time features.

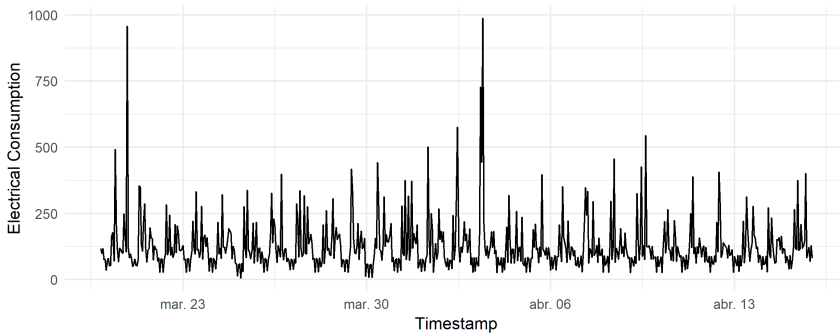
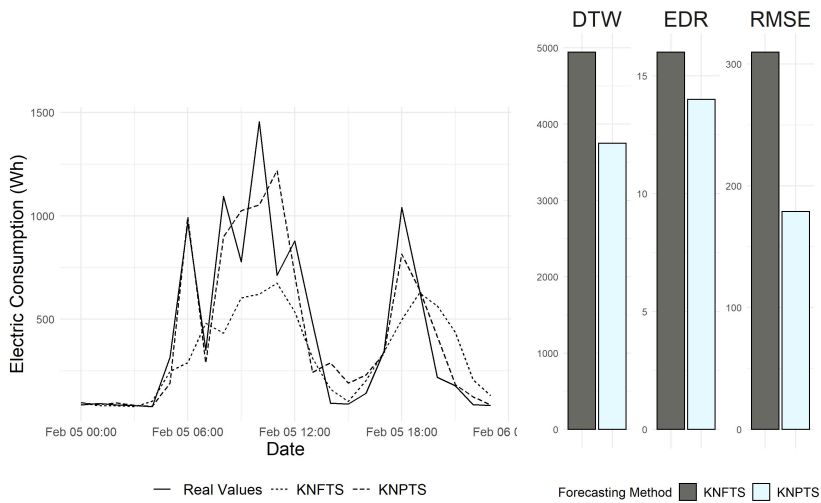


Fig. 3.14.: Extract from the electricity consumption data for households in Madrid

reflected in repetitions in historical consumption values, as can be observed in Figure 3.14.

Therefore, a comparison was conducted to evaluate the effectiveness of two different KNN-based algorithms: the classical KNN using time- and date-related features as inputs named KNFTS and the KNPTS which employed the consumption of the last 24 hours as the pattern. In this case, the chosen strategy was MIMO to provide the estimated values for the next 24 hours in a single run. The performance of the two algorithms was evaluated using three measures of error: RMSE, DTW, and EDR. The DTW was evaluated by allowing the alignment of predicted and actual data with a unit of time margin, i.e. the predictions are compared with the actual value at that instant of



(a) Comparison of the forecasts

(b) Error measurements

Fig. 3.15.: Evaluation of one-day ahead forecasting with KNFTS and KNPTS.

time, in the hour before and in the hour after. The threshold in the calculation of the EDR metric is set at $\epsilon = 30Wh$, considering that two points that are less than this value apart can be considered equal. The results indicated that for this type of data, KNPTS provided more accurate forecasts in terms of similarity between the actual and estimated daily consumption.

This solution was implemented with a daily periodicity in Madrid households, and it was observed that after the outbreak of the COVID-19 pandemic, consumption patterns changed. The impact of confinement on the model results was evident from a reduction in the variability of the historical data. As this change could be associated with the containment measures arising from the pandemic, the date was known, and the models were retrained using only data from the new period. This further increased the accuracy of KNPTS, which now reflected the Spanish date of confinement.

Later on, the comparison between KNFTS and KNPTS was repeated on data from different households and small commercial establishments in Lanzarote, Spain. In this case, predictions were made for one day ahead and one week ahead and the error measures used were the same as in the previous case: RMSE, DTW and EDR. In both cases, it was concluded that the pattern-based algorithm provided more accurate results. Therefore, KNPTS was deployed as the optimal solution for one-day ahead forecasting of energy consumption data.

Another line of research in energy demand concerns the degradation of deployed models. Following the observed change in Madrid's energy consumption data with confinement, the development of PRENERGET was initiated. This is an automatic training tool for electricity consumption forecasting models capable of detecting the drift in the prediction error with two objectives. The first is to inform the user that the model in production is suffering a degradation in its accuracy. On the other hand, the re-training of the model with a new training set to try to minimise the impact of the variability that has occurred in the historical set. This modular tool allows the incorporation of data from different sources, the implementation of different ML models and the measurement of forecast error using different techniques. Currently, the solution implemented to solve model degradation is to update the input data by means of a sliding window that discards the oldest data and incorporates the most recent data. However, the modular structure of PRENERGET allows other techniques to be implemented to solve this drift.

3.3.4 Conclusions and Outcomes

DQ issues related to connection losses resulting in missing values and consumption peaks are automatically resolved using the functions implemented in the `dqts` package in R. This type of solution reduces

data preprocessing costs and significantly improves the quality of input data for the models.

The initial comparisons conducted help guide the selection of algorithms to be used and rule out less accurate or computationally expensive techniques. The results of those comparisons provided by each of the initial models were presented at the Sustainable Places 2020 conference. The complete experimentation was published in the Proceedings of the conference with the title *Short-Term Electric Demand Forecasting for the Residential Sector: Lessons Learned from the RESPOND H2020 Project* and the article can be read in Section 9.

After selecting KNN as the most effective base algorithm, the proposal of KNPTS as a prediction algorithm provides simplicity to the problem, as it relies solely on historical data as input. The experimentation done in Madrid data to compare KNFTS and KNPTS and the results of these predictions were presented at the 40th SGAI International Conference on AI. The article entitled *Short-Term Forecasting Methodology for Energy Demand in Residential Buildings and the Impact of the COVID-19 Pandemic on Forecast* received the Best Application Student Paper award. This publication can be read in Section 9.

That experimentation was extended to validate the conclusions obtained on a more extensive and varied data set, demonstrating the efficiency of KNPTS for this type of problem. This development was published in the Q1 journal *Energy and Buildings* under the title *k-Nearest Patterns for Electrical Demand Forecasting in Residential and Small Commercial Buildings*. This article can be found in Section 6.

Finally, the development of the PRENERGET tool provides a controlled methodology for deploying prediction models. With this development, the detection of drifts in prediction errors is monitored, contributing to increased reliability of the solution by detecting its

degradation. PRENERGET was presented at the 2022 European Conference on Computing in Construction and it was published in the EC3 2022 Proceedings. The article entitled *A Framework for the Inclusion and Adaptation Strategies of Machine Learning Models for Energy Demand Forecasting in Buildings* can be found in Section 9.

3.4 Porosity Detection in Additive Manufacturing

3.4.1 Context and Motivation

Additive Manufacturing (AM) consists of material deposition layer by layer following a sliced Computer-Aided Design (CAD) geometry. Multiple metal AM processes can be found in the literature with different degrees of development and which are classified based on the feedstock format and the energy source. This use case is focused on the direct deposition of material in wire format and a laser beam that melt both the added material and the substrate, called wire Laser Metal Deposition (wLMD). This manufacturing process achieves a balance between the advantages of AM and makes it an appropriate technology for the manufacture of medium and large parts, obtaining near net shape geometries (97; 98). However, due to the complexity of the material deposition process, it has not yet been mastered enough to enable the industrialisation of wLMD.

The complexity increases in 3D geometries, where instability in condition definitions can lead to process failures or defects as pores (99; 100). Porosity is one of the most significant defects in metal AM that affects the mechanical performance of parts, especially fatigue properties (101). The Computed Tomography (CT) analysis is a Non-Destructive Testing (NDT) method that provides information

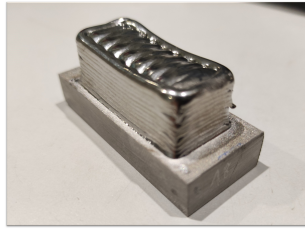


Fig. 3.16.: Part manufactured by a wLMD process.

about the internal pores in the manufactured part (102). However, the in-situ inspection is not possible, and the efforts are focused on the development of alternatives based on data and AI algorithms to generate porosity detection models.

Researchers have dedicated tremendous efforts to monitoring defects during printing to store large amounts of data to train ML models (103). However, these types of techniques require a high amount of data from both healthy and defective parts, and the number of parts with faults is usually not high in the industry. This limitation makes ML models less reliable since they cannot properly validate the proposed solution. In this case, treating the observations separately can enrich the data set as long as the time series approach is not necessary. That is, if the appearance of defects is considered a local failure, the evolution of features over time is not considered essential for defect detection, and therefore observations can be treated as independent records.

3.4.2 Use Case Presentation

This problem aims to identify the most effective data-driven modelling technique, i.e., models trained using in-situ and post-manufacturing characterisation, for two objectives: local detection of porosity and early porosity prediction. The data is obtained from the manufacturing of three 3D geometry parts, as depicted in Figure 3.16.

These parts consist of 12 rectangular layers with dimensions of approximately 16 by 40 mm each, manufactured using a predetermined material deposition trajectory that includes a perimeter strategy and zigzag for interior filling. The manufacturing time for each part ranges between 18,2 and 18,8 seconds. The high-frequency monitoring of the process generates a data set of approximately 19.000 observations per part containing information on the following features:

- The position over time is recorded, which defines the material deposition trajectory performed by the robot. The signal is composed of three univariate variables: X, Y, Z, and Time. Both variables serve as the reference to merging all acquired data, as it includes the relation between space and time information. From the position record, the *Vertical Movement* during material deposition is extracted. This variable describes the positioning inaccuracy of the robot in the vertical direction. It is measured in the Z axis with respect to the preset height value of each layer.
- During the material deposition process, two types of signals are acquired with reference to the processing time. On the one hand, the signals that control the laser power and the wire feed speed are recorded by the Programmable Logic Controllers (PLC). On the other hand, images of the area where the material is being added are acquired by coaxial monitoring of the melt pool. The homogeneity of the growth across the manufacturing process is obtained through geometric scanning. Since the manufacturing is carried out layer by layer, after each layer is deposited, the resulting surface geometry is scanned. The resulting 3D point cloud that represents the surface is spatially referenced to the trajectory and the manufactured part.

The *Overlap Factor* variable is calculated from the overlapping conditions between the beads that form each layer. In fact, the trajectory recorded describes the distance between adjacent beads, which affects the uniformity and effectiveness of the filling of the layers. A bigger gap between two adjacent beads means less overlap between them and therefore less material in that area. Thus, the *Overlap Factor* is calculated based on the distance of the beads to each other along each layer.

After manufacturing of each layer, the surface geometry is scanned and compared with the theoretical growth, from which the geometric deviation is computed. Two variables are extracted when merging the information with the trajectory. On the one hand, if linked to the movements of the last layer deposited, the variable *Z Distortion* describes the resulting geometric distortion. On the other hand, if it is linked to the trajectory of the next layer, the information refers to the geometric conditions of the base surface on which the material is deposited, which is named *Base Distortion*. Notice that in the first layer, this value should be 0 because the material is deposited on a flat surface.

- After the completion of the manufacturing of each part, the porosity is measured offline. The 3D CT shown in Figure 3.17 provides data on the location of the centre of each pore as well as its volume and shape, which is spatially referenced to the part geometry. This information is used to label the recorded data sets and distinguish the pore observations.

Based on the porosity information, each pore is simplified and visualised by ellipsoids calculated from their corresponding length, width and height data. The trajectory and the ellipsoids are analysed together and the points of the trajectory that lie within the volume of the ellipsoids are located. Therefore, two labels are created regarding porosity. The *Inside Pore* variable

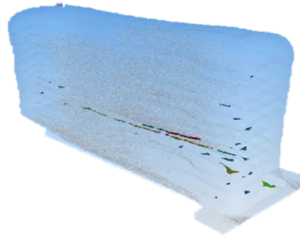


Fig. 3.17.: Scanned part provided by the CT.

takes the value 1 if the observation is within a pore and 0 if not.

3.4.3 Proposal Solution

These data acquisition systems are very accurate, and despite analysing the DQ with the `dqts` package, no waiting times, repetitions, anomalous values, or missing values are identified. Therefore, no DQ improvement technique is needed.

In summary, the data sets available contain the data acquired in-process, some processed signals and the CT scan information as a binary variable that takes values 1 or 0 whether the observation has a pore or not. As previously mentioned, one of the challenges with these industrial cases is that the proportion of failures is minimal compared to the amount of correct data. Table 3.5 displays the number of observations labelled with a pore for each component and the percentage it represents in the entire data set. In none of the three parts does this percentage even reach 1%, which is an inadequate amount of information that could potentially lead the algorithm to make more classification errors than intended.

Tab. 3.5.: Percentage of pores in each of the three parts

Part	ID	N° Records	N° Pores	N° Non Pores	Percentage
1	13	19196	109	19087	0.57%
2	16	19070	144	18926	0.76%
3	18	19281	36	19245	0.19%

To overcome this challenge, a balancing technique is suggested. This approach includes randomly undersampling a large percentage of data from the majority class, followed by employing the SMOTE-Tomek strategy to equalize the two classes. The class imbalance is deemed substantial, where applying SMOTE-Tomek directly would lead to a replication of pores without adding valuable and high-quality information to the study. Hence, the initial decision is to reduce the number of observations without pores, ensuring that relevant information for the classifier's learning is preserved. Each of the classification strategies explained below involves a distinct number of values in the training set. Subsequently, this combined balancing strategy is applied to the processed data to extract the relevant features.

During the phase of creating and selecting the best predictive model, two key aspects are tested. Firstly, three different strategies are proposed for classifying observations to detect porosity: local classification, time series features classification and persistence images classification, which are explained in detail below. Secondly, three methods are suggested for validating the results by employing various train-test splits on the data sets: using the information from the parts separately, mixing them all, or using some parts to train a model that predicts pores in the remaining part.

Each of the classification strategies leads to different conclusions, and the results of all three are compared to determine whether porosity is a manufacturing defect that can be anticipated by analyzing values prior to its occurrence or if, on the contrary, the pores need to be

detected retrospectively after they have already happened. That is, when past features are related to the current failure, it can be concluded that there is a temporal relationship manifested in the values captured before the appearance of the porosity. On the other hand, if the classification is done observation by observation, it is understood that porosity in the parts is a local failure that cannot be anticipated but can be estimated with data-driven models that replace classical CT.

In each of the strategies, the results of four of the most commonly used ML models in this type of classification problem are compared: KNN, SVM, RF, and XGBoost. Each of these algorithms requires a different parameter tuning process, which also depends on the chosen strategy.

1. Local classification is performed in the entire data set to estimate the pores in parts by classifying the captured data during the manufacturing observation by observation. Therefore, raw data shown in Figure 3.18, graph 1, is classified point by point using ML algorithms. In this case, the values of the 4 signals of *Z Distortion*, *Base Distortion*, *Overlap Factor* and *Vertical Movement* are used as inputs, and the presence of the pore is estimated as a binary classification output.

The data balancing in this case is achieved by randomly reducing 80% of the data from the majority class, followed by the application of SMOTETomek to equalise the observations of both classes. This number is chosen due to the large number of values without pores compared to the few values with pores, as mentioned before.

The model tuning is done through an exhaustive search for parameters that provide the highest recall. For each algorithm, an appropriate grid search is designed. For KNN, the optimal

Tab. 3.6.: Percentage of pores in each of the three parts after rolling window processing.

Part	ID	N° Records	N° Pores	N° Non Pores	Percentage
1	13	339	23	316	6.78%
2	16	336	51	285	15.18%
3	18	338	5	333	1.48%

choice of k is tested from 1 to 15. In RF, the set of the number of estimators is $\{50, 200, 500\}$ with a maximum depth of $\{4, 6, 10\}$. The minimum number of samples required to be at a leaf node is tested with values of $\{5, 10, 20\}$, and the Gini criterion is used to evaluate efficiency. The regularisation parameter C of SVC is tested with $\{0.1, 1, 10, 100\}$. The chosen kernel is the exponential function, and its parameter is tested with values of $\{0.01, 0.1, 1\}$. Finally, for XGBoost, different numbers of estimators are tested $\{60, 160, 260, 360, 460, 560\}$ with maximum depths of $\{2, 4, 6, 8, 10\}$. The learning rate is tested for values $\{0.01, 0.1, 0.5\}$.

2. A rolling window of size 100 is applied in the multivariate time series skipping 50 values to avoid unnecessary overlaps. Each of the sub-series is labelled according to the presence or absence of pores, regardless of the quantity. Table 3.6 summarises the number of available sub-series for each part and indicates which contain pores.

The class balancing in this case is done considering the number of sub-series available in the data set, which are 336, 338, and 339 for each part, and the percentage of these sub-series that contain pores. The 40% observations without pores are randomly removed from the data set, and the remaining are balanced using SMOTETomek to have an equal number of sub-series with and without pores.

The model tuning is different in this approach because the number of training instances is not the same. Due to the varying sizes of the datasets in different classification strategies, the process of tuning and optimizing the models may need to be adjusted accordingly. The changes made compared to the local classification strategy are as follows: the depth of the RF is tested for $\{1, 3\}$, and the number of estimators is adjusted to $\{5, 10, 20\}$. For the XGBoost the number of estimators is tested for $\{10, 35, 60, 85, 110, 135, 160, 210, 235, 260\}$.

Then, from the set of sub-series, two different strategies of time series feature extraction are tested: signal-based and persistence images.

- A time series signal-based feature extraction is done to create a set of inputs for ML algorithms from each window shown in Figure 3.18, graph 2. The mean, the standard deviation, the maximum, the minimum, the trend, the entropy and the curvature values are used as explanatory variables to predict porosity.
- The persistence images shown in Figure 3.18, graph 3, are created using TDA for time series following the steps explained in Section 2.3.4 and using `ripser` library for Python (104). The persistence image of each of the available variables is extracted as an array of 20 by 20 pixels dimension. Then, data sets are written as a vector of length 400 and ML techniques to classify the vectorised pixels, using each pixel as an input to the algorithm. Some of these pixels have the same value for all persistence images, so columns with near-zero variance are removed to reduce the size of the input data matrix. In total, information from 367 pixels is retained.

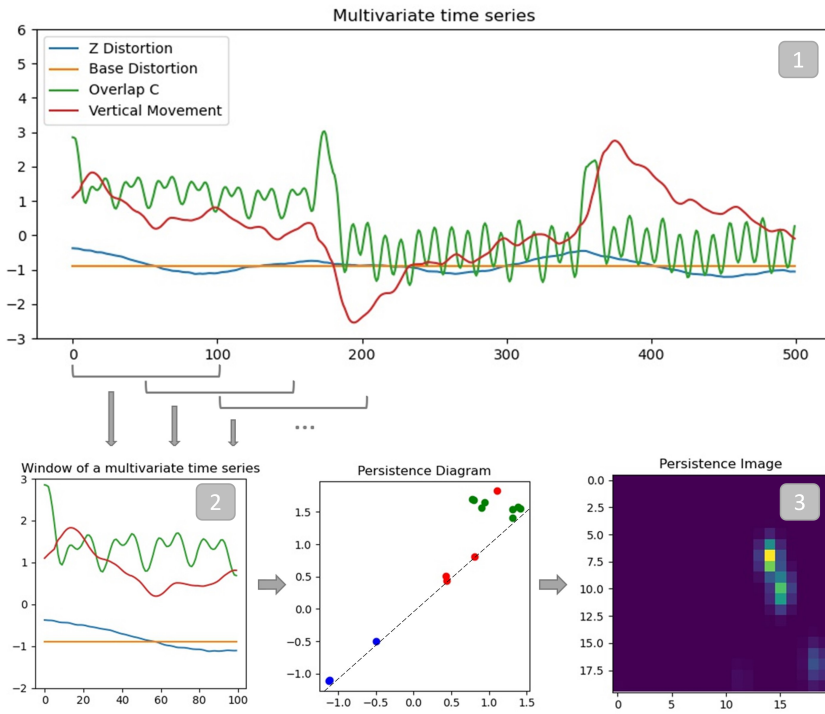


Fig. 3.18.: Creation of the persistence images from the multivariate time series.

As mentioned before, one of the recurring problems in the industry, when classification cases are proposed for fault detection, is the lack of repeated parts to experiment with in model training processes. For this reason, validating data-driven models requires realistic strategies that allow for the extraction of appropriate conclusions for generalisation. For each of the strategies explained below, three different validations are tested to evaluate the efficiency of the approaches.

- For each part, 2/3 of the data is used to train the models and the remaining 1/3 of the part is used to test the fitted model. In the case of classification using features extracted from the sub-series, part 18 is not considered in this validation approach due to having only 5 sub-series with pores.
- Data from all three parts are used together. 2/3 of the data is used to fit the model in training and 1/3 for validation.
- Three tests are done by training the model with the data of 2 complete parts and reserving an entire part for testing.

The results of all the experiments are shown in Table 3.7. Classification errors are measured by accuracy and recall metrics and the mean of the results of the 5 different random train-test partitions are shown.

In general, in any of the strategies and algorithms, classifications are more accurate when using data from the same part to detect pores in the rest of the part. When mixing the data from all three parts and treating them equally, the accuracy tends to remain the same although the recall slightly decreases. However, when using the data from two parts to predict the third part, the recall drastically drops, providing estimations that cannot be considered reliable.

Regarding the forecasting strategies, local prediction achieves promising results in porosity detection. The results obtained with this forecasting strategy and validation approaches are similar for the

Tab. 3.7.: Summary of the performance obtained by detecting porosity using different approaches

Validation Partition	Algorithm	Local Class.		TS features Class.		Image Class.	
		Accuracy	Recall	Accuracy	Recall	Accuracy	Recall
2/3 part to train and 1/3 part to test	KNN	0.954	0.855	0.827	0.665	0.714	0.398
	SVM	0.959	0.855	0.839	0.408	0.641	0.485
	RF	0.937	0.818	0.666	0.702	0.803	0.310
	XGBoost	0.964	0.804	0.730	0.755	0.811	0.299
2/3 total to train and 1/3 total to test	KNN	0.971	0.742	0.814	0.505	0.753	0.418
	SVM	0.956	0.762	0.870	0.264	0.697	0.437
	RF	0.914	0.728	0.620	0.766	0.795	0.400
	XGBoost	0.972	0.694	0.857	0.493	0.801	0.356
2 parts to train and 1 part to test	KNN	0.969	0.393	0.715	0.348	0.748	0.226
	SVM	0.966	0.034	0.871	0.106	0.733	0.295
	RF	0.903	0.267	0.631	0.397	0.813	0.251
	XGBoost	0.975	0.021	0.844	0.211	0.819	0.211

Tab. 3.8.: Confusion matrices from local classification part by part using KNN algorithm.

Id Part		13		16		18	
		Predicted					
		0	1	0	1	0	1
Actual	0	6052	309	5544	762	6258	155
	1	4	34	5	46	1	13
Accuracy		0.951		0.879		0.976	
Recall		0.895		0.902		0.929	

Tab. 3.9.: Confusion matrices from time series features classification part by part using XGBoost algorithm.

Id Part		13		16	
		Predicted			
		0	1	0	1
Actual	0	97	8	77	17
	1	1	7	4	14
Accuracy		0.951		0.813	
Recall		0.895		0.778	

four algorithms. For instance, the confusion matrices in Table 3.8 show classifications performed with KNN in a specific train-test partition. The number of neighbours used are $k = 5$, $k = 7$ and $k = 3$, respectively in parts 13, 16 and 18.

In the case of classifications performed on the features extracted from the temporal subseries of the signals, the difference in results is more noticeable. For instance, the recall obtained with SVM in the first validation strategy is 0.408, the lowest of the four. However, with the same features and validation, XGBoost achieves a recall of 0.755. In the latter case, Table 3.9 displays the classification in one of the train-test splits. In this strategy, the classification remains binary, where 1 indicates the presence of some pore in that time subseries, without specifying the quantity or positions.

Tab. 3.10.: Confusion matrix from time series features classification using the three parts and the RF algorithm.

		Predicted	
		0	1
Actual	0	197	115
	1	6	20

If all the extracted subseries from different parts are considered independent and used to predict porosities, in the second validation strategy, it can be seen that the RF achieves a good classification result with an average accuracy of 2 and an average recall of 9. Table 3.10 displays an example of this classification in the confusion matrix, combining the three parts in a specific train-test split.

Finally, it can be observed that with the extraction of topological features instead of basic features from the signals, both accuracy and recall decrease. This fact suggests that the presence of pores in the parts is related to the absolute values taken by the captured signals, rather than their shape.

3.4.4 Conclusions and outcomes

The accuracy and recall results obtained by all the tested algorithms in the local classification are promising, suggesting an approach for pore detection based on data that could lead to the replacement of the costly tomography techniques currently used for this purpose.

The results obtained by classifying the temporal subseries conclude that there is a way to classify pores using smaller datasets. In this case, the accuracy would decrease as the simplicity of the computation increases.

The poor performance of pore estimations when using information from two parts to predict a third part is due to the context in which

these parts were manufactured. The conclusion that can be drawn in this case is twofold. On one hand, it is necessary to replicate experiments under the same conditions to contextualize the manufacturing process. On the other hand, the variables collected for this problem are incomplete as the algorithms should have some input regarding the manufacturing parameters that make the parts different.

The most relevant results that allowed conclusions to be drawn for future work in this line of research were detailed in Section 8. The article entitled *Optimizing Porosity Detection in wire Laser Metal Deposition Processes through Data-driven AI Classification Techniques* is submitted in 2023 to Engineering Failure Analysis Journal and is now waiting for publication.

3.5 Manufacturing Production Line Diagnosis

3.5.1 Context and Motivation

The diagnosis of manufacturing production lines is a rapidly advancing field, thanks to advances in technology and data analytics. One of the key trends in this area is the increasing use of Predictive Maintenance (PdM), which allows manufacturers to identify potential equipment failures before they occur, minimising downtime and reducing maintenance costs. Another significant development is the incorporation of sensors into production equipment, enabling real-time monitoring of essential KPIs such as temperature, vibration, and power consumption. PdM is a strategy that relies on advanced data analytics techniques, such as ML and AI, to analyse sensor data from production equipment and detect anomalies or deviations from normal operating conditions. By detecting these issues early on, manufacturers can take proactive measures to repair or replace equipment before it fails (105).

In addition, the use of ML and AI tools can analyse large volumes of data from multiple sources, identifying patterns and trends that may not be immediately apparent to human operators. Therefore, the state of the art in manufacturing production line diagnosis is focused on leveraging advanced technologies to increase efficiency, reduce costs, and improve overall product quality.

The choice of algorithm depends on the specific application and the nature of the data being analysed (106). Among the most widely used approaches for their computational efficiency and explainability of results are algorithms such as DT and RF (107). These types of models allow the extraction of the feature importance to understand the influence of the inputs in the predictions. Neural networks such as LSTM are also widely used in this field, especially when data are acquired at high frequency and large data sets are available (108; 109). Hybrid techniques that combine more than one algorithm can also be a good approach to increase model performance (110).

One of the main challenges in PdM, and more specifically in machine diagnosis, is dealing with imbalanced data. Imbalanced data refers to a situation where the number of examples in one class (e.g., healthy equipment) is significantly greater than the number of examples in another class (e.g., faulty equipment). When training an ML model on imbalanced data, the model may become biased towards the majority class, resulting in poor performance in detecting the minority class. This leads to a poor generalisation of the model as it may result in a high number of false positives, as the model may classify healthy equipment as faulty due to the lack of examples of faulty equipment in the training data. To address these issues, various techniques have been developed, including data resampling methods (e.g., oversampling and undersampling), cost-sensitive learning, and ensemble methods. These techniques aim to balance the data distribution and improve the performance of the model in detecting the minority class.

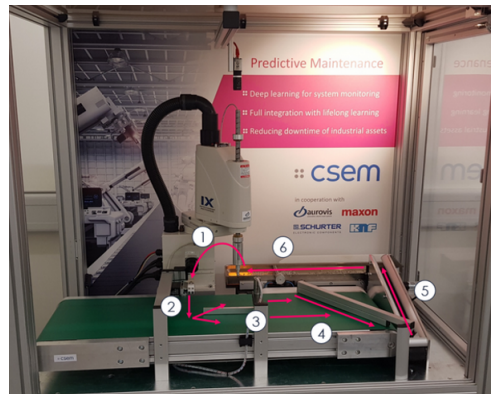


Fig. 3.19.: System of the experimental rig

3.5.2 Use Case Presentation

Numerous associations, universities and industries are putting their efforts and resources into advancing PdM. Among them, the European Conference of the Prognostics and Health Management Society (PHME) ¹ presented in 2021 a diagnostic problem for its annual data analytics challenge ². The experimental bed, courtesy of the Swiss Centre for Electronics and Microtechnology (CSEM) ³, generates data similar to a real-world industrial manufacturing line. The line consists of a 4-axis SCARA robot with a vacuum gripper that picks up fuses from a feeder to a fuse-test-bench. On the test bench, fuse current conductivity is measured, and, if the conductivity is appropriate, heating is applied to the fuse while a thermal camera measures during the heating process. Once the fuse is tested, it is sent back to the feeder with two conveyor belts. Figure 3.19 shows the experimental rig with the flow that the fuses follow during the tests.

¹<https://phm-europe.org/data-challenge>

²<https://github.com/PHME-Datachallenge/Data-Challenge-2021>

³<https://www.csem.ch/>

Participants were tasked with developing data-driven algorithms to accurately predict the remaining useful life (RUL) of the end-effector, with evaluation based on prediction accuracy and computational efficiency. The testbed featured various sensors capturing 50 signals measuring physical properties such as pressure, vacuum, and humidity, with statistical descriptors computed every 10 seconds instead of raw measurements. These descriptors included counts (Cnt), frequency (Freq), maximum (Max), minimum (Min), standard deviation (Std), Trend, and Value, and were not computed for every signal. The experiments lasted between 1 and 3 hours and included disturbances simulating eight different system failure conditions (labelled 2, 3, 4, 5, 7, 9, 11, and 12), resulting in a total of nine classes including a healthy class (labelled 0) with no disturbances introduced. The ultimate objective was to correctly classify each data set corresponding to the different experiments in the shortest possible time.

3.5.3 Proposal Solution

Using the `dqts` package, the quality of the data sets was evaluated and data losses were detected. It was observed that in some variables when the Cnt indicator was 0, the rest of the indicators had no associated value. This implied a large number of missing values in some indicators and it was understood that these indicators could be eliminated from the set. Thus, indicators with more than 80% missing data were eliminated. For the rest, the `handleDQ` function was used for the *Completeness* metric and missing data were imputed with the LOCF method, and in the case of a loss at the beginning of the series, the NOCB method is used. In order to do so, it is necessary to have at least the initial amount of data up to the first observation without missing values. Thus, the minimum initial amount of values

that the algorithm needs to be able to execute the NOCB imputation is saved as T_{NA} .

During the modelling stage, there are 99 data sets comprising 357 to 1081 observations collected over time to carry out the training task, where 70 were healthy experiments. More than two hundred features that are extracted and have passed DQ checks are utilised for the multivariate time series multi-class classification problem. The general approach is based on the idea of dividing the problem into different subproblems and classifying each observation one by one is applied to classify the entire data set, and then the resulting classes are combined to label the data set.

First, an unsupervised classification algorithm called CLARA (Clustering Large Applications) (111) is used to discern two different operating contexts in healthy tests. The CLARA result confirmed that the values taken by one of the variables (Smart Motor Position Error) were sufficient to separate the two operating conditions in which the experimental rig had been operating as can be seen in Figure 3.20. Thus, within class 0 there were data with clearly different behaviours in some of the variables but which did not induce any faults. That is the initial partition: experiments conducted with the SP1 and SP2 parameter configurations. The time required to determine the operation under which the test is performed, i.e., the system configuration parameters, is denoted by T_O . As can be seen, in most cases, this value is expected to be 1, since the separation rule is 100% reliable at least in the training set. This means that with a single value of the separation variable, the algorithm is able to determine whether the experiment is executed with an SP1 or SP2 configuration.

Due to their simplicity and capability to handle multicollinearity, DTs are chosen for the classification task. The features used as inputs for the model are those provided in the initial sets to which quality has been improved through imputations. In addition, to select relevant features, avoiding those that lack useful information,

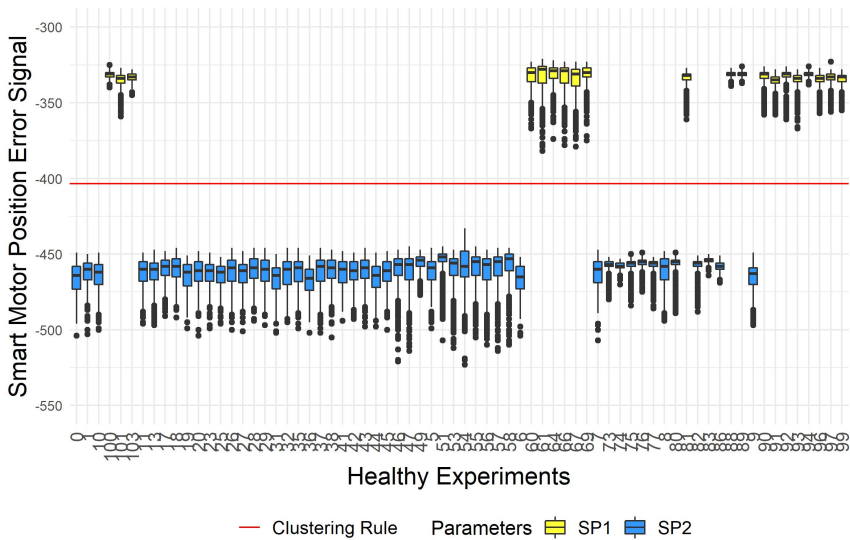


Fig. 3.20.: Box plots displaying the Smart Motor Position Error for 70 healthy test with a red line indicating the decision rule for the operation configuration.

a process based on principal components is carried out and those that do not contribute to DTs are eliminated. In addition, as it was challenging to identify some classes from the raw data, feature engineering is utilised to create more meaningful features that can help to disambiguate them for classes 5 and 7.

To address the issue of imbalanced data, a combination of SMOTE and Tomek Links was used to create a balanced data set. SMOTE was used to oversample the minority classes, while Tomek Links were used to removing the samples that were close to the decision boundary. This ensured that the algorithm would not penalize the failure classes for being underrepresented in the data set.

In order to ensure the robustness of the trees, the tree depth is limited to 5 to avoid overfitting and to use only the most significant signals. Additionally, the minimum number of observations per leaf is set at 450, ensuring that each leaf contained at least one complete

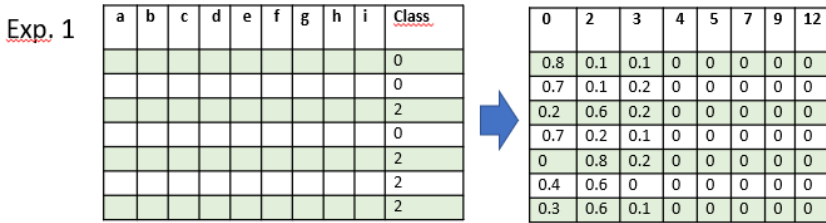


Fig. 3.21.: Example of label assignment using the most probable class resulting from the classification algorithm.

experiment, as the shortest experiment consisted of 360 observations. These values are chosen to achieve a generalised solution, although other values are tested and yielded similar results. The Gini impurity function is used as the measure of the quality of a split. Leave One Group Out cross-validation is used instead of random training/testing partitions to avoid experimental noise and ensure a more reliable estimation of the error. Specifically, one experiment of each class was left out of the training phase, and all of them are used for validation at each iteration, preventing data leakage from the same experiment to the testing set.

The required solution consists of a single class for the entire data set, but the algorithm classifies all the records collected during each experiment observation by observation. However, the output provided by a DT can be the probability of belonging to each of the classes. When the algorithm selects a class for a set instance, it chooses the label that is most probable. This process is illustrated in Figure 3.21, which presents a straightforward example where nine features are used to assign a distinct class for each observation. On the right side of the figure, the probability of belonging to each of the classes is displayed, along with the final class assignment as the most likely of the eight contenders.

To determine the overall classification label of an experiment, the initial assumption is that each experiment has an equal probability

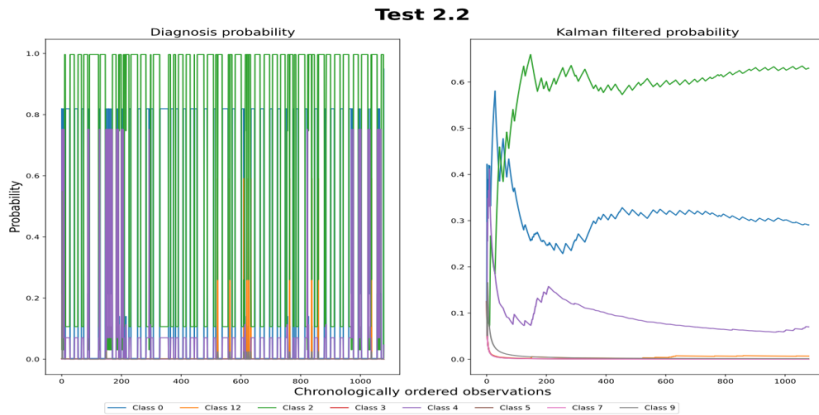


Fig. 3.22.: Execution of the Final Classification Algorithm

of belonging to any of the available classes. As such, at the start of the test run, each experiment is assigned a probability of $1/8$ for each of the eight possible labels. Therefore, using the probabilities provided by the classification algorithm, a propagation process is performed in order to decide the final class label to be assigned to the experiment.

Starting from an equal probability of each class, the Kalman filter with a gain of $K = 0.75^2$ is initially applied taking as the matrix of observations the probabilities of the output of the DT. Over time, one of the classes prevails over the rest, and this is the label that the algorithm decides on for the entire test. Figure 3.22 shows an example of the solution provided by the DT on the left and on the right, the propagation of that classification using this technique. Class 2 can be seen to prevail over the rest at an early point in the experiment. That is, in a set of 1000 observations, with one-fifth of the data, the type of failure can be determined with a probability of about 0.6.

Using this approach of concatenating a DT with class propagation using the Kalman filter provides a sufficiently accurate classification to address this problem. In the case of classes 0, 5 and 7 in SP2

an additional DT is necessary to achieve maximum accuracy. In addition, this type of approach allows for early detection of failures, another of the objectives proposed in the challenge as one of the most sought-after requirements in today's smart industry. The minimum number of observations required to determine the predominant class is extracted from each run and used as an indicator of the time required to make the appropriate labelling decision. The number of observations required to establish the predominant class is labelled as $T_{D'}$, and in case it belongs to classes 0, 5, or 7 of SP2, $T_{D''}$ is also added.

The final flow for the classification of the tests according to their failure can be seen in Figure 3.23. In summary, the resulting algorithm was a concatenation of the operation separation rule and several DTs. Each of these estimates would go through a Kalman filter-style propagation system to reduce the decision time, i.e. classify using as few observations as possible. The algorithm was able to accurately detect the status of all tests in both the training set and the final validation set of the challenge, except for the type 4 failure.

Due to the imbalance of the classes in the training of the models (70 healthy tests and 3 or 4 tests from each fault), the recall is used to evaluate the effectiveness of the classifications. However, accuracy is the metric used to evaluate the goodness of the algorithm as only the proportion of tests that are correctly labelled versus those that are not is considered. Figure 3.24 shows the two confusion matrices from the application of the final classification algorithm. On the left, is the result obtained on the training data sets, which can be controlled during the development tasks. On the right, are the final results of the validation data sets, which are scored in the competition. In both cases, the classification is perfect, except for case 4 which that are forced to be labelled as healthy. Thus, the accuracy in training is 0.9697 and in validation is 0.8824.

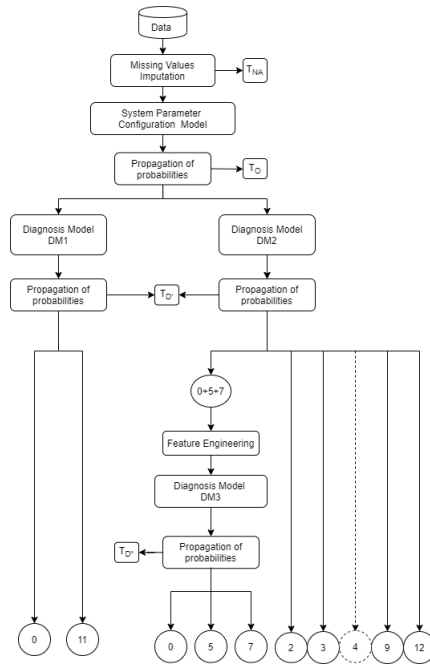


Fig. 3.23.: Final classification algorithm to determine the type of fault present in the shortest possible time.

		Predicted Class									
		Label	0	2	3	4	5	7	9	11	12
Real Class	0	70	0	0	0	0	0	0	0	0	0
	2	0	4	0	0	0	0	0	0	0	0
	3	0	0	4	0	0	0	0	0	0	0
	4	3	0	0	0	0	0	0	0	0	0
	5	0	0	0	0	4	0	0	0	0	0
	7	0	0	0	0	0	4	0	0	0	0
	9	0	0	0	0	0	0	4	0	0	0
	11	0	0	0	0	0	0	0	3	0	0
	12	0	0	0	0	0	0	0	0	0	3

		Predicted Class									
		Label	0	2	3	4	5	7	9	11	12
Real Class	0	3	0	0	0	0	0	0	0	0	0
	2	0	2	0	0	0	0	0	0	0	0
	3	0	0	2	0	0	0	0	0	0	0
	4	2	0	0	0	0	0	0	0	0	0
	5	0	0	0	0	2	0	0	0	0	0
	7	0	0	0	0	0	2	0	0	0	0
	9	0	0	0	0	0	0	2	0	0	0
	11	0	0	0	0	0	0	0	1	0	0
	12	0	0	0	0	0	0	0	0	0	1

(a) Training results

(b) Testing results

Fig. 3.24.: Confusion matrices of the final classification algorithm in training and testing tasks.

The minimum time required to traverse the splitting algorithm in each case without encountering execution errors T_c is the maximum of the observations required to complete each of the concatenated tasks. That is, $T_c = \max\{T_{NA}, T_O, T_{D'} \text{ and } T_{D''}\}$.

3.5.4 Conclusions and Outcomes

The provided solution consists of the concatenation of different DTs. This algorithm stands out for its added advantage of being understandable. In addition, as a novelty in the case of time series classification, the propagation of class probabilities was tested using a Kalman filter algorithm, which achieved the early classification of the experiment. Both the simplicity of explanation, as well as the speed and accuracy in classification, make this proposal a highly reliable data-driven system.

The prediction results provided by the algorithm achieved the highest accuracy with the lowest number of observations compared to other participating teams. For that reason, we were awarded as the winners of the PHME Challenge 2021 and we presented our work at the congress that was held online due to the COVID-19 pandemic in July. In addition, the article *Divide, propagate and conquer: Splitting a complex diagnosis problem for early detection of faults in a manufacturing production line* (25) is attached in Section 9. This work was published in the conference proceedings, where the development of our solution can be found as well as an explanation of the challenge and the evaluation system applied.

Despite that, this proposal does have some limitations. Firstly, concerning the strategy employed to determine the shortest time needed for classification, it should be noted that while it aligns well with the context of the challenge, it is not applicable in a real-world scenario where there are no multiple runs for each test. Secondly, the

interpretation of the proposed solution, in this case, should take into consideration that the problem was designed in a controlled environment. Therefore, the available information was limited to what was provided by the organizers, and the development time was constrained. For those reasons, it has not been possible to detect all faults, rendering this solution incomplete.

Closing Remarks

” *Reserve your right to think, for even to think wrongly is better than not to think at all*

— **Hypatia**
(Mathematician, Inventor, and
Philosopher)

Hypatia was a Greek mathematician, astronomer, and philosopher who lived in Alexandria, Egypt, during the fourth century AD. She received an education from her father, who imparted all his knowledge to her, leading to her developing a deeper understanding of thoughts and ideas beyond his own. Hypatia wrote texts on algebra and geometry, conducted classes, and invented scientific instruments; becoming one of the pioneering women in science.

As a philosopher, Hypatia emphasised the importance of critical and independent thinking. She encouraged her students to question assumptions and to pursue the truth through rational inquiry. She maintained that it was better to think independently and run the risk of being wrong than to accept ideas without criticism or to abstain from thinking altogether.

Hypatia was renowned for her lectures on mathematics, astronomy, and philosophy, and enjoyed a high level of respect within the scientific community for her intelligence and knowledge. However, certain influential individuals rejected mathematicians, whom they viewed as deceitful or magical.

Tragically, an angry mob killed Hypatia, feeling threatened by her advanced scientific ideas. Nevertheless, Hypatia always defended that individuals should be free to explore ideas and make mistakes, as long as they are committed to the pursuit of truth and knowledge.

4.1 Conclusions

Improving reliability in AI systems for the current industry requires different phases and interventions that depend on professionals from various fields. Specifically, the tasks corresponding to data analysts have a significant influence on data-driven solutions. Therefore, their involvement is considered essential in the phases of data preparation, exploration and modelling, and evaluation.

In particular, time series in industrial environments are data sets received throughout a process that can be monitored to obtain ordered information that holds dependencies with past and future information. Therefore, there is an intrinsic characteristic to this type of ordered data that requires specific treatment and a particular approach to improve the quality of time series-based systems.

The proposal to increase reliability in three phases allows us to arrange the interventions temporally and understand that certain actions should be considered in each phase to improve the quality of data, model, and errors separately.

- To improve the quality of the data, the existing gap in the literature has been addressed by unifying the extensive definitions of DQ, allowing them to be applied in different fields. That is, a way to measure the quality of time series data independent of the use case. The R package `dqts` provides a quick and easy way for users from different specializations to execute, analyze, and improve the quality of the data sets they work with.

On the one hand, ensuring high DQ in an early stage of analysis increases the quality of results from AI-based models. On the other hand, saving time and resources in data preprocessing adds efficiency to the tasks of data analysts who often invest significant efforts in cleaning and preprocessing information.

Furthermore, the R package can be embedded in industrial systems and can be used for different purposes. On the one hand, the outputs of the `DQ`, `deepDQ` and `plotDQ` functions allow the automation of periodic quality reports that provide insight into the status of time series data received over time. In addition, it is possible to programme alerts that enable swift action to address any issues that may arise. On the other hand, the `handleDQ` function saves analysis time and facilitates decision-making to increase DQ by modifying the data. This part of the package can be used within a loop to modify or duplicate the database allowing it to store equivalent data but with optimal quality.

Currently, `dqts` allows the reading of data with two types of structure: time series or R data frames. However, thanks to the modular structure with which the package has been designed, it is envisaged to extend its use to other data structures. This would be possible by adding the appropriate transformations that standardise the reading of the data in a step prior to the execution of the metrics. Moreover, thanks to the connectors to different databases available in R, this access to the data could be direct without the need to download the time series to the computers beforehand.

The package has been tested in different industrial scenarios, both simulated and real, and is currently a powerful tool for efficiently knowing the quality of the time series used in different industrial projects. In particular, `dqts` has been used in

the application cases of this research work, as can be seen in Section 3.

Beyond the scope of this work, `dqts` has been integrated into Shiny applications for interactive use by users without specific knowledge of data analytics. In this manner, colleagues in mechanical engineering are capable of comprehending the DQ of their building cooling systems and making the necessary modifications to attain the required DQ in their equipment.

Therefore, it is concluded that the interventions carried out in this work to improve the quality of time series structured data received in industrial environments contribute to increasing the reliability of AI systems. Furthermore, in the long term, the benefits of organisations designing such solutions as well as users implementing and exploiting them are increased.

The advances made in DQ throughout this research work were published in August 2022 in the IEEE Access journal. That article can be read in Section 7.

- There is a wide range of methods for modelling time series, and a proper selection of the modelling approach is essential to achieve accurate results that provide reliability and allow for generalisation.

The efficiency of time series regression methods as well as their limitations in long-term forecasting problems have been emphasised, particularly when the prediction horizon exceeds the periodicity of the data or the number of regressive variables to be used.

In addition, feature engineering and selection tasks have been highlighted as crucial when using classical ML prediction models. Furthermore, they have been classified into statistical, temporal, lagged, and topological features. Each of them has

been used in different industrial problems, demonstrating that there is no intrinsic superiority in any of them, but rather their selection depends on the problem to be addressed. Particularly, cyclic transformations of temporal variables such as hour, day, or month are considered an essential step in preparing the data to be used as inputs for the models. Applying these transformations helps to numerically bring explanatory variables closer to their real behaviour, thus avoiding distance-based algorithms from drawing erroneous conclusions and ignoring nearby values.

Therefore, in addition to the classic methods of time series regression and ML techniques, the use of KNPTS method has been proposed. This algorithm allows the estimation of future data based on the patterns present in the historical time series. It is a useful strategy in univariate time series with significant seasonality, where a periodic repetition of the data behaviour is expected.

The creation of the R package `knpts` allows for the execution of this algorithm in a simple and robust way, including different strategies and validations to test and choose the most accurate approach.

For proper algorithm validation, it is essential to use time series data in an appropriate manner. In other words, in cross-validation of the algorithm, past data cannot be used, so techniques specific to time series are essential: using sliding windows or using expanding windows. This approach avoids overfitting due to the use of information available in training that cannot be used later because it belongs to future data.

Finally, in the modelling phase, the effect of the prediction strategy on the efficiency of the models has been demonstrated. Especially in cases where delays are used as inputs to the model

and in those cases where the forecasting strategy is recursive, error accumulation is a problem that must be approached with great caution. The use of a training strategy different from the deployment strategy can lead to incorrect conclusions due to the selection of an inappropriate model. This error would lead to a lack of reliability in the modelling phase by not obtaining the expected results in the proposed solution.

- Applying ML methods to industrial problems without proper validation is doomed to fail. In fact, in numerous studies, it has been observed that deploying seemingly good solutions may not accurately represent reality. This can be avoided by choosing an appropriate and rigorous validation method and being aware of the representation that these numerical values have in the problem being addressed.

Industry 4.0 is moving towards the automation of systems and providing equipment with intelligence for independent decision-making. However, for these systems to be able to efficiently replace human decisions, all phases of reasoning must be programmed. Therefore, it is not enough to calculate an accuracy metric and choose the one that provides the lowest error to decide the best solution. As discussed in Section 2.4, it depends on the problem we are dealing with. In most cases, the analyst's review is essential in order to understand the decisions made.

In numerical forecasting problems, the use of similarity measures in time series as performance metrics has been proposed. Unlike static metrics such as MSE, these elastic strategies allow for the comparison of non-aligned values over time. This increases reliability by selecting algorithms that provide good predictions based on the shape of the predicted time series, rather than comparing the real and predicted values using Euclidean distances. As an example of these new prediction

error calculation strategies, DTW and EDR have been used, demonstrating their efficiency for this purpose, especially in high-frequency data or in series where the displacement of the prediction should not be penalised, such as the daily electrical consumption.

In classification problems, the impact of using one metric or another when making decisions has been explained, especially in problems with imbalanced classes. It is important to consider metrics such as Recall, Precision, or F1 score, which take into account the imbalance by weighting in favour of one of the classes. This type of decision increases the reliability of classification problems, especially in cases such as predicting machine failures, where detecting the failure is prioritised and customers focus on predicting breakdowns, for example.

In general, the reliability of AI systems has been improved in four current and complex industrial cases through the application of these three phases. Despite the great progress made in the research of complex algorithms, it is essential to keep in mind that the choice of prediction method is not the core of the analysis. In the life cycle of a time series, there are different phases where there are no established rules, and the efficiency of solutions depends on the context and purpose of the problem, DQ and structure, data frequency and periodicity, system deployment and exploitation, etc.

Current research on AI-based systems focuses on the development of automated solutions that can interact with the environment and learn and adjust as new information becomes available over time. However, the tasks of analysts remain essential and solutions still depend on the use case, so automation carries a risk of loss of reliability.

4.2 Future Work

The increase in reliability in AI systems is constantly being researched, as it is considered that high-quality solutions have not yet been achieved in many areas.

Although the gap in standardisation for measuring the quality of time series data using the same criterion has been covered, more research in this direction is necessary. Firstly, the modular structure of the R package `dqts` allows for the incorporation of new quality metrics that can be designed after the detection of new needs. Additionally, the incorporation of new metrics also allows for specialisation. The majority of the research into time series DQ has a primary focus on financial and economic time series data. It is imperative to undertake further research into DQ issues in other domains, such as healthcare, energy, and manufacturing. Additionally, there is a requirement for further research into the impact of DQ issues on downstream tasks, including forecasting, anomaly detection, and decision-making. Furthermore, it is necessary to explore the human factors that impact DQ, such as data entry errors, user interface design, and training for data entry personnel.

In the modelling phase of the time series, problems in predicting random peaks have been detected. Most time series modelling research is focused on forecasting and pattern recognition, but there is a lack of research on causal relationships between variables. Greater comprehension of these causal behaviours would enhance the dependability of time series-based models. In addition, further research is required to investigate the integration of domain knowledge, such as expert opinions or physical laws, into time series models to improve their precision and interpretability. Along similar lines, there is a requirement to develop time series models that are more accessible and comprehensible to a broader audience.

Remaining the measures used to quantify the error of time series forecasting models, more research is needed to cover uncertainty issues associated with the estimate. For example, the use of elastic time series similarity measures allows the alignment of the prediction with the actual series to minimise the effect of predicting peaks with delays. However, there is a need for error measures that are more robust to outliers that can significantly affect the decision by making results less informative.

Bibliography

- [1] P. H. Winston, *Artificial intelligence*. Addison-Wesley Longman Publishing Co., Inc., 1984.
- [2] A. K. Agrawal, J. S. Gans, and A. Goldfarb, “What to expect from artificial intelligence,” *MIT Sloan Management Review*, vol. 58, no. 3, p. 23, 2017.
- [3] J. A. Nichols, H. W. Herbert Chan, and M. A. Baker, “Machine learning: applications of artificial intelligence to imaging and diagnosis,” *Biophysical reviews*, vol. 11, pp. 111–118, 2019.
- [4] Z. Ullah, F. Al-Turjman, L. Mostarda, and R. Gagliardi, “Applications of artificial intelligence and machine learning in smart cities,” *Computer Communications*, vol. 154, pp. 313–323, 2020.
- [5] M. Ghobakhloo, “Industry 4.0, digitization, and opportunities for sustainability,” *Journal of cleaner production*, vol. 252, p. 119869, 2020.
- [6] L. Atzori, A. Iera, and G. Morabito, “The internet of things: A survey,” *Computer networks*, vol. 54, no. 15, pp. 2787–2805, 2010.
- [7] M. U. Farooq, M. Waseem, S. Mazhar, A. Khairi, and T. Kamal, “A review on internet of things (iot),” *International journal of computer applications*, vol. 113, no. 1, pp. 1–7, 2015.

- [8] S. Madakam, V. Lake, V. Lake, V. Lake, *et al.*, “Internet of things (iot): A literature review,” *Journal of Computer and Communications*, vol. 3, no. 05, p. 164, 2015.
- [9] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, “The industrial internet of things (iiot): An analysis framework,” *Computers in industry*, vol. 101, pp. 1–12, 2018.
- [10] H. Jaidka, N. Sharma, and R. Singh, “Evolution of iot to iiot: Applications & challenges,” in *Proceedings of the international conference on innovative computing & communications (ICICC)*, 2020.
- [11] A. Kusiak, “Smart manufacturing must embrace big data,” *Nature*, vol. 544, no. 7648, pp. 23–25, 2017.
- [12] A. Kusiak, “Smart manufacturing,” *International Journal of Production Research*, vol. 56, no. 1-2, pp. 508–517, 2018.
- [13] M. A. Waller and S. E. Fawcett, “Data science, predictive analytics, and big data: a revolution that will transform supply chain design and management,” *Journal of Business Logistics*, vol. 34, no. 2, pp. 77–84, 2013.
- [14] L. Monostori, “Ai and machine learning techniques for managing complexity, changes and uncertainties in manufacturing,” *Engineering applications of artificial intelligence*, vol. 16, no. 4, pp. 277–291, 2003.
- [15] R. H. Shumway, D. S. Stoffer, and D. S. Stoffer, *Time series analysis and its applications*, vol. 3. Springer, 2000.
- [16] S. O. Abioye, L. O. Oyedele, L. Akanbi, A. Ajayi, J. M. D. Delgado, M. Bilal, O. O. Akinade, and A. Ahmed, “Artificial intelligence in the construction industry: A review of present status, opportunities and future challenges,” *Journal of Building Engineering*, vol. 44, p. 103299, 2021.

- [17] T. C. Redman, "The impact of poor data quality on the typical enterprise," *Communications of the ACM*, vol. 41, no. 2, pp. 79–82, 1998.
- [18] B. Butcher and B. J. Smith, "Feature engineering and selection: A practical approach for predictive models," *The American Statistician*, vol. 74, no. 3, pp. 308–309, 2020.
- [19] C. Chatfield, *The analysis of time series: an introduction*. Chapman and hall/CRC, 2003.
- [20] T.-c. Fu, "A review on time series data mining," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 164–181, 2011.
- [21] G.-O. Meritxell, B. Sierra, and S. Ferreiro, "On the evaluation, management and improvement of data quality in streaming time series," *IEEE Access*, vol. 10, pp. 81458–81475, 2022.
- [22] I. Esnaola-Gonzalez, M. Gómez-Omella, S. Ferreiro, I. Fernandez, I. Lázaro, and E. García, "An iot platform towards the enhancement of poultry production chains," *Sensors*, vol. 20, no. 6, 2020.
- [23] M. Gomez-Omella, I. Esnaola-Gonzalez, and S. Ferreiro, "Short-term forecasting methodology for energy demand in residential buildings and the impact of the covid-19 pandemic on forecasts," pp. 227–240, Springer, 2020.
- [24] M. Gómez-Omella, I. Esnaola-Gonzalez, S. Ferreiro, and B. Sierra, "k-nearest patterns for electrical demand forecasting in residential and small commercial buildings," *Energy and Buildings*, vol. 253, p. 111396, 2021.
- [25] K. López de Calle Etxabe, M. Gómez-Omella, and E. Garate Perez, "Divide, propagate and conquer: Splitting a complex diagnosis problem for early detection of faults

in a manufacturing production line,” *PHM Society European Conference*, vol. 6, no. 1, 2021.

- [26] G. Kitagawa, *Introduction to time series modeling*. CRC press, 2010.
- [27] B. Heinrich, D. Hristova, M. Klier, A. Schiller, and M. Szubartowicz, “Requirements for data quality metrics,” *Journal of Data and Information Quality (JDIQ)*, vol. 9, no. 2, pp. 1–32, 2018.
- [28] Y. Wand and R. Y. Wang, “Anchoring data quality dimensions in ontological foundations,” *Communications of the ACM*, vol. 39, no. 11, pp. 86–95, 1996.
- [29] C. Batini and M. Scannapieco, *Data Quality: Concepts, Methodologies and Techniques*. Berlin: Springer, 2006.
- [30] N. Laranjeiro, S. N. Soydemir, and J. Bernardino, “A survey on data quality: classifying poor data,” in *2015 IEEE 21st Pacific rim international symposium on dependable computing (PRDC)*, pp. 179–188, IEEE, 2015.
- [31] C. Batini, C. Cappiello, C. Francalanci, and A. Maurino, “Methodologies for data quality assessment and improvement,” *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–52, 2009.
- [32] A. Maydanchik, *Data Quality Assessment*. Data quality for practitioners series, Technics Publications, 2007.
- [33] L. Ehrlinger and W. Wöß, “A survey of data quality measurement and monitoring tools,” *Frontiers in big data*, p. 28, 2022.
- [34] P. Newbold, “ARIMA model building and the time series analysis approach to forecasting,” *Journal of Forecasting*, vol. 2, no. 1, pp. 23–35, 1983.

- [35] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
- [36] A. Pankratz, *Forecasting with dynamic regression models*. John Wiley & Sons, 2012.
- [37] R. S. Tsay, "Identifying multivariate time series models," *Journal of Time Series Analysis*, vol. 10, no. 4, pp. 357–372, 1989.
- [38] J. J. Commandeur and S. J. Koopman, *An introduction to state space time series analysis*. Oxford university press, 2007.
- [39] A. C. Harvey, *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.
- [40] D. Maulud and A. M. Abdulazeez, "A review on linear regression comprehensive in machine learning," *Journal of Applied Science and Technology Trends*, vol. 1, no. 4, pp. 140–147, 2020.
- [41] N. Cristianini, J. Shawe-Taylor, *et al.*, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.
- [42] K. Taunk, S. De, S. Verma, and A. Swetapadma, "A brief review of nearest neighbor algorithm for learning and classification," in *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pp. 1255–1260, IEEE, 2019.
- [43] W.-Y. Loh, "Classification and regression trees," *Wiley interdisciplinary reviews: data mining and knowledge discovery*, vol. 1, no. 1, pp. 14–23, 2011.
- [44] M. Rakhra, P. Soniya, D. Tanwar, P. Singh, D. Bordoloi, P. Agarwal, S. Takkar, K. Jairath, and N. Verma, "Crop price prediction using random forest and decision tree regression:-a review," *Materials Today: Proceedings*, 2021.

- [45] A. Natekin and A. Knoll, “Gradient boosting machines, a tutorial,” *Frontiers in neurorobotics*, vol. 7, p. 21, 2013.
- [46] G. Bontempi, S. Ben Taieb, and Y.-A. Le Borgne, “Machine learning strategies for time series forecasting,” *Business Information Processing*, vol. 138, pp. 62–77, 2013.
- [47] J. Faouzi, “Time series classification: A review of algorithms and implementations,” *Machine Learning (Emerging Trends and Applications)*, 2022.
- [48] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.
- [49] I. Tomek, “Two modifications of cnn,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 11, pp. 769–772, 1976.
- [50] D. L. Wilson, “Asymptotic properties of nearest neighbor rules using edited data,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-2, no. 3, pp. 408–421, 1972.
- [51] G. E. Batista, R. C. Prati, and M. C. Monard, “A study of the behavior of several methods for balancing machine learning training data,” *SIGKDD Explor. Newsl.*, vol. 6, no. 1, p. 20–29, 2004.
- [52] B. Ghojogh and M. Crowley, “The theory behind overfitting, cross validation, regularization, bagging, and boosting: tutorial,” *arXiv preprint arXiv:1905.12787*, 2019.
- [53] D. R. Roberts, V. Bahn, S. Ciuti, M. S. Boyce, J. Elith, G. Guillera-Arroita, S. Hauenstein, J. J. Lahoz-Monfort, B. Schröder, W. Thuiller, *et al.*, “Cross-validation strategies

for data with temporal, spatial, hierarchical, or phylogenetic structure,” *Ecography*, vol. 40, no. 8, pp. 913–929, 2017.

- [54] R. J. Hyndman and G. Athanasopoulos, *Forecasting: principles and practice*. OTexts, 2018.
- [55] I. Guyon and A. Elisseeff, “An introduction to feature extraction,” *Feature extraction: foundations and applications*, pp. 1–25, 2006.
- [56] R. A. Groeneveld and G. Meeden, “Measuring skewness and kurtosis,” *Journal of the Royal Statistical Society: Series D (The Statistician)*, vol. 33, no. 4, pp. 391–399, 1984.
- [57] M. Rostaghi and H. Azami, “Dispersion entropy: A measure for time-series analysis,” *IEEE Signal Processing Letters*, vol. 23, no. 5, pp. 610–614, 2016.
- [58] R. T. Olszewski, *Generalized feature extraction for structural pattern recognition in time-series data*. Carnegie Mellon University, 2001.
- [59] A. Sherstinsky, “Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network,” *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.
- [60] Y. Umeda, J. Kaneko, and H. Kikuchi, “Topological data analysis and its application to time-series data analysis,” *Fujitsu Scientific & Technical Journal*, vol. 55, no. 2, pp. 65–71, 2019.
- [61] N. Ravishanker and R. Chen, “An introduction to persistent homology for time series,” *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 13, no. 3, p. e1548, 2021.
- [62] S. Bochner, “Curvature and Betti Numbers,” *The Annals of Mathematics*, vol. 49, no. 2, p. 379, 1948.

- [63] G. Singh, F. Mémoli, and G. Carlsson, “Topological Methods for the Analysis of High Dimensional Data Sets and 3D Object Recognition,” *Eurographics Symposium on Point-Based Graphics*, vol. 2, pp. 91–100, 2007.
- [64] M. C. Yesilli, *Topological Data Analysis and Machine Learning Framework for Studying Time Series and Image Data*. Michigan State University, 2022.
- [65] F. Takens, “Detecting strange attractors in turbulence,” *Journal of Animal Ecology*, vol. 84, no. 6, pp. 388–400, 2012.
- [66] G. Damiand, E. Paluzo-Hidalgo, R. Slechta, and R. Gonzalez-Diaz, “Approximating lower-star persistence via 2D combinatorial map simplification,” *Pattern Recognition Letters*, vol. 131, pp. 314–321, 2020.
- [67] R. Mezher, J. Arayro, N. Hascoet, and F. Chinesta, “Study of concentrated short fiber suspensions in flows, using topological data analysis,” *Entropy*, vol. 23, no. 9, p. 1229, 2021.
- [68] S. B. Taieb, G. Bontempi, A. F. Atiya, and A. Sorjamaa, “A review and comparison of strategies for multi-step ahead time series forecasting based on the nn5 forecasting competition,” *Expert systems with applications*, vol. 39, no. 8, pp. 7067–7083, 2012.
- [69] D. M. Kline, “Methods for Multi-Step Time Series Forecasting Neural Networks,” *Neural Networks in Business Forecasting*, pp. 226–250, 2004.
- [70] M. Z. Naser and A. H. Alavi, “Error Metrics and Performance Fitness Indicators for Artificial Intelligence and Machine Learning in Engineering and Sciences,” *Architecture, Structures and Construction*, pp. 1–19, 2021.

- [71] A. Abanda, U. Mori, and J. A. Lozano, “A review on distance based time series classification,” *Data Mining and Knowledge Discovery*, vol. 33, no. 2, pp. 378–412, 2019.
- [72] S. Corbett-Davies and S. Goel, “The measure and mismeasure of fairness: A critical review of fair machine learning,” *arXiv preprint arXiv:1808.00023*, 2018.
- [73] M. D. Schluchter, “Mean square error,” *Encyclopedia of Biostatistics*, vol. 5, 2005.
- [74] A. Botchkarev, “Performance metrics (error measures) in machine learning regression, forecasting and prognostics: Properties and typology,” *arXiv preprint arXiv:1809.03006*, 2018.
- [75] D. Gunopulos and G. Das, “Time series similarity measures and time series indexing,” in *SIGMOD Conference*, p. 624, 2001.
- [76] A. Shifaz, C. Pelletier, F. Petitjean, and G. I. Webb, “Elastic similarity measures for multivariate time series classification,” *arXiv preprint arXiv:2102.10231*, 2021.
- [77] L. Chen, M. T. Özsu, and V. Oria, “Robust and fast similarity search for moving object trajectories,” in *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pp. 491–502, Association for Computing Machinery, 2005.
- [78] C. Goutte and E. Gaussier, “A probabilistic interpretation of precision, recall and f-score, with implication for evaluation,” in *Advances in Information Retrieval: 27th European Conference on IR Research, ECIR 2005, Santiago de Compostela, Spain, March 21-23, 2005. Proceedings 27*, pp. 345–359, Springer, 2005.

- [79] E. V. Burnaev, "Time-Series Classification for Industrial Applications: Road Surface Damage Detection Use Case," *Journal of Communications Technology and Electronics*, vol. 65, pp. 1491–1498, 2020.
- [80] L. Ryan, "A conversation with nan laird," *Statistical Science*, vol. 30, no. 4, pp. 582–596, 2015.
- [81] M. Bertolini, D. Mezzogori, M. Neroni, and F. Zammori, "Machine learning for industrial applications: A comprehensive literature review," *Expert Systems with Applications*, vol. 175, p. 114820, 2021.
- [82] H. Xin and J. D. Harmon, "Livestock industry facilities and environment: Heat stress indices for livestock," 1998.
- [83] M. H. Albadi and E. F. El-Saadany, "Demand response in electricity markets: An overview," in *2007 IEEE power engineering society general meeting*, pp. 1–5, IEEE, 2007.
- [84] N. A. Mohammed, "Modelling of unsuppressed electrical demand forecasting in iraq for long term," *Energy*, vol. 162, pp. 354–363, 2018.
- [85] E. M. de Oliveira and F. L. C. Oliveira, "Forecasting mid-long term electric energy consumption through bagging arima and exponential smoothing methods," *Energy*, vol. 144, pp. 776–788, 2018.
- [86] P. F. Pai and W. C. Hong, "Forecasting regional electricity load based on recurrent support vector machines with genetic algorithms," *Electric Power Systems Research*, vol. 74, no. 3, pp. 417–425, 2005.
- [87] B. J. Chen, M. W. Chang, and C. J. Lin, "Load forecasting using support vector machines: A study on EUNITE Competition

- 2001,” *IEEE Transactions on Power Systems*, vol. 19, no. 4, pp. 1821–1830, 2004.
- [88] O. A. Carpinteiro, R. C. Leme, A. C. de Souza, C. A. Pinheiro, and E. M. Moreira, “Long-term load forecasting via a hierarchical neural model with time integrators,” *Electric Power Systems Research*, vol. 77, no. 3-4, pp. 371–378, 2007.
- [89] Y. Hu, J. Li, M. Hong, J. Ren, R. Lin, Y. Liu, M. Liu, and Y. Man, “Short term electric load forecasting model and its verification for process industrial enterprises based on hybrid GA-PSO-BPNN algorithm—A case study of papermaking process,” *Energy*, vol. 170, pp. 1215–1227, 2019.
- [90] B. Kermanshahi, “Recurrent neural network for forecasting next 10 years loads of nine Japanese utilities,” *Neurocomputing*, vol. 23, no. 1-3, pp. 125–133, 1998.
- [91] G. Mihalakakou, M. Santamouris, and A. Tsangrassoulis, “On the energy consumption in residential buildings,” *Energy and buildings*, vol. 34, no. 7, pp. 727–736, 2002.
- [92] N. J. Johannesen, M. Kolhe, and M. Goodwin, “Relative evaluation of regression tools for urban area electrical energy demand forecasting,” *Journal of Cleaner Production*, vol. 218, pp. 555–564, 2019.
- [93] W. Yu, B. Li, Y. Lei, and M. Liu, “Analysis of a residential building energy consumption demand model,” *Energies*, vol. 4, no. 3, pp. 475–487, 2011.
- [94] M. K. Kim, Y. S. Kim, and J. Srebric, “Predictions of electricity consumption in a campus building using occupant rates and weather elements with sensitivity analysis: Artificial neural network vs. linear regression,” *Sustainable Cities and Society*, vol. 62, p. 102385, 2020.

- [95] G. Escrivá-Escrivá, C. Álvarez-Bel, C. Roldán-Blay, and M. Alcázar-Ortega, “New artificial neural network prediction method for electrical consumption forecasting based on building end-uses,” *Energy and Buildings*, vol. 43, no. 11, pp. 3112–3119, 2011.
- [96] T. Pinto, I. Praça, Z. Vale, and J. Silva, “Ensemble learning for electricity consumption forecasting in office buildings,” *Neurocomputing*, vol. 423, pp. 747–755, 2021.
- [97] C. Leyens and E. Beyer, “Innovations in laser cladding and direct laser metal deposition,” in *Laser Surface Engineering: Processes and Applications*, pp. 181–192, 2015.
- [98] D. Ding, Z. Pan, D. Cuiuri, and H. Li, “Wire-feed additive manufacturing of metal components: technologies, developments and future interests,” *The International Journal of Advanced Manufacturing Technology*, vol. 81, pp. 465–481, 2015.
- [99] M. Akbari and R. Kovacevic, “An investigation on mechanical and microstructural properties of 316LSi parts fabricated by a robotized laser/wire direct metal deposition system,” *Additive Manufacturing*, vol. 23, pp. 487–497, 2018.
- [100] M. Motta, A. G. Demir, and B. Previtali, “High-speed imaging and process characterization of coaxial laser metal wire deposition,” *Additive Manufacturing*, vol. 22, pp. 497–507, 2018.
- [101] J. J. Lewandowski and M. Seifi, “Metal Additive Manufacturing: A Review of Mechanical Properties,” 2016.
- [102] A. Du Plessis, I. Yadroitsava, and I. Yadroitsev, “Effects of defects on mechanical properties in metal additive manufacturing: A review focusing on x-ray tomography insights,” *Materials & Design*, vol. 187, p. 108385, 2020.

- [103] C. Wang, X. Tan, S. B. Tor, and C. Lim, “Machine learning in additive manufacturing: State-of-the-art and perspectives,” *Additive Manufacturing*, vol. 36, p. 101538, 2020.
- [104] C. Tralie, N. Saul, and R. Bar-On, “Ripser.py: A lean persistent homology library for python,” *The Journal of Open Source Software*, vol. 3, p. 925, Sep 2018.
- [105] T. Zonta, C. A. Da Costa, R. da Rosa Righi, M. J. de Lima, E. S. da Trindade, and G. P. Li, “Predictive maintenance in the industry 4.0: A systematic literature review,” *Computers & Industrial Engineering*, vol. 150, p. 106889, 2020.
- [106] T. P. Carvalho, F. A. A. M. N. Soares, R. Vita, R. da P. Francisco, J. P. Basto, and S. G. S. Alcalá, “A systematic literature review of machine learning methods applied to predictive maintenance,” *Computers Industrial Engineering*, vol. 137, p. 106024, 2019.
- [107] R. Kizito, P. Scruggs, X. Li, R. Kress, M. Devinney, and T. Berg, “The application of random forest to predictive maintenance,” in *IIE Annual Conference. Proceedings*, pp. 354–359, Institute of Industrial and Systems Engineers (IISE), 2018.
- [108] Z. Wang, P. Gao, and X. Chu, “Remaining useful life prediction of wind turbine gearbox bearings with limited samples based on prior knowledge and pi-lstm,” *Sustainability*, vol. 14, no. 19, p. 12094, 2022.
- [109] O. Serradilla, E. Zugasti, J. Rodriguez, and U. Zurutuza, “Deep learning models for predictive maintenance: a survey, comparison, challenges and prospects,” *Applied Intelligence*, vol. 52, no. 10, pp. 10934–10964, 2022.
- [110] S. Cho, G. MAY, I. Tourkogiorgis, R. Perez, O. Lazaro, B. Maza, and D. Kiritsis, “A hybrid machine learning approach for pre-

dictive maintenance in smart factories of the future,” pp. 311–317, 08 2018.

- [111] L. Kaufman and P. J. Rousseeuw, “CLUSTERING LARGE DATA SETS,” in *Pattern Recognition in Practice*, pp. 425–437, North Holland, 1986.
- [112] M. Gómez-Omella, I. Esnaola-Gonzalez, and S. Ferreiro, “Short-term electric demand forecasting for the residential sector: Lessons learned from the respond h2020 project,” *Proceedings*, vol. 65, no. 1, 2020.
- [113] I. Esnaola-Gonzalez, M. Gomez-Omella, S. Ferreiro, F. J. Diez, and A. García, “Prenerget: A framework for the inclusion and adaptation strategies of machine learning models for energy demand forecasting in buildings,” in *Proceedings of the 2022 European Conference on Computing in Construction*, vol. 3, University of Turin, 2022.

List of Figures

1.1	Smart specialisation areas in Euskadi based on the interaction of business capabilities, scientific-technological capabilities and market opportunities.	7
1.2	Time series data life cycle and the three phases explored for improving the quality and reliability of data-driven developments.	8
2.1	Additive decomposition of monthly gas consumption time series for Australia.	30
2.2	Multiplicative decomposition of monthly gas consumption time series for Australia	30
2.3	Example of simulated values following a Gaussian distribution and the three boundaries of the CI with confidence levels of 80%, 95% and 99%	34
2.4	Example of integration of the dqts package in an industrial system for the evaluation and correction of manufacturing data.	37
2.5	One year ahead forecasting training ARIMA model with 0.95 in Completeness and 0.99 in Range	38
2.6	One day ahead forecasting using fixed data with the highest quality.	38
2.7	One-week-ahead forecasting in one execution using SARIMA model for energy demand.	42
2.8	Ten-weeks-ahead forecasting in one execution in weekly data using SARIMA model for energy demand.	43

2.9	Ten weeks ahead forecasting in seventy executions in weekly data using SARIMA model.	44
2.10	Two-dimensional example showcasing the effect of the kernel trick in the input space of the SVM model. . . .	49
2.11	Graphic representation of the operation of a DT with a maximum depth of 3.	50
2.12	Nearest-neighbour three-step-ahead forecasts. A window of length five is selected and one neighbour is used to estimate the next three values.	53
2.13	Two different approaches of time series classification problems.	54
2.14	5-fold cross-validation.	59
2.15	Time series cross-validation using sliding windows and expanding windows.	60
2.16	Process of creating persistence images of a univariate time series using TDA	67
2.17	The five different forecasting strategies and their relationships.	68
2.18	Outline of the <i>Recursive</i> strategy for forecasting the next H values of a time series using M lags and n features.	69
2.19	Outline of the <i>Direct</i> strategy for forecasting the next H values of a time series using M lags and n features.	70
2.20	Outline of the <i>MIMO</i> strategy for forecasting the next H values of a time series using M lags and n features.	71
2.21	Comparison of two different ways of mapping two time series	74
2.22	comparison of two time series forecasts with the same value of MSE but different DTW.	76
3.1	Temperature time series of sensor 2 after DQ correction with the <code>dqts</code> package of R	90
3.2	Temperature and comfort bands captured in one of the farms.	91

3.3	HSI boundaries used to evaluate the risk of heat stress in poultry.	92
3.4	Boundaries for detecting anomalous peaks in the truck acceleration module	94
3.5	Summary of features extracted from the time series at different poultry chain steps.	94
3.6	Initial DT for one of the farms simulating data adding noise to replications.	96
3.7	Capture of the DT for quality improvement in the poultry production chain provided by the PUMA tool.	98
3.8	Outlier produced after the temporary disconnection of the sensor in one of the houses in Madrid.	101
3.9	Example of the process of increasing DQ with the <i>dqts</i> package	102
3.10	Additive decomposition of one month's data from hourly electricity consumption series	103
3.11	Hourly forecast of electricity consumption data for ten days using SARIMA(2,0,0)(2,0,0) [24].	104
3.12	Hourly forecast of electricity consumption data for ten days using SARIMA(1,0,1)(2,1,0) [24].	104
3.13	Hourly forecast of electricity consumption data for ten days using the KNN with date and time features.	106
3.14	Extract from the electricity consumption data for households in Madrid	106
3.15	Evaluation of one-day ahead forecasting with KNFTS and KNPTS.	107
3.16	Part manufactured by a wLMD process.	111
3.17	Scanned part provided by the CT.	114
3.18	Creation of the persistence images from the multivariate time series.	119
3.19	System of the experimental rig	126

3.20	Box plots displaying the Smart Motor Position Error for 70 healthy test with a red line indicating the decision rule for the operation configuration.	129
3.21	Example of label assignment using the most probable class resulting from the classification algorithm.	130
3.22	Execution Flow of the Final Classification Algorithm	131
3.23	Final classification algorithm to determine the type of fault present in the shortest possible time.	133
3.24	Confusion matrices of the final classification algorithm in training and testing tasks.	133
A.1	The two modes of execution of the data quality visualisation in an example without normality metrics	326
A.2	Flow for the detection, inspection and resolution of poor DQ problems	330
B.1	One week ahead forecasting using KNPTS with MIMO and Recursive strategies.	333
B.2	Ten weeks ahead forecasting using KNPTS with MIMO and Recursive strategies.	334
B.3	Ten weeks ahead forecasting using KNPTS with ten different executions using MIMO strategy.	334

List of Tables

2.1	Summary of problems identified using each of the 11 proposed DQ metrics grouped in 5 dimensions.	35
2.2	Values of the <i>hour</i> variable to be used as input to the KNN algorithm.	62
2.3	Values of the sine and the cosine of the <i>hour</i> variable to be used as inputs to the KNN algorithm.	63
2.4	Confusion Matrix in a Binary Classification	77
2.5	Two different binary classification results with the same accuracy.	79
3.1	Classification of use cases in industrial applications based on data quality and number of samples.	85
3.2	Result of the classification on the test set.	97
3.3	Output of deepDQ function executed in <i>Timeliness</i> metric	102
3.4	Output of deepDQ function executed in <i>Range</i> metric .	102
3.5	Percentage of pores in each of the three parts	115
3.6	Percentage of pores in each of the three parts after rolling window processing.	117
3.7	Summary of the performance obtained by detecting porosity using different approaches	121
3.8	Confusion matrices from local classification part by part using KNN algorithm.	122
3.9	Confusion matrices from time series features classification part by part using XGBoost algorithm.	122

3.10	Confusion matrix from time series features classification using the three parts and the RF algorithm.	123
A.1	Sample of the data contained in the weatherdf set of the dqts R package.	322
A.2	Ranges values allowed for weatherdf data set of the dqts R package.	322
A.3	Metric scores for the complete execution of DQ function in weatherdf data set.	324

Part II



The research

An IoT Platform towards the Enhancement of Poultry Production Chains

- Title: An IoT Platform towards the Enhancement of Poultry Production Chains
- Authors: Iker Esnaola-Gonzalez, Meritxell Gómez-Omella, Susana Ferreiro, Izaskun Fernandez, Ignacio Lázaro, Elena García
- Journal: Sensors
- Year: 2020
- Quartile: Q1
- DOI: 10.3390/s20061549

Article

An IoT Platform towards the Enhancement of Poultry Production Chains

Iker Esnaola-Gonzalez ^{1,*}, Meritxell Gómez-Omella ^{1,2,†}, Susana Ferreiro ¹,
Izaskun Fernandez ¹, Ignacio Lázaro ¹ and Elena García ¹

¹ TEKNIKER, Basque Research and Technology Alliance (BRTA), C/ Iñaki Goenaga 5, 20600 Eibar, Spain; meritxell.gomez@tekniker.es (M.G.-O.); susana.ferreiro@tekniker.es (S.F.);

izaskun.fernandez@tekniker.es (I.F.); ignacio.lazaro@tekniker.es (I.L.); elena.garcia@tekniker.es (E.G.)

² Faculty of Informatics, University of the Basque Country (UPV/EHU), Paseo Manuel Lardizabal 1, 20018 Donostia-San Sebastián, Spain

* Correspondence: iker.esnaola@tekniker.es

† These authors contributed equally to this work.

Received: 2 February 2020; Accepted: 5 March 2020; Published: 11 March 2020



Abstract: As a consequence of the projected world population growth, world meat consumption is expected to grow. Therefore, meat production needs to be improved, although it cannot be done at any cost. Maintaining the health and welfare status of animals at optimal levels has traditionally been a main concern of farmers, and more recently, consumers. In this article the Poultry Chain Management (PCM) platform is presented. It aims at collecting data across the different phases of the poultry production chain. The collection of this data not only contributes to determine the quality of each phase and the poultry production chain as a whole, but more importantly, to identify critical issues causing process inefficiencies and to support decision-making towards the holistic improvement of the production chain. Results showed that the information gathered can be exploited to make different suggestions to guarantee poultry welfare, and ultimately, improve the quality of the meat.

Keywords: IoT; agriculture; poultry welfare

1. Introduction

The population of the world is growing at exponential rates and, according to United Nation's Revision of World Population Prospects (<https://esa.un.org/unpd/wpp/>), it is projected to reach a number of over 9.7 billion by 2050. This population growth poses issues that may affect the sustainability of demographic, social and economic systems. More specifically, one of the main challenges consists of finding a way to feed all these people and agriculture, which can be understood as the cultivation and breeding of animals and plants to provide food and other products to sustain and enhance life, plays a vital role in addressing this issue.

Due to the aforementioned population growth, world food consumption will increase accordingly, and consequently, so will meat consumption. As a matter of fact, meat consumption per capita is expected to increase from 38.7 kg in 2005, to 49.4 kg in 2050 and overall meat consumption, is expected to grow by 70% in the period of 2000–2030 and by 120% in the period of 2030–2050 [1]. Meat production is one of the most important sectors at a worldwide agriculture level and also in Europe. According to Eurostat (http://ec.europa.eu/eurostat/statistics-explained/index.php/Meat_production_statistics) there are almost 7 million livestock farms in the EU, and the four main types of farms are those rearing pigs, bovine animals, poultry, and sheep and goats. However, in order to satisfy the future meat demand increase, there is a dire need to increment meat production.

However, this meat production improvement cannot be made at any cost. Maintaining the health and welfare status of livestock at optimal levels has traditionally been a main concern of farmers [2], and more recently, consumers [3–5]. Comfort is one of the main factors that influences the well-being and health of animals during their lifetime [6], hence it cannot be neglected. Providing a comfortable environment not only maximizes the profit garnered from each animal, but also reduces mortality, which in turn allows the amount of wasted water and feed resources to be reduced. However, ensuring livestock comfort within farms may not be enough to guarantee their well-being and optimal meat quality when arriving at slaughterhouses. The way animals are transported from farms to slaughterhouses and the way they are loaded and unloaded in transport vehicles are stressful operations that might affect welfare and increase animal mortality [7]. Therefore, all these facts reinforce the need to ensure comfort of livestock through the whole production chain, from their breeding in farms until their arrival at slaughterhouses.

In the context of the Internet of Food & Farm 2020 (IoF2020) H2020 project <https://www.iof2020.eu/>, one of the trials is aimed at optimizing animal health, production chain transparency and traceability. Within this trial, there is a poultry production chain use case. In 2014, poultry meat production in the EU reached 10.5 million tonnes, representing around 12% of world production [8] and, similar to the meat production of other livestock farms, the poultry meat production is expected to grow at a worldwide level, reaching 181 million tonnes in 2050 [1]. 70% of EU poultry meat production is concentrated in seven member states (Poland, France, United Kingdom, Germany, Spain, Italy and the Netherlands) and chicken consumption has overtaken that of the pork or beef in some places such as the USA <https://www.nationalchickencouncil.org/about-the-industry/statistics/per-capita-consumption-of-poultry-and-livestock-1965-to-estimated-2012-in-pounds/>.

Nowadays, poultry chain managers get paid according to the quality of the meat that is obtained at the slaughterhouse. Therefore, one of the main criteria used to evaluate the whole production chain is the final quality of the meat. However, the quality of meat is strongly influenced by the stressful situations to which poultry is exposed throughout the production chain phases [9], even though market pressure does not sufficiently encourage breeding companies to give welfare traits greater weighting in their programs [8]. Consequently, a platform where information throughout the whole production chain is collected, establishes how to exploit that information to evaluate the quality of each phase and support the decision-making towards improving the final meat quality while ensuring animal welfare. Nevertheless, the development of such a platform is not an easy task due to the different inherent characteristics of the data sources (e.g., heterogeneity in terms of format and structures) to be integrated, and the need to ensure a secure data exchange environment.

In this article the Poultry Chain Management (PCM) platform is presented. It aims at collecting data across the different phases of the poultry production chain. The collection of this data not only contributes to determine the quality of each phase and the poultry production chain as a whole, but more importantly, to identify critical issues causing process inefficiencies and to support decision-making towards the holistic improvement of the production chain. Although it is motivated by the IoF2020 project, the PCM platform is based on open-source components and open standards towards its replicability in other projects and other livestock production chains. The rest of this article is structured as follows. Section 2 reviews related work. Section 3 presents the PCM platform and describes the flow of the data throughout the whole poultry production chain. Section 4 explains the KPIs used to evaluate each poultry production chain phase and showcases them in a real-world use case. Section 5 demonstrates the exploitation of the collected data with an analytical tool. Finally, the conclusions of this work are presented in Section 6.

2. Related Work

Nowadays, food industries invest a considerable part of their resources to ensure the quality of their products [10], and the poultry industry is no exception. As a result, many research efforts have

been dedicated to studying the influence that situations occurring in the different poultry production chain phases have in the poultry welfare and final meat quality.

Different solutions are proposed for ensuring poultry welfare and good rearing conditions within farms. A study used infrared images to discover that poultry under cold stress conditions spent about four times more energy trying to maintain body temperature [11]. Infrared thermography has also been used to evaluate metabolic heat loss of poultry fed with different energy densities [12]. There is work that develops predictive models to predict poultry growth [13] and death rates [14] within farms. Furthermore, the effect the indoor conditions of the farm has on poultry welfare has been researched. A laboratory experiment determined the influence of different temperature and humidity combinations on the physiological response of poultry [15]. Another study proposed the diagram shown in Figure 1, indicating how the relationship between ambient temperature and relative humidity affects poultry heat stress [16]. Towards ensuring optimal poultry rearing conditions, there is a system combining Knowledge Discovery and Semantic Technologies [17]. This system sends farmers notifications when a stressful thermal situation is predicted, so that they can anticipate such undesirable situations by taking actions on the farm beforehand.

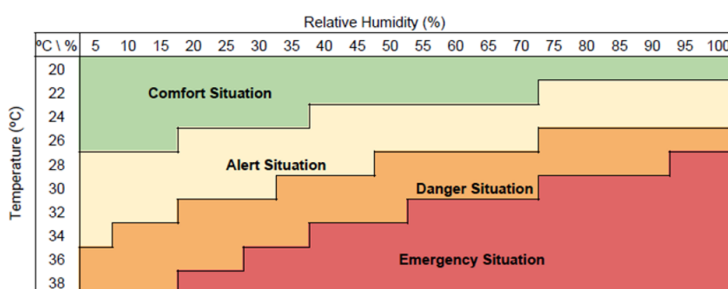


Figure 1. HIS (Heat Stress Index) for poultry [16].

Few scientific studies have been conducted to analyze the effect that catching and loading has on the welfare of poultry, although it is reasonable to assume that it generates a particular pattern and level of injury and stress. A study recorded reduced heart rates and catching damage in birds when using a semi-automatic loading system, in comparison to 3 other manual catch and loading approaches [18]. Other studies compared the death rates between a mechanical poultry catching system and a manual one, registering an increase in the manual system [19,20]. Furthermore, the rate of injuries has also been compared between manual and automatic poultry catching systems. There are studies that recorded significantly reduced injuries for mechanically caught poultry (3.1%) compared with manually caught (4.4%), especially with respect to leg injuries [19]. However, another study found no differences in the percentage of bruises, meat quality or corticosterone levels between manual and automatic catching methods [20]. There are studies that also attribute the decrease in injury rates to the reduction in speed of conveyors in automatic loading systems [19]. According to another study, the slower catching of automatic systems compared with manual catching may result in more heat stress mortality [21]. Furthermore, a study showed no significant difference between catching poultry from one leg (grabbing 2–3 birds per hand) or two legs (grabbing a maximum of 2 birds per hand) [22]. Additionally, the benefits of training and incentivising farm operators to adequately interact with animals in farms are recognized [23–25]. From these studies it can be concluded that, when properly carried out, using optimal equipment and trained personnel, both methods can result in low levels of injury and low levels of stress to the birds. Conversely, both manual and mechanical catching can result in unacceptably high levels of bruises, fractures and other injuries, as well as high stress levels, if carried out in an improper way [26].

Although loading procedures are more likely to cause physical injuries, the transport phase has also been reported to be stressful to poultry and have a direct effect on final meat quality. A research focused on the effect of truck temperature and relative humidity [27]. A study conducted with simulated electronic chickens suggested that ambient temperatures in the range of 11 °C to 25.1 °C during transport phases are thermally comfortable as they allow poultry to regulate heat with their metabolism in order to stay comfortable [28]. Conducted research also shows that poultry deaths on transport differs depending on the geographical point and the season of the year. For example, a study reported that poultry deaths increased from 0.28% in winter, to 0.42% in summer for transport in a subtropical climate of southeastern Brazil [7]. Another research, in this case conducted in Italy, showed that death rates in transport increased from 0.35% in winter months to 0.47% in summertime [29]. Transport duration has also a direct effect on the poultry welfare [30], as a study showed that the lowest poultry death rates happened in journeys of less than 2 h (0.29%) and the highest when transport times exceeded 5 h (0.46%) [31]. Additionally, in another different research, death rates also increased from 0.24% (in routes of less than three and a half hours) to 0.45% (in routes of more than 5 h) [32]. Likewise, the transport distance traveled from the farm to the slaughterhouse influences the welfare of the poultry. A study conducted in the Czech Republic recorded that for journeys up to 50 km death rate was 0.15%, and 0.86% for distances exceeding 300 km [33]. Last but not least, the vibrations related with the sudden acceleration of trucks have also been studied and showed a negative effect on poultry welfare due to the impacts and muscle strain suffered [34–36].

A certain number of individuals from each flock may be rejected in the slaughterhouse due to status as a result of poor welfare conditions, such as abnormal levels of contact dermatitis, parasitism or systemic illness in the house [21]. Other causes for rejection include infectious diseases such as colisepticaemia [37], skin injuries such as cellulite or dermal squamous cell carcinoma [38], and fractures or bruises [39]. Furthermore, the tasks performed in the slaughterhouse phase may also result in the rejection of the animal, such as contamination produced by evisceration [39].

According to the European Food Safety Authority, a monitoring system for animal welfare is made up of the following steps: (I) identification of the goal; (II) identification of the population concerned, and definition and selection of the survey population; and (III) selection of the indicators and the systematic collection of data [40]. Furthermore, the systematic collection of animal-based measures and its subsequent storage in well-defined databases could pave the way to better assessing the validity and robustness of those measures, thus moving towards quantitative risk assessment of animal welfare [41]. Moreover, in animal health research, the visualization of relationships between risk factor and health outcomes is based on the correction of indicators, followed by an analysis to investigate relevant associations. There are data collection systems at specific poultry production chain phases, especially in the breeding phase [42–45]. However, to the extent of our knowledge, there is no platform that collects information throughout the whole poultry production chain for its exploitation towards poultry welfare improvement.

3. The Poultry Chain Management Platform

Chickens raised for meat (also known as broiler chickens) arrive at the farm around 21 days of after hatching and it typically takes around 7 weeks until animals reach the proper size and weight to be sent to the slaughterhouse <https://www.chickencheck.in>. In order to ensure adequate growth and welfare, there are dietary needs and comfort requirements that need to be fulfilled. On the one hand, these comfort requirements vary with age. For example, the first days of breeding, poultry require a higher ambient temperature because their body temperature, metabolic rate, insulation from feathering and thermoregulatory ability are relatively low [46,47]. On the other hand, these comfort requirements may also change if, for certain reasons, the growth of the poultry flock is slower or faster than expected. This phase, where poultry are raised, is referred to as the breeding phase. Once the flock reaches the determined size and weight and they are ready to be sent to the slaughterhouse, farm operators catch and load them into holding cages or modular bins. These cages are specifically

designed to ensure that birds cannot hurt themselves or other birds, and that the air is able to circulate. This phase is referred to as the loading phase. These cages are then placed on a truck which transports the flock from farms to the slaughterhouse. This is the transport phase. Finally, once the animals arrive at the slaughterhouse, they are unloaded and handled by slaughterhouse operators in the phase referred to as the slaughterhouse phase.

Summarizing, there are four phases that comprise the whole poultry production chain from when baby chicks arrive at the farm until they arrive at the slaughterhouse: breeding, loading, transport and slaughterhouse phases. Figure 2 depicts the four main poultry production chain phases.



Figure 2. The four phases involved in a poultry production chain.

The variety of situations that may occur throughout the different phases has a direct impact on the health and welfare of animals, as well as on the quality of their meat by the time they arrive at the slaughterhouse. Therefore, monitoring each of these phases can provide relevant data that can later be exploited to detect causalities behind these issues and adopt measures towards the holistic improvement of the poultry production chain. Therefore, a platform that collects and manages information across the whole production chain is considered of utmost importance.

3.1. The PCM Platform Workflow

The PCM is a cloud-based platform supported by FIWARE open-source components, and implemented based on open standards in charge of collecting, processing and storing data coming from different phases of the poultry production chain in a secure way. The PCM platform is divided into four different stages as can be seen in Figure 3. It is worth mentioning that the exploitation of the collected information is left out of the architecture.

Data Sources

This is the first stage of the platform where methods to access, assess, convert and aggregate signals are employed by different devices, machines or systems to represent real-world parameters as communicable data assets. In the context of the poultry production chain, data sources are heterogeneous ranging from sensors to measure the environmental conditions within farms or trucks, to wristbands measuring the way operators handle animals. Furthermore, each data source may have its own method to collect the information.

Data Acquisition

This is the stage where the data coming from the various monitoring devices and systems is received. This stage consists of different agents and components to enable the adequate data handling coming from heterogeneous data sources.

The central component is the FIWARE Orion Context Broker <https://fiware-orion.readthedocs.io/> (OCB), a C++ implementation of the NGSiv2 REST API binding. The OCB allows the management of the entire lifecycle of context information including updates, queries, registrations and subscriptions.

The recurrent news about security breaches, private data leaks and the inappropriate use of data, makes the security of the IoT platforms a vital requisite nowadays. Therefore, the security of the PCM platform is of utmost importance. Due to the multitude of different devices, sensors and services involved in the data flow, the security of the PCM platform has to be handled by different agents. FIWARE Keyrock <https://fiware-idm.readthedocs.io/> is responsible for identifying,

authenticating and authorizing devices and systems to publish their information in the OCB, by associating them rights and restrictions with established identities. It is based on OpenStack Keystone <https://docs.openstack.org/keystone>, a service that provides API client authentication, service discovery, and distributed multi-tenant authorization by implementing OpenStack's Identity API, and OpenStack Horizon <https://docs.openstack.org/horizon/>, which provides a web-based user interface to OpenStack Keystone. Additionally, to complete the security module of the PCM platform, the FIWARE PEP Proxy <https://fiware-pek-proxy.readthedocs.io/> provides a security layer for adding authentication and authorization filters, and it is combined with Keyrock to enforce access control to backend applications.

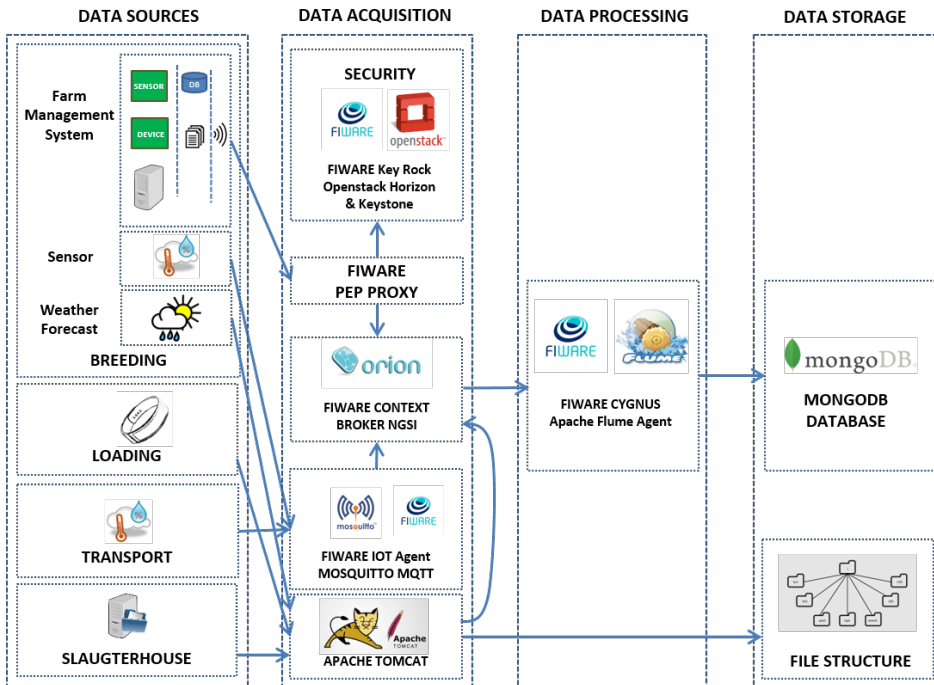


Figure 3. The PCM platform data flow.

Furthermore, data coming from IoT environmental sensors (explained in Section 3.2) is collected by a FIWARE IoT Agent <https://fiware-iotagent-json.readthedocs.io/>, which has a Mosquitto MQTT broker <https://mosquitto.org/> embedded. The MQTT protocol is envisioned as a high performing solution for data acquisition, not only because of the low power and memory needed for the implementation of different clients in small devices, but also due to the low bandwidth needed. The FIWARE IoT Agent is used as a bridge to publish sensor data to the central OCB.

Last but not least, there is an Apache Tomcat server in charge of executing periodic tasks for sending CSV-like files to store them in file structures, and get data from external sources (e.g., meteorological prediction from weather forecasting services) in order to publish it in the central OCB.

Data Processing

This is the stage where the data acquired in the previous stage is sent to be stored in the corresponding data repository. The main component of this stage is a FIWARE Cygnus <https://fiware-cygnus.readthedocs.io/> agent. Cygnus is a connector in charge of storing certain sources

of data in certain configured third-party storage, creating a historical view of such data. Internally, Cygnus is based on Apache Flume <http://flume.apache.org/>, a technology addressing the design and execution of data collection and storage agents. An agent is composed of a listener or source in charge of receiving the data, a channel where the source puts the data once it has been transformed into a Flume event, and a sink, which takes Flume events from the channel in order to store the data within its body into a third-party storage.

Data Storage

This is the stage where the collected data is stored and remains accessible for its future exploitation. Two main data storage repositories are considered in this stage. On the one hand, MongoDB <https://www.mongodb.com/>, a NoSQL database which uses JSON-like documents and is the adequate option to store data coming from heterogeneous sources. On the other hand, a file structure to store files created in the different poultry production chain phases and that are not considered to be worth passing through the OCB for various reasons. For example, the amount of data acquired during the loading of the animals with the wearable devices is too big, and in this case, the CSV structure is more suitable for data analysis tasks.

3.2. the Information Collection from the Different Phases

The poultry production chain is characterized by four phases: breeding, loading, transport and slaughterhouse phases. In each of these phases, different information is retrieved, which could potentially be exploited to both evaluate the quality of each phase and support the decision-making towards improving the final meat quality while ensuring animal welfare. This section explains the information collected from each phase, and the way in which data from each phase is obtained and stored in the PCM platform.

Breeding Phase

This is the phase where chickens arrive at the farm 21 days after hatching and spend around 7 weeks until they reach the required weight and size. During these 7 weeks the comfort requirements of the flock including temperature and humidity vary, and if they are not satisfied, animals may be exposed to stressful situations that may result in deficient growth. For example, if relative humidity is too low, there is a higher production of dust and an increase in the number of airborne microorganisms, which may increase susceptibility to respiratory diseases especially during the early days of the broiler. However, if relative humidity levels which are too high are combined with high temperatures, broilers may die from hyperthermia or hypoxia [26]. Although there are different guidelines to set minimum comfort requirements, in the context of the IoF2020 project, they are defined by farmers, and they can be modified at any moment during the breeding phase for different reasons, including a slower or faster rate of flock growth.

The information generated in the breeding phase is retrieved from different data sources. On the one hand, IoF environmental sensors are installed throughout the farm to measure the climatic conditions in different points of the farm. In the initial deployment plan of IoF2020 project, Tibucon sensors (which were developed as part of the TIBUCON FP-7 project <https://cordis.europa.eu/project/rcn/95501/en>) were deployed, which measured temperature, humidity and luminosity values. These sensors were then upgraded in terms of hardware and software, and CO₂ and ammonia level measurement capabilities were added. The resulting IoF environmental sensors are easy-to-install, have a battery supply and offer wireless connectivity. They measure farm conditions every 30 s and this data is sent through a low-power multi-hop wireless network based on the standard IEEE 802.15.4 https://standards.ieee.org/standard/802_15_4-2015-Cor1-2018.html) to a gateway installed in the control room next to the farm. The gateway then sends the data to the Mosquitto MQTT Broker of the PCM platform which will in turn send it to the central OCB.

In addition, there is an existing third-party Farm Management System deployed within farms. This system collects information from different devices and sensors installed within farms, including weight scales, temperature and humidity sensors. The information collected by these sensors is stored in a centralized database. In order to integrate this information within the PCM platform, a Visual Basic application periodically retrieves the latest data stored in the central database of the Farm Management System and sends it to the FIWARE PEP Proxy. Once this Proxy authenticates and authorizes the data delivery, it is forwarded to the central OCB.

The external weather conditions have a direct impact on the indoor conditions of the farm. Therefore, a weather forecasting service is leveraged to retrieve this information. More precisely, Tiempo.com <https://www.tiempo.com/> service's API is accessed executing a Java-based periodic task which collects the weather forecasting for the location where the IoF2020 project's use case farms are located. Tiempo.com offers predictions with different time-horizons and formats, among which XML files with hourly predictions for the next 5 days are leveraged. These files include the predicted temperature, relative humidity, sky status (e.g., cloudy or sunny) and wind speed for a given location, and this data is sent to the central OCB.

Loading Phase

In most European countries, poultry loading is performed by catching the birds by one or two legs and carrying three or four birds in each hand [48,49]. The poultry handling is recommended to be carried out in a careful and conscientious way in order to avoid stressful situations, injuries and subsequent downgrading of the meat. However, in practice, the loading phase is often rather rough due to the poor working conditions of the personnel, consisting of arduous and repetitive work in a dusty environment [50]. In summary, the loading of animals into transport trucks has a direct effect on the final quality of the meat and its supervision may ensure poultry welfare and prevent injuries.

In order to monitor how the flock is loaded into trucks, operators wear electronic wristbands which measure arm sway acceleration. It is worth mentioning that operators work in an environment without connectivity, so wristbands cannot send data periodically and are required to have large data storage capabilities. Initially, operators used Wear OS smart watches <https://wearos.google.com/>, however, they were later replaced by Axivity AX3 3-axis logging accelerometer wristbands <https://axivity.com/product/wrist-band> due to difficulties capturing data at a continuous rate. The wristbands collect information during the whole loading phase (which may last from 3 to 5 h), and then these wristbands are sent to the slaughterhouse in the transport trucks. Once they arrive at the slaughterhouse, a farm operator plugs the wristbands into the USB port of a PC and the collected information is managed using the OMGUI (Open Movement GUI) software. This software allows both the visualization and the download of the collected data into a CWA binary format file. This file is not compatible with Microsoft Excel or other third-party software, so the collected data needs to be exported into a CSV file using the OMGUI. As these CSV files may be too large (up to 4 GBs of data), the storage of information through the OCB is discarded. Instead, these CSV files are sent via WeTransfer to the PCM platform managers. Next, they store these files in the adequate folder of a file structure where they remain accessible to be exploited.

Transport Phase

This is the phase where the poultry flock is transported from farms to slaughterhouses. The transport phase has become a cause for concern because of animal welfare consideration, associated chicken mortality and consequential economic losses [9,51]. The transport journey duration is directed linked to the fasting duration of the poultry, so it has to be correctly estimated to ensure that the feed privation will be as short as possible. As a matter of fact, domestic birds can be transported without food and water up to 12 h [52]. Furthermore, the transport vehicle must ensure the safety of the animals and their welfare [53], for example by using side covers to protect birds from cold and wet weather while not impeding the air circulation. Last but not least, the driving style is directly related

to the amount of stress perceived by poultry. Smooth and consistent speed driving habits allow the poultry to relax more during a journey, thus ensuring their welfare and the final meat quality [54]. Therefore, a good transport preparation is essential to avoid causing different degrees of injuries and stress to poultry, ranging from mild discomfort to more severe situations that may terminate in death.

Trucks transporting animals from farms to slaughterhouses are equipped with sensors that measure certain environmental properties. These are IoF transport sensors (similar to the IoF environmental sensors installed within farms) and they are attached to the cages or modular bins where poultry is previously loaded. The IoF transport sensors measure temperature, humidity, acceleration, ammonia and CO₂ values. Taking into account that the transport trucks are vehicles without connectivity, sensors cannot send the collected information to the MQTT broker in real-time, but instead, they record all the information until the trucks arrive at the slaughterhouse. Once there, sensors are unloaded from the truck and placed near the slaughterhouse gateway. When the sensor detects the wireless network created by the gateway, the previously recorded information is sent to it, and afterwards, to the MQTT broker. From there, to the OCB and then, data is stored in MongoDB.

Slaughterhouse Phase

This is the phase where, once the poultry flock is unloaded from transport trucks, they are processed and packaged. This stage includes, in turn, different sub stages [55]. First of all, chickens are slaughtered and completely bled. Then, in the scalding step, carcasses are immersed into hot water to ease the elimination of feathers. Afterwards, the carcasses are submitted to gutting and washing processes, and carcasses are classified according to their weights and quality. This process is followed by the chilling of carcasses and entrails. Finally, the carcasses are sent to the cutting stage where different pieces or products of poultry meat (e.g., wings and breasts) are produced and packaged. Therefore, the overall quality of the slaughterhouse phase is based on the quality of the final poultry meat.

Once the transported poultry flock is unloaded in the slaughterhouse, operators evaluate their state. They take a random sample of 200 chickens of which the number of dead animals and physical conditions (e.g., broken wings and bruises) are assessed. When the flock is processed and packaged, the status of the plucking or evisceration is also evaluated. The overall quality of the slaughterhouse phase is assessed based on the aforementioned criteria and it is registered in an Excel file. This file contains information of flocks coming from different farms and belonging to different poultry production chains and it is sent to the PCM platform managers once a week. Afterwards, this file is stored in the file structure of the platform. It is worth mentioning that, within this file, each flock is correctly identified so that it can be related to the corresponding information of the previous phases.

4. The Poultry Chain Quality Indicators

The data collected throughout the different phases of the poultry production chain remains accessible in the storage systems of the PCM platform. As a matter of fact, this data is retrieved and exploited by different data analytic services for various purposes. One of those services generates quality indicators for each phase. These indicators are used to determine the overall quality of the poultry production chain, and may support the adoption of specific actions towards the improvement of future production chains.

An indicator is an objectively verifiable measurement which reflects the activity, assumption, or effect being measured and allows for comparisons both between different populations or individuals and between measures of the same population or individual at different points in time [56]. Furthermore, the crucial factors when defining an indicator are that it is valid (or appropriate) and reliable (or trustworthy) as well as feasible to measure, given the resource constraints [40]. This section details the exploitation of the data available in each phase for developing the indicators used to determine their quality. In addition, a real-world poultry production chain use case is employed to showcase the extraction and meaning of such indicators. Namely, one that started on 30 April 2019

and ended on 01 July 2019 with a flock of around 33,000 chickens raised in the poultry farm shown in Figure 4. For each of the four phases, the conditions evaluated are described, defined Key Performance Indicators (KPIs) are specified and the results obtained for the aforementioned use case are detailed.

The KPI generation for each phase is automatized with R scripts that are automatically executed in an Rserve <https://www.rforge.net/Rserve/> version 3.2.5 deployed in a Docker <https://www.docker.com/> container. The only exception are the slaughterhouse phase KPIs, which are manually calculated by slaughterhouse operators and saved in an Excel file. Furthermore, obtained KPIs are stored in the MongoDB database of the PCM platform, thus remaining accessible for their analysis or further exploitation. An example of the generated exploitation of the KPIs is described in Section 5.



Figure 4. Real-world poultry farm used for demonstration purposes.

4.1. Poultry Breeding Phase

In order to determine the quality of the poultry breeding phase, five KPIs are defined. The first set of KPIs comprises two KPIs which are calculated based on the optimal farm temperature for the flock rearing. These temperatures, which are provided by farmers, vary for the different poultry growth stages (e.g., chicklings or adult stages) and other factors (e.g., a slower or faster growth pace of the flock). Namely, the aforementioned two KPIs are *Temperature Warning* and *Temperature Alarms*. The *Temperature Warnings* determines the percentage of time when farm temperatures deviate between 1.5°C and 3°C from the optimal temperature. This KPI determines the period of time in which animals have been exposed to mild uncomfortable temperatures while they are in the farm. *Temperature Alarms* is the percentage of time when farm temperatures deviate more than 3°C from the optimal temperature. When this situation occurs, it is considered that the flock is exposed to severe thermal discomfort that may terminate in considerable heat stress. Therefore, this KPI determines the percentage of time in which animals have been exposed to severe uncomfortable temperatures in the farm. The second set of KPIs is based on the HIS, which combine the effects of both temperature and relative humidity of the air to determine the stress to which the flock is exposed within the farm [16]. Depending on the value of the HIS, a level of stress is assumed for the flock. A HIS value between 70 and 75, establishes the *Alert Situation* KPI where poultry may start to pant. A HIS value between 76 and 81 is considered a dangerous situation (*Danger Situation* KPI), meaning that a considerable heat stress condition exists for the flock. A HIS value higher than 81 triggers an emergency state (*Emergency Situation* KPI), indicating that extreme heat conditions exist.

The KPI values obtained for the previously presented case of real-world poultry production chain use breeding phase are shown in Table 1. As shown by the *Temperature Warnings* KPI, 18.31% of the time when the flock was within the farm, the temperature was moderately distant compared with the optimal. Furthermore, for the *Temperature Alarms* KPI, a value of 34.48 was obtained, which means that 34.48% of the time, the flock was exposed to potentially harmful situations where the temperature within the farm was very different from the desired one. This means that, overall, the flock has been exposed to undesirable temperatures more than half of the time it stayed in the farm. Regarding the heat stress determined by the HIS, 7.89% of the time the flock has been exposed to alert situations,

5.62% of the time to a more severe danger situation, and 68.30% to emergency situations. That is, the flock has been exposed to undesirable heat stress situations a combined 81.81% of the whole rearing period.

Table 1. KPI results obtained in poultry breeding phase.

KPI	Value
Temperature Warning	18.31 %
Temperature Alarm	34.48 %
Alert Situation	7.89%
Danger Situation	5.62 %
Emergency Situation	68.30 %

4.2. Loading Phase

The objective of the loading phase indicators is to identify the quality of the conditions in which the flock is loaded into transport trucks. Within the context of the IoF2020 project, the poultry catching and loading is performed manually, so, in this phase, the force with which the operators carry out these tasks is controlled. Force is directly proportional to acceleration (by Newton's second law), therefore, operators arm sway acceleration is captured with electronic wristbands. These wristbands contain a sensor that measures acceleration on three axes, that is, they measure three components of acceleration: a_x, a_y, a_z . The acceleration is measured in g-force, or the gravitational force, which is a measurement of the type of force per unit mass that causes a perception of weight. A g-force of 1 g is equivalent to the conventional value of gravitational acceleration on Earth, about 9.8 m/s^2 . The wristband sensor has a range of forces that can measure, being -8.00 g the lowest measurable value, and 7.98 g the highest. Furthermore, the sensor has a sample rate of 1000 Hz , that is, it logs data with a frequency of 1 ms .

In order to determine the overall quality of a loading phase, 3 KPIs are defined. The first KPI called *Saturation Rate*, is based on the aggregation of the acceleration data on every second, and determines the average of the amount of acceleration data that exceeds the wristband sensor range of forces per minute. In other words, the average of the amount of saturated status. The remaining two KPIs are based on the acceleration module a , which is measured in g and is calculated as follows:

$$a = \sqrt{a_x^2 + a_y^2 + a_z^2} \quad (1)$$

Namely, these two KPIs are the *Mean Accumulation* and the *Standard Deviation* of acceleration. They are both based on the amount of acceleration every minute, calculated from the maximum value of acceleration module every second.

The KPI values obtained for the previously presented real-world poultry production chain use case's transport phase are shown in Table 2. The 0.98 value for the *Saturation Rate* means that the wristband sensor gets saturated almost once per minute.

Table 2. KPI results obtained in loading phase.

KPI	Value
Saturation Rate	0.98
Mean Accumulation	121.67
Standard Deviation	48.99

4.3. Transport Phase

In order to determine the overall quality of a transport phase, five KPIs are defined, which can be classified in two groups according to the environmental aspect that they specify. The first set of

KPIs determines the flock's thermal comfort and it is composed of 4 KPIs: *High Temperature*, *Low Temperature*, *High Relative Humidity* and *Low Relative Humidity*. These KPIs determine the percentage of the transport time under which the flock is exposed to temperatures over 31°C (*High Temperature*) or below 18°C (*Low Temperature*) and to relative humidity of over 80% (*High Relative Humidity*) or below 60% (*Low Relative Humidity*), which are considered to be stressful and harmful situations for the animals. The fifth KPI, the *Abrupt Movements* KPI, determines the quality of the transport in terms of the driver's abruptness when driving. This KPI derives from the acceleration measurements of the IoF transport sensor deployed in the transport truck and calculates the percentage of time in which flocks suffer from the drivers sudden speed changes. In order to calculate this, a peak detection algorithm called The Smoothed Z-score Peak Detection Algorithm [57] is defined, based on the theoretical normal distribution of the acceleration. If a new acceleration measurement is a given x number of standard deviations away from a given moving mean, the algorithm generates a signal for this value.

This algorithm requires three parameters to be configured: lag, threshold and influence. The lag parameter is the size of the moving window and determines, on the one hand, the amount of data to be smoothed, and on the other hand, how adaptive the algorithm is to the changes in the long-term average of the data. The threshold parameter is the z-score at which algorithm generates signals, that is, the number of standard deviations from the moving mean above which the algorithm will classify a new data point as being a signal. Finally, the influence parameter indicates the effect of new signals on the mean and standard deviation and it has a value between 0 and 1. That is, it determines the effect of signals on the detection threshold of the algorithm. If value is 0, the signals have no influence on the detection threshold, therefore, future signals will be detected based on a detection threshold that is calculated with a mean and standard deviation that are not influenced by previous signals.

Let a be the acceleration module vector of length t , which is calculated with Equation (1), where a_x , a_y and a_z are the acceleration components measured in g along the X, Y, Z axes.

Once the acceleration module is calculated, the peak detection algorithm works as follows shown in Algorithm 1:

Algorithm 1: The Smoothed Z-score Peak Detection Algorithm.

```

1 a, lag, threshold, influence Signals, avgFilter, stdFilter
  /* Initialize Variables */
2 set signals to vector 0, ..., 0 of length of a;
3 set filtered_a to a(1), ..., a(lag);
4 set avgFilter to null;
5 set stdFilter to null;
6 set avgFilter(lag) to mean(a(1), ..., a(lag));
7 set stdFilter(lag) to std(a(1), ..., a(lag));
  /* For loop to calculate signals */
8 for i ∈ lag + 1, ..., t do
9   if a(i) - avgFilter(i - 1) > threshold * stdFilter(i - 1) then
10    /* Positive signal */
11    if a(i) - avgFilter(i - 1) > threshold * stdFilter(i - 1) then
12     set signals(i) to +1
13    /* Negative signal */
14    else if a(i) - avgFilter(i - 1) < threshold * stdFilter(i - 1) then
15     set signals(i) to -1
16    /* Reduce influence of signal */
17    set filtered_a(i) to influence * a(i) + (1 - influence) * filtered_a(i - 1)
18  else
19    /* No signal */
20    set signals(i) to 0
21    set filtered_a(i) to a(i);
  /* Adjust the filters */
22 set avgFilter(i) to mean(filtered_a(i - lag), ..., filtered_a(i))
23 set stdFilter(i) to std(filtered_a(i - lag), ..., filtered_a(i))

```

The algorithm returns three vectors called *signals*, *avgFilter* and *stdFilter*. The first one takes the values -1 , 0 or $+1$ depending on whether the acceleration is considered stable (value 0) or not (values -1 or $+1$). If a new acceleration point exceeds the threshold of standard deviations away from the moving mean, it will assign $+1$ when it is above the upper limit, and -1 when it is below the lower limit. Otherwise, the algorithm will assign the value 0 to the signal variable. The other two correspond to the moving average (*avgFilter*) and standard deviation (*stdFilter*) of the previous data window, and they are calculated each time a new acceleration point is analyzed by the algorithm.

With views to ease the understanding to the Smoothed Z-score Peak Detection algorithm, an illustrative example will be provided next. Given $lag = 25$, $threshold = 5$ and $influence = 0$ as parameters of the Smoothed Z-score Peak Detection Algorithm, and a the acceleration module calculated using the a_x , a_y and a_z data captured for 60 s, a simulation of that algorithm is computed. Acceleration values are shown in black in Figure 5. The blue discontinued line is obtained from *avgFilter* and two red discontinued lines are the limits of the confidence interval that is built using the algorithm. Finally, the signal results are represented in Figure 6 using a simple line graph to illustrate the abruptness score of the driver when transporting the animals.

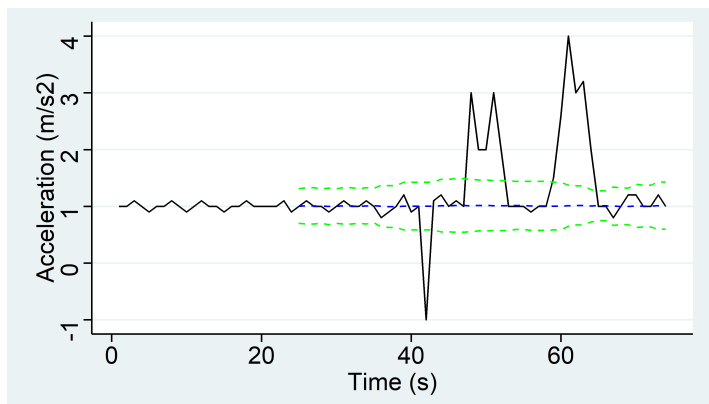


Figure 5. Results of simulation of the Z-Peak Algorithm.

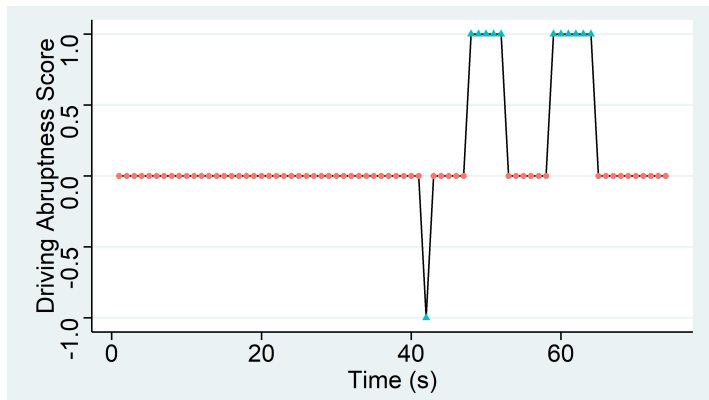


Figure 6. Representation of a driving abruptness score.

Table 3 shows KPI values for the case of the previously presented use transport phase. It can be stated that temperature has been in comfort ranges as it has never reached values which are too high or too low. As for the relative humidity values measured, 17% of the transport time they have surpassed the upper comfort limit (i.e., 80%), thus exposing the flock to excessive relative humidity.

Finally, the truck that transports the flock from the farm to the slaughterhouse has undergone abrupt changes of acceleration during 9.32% of the whole journey time.

Table 3. KPI results obtained in transport phase.

KPI	Value
Low Temperature	0%
High Temperature	0%
Low Relative Humidity	0%
High Relative Humidity	17%
Abrupt Movements	9.32%

4.4. Slaughterhouse Phase

In total, a set of 19 KPIs are defined to determine the slaughterhouse phase quality. This set includes KPIs to evaluate the physical conditions of the flock on its arrival (e.g., broken wings, bruises and broken bones) and the processing and packaging tasks performed (e.g., percentage of bad eviscerated and bad showered poultry). The last KPI, *Meat Quality*, takes only two categorical values depending on the quality of the product: *A* if the majority of the final meat is of a high-level, and *B* otherwise. These KPIs are manually evaluated by slaughterhouse operators, and stored in an Excel file that follows a predefined template.

Regarding the KPI values obtained for the slaughterhouse phase of the use case, they are shown in Table 4. These results refer to a sample of 200 chicken from a lot of 5040.

Table 4. KPI results obtained in slaughterhouse phase.

KPI	Value
Weight Range	3.42 Kg
Farm Weight	3.42 Kg
Total Hematoma	23
Broken Wing	21
Hematoma Wings	11
Hematoma Armpit	0
Breast	12
Broken Bones	9
Overscalded	0
Bad Extraction Viscera	1
Bad Plucked	0
Bad Wash	0
Scab	0
Crops	0
Knuckles	0
Dead in transport	3
Confiscated	8
Numbers of chickens	5040
Meat Quality	A

5. PUMA: A Decision Support System for Poultry Chain Managers

The PCM platform stores both the data collected through the poultry production chain and the generated KPIs in order to enable its further exploitation by different services and tools. One of those tools is PUMA (PoUltrY Management Advisor), an Artificial Intelligent system inspired by the inherent requirements of the improvement of the whole poultry production chain. Namely, it is a Decision Support System for poultry chain managers developed with Shiny <https://shiny.rstudio.com/>, an R package that supports the building of interactive web apps straight from R. A screenshot of PUMA is shown in Figure 7.

PUMA is based on machine learning algorithms to extract knowledge from the KPIs generated through the different poultry production chain phases (explained in Section 4). It leverages a Decision Tree type of algorithm, namely Classification and Regression Tree (CART), which is a supervised learning method that uses a tree structure in order to go from features of an item (represented as branches) to conclusions about the item’s final value (represented as leaves). The CART algorithm is implemented using R’s Recursive Partitioning And Regression Tree (RPART) package <https://www.rdocumentation.org/packages/rpart/versions/4.1-15>. Decision trees can be thought as a disjunction of conjunctions that result in IF-THEN-ELSE type of rules. These rules can also be represented in a more formal way by means of Disjunctive Normal Forms (DNF). Furthermore, the use of a Decision Tree favors the transparency and the explainability in decision-making, thus aligning with the European Commission’s Ethics guidelines for trustworthy Artificial Intelligence systems <https://ec.europa.eu/digital-single-market/en/news/ethics-guidelines-trustworthy-ai>.

With views to making PUMA flexible and applicable to different use cases, it is developed so that it allows the manual selection of the variable to be predicted and the set of explanatory variables to be used for creating the decision tree. Furthermore, data used for the creation of the decision tree can be retrieved from a database or a CSV file. For example, a PUMA user may select the “number of deaths” variable as the feature to be predicted, and the transport phase KPIs to see how the transport conditions may influence the number of deaths of poultry in transport. For demonstration purposes, let us consider the following scenario: a chain manager wants to receive recommendations for the different poultry production chain phases in order to maximize the final meat quality. To do so, the Meat Quality variable will be selected as the feature to be predicted, and the Breeding, Loading and Transport KPIs as the explanatory variables.

PUMA: A DSS for Poultry Chain Managers (IoF2020)

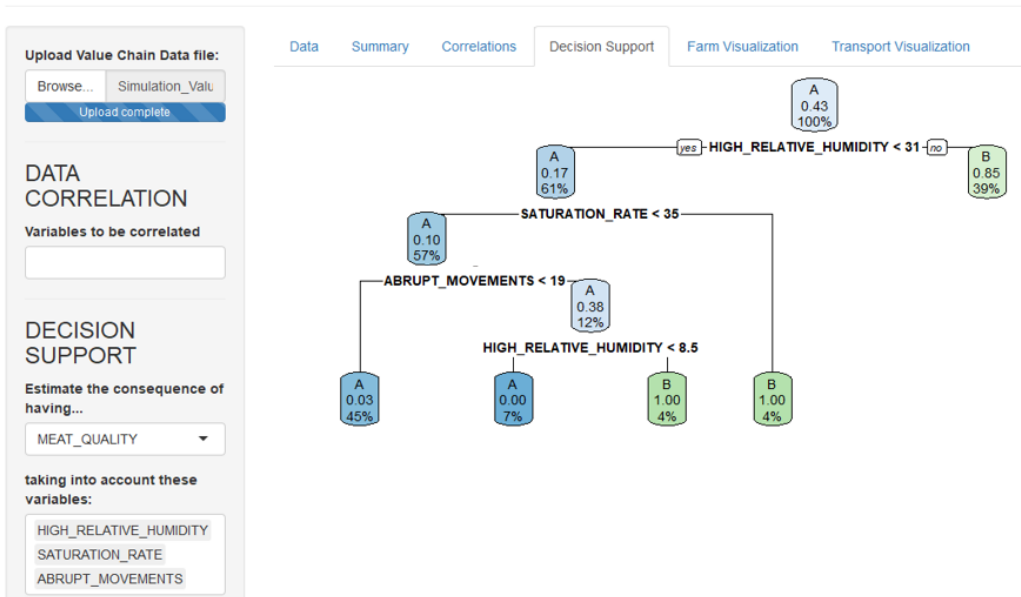


Figure 7. Screenshot of the PUMA tool.

PUMA leverages rules generated from decision trees derived from historical KPIs. As more poultry production chains are observed and the corresponding KPIs are calculated, decision trees and the derived rules are updated. The more the poultry production chains that are observed, the bigger the reliability of the generated rules. These rules capture the real behavior of the breeding carried

out on that farm, their respective loads of the animals in the trucks, transport to the slaughterhouses, and their physical status on arrival as well as the final meat quality.

For the purpose of demonstrating PUMA, a set of poultry production chains were simulated and the corresponding set of KPIs were generated. The decision tree shown in Figure 8 is the result of this simulation.

Furthermore, the following rules expressed in DNF can be derived by searching a path through the created decision tree:

- Rule (1): $ES > 65\% \wedge AM > 40\% \wedge HH > 20\% \rightarrow MQ = B$
- Rule (2): $ES > 65\% \wedge AM > 40\% \wedge HH \leq 20\% \rightarrow MQ = A$
- Rule (3): $ES > 65\% \wedge AM \leq 40\% \wedge HT > 50\% \rightarrow MQ = B$
- Rule (4): $ES > 65\% \wedge AM \leq 40\% \wedge HT \leq 50\% \rightarrow MQ = A$
- Rule (5): $ES \leq 65\% \wedge LT > 35\% \rightarrow MQ = B$
- Rule (6): $ES \leq 65\% \wedge LT \leq 35\% \rightarrow MQ = A$

where ES: Emergency Situation, AM: Abrupt Movements, HH: High Relative Humidity, HT: High Temperature, LT: Low Temperature and MQ: Meat Quality.

In order to showcase the use of this decision tree, the KPIs generated from a real-world poultry production chain use case (described in Section 4) will be used. In this use case, the value of ES is 60.30%, the preconditions of rules (5) and (6) cannot be satisfied and therefore they are discarded. The objective of the chain manager using PUMA is to get the A value for variable MQ, so rules (2) and (4) are those followed. Depending on the quality of the transport trucks and the routes followed by them, experts will decide between giving some recommendations or others. There are two ways to get meat to reach quality A, either $AM > 40$ and $HH \leq 20$ (2) or $AM \leq 40$ and $HT \leq 50$ (4).

Let us consider that an optimal form to control the temperature is available in trucks but there is no adequate relative humidity control, so following the rule (4) is the best option to get high meat quality and the decision is made to take the truck along a smoother road, such as a highway, so that the AM KPI does not exceed 40%. The simulation results show that $AM = 9.32\%$ and $HT = 0\%$, so this lot will be cataloged with the meat quality of type A.

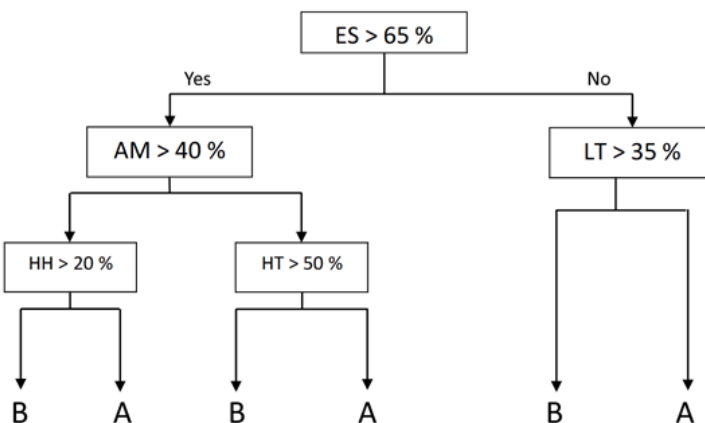


Figure 8. Decision Tree obtained training the algorithm with a simulated KPI.

6. Conclusions

The population of the world is growing at exponential rates and one of the main challenges consists of finding a way to feed all these people. One of the potential answers to this challenge considers meat production improvement, although it cannot be done at any cost. Maintaining the

health and welfare status of animals at optimal levels has traditionally been one of the main concern of farmers, and, more recently, consumers. One of the most relevant types of meat is poultry, which is expected to grow at a worldwide level, reaching 181 million tonnes in 2050.

Although there are solutions that aim at monitoring different poultry production chain phases, there is no existing solution that collects data throughout the whole chain. In this article, the Poultry Chain Management (PCM) platform is presented. It collects data across the different phases of the poultry production chain in a centralized and secure way. The collection of this data establishes the base for the definition of a set of KPIs that determine the quality of each phase. Specifically, a total of 32 indicators are defined: 5 in the breeding phase, 3 in the loading phase, 5 in the transport phase, and 19 in the slaughterhouse phase. These KPIs contribute to determining the quality of each phase and the poultry production chain as a whole.

Furthermore, the exploitation of these KPIs paves the way towards the identification of critical issues causing process inefficiencies. As a matter of fact, this exploitation is demonstrated with services which exist already such as PUMA, which supports the decision-making towards the holistic improvement of the poultry production chain.

Last but not least, the PCM platform is based on open-source components and open standards, with views to make it reusable for other livestock production chains with minimum modifications that can be methodically approached and are expected to be of bounded complexity.

Future Work

The contribution presented in this article tries to enhance the poultry production chain phases as well as the chain as a whole. However, it also opens up new paths for research and improvements of the existing solution.

On the one hand, the data collection of the poultry production chain in some phases could be automatized and better integrated with the PCM platform. Namely, the data generated in the slaughterhouse phase should ideally be collected in a seamless way. Instead of using an Excel file, an interface with a simple form could be developed, so that slaughterhouse operators could fill in the KPI information and send it to the PCM platform. This would contribute to simplifying the storage system design by having a unique centralized repository, instead of having a database and a file storage system.

On the other hand, the set of defined KPIs could be extended with additional ones that cover other relevant aspects affecting poultry welfare. For example, IoF environmental sensors installed in farms measure CO₂ and ammonia levels (which are demonstrated to affect poultry welfare [58–60]), therefore, they could be exploited. Furthermore, platform design of the PCM enables the addition of new KPIs with minimal development requirements.

Author Contributions: Conceptualization, I.E.-G.; Data curation, M.G.-O.; Formal analysis, S.F.; Investigation, I.E.-G. and M.G.-O.; Project administration, E.G.; Software, I.L.; Supervision, I.F.; Writing—original draft, I.E.-G. and M.G.-O.; Writing—review & editing, S.F., I.F. and E.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partly supported by the project Internet of Food 2020 which has received funding from the European Union Horizon 2020 research and innovation programme under grant agreement number 731884.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

HIS	Heat Stress Index
KPI	Key Performance Indicator
OCB	FIWARE Orion Context Broker
PCM	Poultry Chain Management

References

- Alexandratos, N.; Bruinsma, J. *World Agriculture Towards 2030/2050: The 2012 Revision*; ESA Working Papers 12-03; Food and Agriculture Organization of the United Nations: Rome, Italy, 2012. [CrossRef]
- Vanhonacker, F.; Verbeke, W.; Van Poucke, E.; Tuytens, F.A. Do citizens and farmers interpret the concept of farm animal welfare differently? *Livest. Sci.* **2008**, *116*, 126–136. [CrossRef]
- Harper, G.C.; Makatouni, A. Consumer perception of organic food production and farm animal welfare. *Br. Food J.* **2002**, *104*, 287–299. [CrossRef]
- María, G.A. Public perception of farm animal welfare in Spain. *Livest. Sci.* **2006**, *103*, 250–256. [CrossRef]
- Ingenbleek, P.T.; Immink, V.M. Consumer decision-making for animal-friendly products: Synthesis and implications. *Anim. Welf.* **2011**, *20*, 11–19.
- Hulzebosch, J. Effective heating systems for poultry houses. *World Poult.* **2005**, *22*, 212–216.
- Vieira, F.; Silva, I.; Barbosa Filho, J.; Vieira, A.; Broom, D. Preslaughter mortality of broilers in relation to lairage and season in a subtropical climate. *Poult. Sci.* **2011**, *90*, 2127–2133. [CrossRef]
- Commission to the European Parliament and the Council. The Impact of Genetic Selection on the Welfare of Chickens Kept for Meat Production COM/2016/0182 Final, 2016. Available online: <https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52016DC0182> (accessed on 10 March 2020).
- Ali, M.; Kang, G.H.; Joo, S.T.; others. A review: Influences of pre-slaughter stress on poultry meat quality. *Asian-Australas. J. Anim. Sci.* **2008**, *21*, 912–916. [CrossRef]
- Tsola, E.; Drosinos, E.; Zoiopoulos, P. Impact of poultry slaughter house modernisation and updating of food safety management systems on the microbiological quality and safety of products. *Food Control* **2008**, *19*, 423–431. [CrossRef]
- Alves, F.; Felix, G.; Almeida Paz, I.; Nääs, I.; Souza, G.; Caldara, F.; Garcia, R. Impact of exposure to cold on layer production. *Braz. J. Poult. Sci.* **2012**, *14*, 223–226. [CrossRef]
- Ferreira, V.; Francisco, N.; Belloni, M.; Aguirre, G.; Caldara, F.R.; Nääs, I.; Garcia, R.; Almeida Paz, I.; Polycarpo, G. Infrared thermography applied to the evaluation of metabolic heat loss of chicks fed with different energy densities. *Braz. J. Poult. Sci.* **2011**, *13*, 113–118. [CrossRef]
- Huang, P.; Lin, P.; Yan, S.; Xiao, M. Seasonal Broiler Growth Performance Prediction Based on Observational Study. *JCP* **2012**, *7*, 1895–1902. [CrossRef]
- De Moura, D.J.; do Vale, M.M.; de Alencar Nääs, I.; Rodrigues, L.H.A.; de Oliveira Medeiros, S.R. Estimating Poultry Production Mortality Exposed to Heat Wave Using Data Mining. In Proceedings of the Livestock Environment VIII, Iguassu Falls, Brazil, 31 August–4 September 2008; American Society of Agricultural and Biological Engineers: St. Joseph, MI, USA, 2009; p. 125. [CrossRef]
- Genç, L.; Portier, K.M. Sensible and latent heat productions from broilers in laboratory conditions. *Turk. J. Vet. Anim. Sci.* **2005**, *29*, 635–643.
- Xin, H.; Harmon, J.D. *Livestock Industry Facilities and Environment: Heat Stress Indices for Livestock*; Iowa State University Extension and Outreach: Ames, IA, USA, 1998.
- Esnaola-Gonzalez, I.; Fernandez, I.; García, E.; Ferreira, S.; Gomez, M.; Lázaro, I.; García, A. Towards Animal Welfare in Poultry Farms through Semantic Technologies. In Proceedings of the IoT Connected World & Semantic Interoperability Workshop (IoT-CWSI), Bilbao, Spain, 22–25 October 2019.
- Prescott, N.; Berry, P.; Haslam, S.; Tinker, D. Catching and crating turkeys: Effects on carcass damage, heart rate, and other welfare parameters. *J. Appl. Poult. Res.* **2000**, *9*, 424–432. [CrossRef]
- Knierim, U.; Gocke, A. Effect of catching broilers by hand or machine on rates of injuries and dead-on-arrivals. *Anim. Welf.* **2003**, *12*, 63–73.
- Nijdam, E.; Delezie, E.; Lambooi, E.; Nabuurs, M.; Decuypere, E.; Stegeman, J. Comparison of bruises and mortality, stress parameters, and meat quality in manually and mechanically caught broilers. *Poult. Sci.* **2005**, *84*, 467–474. [CrossRef] [PubMed]
- Weeks, C.A.; Nicol, C. Poultry handling and transport. In *Livestock Handling and Transport*; CABI Book: Oxfordshire, UK, 2000; pp. 363–384.
- Langkabel, N.; Baumann, M.P.; Feiler, A.; Sanguankiat, A.; Fries, R. Influence of two catching methods on the occurrence of lesions in broilers. *Poult. Sci.* **2015**, *94*, 1735–1741. [CrossRef]
- Hemsworth, P.H. Human–animal interactions in livestock production. *Appl. Anim. Behav. Sci.* **2003**, *81*, 185–198. [CrossRef]

24. Broom, D.M. The effects of land transport on animal welfare. *Rev. Sci. Tech.* **2005**, *24*, 683–691. [[CrossRef](#)]
25. Hester, P. Impact of science and management on the welfare of egg laying strains of hens. *Poult. Sci.* **2005**, *84*, 687–696. [[CrossRef](#)]
26. EU Scientific Committee on Animal Health and Animal Welfare. The Welfare of Chickens Kept for Meat Production (Broilers), 2016. Available online: https://ec.europa.eu/food/sites/food/files/safety/docs/sci-com_scah_out39_en.pdf (accessed on 10 March 2020).
27. Warriss, P.; Pagazaurtundua, A.; Brown, S. Relationship between maximum daily temperature and mortality of broiler chickens during transport and lairage. *Br. Poult. Sci.* **2005**, *46*, 647–651. [[CrossRef](#)]
28. Luthra, K. Evaluating Thermal Comfort of Broiler Chickens during Transportation Using Heat Index and Simulated Electronic Chickens. Master's Thesis, University of Arkansas, Fayetteville, AR, USA, 2017.
29. Petracci, M.; Bianchi, M.; Cavani, C.; Gaspari, P.; Lavazza, A. Preslaughter mortality in broiler chickens, turkeys, and spent hens under commercial slaughtering. *Poult. Sci.* **2006**, *85*, 1660–1664. [[CrossRef](#)] [[PubMed](#)]
30. Caffrey, N.; Dohoo, I.; Cockram, M. Factors affecting mortality risk during transportation of broiler chickens for slaughter in Atlantic Canada. *Prev. Vet. Med.* **2017**, *147*, 199–208. [[CrossRef](#)] [[PubMed](#)]
31. Yılmaz, A.; Arıkan, M.S.; Akin, A.C.; Kuyulu, Ç.Y.K.; Güloğlu, S.C.; Sakarya, E. Economic losses due to live weight shrinkage and mortality during the broiler transport. *Ankara Üniv. Vet. Fak. Derg.* **2014**, *61*, 205–210. [[CrossRef](#)]
32. Bianchi, M.; Petracci, M.; Cavani, C. Effects of transport and lairage on mortality, liveweight loss and carcass quality in broiler chickens. *Ital. J. Anim. Sci.* **2005**, *4*, 516–518. [[CrossRef](#)]
33. Vecerek, V.; Grbalova, S.; Voslarova, E.; Janackova, B.; Malena, M. Effects of travel distance and the season of the year on death rates of broilers transported to poultry processing plants. *Poult. Sci.* **2006**, *85*, 1881–1884. [[CrossRef](#)] [[PubMed](#)]
34. MacCaluim, J.; Abeyasinghe, S.; White, R.; Wathes, C. A continuous-choice assessment of the domestic fowl's aversion to concurrent transport stressors. *Anim. Welf.* **2003**, *12*, 95–107.
35. Randall, J.; Duggan, J.; Alami, M.; White, R. Frequency weightings for the aversion of broiler chickens to horizontal and vertical vibration. *J. Agric. Eng. Res.* **1997**, *68*, 387–397. [[CrossRef](#)]
36. Carlisle, A. Physiological responses of broiler chickens to the vibrations experienced during road transportation. *Br. Poult. Sci.* **1998**, *39*, 48–49. [[CrossRef](#)]
37. Yogaratnam, V. Analysis of the causes of high rates of carcase rejection at a poultry processing plant. *Vet. Rec.* **1995**, *137*, 215–217. [[CrossRef](#)]
38. Fallavena, L.C.; Moraes, H.L.; Salle, C.T.; Da Silva, A.B.; Vargas, R.S.; Do Nascimento, V.P.; Canal, C.W. Diagnosis of skin lesions in condemned or downgraded broiler carcasses—A microscopic and macroscopic study. *Avian Pathol.* **2000**, *29*, 557–562. [[CrossRef](#)]
39. Santana, Â.P.; Murata, L.S.; Freitas, C.G.d.; Delphino, M.K.; Pimentel, C.M. Causes of condemnation of carcasses from poultry in slaughterhouses located in State of Goiás, Brazil. *Ciência Rural* **2008**, *38*, 2587–2592. [[CrossRef](#)]
40. European Food Safety Authority. Technical assistance to the Commission (Article 31 of Regulation (EC) No 178/2002) for the preparation of a data collection system of welfare indicators in EU broilers' slaughterhouses. *EFSA J.* **2013**, *11*, 3299. [[CrossRef](#)]
41. EFSA Panel on Animal Health and Welfare (AHAW). Statement on the use of animal-based measures to assess the welfare of animals. *EFSA J.* **2012**, *10*, 2767. [[CrossRef](#)]
42. Murad, M.; Yahya, K.M.; Hassan, G.M. Web based poultry farm monitoring system using wireless sensor network. In Proceedings of the 7th International Conference on Frontiers of Information Technology, Abbottabad, Pakistan, 16–18 December 2009; ACM: New York, NY, USA; p. 7. [[CrossRef](#)]
43. So-In, C.; Poolsanguan, S.; Poonriboon, C.; Rujirakul, K.; Phasuk, Y.; Haitook, T. Smart mobile poultry farming systems in Tmote sky WSNs. *Int. J. Digit. Content Technol. Its Appl.* **2013**, *7*, 508.
44. Mahale, R.B.; Sonavane, S. Smart Poultry Farm Monitoring Using IOT and Wireless Sensor Networks. *Int. J. Adv. Res. Comput. Sci.* **2016**, *7*, 187–190.
45. Manshor, N.; Rahiman, A.R.A.; Yazed, M.K. IoT Based Poultry House Monitoring. In Proceeding of the 2019 2nd International Conference on Communication Engineering and Technology (ICCET), Nagoya, Japan, 12–15 April 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 72–75. [[CrossRef](#)]

46. Freeman, B. The relationship between oxygen consumption, body temperature and surface area in the hatching and young chick. *Br. Poult. Sci.* **1965**, *6*, 67–72. [[CrossRef](#)]
47. Jurkschat, M.; Burmeister, A.; Nichelmann, M. The development of thermoregulation in white beltville turkeys (*Meleagris gallopavo*) between day 10 and 50. *J. Therm. Biol.* **1989**, *14*, 83–86. [[CrossRef](#)]
48. Gerrits, A.; De Koning, K.; Migchels, A. Catching broilers. *Poultry* **1985**, *1*, 20–23.
49. Bayliss, P.; Hinton, M. Transportation of broilers with special reference to mortality rates. *Appl. Anim. Behav. Sci.* **1990**, *28*, 93–118. [[CrossRef](#)]
50. Berry, P.; Kettlewell, P.; Moran, P. The AFRC mark I experimental broiler harvester. *J. Agric. Eng. Res.* **1990**, *47*, 153–163. [[CrossRef](#)]
51. Chauvin, C.; Hillion, S.; Balaine, L.; Michel, V.; Peraste, J.; Petetin, I.; Lupo, C.; Le Bouquin, S. Factors associated with mortality of broilers during transport to slaughterhouse. *Animal* **2011**, *5*, 287–293. [[CrossRef](#)] [[PubMed](#)]
52. Council of the European Union. Council Regulation (EC) No 1/2005 of 22 December 2004 on the protection of animals during transport and related operations and amending Directives 64/432/EEC and 93/119/EC and Regulation (EC) No 1255/97, 2004. Available online: <https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX%3A32005R0001> (accessed on 10 March 2020).
53. Schwartzkopf-Genswein, K.; Faucitano, L.; Dadgar, S.; Shand, P.; González, L.; Crowe, T. Road transport of cattle, swine and poultry in North America and its impact on animal welfare, carcass and meat quality: A review. *Meat Sci.* **2012**, *92*, 227–243. [[CrossRef](#)] [[PubMed](#)]
54. Consortium of the Animal Transport Guides Project (2017). Guide to Good Practices for the Transport of Poultry, 2017. Available online: <http://animaltransportguides.eu/wp-content/uploads/2016/05/EN-Guides-Poultry-final.pdf> (accessed on 10 March 2020).
55. Escudero-Gilete, M.; González-Miret, M.; Temprano, R.M.; Heredia, F. Application of a multivariate concentric method system for the location of *Listeria monocytogenes* in a poultry slaughterhouse. *Food Control* **2007**, *18*, 69–75. [[CrossRef](#)]
56. Levinson, F.J.; Hicks, K.M.; Rogers, B.L.; Schaetzel, T.; Troy, L.M.; Young, C. *Monitoring and Evaluation: A Guidebook for Nutrition Project Managers in Developing Countries*; Human Development Network, The World Bank: Boston, MA, USA, 1999.
57. Brakel, J. Smoothed Z-Score Algorithm, 2016. Available online: <http://stackoverflow.com/questions/22583391/peak-signal-detection-in-realtime-timeseries-data> (accessed on 10 March 2020).
58. Kristensen, H.; Wathes, C. Ammonia and poultry welfare: A review. *World's Poult. Sci. J.* **2000**, *56*, 235–245. [[CrossRef](#)]
59. Yahav, S. Ammonia affects performance and thermoregulation of male broiler chickens. *Anim. Res.* **2004**, *53*, 289–293. [[CrossRef](#)]
60. Lin, H.; Jiao, H.; Buyse, J.; Decuyper, E. Strategies for preventing heat stress in poultry. *World's Poult. Sci. J.* **2006**, *62*, 71–86. [[CrossRef](#)]



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).

k-Nearest Patterns for Electrical Demand Forecasting in Residential and Small Commercial Buildings

- Title: k-Nearest Patterns for Electrical Demand Forecasting in Residential and Small Commercial Buildings
- Authors: Meritxell Gómez-Omella, Iker Esnaola-Gonzalez, Susana Ferreiro, Basilio Sierra
- Journal: Energy and Buildings
- Year: 2021
- Quartile: Q1
- DOI: [10.1016/j.enbuild.2021.111396](https://doi.org/10.1016/j.enbuild.2021.111396)

k-Nearest Patterns for Electrical Demand Forecasting in Residential and Small Commercial Buildings

Meritxell Gómez-Omella^{a,b,*}, Iker Esnaola-Gonzalez^a, Susana Ferreiro^a,
Basilio Sierra^b

^a*TEKNIKER, Basque Research and Technology Alliance (BRTA),
C/Iñaki Goenaga, 5, 20600 Eibar, Spain*

^b*Faculty of Informatics, University on the Basque Country (UPV/EHU),
Paseo Manuel Lardizabal, 1, 20018 Donostia-San Sebastian, Spain*

Abstract

This work presents a case study of Big Data and Machine Learning whose objective is to improve energy Demand Response (DR) programs by providing accurate energy demand forecasts. Given the present state of the art, this research work introduces the proposed methodology for Time Series Forecasting based on two variants of the K-neighbours method (KNN): K-Nearest Features in Time Series (KNFTS) and K-Nearest Patterns in Time Series (KNPTS) algorithms. These algorithms are valuable in this field since only a historical data set consisting the time and energy consumption variables are used to find similar patterns of electricity consumption and then make future forecasts. Furthermore, the proposal to use elastic similarity measures such as DTW and EDR shows to have advantages over the use of common error metrics. It has been proven on data from 122 houses and small commercial buildings located on the island of Lanzarote that the KNPTS achieves minor errors in 89% of cases. Therefore, it shows that the KNPTS algorithm provides a good accuracy, more efficient in prediction than the KNFTS algorithm, to improve DR programs.

Keywords: Time Series Forecasting, Energy Forecasting, Demand Response, Smart Buildings, Sustainable Cities

*Corresponding author

Email address: meritxell.gomez@tekniker.es (Meritxell Gómez-Omella^{id})

1. Introduction

Ensuring an efficient usage of energy within buildings is key to ensure a sustainable development and one of the major concerns among governments, scientists and researchers since, according to the UNEP (United Nations Environment Programme), the building sector consumes about 40% of global energy. Furthermore, the energy required for maintaining indoor comfort conditions and daily maintenance tasks entails about 90% of the total energy consumed throughout the building life cycle [1]. These activities' consumption patterns show a high variance as they are strongly influenced by occupants' behaviour, so understanding and changing this behaviour is considered an effective way to improve energy efficiency in buildings [2]. This statement is supported by a study which showed that the EU residential sector's potential electricity savings can reach 48% [3].

The increasing penetration of ubiquitous sensing enabled by Wireless Sensor Network (WSN) technologies and the proliferation of IoT (Internet of Things) devices facilitates the monitoring of building occupants' energy consumption patterns. Furthermore, this information can later be exploited for forecasting future inefficient energy usages and avoid them by suggesting more adequate DSM (Demand Side Management) actions such as load curtailment (i.e. a reduction of electricity usage) or reallocation (i.e. a shift of energy usage to other off-peak periods). In combination with DSM activities, the Demand Response (DR) can influence customers' use of electricity in effective ways. DR can be understood as the set of technologies or programs that concentrate on shifting energy use [4], and although traditionally the DR programs' implementation has been limited to large industrial buildings due to their high energy demand, residential and small commercial buildings are specially promising due to their potential to reduce demand peaks [5].

The effectiveness of DR programs heavily depends the management of collectively shared energy assets such as renewable energy generation as well as variable pricing tariffs and specific demand flexibility constraints. This is certainly a complex problem, where the accurate prediction of the energy to be consumed plays a key role [6].

Decisions are made based on the values of future electricity demand that are expected to occur. These estimates are commonly generated as a result of Machine Learning (ML) algorithms trained with historical values of different input variables. These variables called features can be the past electricity consumption, the ambient temperature or the occupation of the

building, among others that have a relationship with the electricity consumption. These values can be measured and stored in some cases, although in others they must be estimated, so that an accumulation of error can be generated in the demand forecasts. The alternative, little evaluated so far, is to use the information provided by the electrical consumption history to train the forecasting models. In this article, focus is placed on this issue, and the development of forecasting models for the electric demand for different time horizons in residential and small commercial buildings is addressed. Furthermore, the effectiveness of forecasting models has been measured mostly using classical metrics that do not allow the comparison of data at different instants of time. However, elastic measures of similarity in time series are a promising proposal in the study of the performance of prediction algorithms for ordered data.

The rest of the article is organised as follows. Section 2 summarises the methodologies used in the literature to make electrical consumption forecasting in different scenarios and using different data sets. Section 3 shows the proposed methodology for making forecasts in time series, including a definition of each of the steps to be considered. Then, the experimental setup is described in Section 4 and in Section 5 the obtained results are discussed. Finally, Section 6 summarises the conclusions of this work and proposes some potential future work.

2. Related work

There is a large number of studies in the literature in relation to the analysis of the energy consumption forecasting, where computational intelligence techniques for the prediction of time series have become competitive prediction methods, compared to traditional statistical models such as smoothing Exponential (ETS) or Autoregressive Integrated Moving Average (ARIMA), as indicated by Hewamalage et al. [7].

In the work presented by Mat Daut et.al [8], a review of the forecasting methods of the electrical energy consumption of buildings is provided. It includes conventional and artificial intelligence (AI) methods. Each of these studies follows a different approach and varies according to the typology of the elements or characteristics of the case study. Some of these research articles are mentioned below. They have been categorised based on the type of building, the prediction time interval, the learning method, the validation metric, and the type of information considered for the analysis (Table 1).

Table 1: Summary of electrical consumption forecasting problems in the literature

Forecast Problem Characteristics		References
Type of building	Residential	[9], [10], [11]
	Non-residential	[12], [13], [14], [15], [16], [17]
Time interval	Per Hour	[9], [12], [14], [18]
	Per Day(s)	[7], [15], [16], [17]
	Per Year	[10], [11], [13],
Method	Recurrent Neural Networks	[7]
	Neural Networks	[9], [12], [13], [14], [16], [17]
	Back Propagation Neural Network	[10]
	RBFNN and GRNN	[11]
	LR	[17]
	KNN, LR and RF	[18]
	SVM	[8], [16], [19]
	Time Series Models (ARIMA, Exponential Smoothing,...)	[15], [20]
	SMAPE and MASE	[7]
	RE	[9], [19]
Metric	RMSE and MRE	[11]
	MAPE	[12], [13], [15], [16], [18]
	NMBE and CVRMSE	[17]
	Energy Mean Error (EME)	[12]
	Historical Data	Electrical Consumption
Weather	Solar Radiation and Air Temperature	[9], [10], [11], [12], [16], [18]
	Temperature	[9], [11], [14], [16]
	Humidity Ratio and Wind Speed	[12], [15], [17]
	Temperature, humidity, wet-bulb, dew-bulb	[14], [17]
Indoor Data	Occupancy	[18]
	The Living Standard, Social Development, Urban Construction and Development, the Natural Condition	[10], [17]
Others	Thermal Index, Heat Transfer and Size of Building	[10]
	Calendar and Type of Day	[11]
	Electricity Price and number of consumers	[12], [16]
	Energy Performance Indicators	[13]
	Season	[15]
		[18]

Most of this work focuses on non-residential buildings such as universities [12], industries [13], libraries [14], restaurants [15], offices [16], schools [21] or campus [17]. These facilities often present electrical consumption curves with estimable patterns due to the cyclical behavior in their use. However, there is also some work in relation to the forecasting of the consumption of residential buildings. This work normally proposes the estimation of housing consumption at the neighbourhood level. The work presented by Mihalakakou et al. [9], Yu et al. [10] and Li et al. [11] are some examples.

Regarding the time interval, there are a variety of works with different objectives that focus on forecasts of different lengths. There is work that focuses on one-hour ahead ([9] and [12]), one-day ahead ([16] and [20]) or more than one-day ahead ([15] and [17]) and one-year ahead ([10], [11] and [13]) forecasts.

The wide variety of the methods used in the work is diverse, but there is a tendency to use neural networks and their variants ([7], [9], [10], [11], [12], [16]). Some authors justified the effectiveness of neural networks by comparing the results with those obtained with a Linear Regression (LR) [17]. However, more traditional methods such as Autorregressive Integrated Moving Average (ARIMA) [15],[20] or Support Vector Machine (SVM) such as in Shine et al. [19] and Mat Daut et al. [8] are also applied. Furthermore, a comparison between Random Forest (RF), Linear Regression (LR) and K-Nearest Neighbours (KNN) is done in Johannesen et al. [18] and RF provides better results for the next 30 minute forecast and KNN offers relatively better results for the 24 hour forecast.

Most researchers prefer to use Mean Absolute Percentage Error (MAPE) as a reference to compare the performance of various methods in the field of electrical power consumption forecasting because it is dimensionless and allows the comparison of results from different sources ([12], [13], [15], [16], [18]). For the same reason, the Relative Error (RE) is also quite used in this field ([9], [19]). Other authors prefer to compare more than one metric in the same study. For instance, Symmetric Mean Average Percentage Error (SMAPE) and Mean Absolute Scaled Error (MASE) are used by Hewamalage et al. [7], Root Mean Square Error (RMSE) and Mean Relative Error (MRE) are compared by Li et al. [11] and Normalized Mean Bias Error (NMBE) and Coefficient of Variation of the Root Mean Square Error (CVRMSE) in [17]. As there is a metric to measure differences in electricity data, the Energy Mean Error is used in some cases [12].

Furthermore, most of the application-related work relies on the use of

historical consumption data ([9], [10], [11], [12], [16], [18]). But there is work that includes a few additional variables to the initial data set to make forecasts and obtain more accurate results. These variables are usually related to the weather. For instance, ambient air temperature or total solar ([9], [11], [12], [14], [16], [18], [17]). The occupation of the building is another input variable used to improve the performance of the models ([10], [17]). There are other types of information that are less used such as the living standards of residents, urban construction and development level, social development level or natural condition ([10], [11]), the calendar ([12], [16]), the price of the electricity or the number of consumers ([13]), energy performance indicators ([15]) or the season of the year ([18]).

Finally, in Gómez-Omella et al. [22] some machine learning algorithms (KNN, LR and SVM) were compared to the classical model ARIMA using only the temporal variable to make the predictions. The result of this research revealed that KNN algorithm reached the highest accuracy. Further research was done and in Gomez-Omella et al. [23] two variants of the KNN algorithm were presented, the KNFTS and the KNPTS. The authors focus their attention on two variants of the KNN algorithm after having carried out a selection process that concluded with a better performance of the KNN algorithm. One of the conclusions derived from this work was that the KNPTS variant turned out to have a higher forecast precision. In addition, given the impact of the COVID-19 pandemic on energy demand during the preparation of this work, it presents a retraining method for this model that reveals the improvement in the forecasting accuracy.

That said, the work presented below presents a more generic comparative evaluation to demonstrate that the results obtained by the KNPTS do not depend on the data of the building itself but can be extrapolated to other buildings. To that end, an experimentation has been carried out with a large amount of data obtained from different types of buildings located in Lanzarote island (Spain). That island is chosen for the study because it is an area with low thermal amplitude so that the climate does not have a direct influence on electricity consumption since most buildings are not air-conditioned. The amount of data has enough variability so that the effectiveness of the methods does not depend on the data used for model training. In addition, the work emphasises how to train and validate these models properly, using appropriate metrics, so that the results obtained are reliable and more accurate interpretation.

3. Forecasting Methodology for Time Series

A Time Series is a collection of values $Y = \{y_t\}$ captured in a certain order over time where $t = 1, \dots, T$ represents the time elapsed. These time slots are often equidistant. Data can be captured hourly, daily or annually for example, depending on the frequency. [24].

The interest in the analysis of time series lies in the estimation of the behaviour of the historical data, that is, to model the time series, and then, use that knowledge to forecast future values. Given a univariate time series $\{y_t\}$ where $t = 1, \dots, T$ the interest of this work is focused on the forecasting of future values $\{y_{T+h}\}$ where $h \in \{1, \dots, H\}$. The process of forecasting future values in a time series can be classified into single-step ahead or multi-step ahead according to whether the number of points that are estimates is $H = 1$ or $H > 1$.

This section presents the steps to be considered in a time series forecasting problem. These steps include the strategy selection to be used to make multi-step ahead time series forecasting, the choice of the most suitable cross-validation for time series, the model generation to estimate future values and the decision of the evaluation criteria to measure the model performance focused on time series data. Since the preprocessing and the cleansing of the data is considered as a previous task and is supposed particular for each problem, it is not include in this section.

3.1. Strategies for Multi-Step Ahead Time Series Forecasting

A multi-step ahead time series forecasting consists in estimating the next H values $\{y_{T+1}, \dots, y_{T+H}\}$ using the previous values $\{y_1, \dots, y_T\}$. In order to predict more than one value ($H > 1$) in time series, five strategies can be chosen [25].

- *Recursive Strategy.* This strategy trains a single model to forecast only one future value. Then, that model is recursively used to forecast all the H future points. So the output of the model in the first step is \hat{y}_{T+1} is used as a part of the input variables for forecasting the second value. This process is repeated recursively to estimate the H future values. This fact causes the model to be prone to the accumulation of errors.
- *Direct Strategy.* This strategy consists in training H separate models, one for each point to be predicted. A multi-step ahead forecast is done

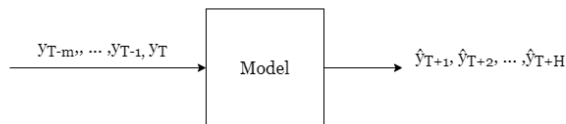


Figure 1: Scheme of the forecasts obtained using MIMO strategy in time series

by concatenating the H predictions, $\hat{y}_{T+1}, \hat{y}_{T+2}, \dots, \hat{y}_{T+H}$, obtained from each model.

- *DirRec Strategy.* The *DirRec Strategy* combines the *Direct* and *Recursive Strategies*. It learns H different models as in *Direct Strategy*, one for each value to be forecast and one step is done for each value. In each step, the output of the previous model is used as an input to train the current model, as in *Recursive Strategy*.
- *MIMO Strategy.* The *Multi-Input Multi-Output (MIMO) Strategy* executes one prediction after learning a single multiple-output model. It consists of a target vector of length equal to the number of points to be predicted H and a feature vector containing the m previous values. The H forecasts values are returned in one-step.
- *DIRMO Strategy* The *DIRMO Strategy* is a combination of *Direct* and *MIMO Strategies*. It forecasts the H future values in s blocks, where each block is forecast in a *MIMO* approach. Then, $\frac{H}{s}$ are trained in the *DIRMO Strategy*.

3.2. Cross-Validation in Time Series

Cross-validation is an important procedure in evaluating the effectiveness of predictive models [26]. It is used for the adequate estimation of the hyper-parameters and to ensure the generalisation of the model. The methodology of evaluation in forecasting when ML methods are used for time series considers the temporal dependence of the observations. The general idea is to prevent future data from being used in the prediction of previous data to respect the order in which the data are received. A detailed study on the effect of using the different approaches to cross validation in time series is found in [27]. Using the standard k -fold cross-validation, no temporal effects of the dependencies within the data can be identified [28].

The Cross-Validation in Time Series (CVTS) is inspired by this partition of the training set and the subsequent combination of the errors obtained. The main difference compared with the traditional k -fold cross-validation, is that the test set must always be later in time than the training set. CVTS can be classified into two types according to whether the training set starts on the same date in each partition or not. On the one hand, in the Forward Chaining CVTS (FCCVTS), an initial training set and an initial testing set are chosen. After evaluating the forecast model with these two sets, the test set is added to the training set to form the new training set. Another test set is taken after the new training set and the process is repeated. It ends when the entire original training set has been used. On the other hand, if the size of the training set is the same in all steps, it is called Moving Window CVTS (MWCVTS). In that case, the initial value of the training set in each iteration is different [29].

3.3. Theoretical Machine Learning Algorithms

ML consists in programming computers to optimise a performance criterion using example data or past experience [30]. A previous experimentation was done in order to compare different features engineering [31] and several ML models and it was concluded that the K-Nearest Neighbours (KNN) provided more accurate results than the rest. Therefore, the methodology proposed in this work is based on the KNN algorithm. The general idea is to examine the distance between the independent variables, then choose the K nearest observations and finally use a combination of their response values to estimate the next value of the output variable. In order to forecast future values using a ML algorithm, it is necessary to make some decisions including the multi-step ahead strategy and the selection of input variables. Particularly for the KNN, other decisions are required: the distance function to evaluate the similarity between instances, the value of K nearest neighbours and the way output points are combined [32].

Different distance functions can be used to define the similarity between instances, although the most commonly used one is the Euclidean distance [33]. A CVTS is implemented to evaluate different values of K , looking for the one that minimises the error. This error is a metric chosen to measure the difference between the estimated and actual values. The optimal value of the K parameter indicates the number of target values to be combined to estimate the future values. The arithmetic mean is the most used method to combine values with equal weights. Alternatively, the weighted average can be used,

where some target values contribute more than others to the estimated value. In this case, it is proposed that the weights are proportional to the distances between the chosen neighbours. Assuming that d_λ is the distance between the current observation and the λ -th nearest neighbour, $\omega_\lambda = 1/d_\lambda^2$ is defined to be its weight. The weights are normalised, dividing by their sum.

Two versions of the KNN algorithm to make time series forecasting are presented in this work. On the one hand, KNFTS which is an approach based on the estimation of a ML model where the features extracted from the time variable are used as exploratory variables. On the other hand, KNPTS which is a method based on the recognition of similar patterns in the time series.

3.3.1. *K-Nearest Features for Time Series (KNFTS)*

An important step prior to training any ML model is the choice of n exploratory variables of T values X_t^i , $i = 1, \dots, n$, $t = 1, \dots, T$ extracted from the time variable when the data are a univariate time series. Some date and time related features X_t^i used to calculate the distance between observations are described below.

- Date-related features are numerical values containing information about the day, month or year. These features also include the day of the week, the season of the year or any other information that can be extracted from the date and that can be related to the values that the response variable takes.
- Time-related features are extracted for the time stamp and are numerical variables like hour, minute, second or any smaller unit of time.

The nearest neighbours are obtained by comparing the Euclidean distance between X_t^i variables in each observation. Some of these variables have a cyclical meaning that is not reflected in the calculation of distances. For example, the Euclidean distance between December and January (represented by the cardinal numbers 12 and 1 respectively) is 11 but they are two consecutive months. To avoid this fact, a trigonometric transformation v is done in this kind of variables¹. Once the periodicity of the variable is found (in this case $P = 12$), the Equation 1 is applied to each value X_t^i , $t = 1, \dots, T$. Two variables are obtained from each of the transformations made (one from

¹<https://www.avanwyk.com/encoding-cyclical-features-for-deep-learning/>

the sine and the other from the cosine). The same situation happens with the hour variable, therefore, same procedure is applied but with a periodicity of $P = 24$.

$$\begin{aligned} v : [0, P] &\longrightarrow [-1, 1] \\ X_t^i &\rightarrow v(X_t^i) = \left(\sin\left(\frac{2\pi X_t^i}{P}\right), \cos\left(\frac{2\pi X_t^i}{P}\right) \right) \end{aligned} \quad (1)$$

The Euclidean distance between $(X_{T+1}^1, \dots, X_{T+1}^n)$ and (X_t^1, \dots, X_t^n) , for all $t = 1, \dots, T$ is calculated to choose the K nearest neighbours. The next value Y_{T+1} is obtained as a weighted combination of the target values of the K nearest neighbour.

$$Y_{T+1} = \frac{\sum_{\lambda=1}^K \omega_\lambda Y^\lambda}{\sum_{\lambda=1}^K \omega_\lambda} \quad (2)$$

where Y^λ is the value of the target variable corresponding on the λ nearest neighbour. This argument applies to all $h = 1, \dots, H$ horizon values to be forecast. The general idea of this method is that in two similar moments of time in terms of date and time, the value of the response variable should be similar.

3.3.2. *K-Nearest Patterns in Time Series (KNPTS)*

In KNPTS, the nearest neighbours are defined as the most similar subsets of data in the time series. Given the window size $W \in \mathbb{N}$, the last values of the output variable (y_{T-W}, \dots, y_T) are used as a reference pattern in the training process. This subset of data is compared with all the other subsets of length m in the series, that is, $Y^j = (y_j, \dots, y_{j+W})$, for all $j = 1, \dots, T - W - H$. The similarity measure used is the Euclidean distance. Therefore, the K nearest neighbours patterns are the K subsets with the lowest Euclidean distance [34].

Figure 2 shows an example of nearest-neighbour multiple-step forecasting using MIMO strategy. It represents a simple case in which $W = 5$ previous values are used as reference pattern to find the most similar subset of data in the time series. Therefore, in that case, $K = 1$ neighbour is used to forecast the $H = 3$ future values of the time series.

The next value Y_{T+1} is calculated by the weighted average of the next

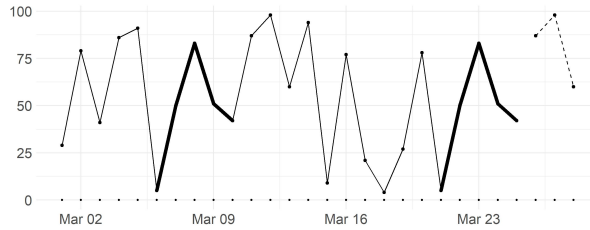


Figure 2: Nearest-neighbour three-step-ahead forecasts. A window of length five is selected and one neighbour is used to estimate the next three values.

values of each nearest neighbour $\lambda = 1, \dots, K$.

$$Y_{T+1} = \frac{\sum_{\lambda=1}^K \omega_{\lambda} Y_{\lambda+W+1}^{\lambda}}{\sum_{\lambda=1}^K \omega_{\lambda}} \quad (3)$$

This argument is applied to all $h = 1, \dots, H$ horizon values to be forecast. The rationale behind this method is that, in two different periods of time with similar values, the following value should be similar as well.

This approach can be posed differently using a lagged set of data and the KNFTS. In that case, the set of explanatory variables X^i should be the M lagged values of Y , including the current value. Following the example of Figure 2, Table 2 presents the data structure. On the left side, data corresponding to the input variables are shown, and on the right, the output variables. In the first row, the reference pattern is highlighted in blue and the estimated values in yellow. The method would establish that the most similar pattern is $\{Y_{T-17}, Y_{T-16}, Y_{T-15}, Y_{T-14}, Y_{T-13}\}$ and its corresponding future values $\{Y_{T-12}, Y_{T-11}, Y_{T-10}\}$ would be used to estimate $\{\hat{Y}_{T+1}, \hat{Y}_{T+2}, \hat{Y}_{T+3}\}$.

3.4. Evaluation Criteria of the Forecast in Time Series

Evaluation measures are functions that are used to quantify the effectiveness of the forecast model. The problem of forecasting future values in a time series is a particular case of forecasting problems. It has its own characteristics derived from the dependency between the values of the series and the temporal variable. The distances between time series are classified into two groups, lockstep and elastic, depending on whether the values are compared one by one or not. Depending on the frequency with which the data are received, lockstep metrics may quantify errors that, despite existing, may not

Table 2: Structure of the data sets to train a KNPTS with a window of length five and make three-step-ahead forecasting

X^5	X^4	X^3	X^2	X^1	h=1	h=2	h=3
Y_{T-4}	Y_{T-3}	Y_{T-2}	Y_{T-1}	Y_T	\hat{Y}_{T+1}	\hat{Y}_{T+2}	\hat{Y}_{T+3}
Y_{T-7}	Y_{T-6}	Y_{T-5}	Y_{T-4}	Y_{T-3}	Y_{T-2}	Y_{T-1}	Y_T
Y_{T-8}	Y_{T-7}	Y_{T-6}	Y_{T-5}	Y_{T-4}	Y_{T-3}	Y_{T-2}	Y_{T-1}
Y_{T-9}	Y_{T-8}	Y_{T-7}	Y_{T-6}	Y_{T-5}	Y_{T-4}	Y_{T-3}	Y_{T-2}
...
Y_{T-17}	Y_{T-16}	Y_{T-15}	Y_{T-14}	Y_{T-13}	Y_{T-12}	Y_{T-11}	Y_{T-10}
...
Y_{T-22}	Y_{T-21}	Y_{T-20}	Y_{T-19}	Y_{T-18}	Y_{T-17}	Y_{T-16}	Y_{T-15}

have such a significant effect. For this reason, elastic measures are also used to evaluate the forecasts. Let $\{y_{T+1}, y_{T+2}, \dots, y_{T+H}\}$ be the future values of a given time series $\{y_1, \dots, y_T\}$ that we try to fit and $\{\hat{y}_{T+1}, \hat{y}_{T+2}, \dots, \hat{y}_{T+H}\}$ the forecast values obtained with a certain method. In order to simplify the notation, we denote real observations as $Y_H = \{y_{T+1}, y_{T+2}, \dots, y_{T+H}\}$ and forecast values as $\hat{Y}_H = \{\hat{y}_{T+1}, \hat{y}_{T+2}, \dots, \hat{y}_{T+H}\}$.

- Lockstep measures consists in the point-to-point comparison between Y_H and \hat{Y}_H . There are many metric used to measure the effectiveness of a forecasting regression model that are based on these Lp distances, specially on the Euclidean distance. Root Mean Square Error (RMSE) is the most common error metric. RMSE is an indicator of the mean deviation of the forecasts against the real series. As it can be seen in Equation 4 represents the square root of the second sample moment of the differences between forecast values and observed values or the quadratic mean of these Euclidean distances.

$$RMSE(Y_H, \hat{Y}_H) = \sqrt{\frac{\sum_{t=h}^H (y_{t+h} - \hat{y}_{t+h})^2}{H}} \quad (4)$$

- Elastic measures consist of different techniques to measure similarity in time series that allow to compare multiple points and series of different length.

Dynamic Time Warping (DTW) algorithm is used to compare the similarity or calculate the distance between two time series and minimises

the effects of shifting and distortion in time [35]. A dynamic programming approach is used to align the series and allows to compare series of different length [36]. The lower the DTW value, the more similar the time series are.

Edit Distance for Real Sequences (EDR) is an adaptation of the Edit Distance (ED) measure commonly used to measure the cost to convert one string to the other using only insertions, deletions and replacements. In EDR, given a positive threshold ϵ , the distance between two points is reduced to 0 or 1 in order to adapt it to numerical values [37]. Two values are considered equals if the Euclidean distance between them is less than ϵ . The lower the EDR value, the more similar the time series are.

4. Experimental Development

In order to explain the experimental part, this section is divided into three parts. First, data acquired for the demonstrations are presented. Then the data processing is explained in detail. Finally, the approach that was given to the experiment and all the steps that were followed to obtain the results are explained.

4.1. Experimental Data

Electric consumption from buildings located in the island of Lanzarote (Spain) have been used for the experimentation. Data of hourly electric consumption have been provided by the Spanish utility Fenie Energia². The buildings have a label that identifies them as dwellings or small commercial, but it is not possible to know the area, population, number of rooms, opening hours, lifestyle of the users, etc. The date on which the data collection begins varies for each data set, although the majority are from the second half of 2018. The data export date was 2020/03/15 so all time series end on that date. Hourly electrical consumption data (Wh) are collected for each of the buildings. Therefore, 364 data sets with records of very different duration from 1 week (168 records) to more than 2 years (18253 records) are available. Due to the privacy of the data, the used data cannot be published here.

²<https://www.fenieenergia.es/>

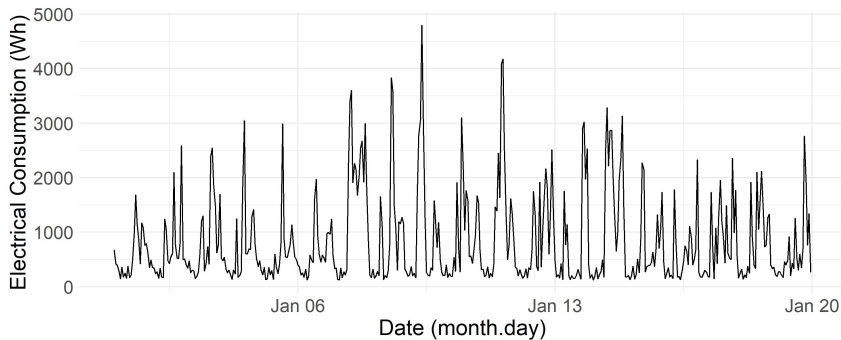


Figure 3: A snippet of the electric consumption data set for a participant building.

However, a snippet of this data is shown in Figure 3. Visually no pattern can be established for these elevated data, but the forecasting model should be able to predict those values as best as possible.

In a previous processing, a set of statistical indicators are generated to characterise each time series. These features extracted from the electrical consumption allow filtering the data sets. The goal of the preprocessing task filter buildings has been to work with complete data with rich variability. For that reason, long sets (buildings with more than one year of data), complete (buildings with less than three months of missing values) and informative (buildings with less than 80% of 0 values) are required. This decision was made to ensure the high quality of data and a properly amount of data to train the models. The filtering process is summarised in the diagram shown in Figure 4. Consequently, the experimental task was done using data from the 122 remaining buildings, where 54 are small business and 68 are dwellings. Therefore, the data sets contain a more homogeneous number of records, from 8424 to 14136 observations. Equivalently, the smallest set contains one year of data and the longest is 1 year and 7 months.

4.2. Data processing

Two forecasting models are trained for each building to be analysed, one with the KNFTS and the other with the KNPTS . All the steps involved in this process have been done using the statistical software R.

A set of date-related and time-related features are generated to train KNFTS model as explained in Section 3.3.1. The objective of the feature

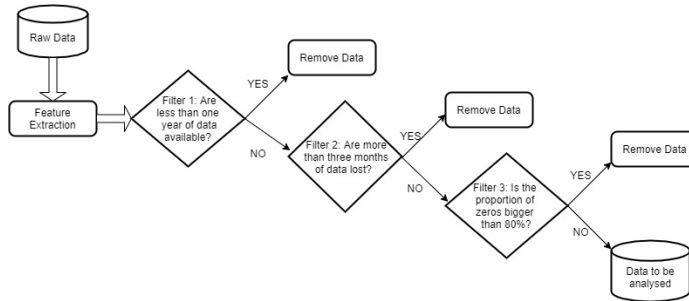


Figure 4: Filtering process to obtain valuable data to be analysed

engineering process is the creation of variables that have a strong relationship with the output variable and provide more information about electricity consumption. This process does not consist in introducing the maximum possible number of variables but rather in identifying the most appropriate ones according to the context of the problem. Lanzarote is an area with low thermal amplitude, therefore weather variables are not chosen as possible inputs of the forecasting models.

As the experimentation data are hourly, the only time-related feature that is extracted is *Hour*. It is known that electrical consumption varies according to the time of day and the consumption is lower at nights. This variable will collect the information that has just been mentioned and will help to identify the daily cycles of data. Furthermore, date related features are also created. First, *Month* information is also used as exploratory variable. Depending on the month of the year in which the electricity consumption data are measured, differences can be seen. This fact is due to the use of some electrical appliances in certain months of the year, such as air conditioning. This variable is able to collect the annual cycles of data. The *Day* variable is decided not to be introduced as input to the model because a direct change in electricity consumption is not recognised according to the day of the month in which data are collected. That is, data are not believed to have a monthly cyclical behaviour. The possibility of a cyclical behaviour for weeks is recognised since a difference in consumption is appreciated on weekends. For that reason, a variable called *Weekday* is generated. It takes values from 0 to 6 to indicate the days of the week from Monday to Sunday. *Season* variable is created indicating the season of the year where 0 is winter, 1 spring, 2

summer and 3 fall. Finally, the *WorkingDay* variable is generated, which takes the value 0 to indicate that the day is a working day and 1 if the day is a holiday. Its purpose is to capture the holidays in which customers usually go on vacation and change their consumption.

The *Hour* and *Month* variables are considered variables with cyclical behaviour. Equation 1 is applied in order to obtain two new variables from each transformation. In summary, the variables used in the training step of KNFTS model are: *sin.Hour*, *cos.Hour*, *sin.Month*, *cos.Month*, *Season*, *Weekday*, and *WorkingDay*.

Regarding the KNPTS method, it is no necessary to create supporting features. The information to be used is already available because the distances are calculated in the consumption data set. The data set is structured appropriately so that the reference pattern is the last 24 values of the time series. That is, all the patterns of the training series are compared with the last day of consumption data. Therefore, following the notation explained in Section 3.3.2, the window size is $W = 24$ h.

4.3. Experimental Setup

The DTW, EDR and RMSE measures are used to evaluate the forecasting effectiveness of the models proposed in this work. RMSE is the classic error metric used to quantify the mean deviation of the forecasts versus the actual observations. However, it is convenient to bear in mind that this metric compares each forecast value with the observed value in that same position. In time series this means that it compares two points at the same instant of time. In addition to RMSE, DTW and EDR are computed. DTW and EDR are two measures that allow comparing values in different positions of time. The error made in the forecast is softened if the forecast value can be similar to the previous or later observed value in time. Hourly electricity consumption data show peaks at some time of the day, as mentioned before. Since data are hourly, a comparison with a window of size 1 is allowed. Therefore, the fact of estimating a consumption value with a variation of one hour will not be considered a relevant failure of the forecasting model. Regarding the EDR, a threshold of $\epsilon = 40$ Wh is established in the Formula ???. That decision has been supported by two aspects. On the one hand, the common and general electrical consumption of certain appliance and electrical devices on a

building³. This threshold is designed to consider error in the prediction of the consumption not collected due to the use of energy-efficient appliances, but not to consider the energy consumed from other less expensive devices such as light bulbs. As the power of an incandescent bulb ranges between 40W and 100W, $\epsilon = 40Wh$ was proposed to be the threshold. On the other hand, the distribution of the electrical consumption data in the experimental case of this work and the general objective of the project involved. The maximum value ranges from 2000Wh to 3500 Wh, the minimum value ranges from 0 to 50Wh, and the mean value from 250 to 350Wh for the original time series. According to the variability of electricity consumption within the same house, a difference of 40Wh between the real value and the predicted one can be considered insignificant and so, it was treated as noise. DTW and EDR are performance measures of the complete forecast, that is, they do not give an estimate of the average error committed per point, but the accumulated error during the entire estimated series. Instead, the RMSE is a measure of the average error per point forecast.

The evaluation of the effectiveness of both methods is made by one-day and one-week ahead forecasting. Five different dates from 2020-01-01 and 2020-03-15 are generated in a random way. These dates are used as initial values for the test set. Each test is named with a label from P1 to P10 as it can be seen in Table 3. The results are analysed separately depending on the number of points to be forecast. Therefore, two different experiments are defined. The tests made with both forecasting methods to predict one day ahead are considered as the first experiment (*E1*). The second experiment (*E2*) consists in making predictions of the values one week ahead. It is natural that the magnitude of the errors is greater the more points are predicted. This is the reason to divide the comparison. Consequently, evaluation criteria are computed as the average of the five values obtained for each of the experiments. On the one hand, the results of the experiment *E1* (tests from P1 to P5) are discussed in Section 5.1. On the other hand, the results of the experiment *E2* (tests from P6 to P10) are discussed in Section 5.2.

Regarding the Forward Chaining Cross Validation for Time Series, the number of repetitions in each iteration was 10, that is, 10 different partitions are done in the train set. In both methods, the value of the parameter K indicating the number of neighbours is tested from 1 to 5 in each iteration. The

³<https://unboundsolar.com/solar-information/power-table>

Table 3: Random partitions of train set and test set

Name	Initial Date	Final Date	Ahead	N ^o points	Day
P1	2020-01-01 00:00:00	2020-01-01 23:00:00	1 day	24	Wednesday
P2	2020-01-18 00:00:00	2020-01-18 23:00:00	1 day	24	Sunday
P3	2020-02-01 00:00:00	2020-02-01 23:00:00	1 day	24	Saturday
P4	2020-02-10 00:00:00	2020-02-10 23:00:00	1 day	24	Monday
P5	2020-03-05 00:00:00	2020-03-05 23:00:00	1 day	24	Thursday
P6	2020-01-01 00:00:00	2020-01-07 23:00:00	1 week	168	Wednesday
P7	2020-01-18 00:00:00	2020-01-24 23:00:00	1 week	168	Sunday
P8	2020-02-01 00:00:00	2020-02-07 23:00:00	1 week	168	Saturday
P9	2020-02-10 00:00:00	2020-02-16 23:00:00	1 week	168	Monday
P10	2020-03-05 00:00:00	2020-03-11 23:00:00	1 week	168	Thursday

reason for choosing this kind of cross validation is the number of historical data available and its cyclical behaviour. There are about one year of data from each building so older data are not expected to negatively influence the forecast. In general, no abrupt changes in the distribution of the data are expected so significant as to discard the first available data. Furthermore, it is not expected that many neighbours will be necessary, since the electrical energy consumption data have a repetitive nature. The combination of the target values to provide the forecasting of the future values is done by a weighted average. That is, each neighbour got a weight equal to the inverse of the square of the distance with the corresponding pattern or set of features. This process has been explained in detail in Section 3.3.

Algorithm 1 summarises the steps needed to make time series forecasting using KNFTS when a train set and test set are done. First, the support features extracted from the time variable are created for both sets. Then, the optimal value of k is found making forward chaining time series cross validation using the train set. Once the model is trained, all distances between the features of the future points and the features of train set are computed and saved. Next, the value of k obtained in FCTSCV is used to choose the nearest neighbours and their n next values are combined with a weighted average. Values obtained are used as the forecast ones. Finally, comparing the results and the actual values of the test set; DTW, EDR and RMSE are computed and returned in a 3-tuple.

Algorithm 1 How to make time series forecasting using KNFTS

Data: Data: Trainset time series, Testset time series

Result: 3-tuple (DTW, EDR, RMSE)

Create support features in train set and test set

$Trainset_supp \leftarrow CreateFeatures(Trainset)$

$Testset_supp \leftarrow CreateFeatures(Testset)$

Execute forward chaining time series cross validation

$K_{opt} \leftarrow FCTSCV(Trainset_supp)$

for ($h = 1; h \leq H; h = h + 1$) {

 Calculate distances between instances

$Dist \leftarrow D(Trainset_supp, Testset_supp[h])$

 Choose the K_{opt} -nearest features

$(\mathbb{X}_{K_{opt}}, Y_{K_{opt}}) \leftarrow Select((\mathbb{X}_t, Y_t), Dist, K_{opt})$

 Compute the h forecast value

$Forecasts[h] \leftarrow WeightedAverage(Y_{K_{opt}})$

}

Save the performance measures

$KNFTS_errors \leftarrow (DTW(Testset, Forecasts), EDR(Testset, Forecasts), RMSE(Testset, Forecasts))$

In Algorithm 2, the steps to make forecasts in a time series using KNPTS are defined. First, the optimal value of k is searched by forward chaining time series cross validation. Second, the distances between the horizon pattern and all other patterns in the train set are computed. Third, the k optimal number of nearest patterns are extracted from the historical data. Fourth, a weighted average of the values of each pattern selected is done to make the forecast of future values. Finally, the DTW, EDR and RMSE are calculated with the forecast obtained and the actual values of the test set. These values are returned in a 3-tuple.

Algorithm 2 How to make time series forecasting using KNPTS

Data: Data: Trainset time series, Testset time series

Result: 3-tuple (DTW, EDR, RMSE)

Execute forward chaining time series cross validation

K_{opt} j - $FCTSCV(Trainset_supp)$

Calculate distances between horizon and all patterns

$Dist$ j - $D(Trainset_{(T-m-1,T)}, Testset)$

Choose the K_{opt} -nearest patterns

$Y_{K_{opt}}$ j - $Select(Y_t, Dist, K_{opt})$

Make h -ahead forecasting

$Forecasts$ j - $WeightedAverage(Y_{K_{opt}})$

Save the performance measures

$KNPTS_errors$ j - $(DTW(Testset, Forecasts), EDR(Testset, Forecasts), RMSE(Testset, Forecasts))$

The process used to obtain the measurements to compare the performance of the KNPTS and KNFTS is summarised in Algorithm 3. For each time series coming from a different building, two methods are executed for each partition explained in Table 3. First, the test set is chosen as one of the P partitions and the train set is chosen as all the previous available data. Second, KNFTS is computed and a 3-tuple containing the value of DTW, EDR and RMSE is provided. All this information is stored in a $B \times P$ array, where B is the number of buildings and P the number of test sets. In that case, there are $B = 122$ time series data sets and $P = 10$ partitions to test. Finally, the same process is repeated with the KNPTS algorithm and another $B \times P$ array containing the performance values is obtained.

Algorithm 3 Comparison process between KNFTS and KNPTS

Data: B building files

Result: two $B \times P$ arrays where each element is a 3-tuple of (DTW, EDR, RMSE) values

forall $b \in B$ **do**

```
    for (  $p = 1; p \leq P; p = p + 1$  ) {  
        Create  $trainset[b, p]$  and  $testset[b, p]$   
        KNFTS.errors[b][p]  $\leftarrow$  KNFTS( $trainset[b, p]$ ,  $testset[b, p]$ )  
        KNPTS.errors[b][p]  $\leftarrow$  KNPTS( $trainset[b, p]$ ,  $testset[b, p]$ )  
    }
```

end

5. Results and Discussion

In this section the effectiveness of the KNFTS and KNPTS forecasting methods is evaluated according to the three established metrics. This evaluation is performed in two different experiments: a one-day ahead forecasting and a one-week ahead forecasting. In both experiments, a graph comparing the DTW values is shown, as it is the performance measure that is considered most representative in the context of the problem we are dealing with. However, the average values of RMSE and EDR are also compared.

5.1. One-day ahead forecasting results

Figure 5 shows the DTW values obtained for each building using the KNFTS and the KNPTS methods for one-day ahead forecasting. Four of the analysed buildings are removed from the visualisation because their values in the KNFTS are extremely high (147,073.40 Wh, 39,971.85 Wh, 27,321.60 Wh, 23,097.07 Wh). Therefore, on the x-axis are the remaining 118 buildings after being ordered so that the errors achieved are shown in order from highest to lowest to facilitate their understanding. On the y-axis are the error values DTW obtained for each of the buildings. These values are the mean of the errors achieved in each of the partitions made to generalize the results. The visual comparison of the area under the curve of the interpolation of the errors allows to draw a preliminary conclusion about this paper's experimentation. Namely, since the area under KNPTS' curve is less than the area under KNFTS' curve, it can be stated that the KNPTS method performs better than the KNFTS in terms of accuracy.

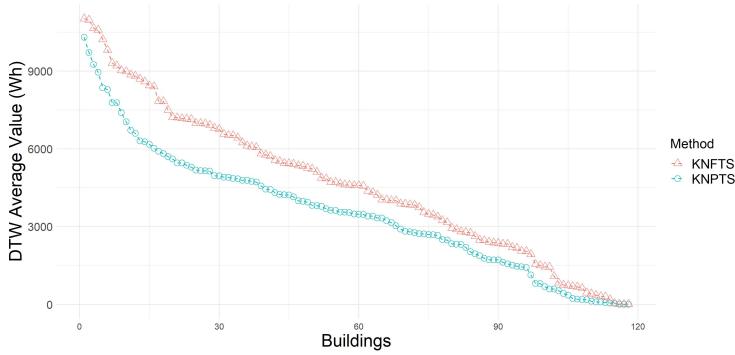


Figure 5: Average of the DTW error achieved by buildings making one day ahead forecasting using both methods (KNPTS and KNFTS)

Table 4: Error average in one day ahead forecasting

	DTW	EDR	RMSE
KNPTS	3,500.94	12.67	277.94
KNFTS	6,427.64	15.59	336.40

Table 4 shows the averages of the three performance criteria used for evaluating the two tested forecasting methods in the one-day ahead experiment. For instance, when the KNPTS method has been used, the mean error that has been obtained from DTW has been 3,500.94 Wh. The KNPTS method showed a better performance in all the cases. Regarding the DTW values, the KNPTS method's was 2,926.7 Wh lower on average, which means a reduction of 46% compared to the KNFTS method. As for the EDR metric, compared to KNFTS, KNPTS needs on average nearly 3 fewer edit operations convert the forecast series into the actual series, which represents an improvement of 19%. It is worth noting that, as mentioned before, the EDR value is subject to the threshold established, which in this experiment, has been of 40 Wh. Finally, the RMSE value provides an hourly approximation of the difference between the forecast and actual values. On average, the KNPTS improves 58.46 Wh per estimated value, that is a reduction of 17%.

Table 5 shows the number of buildings that obtained better results for the given metric in the one-day ahead forecasting experiment. Each column in the table adds up to 122, which is equal to the total number of buildings

Table 5: Best one day ahead forecasting method

	DTW	EDR	RMSE
KNPTS	109	106	66
KNFTS	13	16	56

analysed. Each cell in the table is a counter for the number of buildings that have separately received best results for each of the performance measures. As it can be seen, the KNPTS method performs better in most of the analysed buildings according to the three evaluation metrics. Namely, the KNPTS method is more accurate the 89% of the cases regarding the DTW values, the 87% of cases for the EDR operations and the 54% of the cases for the RMSE. The RMSE metric is the one in which KNPTS method's advantage is not as evident, but it should be considered that, since the RMSE is based on a static distance it sharply penalises deviant estimates of consumption peaks.

In addition, using KNPTS the optimal k number of neighbours to use is less than using KNFTS. The average of the k value is 2.73 in the KNPTS and 32% of those values are 1 whereas the average of k is 4.96 in KNFTS and 98% of models used 5 neighbours. In consequence, the time spent to compute the forecasts is less.

5.2. One-week ahead forecasting results

The comparison of the DTW values achieved for each building using KNFTS and KNPTS for one-week ahead forecasting are shown in Figure 6. There are four buildings which obtained extremely high DTW values for KNFTS (990,148.61 Wh, 202,353.68 Wh, 142,894.86 Wh, 138,205.64 Wh), so they were removed from the visualisation to ease its interpretation. This graph shows the remaining 118 buildings analysed on the x-axis arranged in such a way that the points are drawn in a decreasing way. In this case, although the difference is not as significant as in the one-day ahead experiment, the area under KNPTS' curve with is still less than the area under KNFTS's curve. Therefore, it can be considered that the forecasting errors achieved with KNPTS are less than with the KNFTS.

Table 6 shows the averages of the three performance criteria used for evaluating each the KNFTS and KNPTS methods in the one-week ahead experiment. Therefore, when the KNPTS was used, the mean error obtained for the DTW measurement in the 122 buildings was 24,068.75 Wh. For all the

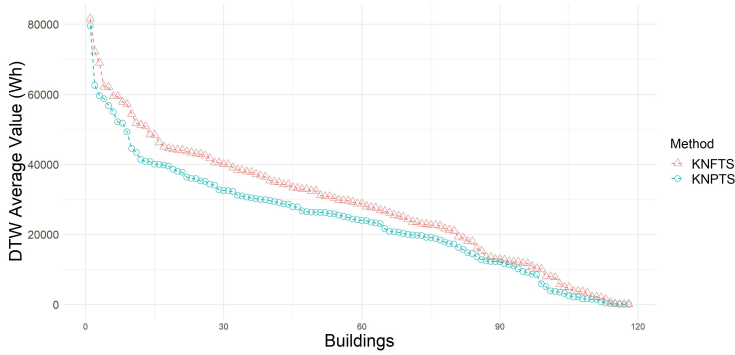


Figure 6: Average of the DTW error achieved by buildings making one week ahead forecasting using both methods (KNPTS and KNFTS)

Table 6: Error average in one week ahead forecasting

	DTW	EDR	RMSE
KNPTS	24,068.75	87.70	316.83
KNFTS	39,611.69	104.01	341.53

metrics evaluated, the KNPTS method performs best. First of all, regarding the DTW values, KNPTS' error is 15,542.94 Wh lower compared to the KNFTS, that is, DTW values are reduced by 39%. Secondly, regarding the EDR metric, an average of 16 fewer edit operations are required to convert the KNPTS forecast series into the real series, compared with the KNFTS forecast. This entails a reduction of 16% edit operations. Finally, as for the RMSE values, KNPTS reduces them by 24.7 Wh on average compared with KNFTS, which means a lowering of the errors in a point-to-point comparison of 7%. Looking at these results, it can be concluded that the KNPTS method is more effective than the KNFTS for forecasting one-week ahead values, similar to forecasting one-day ahead values. Compared with the one-day ahead results, KNPTS' performance improvement is not so big, which can be a consequence of the assumption that, the farther the point to be forecast, the higher the chance of incurring bigger errors.

The number of buildings in which one of the tested methods obtains a lower error for the given metric in a one-week ahead forecasting are summarised in Table 7. It is verified that the columns of the table add up to 122

Table 7: Best one week ahead forecasting method

	DTW	EDR	RMSE
KNPTS	109	105	41
KNFTS	13	17	81

because it is the number of buildings analysed in this study. The one-day ahead and the one-week ahead results are very similar regarding the elastic performance measures. Lower DTW values are obtained with the KNPTS method in 89% of cases and less EDR operations in 86% of cases. However, looking at the point-for-point metric, the RMSE suggests that the KNFTS can be better, as is the case for the 66% of the buildings. This result may be counter-intuitive to the result shown in Table 6, which shows that the KNPTS method reduces the average RMSE. In cases where the KNFTS error is lower, the difference obtained with both methods is small whereas the difference of the errors obtained by both methods is significantly greater when the KNPTS obtains better estimates. Therefore, for a specific case in which these values were obtained, the method to be applied could be debated. However, as the objective is to find a general method that performs better, the KNPTS would be chosen again since the average error achieved for all use cases is 7% lower than with KNFTS.

Finally, a less number of k neighbours are used to forecast in KNPTS. The average of the k value is 2.35 in the KNPTS and 41% of those values were 1 whereas the average of k is 4.96 in KNFTS and 98% were 5. Consequently, the time spent to compute the forecasts is less with KNPTS.

After analysing the results from both experiments, it can be concluded that KNPTS provides an improvement in the forecast of electrical energy consumption compared to KNFTS.

6. Conclusions

The effectiveness of DR programs heavily depends on an accurate prediction of the energy to be consumed and especially on the precision of estimates of peaks in customer consumption. Electricity consumption data are captured with a time series structures and need specific methodology to get accurate forecasts from existing data.

First, in the case of doing multi step ahead forecasting, MIMO is a strategy that avoids the accumulation of errors in future values and builds a single

model to estimate the desired points.

Then, to select the best time series cross validation approach, considering two essential aspects of the historical data: stability and quantity. In the case that data are cyclical, such as the electrical consumption of buildings, and that the amount of data does not imply a sudden change in behaviour, the optimal approach is the forward chaining cross validation.

The third decision is related to the forecasting model. It is commonly thought that variables referring to time and occupation are essential to achieve good forecasts, but this work demonstrates the effectiveness of KNPTS and KNFTS in tackling this problem. These algorithms are especially effective when a large historical data set is available.

The study carried out with 122 buildings on the island of Lanzarote showed that the KNPTS is more effective in forecasting one day ahead and one week ahead than the KNFTS. That improvement implies that the best forecasts can be obtained with similar patterns of consumption in the historical set without any feature engineering preprocessing. This represents an important advance for cases in which no extra information about the building is available.

The values achieved for the three chosen performance criteria shows that the forecasts obtained with KNPTS are better adjusted to the real values than those obtained with the KNFTS. In both experiments, 89% of the cases achieved a better result of DTW value using the KNPTS method. Regarding the EDR and RMSE measures, they also achieved lower values on average when making forecasts with KNPTS instead of KNFTS. Furthermore, the number of neighbours needed to estimate the values in the KNPTS is less than in the KNFTS. This fact and the number of steps required to train both models affects the computation time of the methods and this is an important issue that must be considered when trying to put the algorithms into production. The computation time grows when the number of cases to be estimated and the size of the available historical data set increase.

Finally, the novel approach of using measures of similarity in time series to evaluate the effectiveness of the models shows that lockstep metrics are not always adequate. It is necessary to choose the performance measures that best represent the nature and purpose of each problem. In the case of hourly data in which a shift of one hour is allowed, the DTW is considered the best measure to evaluate the effectiveness of forecasting models.

In summary, the KNPTS is an algorithm that finds similar patterns in the historical consumption in order to estimate the incoming values. This

method performs better in time series with a very cyclical behaviour and a low variability insight their data. The improvement in the estimation of future electricity consumption values allows intervention in user decisions to reduce consumption peaks. In this way, the electrical consumption of buildings can be adapted to the curve of electrical energy production from renewable sources. All of this contributes to energy saving and environmental sustainability.

6.1. Future Work

Electrical consumption data in residential and commercial buildings have a very cyclical and steady behaviour, therefore, algorithms based on the recognition of patterns over historical data perform well. Sometimes, variations in the distribution of the data occur and it could be interesting to adapt the models to the new situations. For instance, the users' lifestyle changed during the COVID-19 pandemic. Some governments activated residential confinement measures and the lifestyle of users was affected. These changes are reflected in the electrical consumption causing concept drift or deviations in the precision of the models. Therefore, future research related to concept-drift methods is expected to detect and deal with these changes to avoid model degradation when these situations happen.

Moreover, a broader study is required for a more exhaustive and comprehensive comparison between the KNPTS algorithm and other Machine Learning algorithms. This comparison should be done using different time series data sets and not only evaluated in terms of the accuracy of the algorithms, but also considering their interpretation and computational cost. This would provide a strong argument for the effectiveness of this method in forecasting energy consumption compared to others.

Nomenclature

X	The set of explanatory variables
ω_λ	Weight of the λ -th nearest neighbour
v	Trigonometric transformation for cyclical variables
$\{\hat{y}_t\}$	Forecast Time Series

$\{y_t\}$	Time Series
H	Number of points to be forecast
K	Number of neighbours in KNN
T	Last value of time in a time series
t	Time variable
P	Periodicity of a cyclical variable
W	Window size of the last values in a time series
X^i	The i -th explanatory variable of a data set
Y	Response variable of a model

Acknowledgements: This work is partly supported by the REACT (Renewable Energy for self-sustainable island Communities) project, which has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement no. 824395 and by the project 3KIA (KK-2020/00049), financed by the SPRI-Basque Government through the ELKARTEK program.

References

- [1] T. Ramesh, R. Prakash, K. Shukla, Life cycle energy analysis of buildings: An overview, *Energy and buildings* 42 (2010) 1592–1600. doi:10.1016/j.enbuild.2010.05.007.
- [2] B. K. Sovacool, What are we doing here? analyzing fifteen years of energy scholarship and proposing a social science research agenda, *Energy Research & Social Science* 1 (2014) 1–29. doi:10.1016/j.erss.2014.02.003.
- [3] A. De Almeida, P. Fonseca, B. Schlomann, N. Feilberg, Characterization of the household electricity consumption in the eu, potential energy savings and specific policy recommendations, *Energy and Buildings* 43 (2011) 1884–1894. doi:10.1016/j.enbuild.2011.03.027.

- [4] P. Warren, A review of demand-side management policy in the uk, *Renewable and Sustainable Energy Reviews* 29 (2014) 941 – 951. doi:10.1016/j.rser.2013.09.009.
- [5] C. Bartusch, K. Alvehag, Further exploring the potential of residential demand response programs in electricity distribution, *Applied Energy* 125 (2014) 39–59. doi:10.1016/j.apenergy.2014.03.054.
- [6] I. Dusparic, C. Harris, A. Marinescu, V. Cahill, S. Clarke, Multi-agent residential demand response based on load forecasting, in: 2013 1st IEEE conference on technologies for sustainability (SusTech), IEEE, 2013, pp. 90–96. doi:10.1109/SusTech.2013.6617303.
- [7] H. Hewamalage, C. Bergmeir, K. Bandara, Recurrent Neural Networks for Time Series Forecasting: Current status and future directions, *International Journal of Forecasting* (2021). doi:10.1016/j.ijforecast.2020.06.008. arXiv:1909.00590.
- [8] M. A. Mat Daut, M. Y. Hassan, H. Abdullah, H. A. Rahman, M. P. Abdullah, F. Hussin, Building electrical energy consumption forecasting analysis using conventional and artificial intelligence methods: A review, 2017. doi:10.1016/j.rser.2016.12.015.
- [9] G. Mihalakakou, M. Santamouris, A. Tsangrassoulis, On the energy consumption in residential buildings, *Energy and Buildings* 34 (2002) 727–736. doi:10.1016/S0378-7788(01)00137-2.
- [10] W. Yu, B. Li, Y. Lei, M. Liu, Analysis of a residential building energy consumption demand model, *Energies* 4 (2011) 475–487. doi:10.3390/en4030475.
- [11] Q. Li, P. Ren, Q. Meng, Prediction model of annual energy consumption of residential buildings, in: 2010 International Conference on Advances in Energy Engineering, ICAEE 2010, 2010. doi:10.1109/ICAEE.2010.5557576.
- [12] G. Escrivá-Escrivá, C. Álvarez-Bel, C. Roldán-Blay, M. Alcázar-Ortega, New artificial neural network prediction method for electrical consumption forecasting based on building end-uses, *Energy and Buildings* 43 (2011) 3112–3119. doi:10.1016/j.enbuild.2011.08.008.

- [13] A. Azadeh, S. F. Ghaderi, S. Sohrabkhani, Annual electricity consumption forecasting by neural network in high energy consuming industrial sectors, *Energy Conversion and Management* 49 (2008) 2272–2278. doi:10.1016/j.enconman.2008.01.035.
- [14] K. Li, H. Su, J. Chu, Forecasting building energy consumption using neural networks and hybrid neuro-fuzzy system: A comparative study, *Energy and Buildings* 43 (2011) 2893–2899. doi:10.1016/j.enbuild.2011.07.010.
- [15] E. Spiliotis, Z. Raptis, Axilleas nad Nikoleta Legaki, V. Assimakopoulos, Forecasting electrical consumption of commercial buildings using energy performance indicators, *International Journal of Decision Support Systems* 1 (2015) 164–182. doi:10.1504/IJDS.2015.067556.
- [16] A. Jozi, T. Pinto, G. Marreiros, Z. Vale, Electricity consumption forecasting in office buildings: An artificial intelligence approach, in: 2019 IEEE Milan PowerTech, PowerTech 2019, 2019. doi:10.1109/PTC.2019.8810503.
- [17] M. K. Kim, Y. S. Kim, J. Srebric, Predictions of electricity consumption in a campus building using occupant rates and weather elements with sensitivity analysis: Artificial neural network vs. linear regression, *Sustainable Cities and Society* (2020). doi:10.1016/j.scs.2020.102385.
- [18] N. J. Johannesen, M. Kolhe, M. Goodwin, Relative evaluation of regression tools for urban area electrical energy demand forecasting, *Journal of Cleaner Production* (2019). doi:10.1016/j.jclepro.2019.01.108.
- [19] P. Shine, M. D. Murphy, J. Upton, T. Scully, Machine-learning algorithms for predicting on-farm direct water and electricity consumption on pasture based dairy farms, *Computers and Electronics in Agriculture* 150 (2018) 74–87. doi:10.1016/j.compag.2018.03.023.
- [20] J. W. Taylor, Triple seasonal methods for short-term electricity demand forecasting, *European Journal of Operational Research* 204 (2010) 139–152. doi:10.1016/j.ejor.2009.10.003.
- [21] S. Gerber, A. J. Rix, M. J. Booyesen, Towards sustainable developing cities: A simplified forecasting model for sizing grid-tied PV us-

- ing monthly electricity bills, *Sustainable Cities and Society* (2020). doi:10.1016/j.scs.2019.101994.
- [22] M. Gómez-Omella, I. Esnaola-Gonzalez, S. Ferreiro, Short-Term Electric Demand Forecasting for the Residential Sector: Lessons Learned from the RESPOND H2020 Project, *Proceedings* (2021). doi:10.3390/proceedings2020065024.
- [23] M. Gomez-Omella, I. Esnaola-Gonzalez, S. Ferreiro, Short-term forecasting methodology for energy demand in residential buildings and the impact of the COVID-19 pandemic on forecasts, in: Bramer M., Ellis R. (eds) *Artificial Intelligence XXXVII. SGA I 2020. Lecture Notes in Computer Science*, volume 12498, 2020, pp. 227–240. doi:10.1007/978-3-030-63799-6_18.
- [24] R. Adhikari, R. Agrawal, An Introductory Study on Time Series Modeling and Forecasting Ratnadip Adhikari R. K. Agrawal, Ph.D. thesis, 2013. arXiv:1302.6613.
- [25] S. Ben Taieb, G. Bontempi, A. F. Atiya, A. Sorjamaa, A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition, *Expert Systems with Applications* (2012). doi:10.1016/j.eswa.2012.01.039. arXiv:1108.3259.
- [26] M. W. Browne, Cross-validation methods, *Journal of Mathematical Psychology* 44 (2000) 108 – 132. doi:10.1006/jmps.1999.1279.
- [27] C. Bergmeir, J. M. Benítez, On the use of cross-validation for time series predictor evaluation, *Information Sciences* 191 (2012) 192–213. doi:10.1016/j.ins.2011.12.028.
- [28] D. Berrar, Cross-validation, in: *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*, 2018. doi:10.1016/B978-0-12-809633-8.20349-X. arXiv:1703.03167.
- [29] R. J. Hyndman, G. Athanasopoulos, *Forecasting: Principles and Practice, Principles of Optimal Design* (2018).
- [30] E. Alpaydin, *Introduction to machine learning*, MIT press, 2020.

- [31] A. Zheng, A. Casari, Feature engineering for machine learning, 2018. doi:10.13140/RG.2.1.3564.3367.
- [32] F. Martínez, M. P. Frías, M. D. Pérez, A. J. Rivera, A methodology for applying k-nearest neighbor to time series forecasting, *Artificial Intelligence Review* 52 (2019) 2019–2037. doi:10.1007/s10462-017-9593-z.
- [33] H. A. Abu Alfeilat, A. B. Hassanat, O. Lasassmeh, A. S. Tarawneh, M. B. Alhasanat, H. S. Eyal Salman, V. B. Prasath, Effects of Distance Measure Choice on K-Nearest Neighbor Classifier Performance: A Review 7 (2019) 221–248. doi:10.1089/big.2018.0175.
- [34] G. Bontempi, S. Ben Taieb, Y. A. Le Borgne, Machine learning strategies for time series forecasting, in: *usiness Intelligence: Second European Summer School, eBISS 2012, Brussels, Belgium, July 15-21, 2012, Tutorial Lectures*, Springer Berlin Heidelberg, 2013, pp. 62–77. doi:10.1007/978-3-642-36318-4_3.
- [35] P. Senin, Dynamic Time Warping Algorithm Review, Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA 855 (2008).
- [36] D. Berndt, J. Clifford, Using dynamic time warping to find patterns in time series, *AAAIWS'94*, AAAI Press, 1994, p. 359–370.
- [37] L. Chen, M. T. Özsu, V. Oria, Robust and fast similarity search for moving object trajectories, in: *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2005, p. 491–502. doi:10.1145/1066157.1066213.

On the Evaluation, Management and Improvement of Data Quality in Streaming Time Series

- Title: On the Evaluation, Management and Improvement of Data Quality in Streaming Time Series
- Authors: Meritxell Gómez-Omella, Basilio Sierra, Susana Ferreira
- Journal: IEEE Access
- Year: 2022
- Quartile: Q1
- DOI: [10.1109/ACCESS.2022.3195338](https://doi.org/10.1109/ACCESS.2022.3195338)

RESEARCH ARTICLE

On the Evaluation, Management and Improvement of Data Quality in Streaming Time Series

MERITXELL GÓMEZ-OMELLA^{1,2}, BASILIO SIERRA^{1,2}, AND SUSANA FERREIRO¹

¹Tekniker, Basque Research and Technology Alliance (BRTA), 20600 Eibar, Spain

²Faculty of Informatics, University of the Basque Country (UPV/EHU), 20018 Donostia-San Sebastian, Spain

Corresponding author: Meritxell Gómez-Omella (meritxell.gomez@tekniker.es)

This work was supported in part by the SPRI-775 Basque Government through the ELKARTEK Program through the Project 3KIA under Grant KK-2020/00049.


ABSTRACT The Internet of Things (IoT) technologies plays a key role in the Fourth Industrial Revolution (Industry 4.0). This implies the digitisation of the industry and its services to improve productivity. To obtain the necessary information throughout the different processes, useful data streams are obtained to provide Artificial Intelligence and Big Data algorithms. However, strategic decision-making based on these algorithms may not be successful if they have been developed based on inadequate low-quality data. This research work proposes a set of metrics to measure Data Quality (DQ) in streaming time series, and implements and validates a set of techniques and tools that allow monitoring and improving the quality of the information. These techniques allow the early detection of problems that arise in relation to the quality of the data collected; and, in addition, they provide some mechanisms to solve these problems. Later, as part of the work, a use case related to industrial field is presented, where these techniques and tools have been deployed into a data management, monitoring and data analysis platform. This integration provides additional functionality to the platform, a Decision Support System (DSS) named *DQ-REMAIN* (*Data Quality REport MANAGEMENT and ImproveMeNt*), for decision-making regarding the quality of data obtained from streaming time series.

INDEX TERMS Data quality, streaming time series, decision support system.

I. INTRODUCTION

The *Internet of Things (IoT)* is a new evolution of the Internet that includes many applications in different domains such as transportation and logistics, healthcare, smart environments and personal and social interactions as explained in [1].

Large amounts of data have been captured with the recent digitisation of the industry, which represents a link between the physical and cyber world [2]. The analysis of the large amount of available data from historical data bases is an important step in obtaining information in different fields. This type of information can be used for anomaly detection, diagnosis, and/or forecasting as shown in [3] and [4] to obtain knowledge about the behaviour or conditions of a system.

The associate editor coordinating the review of this manuscript and approving it for publication was Justin Zhang .

Despite the many types of analysis that can be carried out, the common goal of any of these studies based on the Data Information-Knowledge-Wisdom (DIKW) model [5] is wisdom. Data are the basis of the DIKW pyramid (Figure 1). Therefore, Data Quality (DQ) is a crucial requirement for any data analysis.

Poor DQ has a negative effect on these activities. Therefore, the accuracy of the techniques and algorithms can decrease significantly incorrect or poor-quality data have been used as inputs. Consequently, the conclusions drawn by understanding the results may be incorrect. Some studies have revealed that poor data quality is responsible for millions of annual losses [6]. Data gathered from the global-scale deployment of smart-things are the base for making intelligent decisions and providing services in IoT applications. Low quality has a high impact ranging from increased

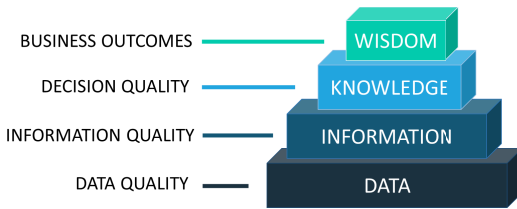


FIGURE 1. The basic structure of the Data-Information-Knowledge-Wisdom pyramid.

difficulty in setting strategies, derived from data analysis, to reduced customer satisfaction. [7]. The use of low-quality data leads to unsustainable decision and unsuccessful strategies and induces inefficient decision-making. The study of DQ is crucial for achieving user participation and acceptance of the IoT paradigm and services [2].

Organisations are aware of the importance of measuring the quality of data to identify errors and face losses. In addition, identifying these problems provides information on whether the data can be useful for their purposes. However, according to Gartner, nearly the 60 % of organisations do not measure the annual financial cost of poor-quality data. the annual spending on on-premises DQ tools remains high, with an average of \$208000 and a median of \$150000, and it prevents more pervasive adoption of tools [8]. Measuring and understanding DQ with the right tools is therefore necessary to improve outcomes and increase confidence in data-driven decisions [9].

Many definitions of the DQ can be found in the literature. Because of this variety, choosing suitable methods, that are advantageous for the DQ of a certain problem or in a particular context is a challenge [10]. These definitions usually refer to technical documents (standards) established by analysts or relevant organisations. In these cases, controlling the DQ simply ensures that the data follow that standard. However, to assess the quality of the data (DQ), it is not only necessary to define the influencing aspects, but also to associate them with numerical scores. It is a scientific and statistical evaluation process that allows the calculation of a numerical data quality value for each problem or factor that influences DQ [11]. It can be considered a set of techniques or equations used to quantify and improve the quality of the data. The data set can be of low quality owing to a set of problems of various types and nature (e.g. loss of data, low accuracy, etc.). Therefore, it is necessary to have the required mechanisms to identify and quantify the problems that may exist in the data set with respect to its low quality, as well as having a set of corrective functions that allow handling each of those problems to improve the quality of the data. This is the focus of this work, the definition and implementation of some metrics to measure the quality of the data in streaming time series, and proposals to increase it.

The remainder of this paper is organised as follows. An extensive review of the methods proposed in the literature

is presented in Section II. A mathematical representation to assess DQ in streaming time series is presented in Section III. Section IV describes the complete flow of the proposed methodology. The implementation of these metrics and functionalities resulted in an R package called `dqts`. Main functions available in `dqts` R package are explained in Section V. The methodology has been validated for different datasets (simulated, open-source and real) in Section VI. Finally, the design of the *DQ-REMAIN DSS* (*Data Quality REport Management and Improvement Decision Support System*) for DQ analysis is presented in Section VII. The DSS has been described in functional blocks based on the use of `dqts` package and it provides an easy and intuitive solution for user interaction with DQ analysis for an industrial use case.

II. RELATED WORK

In this section, existing studies on definitions, approaches and implementations of DQ are analysed.

The need for the research community to define DQ was born at the end of the twentieth century when the quantity of data flows increased considerably with the digitisation of the industry. Initially, DQ focused on measuring dispersion using statistics when the data follow a known distribution. The confidence interval (CI) of the mean can be defined and precision can be evaluated according to the allowed variability [12].

Currently, DQ is not limited to traditional techniques based on the study of the standard deviation. There are other aspects to consider to achieve a high DQ. The main aim of most DQ publications in the late 20th century was to provide a formal definition of the term. Since then, the concept has been most often associated with the 'fitness for use' principle [13].

Therefore, it was wanted to delve into the aspects that define the concept of DQ, and several authors gave different sets of DQ dimensions such as *Accuracy*, *Timeliness*, *Interpretability* and *Accessibility* [14], [15]. Data dimensions are attributes of the DQ that can, when measured correctly, indicate the overall quality level of the data. There are many possible dimensions of DQ depending on the context and nature of the data. These dimensions come from the issues specific to each field [16]–[18]. An overview of dimension definitions can be found in [19]. Furthermore, as the topic became more interesting, the quality of the data in specific sectors began to be defined and some authors focused their work on defining dimensions of quality for data received by sensors [20]. Essentially, the data collected by the sensors are streaming time series because the data are recorded together with the moment of time in which it was received.

Increased interest in the search for quality standards has led to the creation of DQ metrics. Metrics are formulas that allow quantification of different quality aspects within a dimension. Therefore, the most common approach to measure DQ is to define a set of metrics that provide numerical results to detect and correct data failures, and combine them to provide a numerical score of the overall DQ [21], [22]. The range of metrics available varies widely, because of the several definitions of the DQ concept depending on the context.

Nevertheless, one of the aspects in which the authors of research on DQ agree is the study of reliability [23]. It focuses on the use and trust of data. In other words, the study of metrics that can characterise the quality of the data received to provide an indicator of quality for future studies on the exploitation of these data.

The maximum knowledge of the domain and the problems presented is required for the correct definition of the metrics. It is common to find different definitions of metrics in relation to the problems presented by the data in different fields. Some authors place more emphasis on the diversity and volume of the data and the problem that the data change very quickly [23] whereas others consider missing values as the main quality problem. Furthermore, different definitions of DQ metrics can be found according to the origin of the data; for instance, [24] provided a specific approach when data come from sensors.

Then, the need arose to define complete methodologies for the study of DQ. This is divided into four activities: (1) state reconstruction, (2) DQ measurement or assessment, (3) data cleansing or improvement and (4) continuous data monitoring [25], [26]. The first phase of state reconstruction refers to the collection of contextual information, which is beyond the scope of this work. Although measurement and evaluation are concepts often treated at the same level, it is important to differentiate them, in terms of DQ. The term ‘measure’ describes the assignment of a numerical value or degree that allows quantification. Instead, assessment is the evaluation of the nature, ability, or quality of something and consists of analysing the results of measurements to draw a conclusion. Step (3) concerns the strategies to achieve the highest DQ. Finally, the techniques proposed for the periodic report and control of the temporal evolution are monitored.

The studies [27], [28] and [29] mathematically approximated some of the metrics proposed in the literature to quantify the quality of the data in a time series. However, the mathematical formulation of the mathematical formulation of the complete methodology to measure DQ in time series is still not available in the community of data analysts, who must manually adapt their own analysis methodology for DQ to the problem.

On the other hand, a work that is still to be solved is building a personalised DQ management platform. Each data consumer has a unique vision of how “good” data should depend on their core business and needs [2]. The first step in the design of this platform is to define a general methodology for calculating the DQ score in a time series. Thus, a methodology that combines DQ measurement, assessment, improvement and monitoring is still unavailable in the literature.

After a review of the existing studies, we can conclude that there are an extensive number of definitions of metrics and dimensions depending on the subsequent use of the data and the context in which they are being analysed. In addition, no methods have been found to calculate metrics that are not based on a reference data set provided by the user. The need for a practical methodology for the treatment of

DQ has been identified, which focuses on measuring and treating its different aspects instead of providing abstract definitions. Regarding the R packages available for data quality assessment, as far as we know, we conclude that there are no specific R packages for the quality of time series data and although some authors mention the dimensions of data quality, definitions of most of them are lack. Packages can be found in CRAN that are focused on a specific use of data in different fields such as *dataquieR* [30] to calculate the quality of epidemiological research data or *RawHumus* [31] in metabolomics. In addition, the *daqapo* R package offers a DQ assessment for process-oriented data that allows the detection of violations in frequency, order and range, detect outliers and missing values, incorrect names and unique values but it requires a preprocessing step consisting of creating a certain type of data set [32]. The available R packages to assist data quality in general data sets calculate statistical indicators such as the mean and standard deviation, report unique values, and evaluate the number of missing values. These are *StatMeasures* [33], *dlookr* [34], *skimr* [35] and *xplorerr* [36]. In addition, *dlookr* include outliers indicators and *xplorerr* provides an interactive application in Shiny to show these results with open data sets or evaluate the data provided by the user. The only correction function found was provided by *StatMeasures*. It is a simple imputation function that replaces missing values with the value that the user enters as an argument of the function. The remaining packages are excluded from this comparison because the available documentation is not updated, or it is not possible to access the functions to use them in R.

This work was motivated by the need to monitor quality metrics in time series to ensure high DQ over time through the possibility of correcting the problems identified [37]. In addition, the available tools do not address the implementation of quality metrics. There is a gap in the relationship between the theory of data quality and tools available for its exploitation. Furthermore, for the DQ tools generated, approximately half are domain specific and of those that provide automatic support, there are no definitions of the functions. Finally, for the best of our knowledge, there are no tools that allow the correction of the metrics with the worst scores, nor the interactive design of DSS for the management of the DQ of streaming time series.

III. DATA QUALITY METRICS IN TIME SERIES

The analysis of the quality of the data received for subsequent statistical modelling and prediction studies is an important preliminary step in any data analysis. Although it is a matter that resides in the characteristics of each data set and in the intention and objective of the subsequent study, a generalisation of this concept is desired. Exact guidelines cannot be provided, however, the data are expected to conform to established standards. This section describes in detail the concept of DQ when data have an ordered structure, that is, a particular definition for DQ in time series is given by mathematical formulation and some proposed solutions.

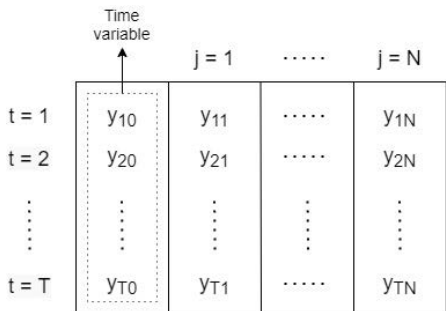


FIGURE 2. Schematic representation of the structure of a multivariate time series.

This type of data has special characteristics and should be treated in a special manner.

First, the necessary notation is introduced to understand the formulation given below. A time series is a collection of values obtained over time, often at regular time intervals. Therefore, consecutive observations can usually be recorded at equally spaced time intervals such as hourly, daily, weekly, monthly or yearly time intervals [38]. A time series is called univariate when it collects information about one characteristic and can be written as $Y = \{y_t\}$, where $t = 1, \dots, T$ represents the elapsed time. A multivariate time series contains information from more than one characteristic, each of them having a univariate time series structure. In that case, the data set can be written as a table of dimension $T \times N$, where T is the number of observations, that is, the time elapsed, and N is the number of variables or univariate time series available. Often, a multivariate time series can be represented as a data table, as shown in Figure 2, where the first column is the time data and the remaining columns are the N variables to be analysed. The following methodology can be applied to multivariate time series data sets, considering each variable as a univariate time series. To simplify the notation, the time variable represented by $\{y_{10}, \dots, y_{T0}\}$ is denoted by $\{t_1, \dots, t_T\}$ from this point forward. The data set $Y = \{y_{ij}\}$, where y_{ij} is the value of variable j at time t , and $t \in \{1, \dots, T\}$ and $j \in \{1, \dots, N\}$, is denoted by Y in the remainder of this work, regardless of whether the time series is univariate ($N = 1$) or multivariate ($N > 1$).

The metrics are adapted to time series, taking the definitions provided by other authors summarised in Section II. Eleven metrics necessary to calculate the DQ in time series are presented below, classified into five dimensions. These metrics are functions that return values between 0 and 1 where 0 represents poor quality of the data and 1 represents the highest quality. Once the results of each quality metric have been calculated, a final DQ indicator can be provided by the arithmetic or weighted mean of the eleven metrics. The problem to be identified is described before each formula used to calculate the DQ scores. An optimal solution is proposed to deal with the poor quality identified by each of the metrics. This information is summarised in Table 1.

A. CONFORMITY

1) PROBLEM IDENTIFICATION

Conformity measures the amount of data stored in a standard format. Then, it determines the proportion of variables that are in the correct format and have the correct names. Conformity analysis is performed in two steps and both use a reference data set that can be provided or simulated before executing these metrics. The reference data set contains the same variables as the time series to be analysed, with the same names and in the same order. Each contains only one string element indicating the type of variable that is expected.

First, it is analysed that the names with which the variables have been labelled are correct. This is done by comparing the names with the names of the variables in the reference data set. It is then checked whether the format of the values contained in each of the variables is correct. The variables can be numerical, categorical or dates. This check is carried out by comparison with the reference data set.

Let $\{l_1, \dots, l_N\}$ denote the names of the N variables available, and $\{l_1^*, \dots, l_N^*\}$ denote the names of the variables in the reference data set. Similarly, $\{f_1, \dots, f_N\}$ are the types of the N variables available and $\{f_1^*, \dots, f_N^*\}$ are the correct formats of the same N variables in the same order in the reference data. The coincidence sets are defined as $\{c_1^F, \dots, c_n^F\}$ and $\{c_1^L, \dots, c_n^L\}$, where

$$c_j^L = \begin{cases} 1, & \text{if } l_j = l_j^* \\ 0, & \text{if } l_j \neq l_j^* \end{cases} \quad \text{and} \quad c_j^F = \begin{cases} 1, & \text{if } f_j = f_j^* \\ 0, & \text{if } f_j \neq f_j^* \end{cases} \quad (1)$$

So using that notation,

$$Names = \frac{\sum_{j=1}^N c_j^L}{N} \quad \text{and} \quad Format = \frac{\sum_{j=1}^N c_j^F}{N} \quad (2)$$

2) SOLUTION

There are two possible solutions when the Conformity value is low. The first is the transformation of the malformed variables into the desired format, taking those of the reference set as valid formats. The same is true for the names of variables. This process is not always possible because an external interpretation is sometimes necessary to convert an element from one type to another. Alternatively, the deletion of the conflicting variable from both the analysis and the reference data set is considered as a solution.

B. UNIQUENESS

1) PROBLEM IDENTIFICATION

The second dimension defined by some authors for the DQ is Uniqueness. The definition of this metric can vary according to the characteristics of the data set and objectives of the study. Each data set requires a different uniqueness in the captured variables. The uniqueness in the time variable is an important point regarding the DQ in time series because the repetition of the timestamps is not allowed. Time Uniqueness is the metric proposed to calculate the proportion of unique

values in the time variable and it is complementary to the duplicated timestamps in the data set. Let $\tau \in \mathbb{N}$ be the number of unique values of the time variable,

$$\text{Time Uniqueness} = \frac{\tau}{T}, \quad \tau \leq T \quad (3)$$

2) SOLUTION

Two solutions were proposed to address the low scores obtained in this metric. A straightforward method to increase the value of the *Time Uniqueness* score is to delete observations with duplicate values in the time variable. Another more complex method is the combination of repeated observations, for instance averaging. The second method is equivalent to the first one when the values of the other variables are repeated. An example of timestamp repeated three times is shown in Figure 3.

C. TIMELINESS

1) PROBLEM IDENTIFICATION

Time series data are typically saved at uniform time intervals. However, when data are received by sensors, there are usually small imbalances that should not be alarming, causing the time waits longer than what you want to allow. Therefore, the following metric calculates the proportion of observations received without a waiting time. *Timeliness* provides information on whether the data are available at the right time. Let Y_{t0} be the observations of the time variable where $t = 1, \dots, T$. A set containing the difference between the time values was performed using

$$\mathcal{D} = \{\delta_1, \dots, \delta_{T-1}\} \quad (4)$$

where $\delta_t = Y_{t+10} - Y_{t0}$ for $t = 1, \dots, T - 1$. Let δ_{max} be the maximum time difference allowed by two consecutive observations, *Timeliness* is computed as follows

$$\text{Timeliness} = \frac{\mathcal{D}^*}{T - 1} \quad (5)$$

where $\mathcal{D}^* = \{\delta_t \in \mathcal{D} \mid \delta_t \leq \delta_{max}\}$, $t = 1, \dots, T - 1$.

2) SOLUTION

Timeliness is the complementary value of the events of time that had been lost during the acquisition of time series data. The methods proposed to increase *Timeliness* value are based on the artificial generation of missing intermediate timestamps. Waiting times were examined and the necessary values were created for the time variable. Three methods are proposed to address the other variables. In the first method, no value is assigned to the rest of the variables, so the value of *Completeness* decreases after applying this method. The other two, take the average and median of the available data for each variable and use them to complete the observations.

D. COMPLETENESS

1) PROBLEM IDENTIFICATION

This concept refers to the degree of complete and present elements in the data set and it is one of the most important

points of DQ. This value is the complementary of the degree of missing values, that are present in many studies and create uncertainty in research results. Given a multivariate time series in a data set Y , $M = \{m_{ij}\}$ is defined as the set of missing values where

$$m_{ij} = \begin{cases} 0, & \text{if } y_{ij} \text{ is missed} \\ 1, & \text{if } y_{ij} \text{ is known} \end{cases} \quad (6)$$

The first proposed metric has the same name as that of the dimension to be calculated. The score obtained by the following formula refers to the number of values present in the global view of the data set.

$$\text{Completeness} = \left(\frac{\sum_{t=1}^T \sum_{j=1}^N m_{tj}}{T \times N} \right) \quad (7)$$

Depending on the objective of each study, knowing whether the loss of data occurs at the same time for different variables may be of interest. Therefore, it is necessary to calculate the present values by observations. Similarly, in some cases it would be interesting to identify if missing data appear in the same variable over time. Thus, it is possible to identify faults in a specific characteristics of the set.

The subsets $T_{obs} \subseteq T$, $N_{var} \subseteq N$ are defined to calculate completeness by observations and variables, respectively.

$$T_{obs} = \left\{ t \in T \mid \sum_{j=1}^N m_{tj} = 0 \right\} \quad (8)$$

$$N_{var} = \left\{ j \in N \mid \sum_{t=1}^T m_{tj} = 0 \right\} \quad (9)$$

The *Completeness by observations* can be calculated as

$$\text{Completeness}_{obs} = \left(1 - \frac{|T_{obs}|}{T} \right) \quad (10)$$

and the *Completeness by variables* can be calculated as

$$\text{Completeness}_{var} = \left(1 - \frac{|N_{var}|}{N} \right) \quad (11)$$

2) SOLUTION

To increase the quality of the *Completeness* metric, different methods of handling missing values can be found in the literature. Methods of dealing with missing data can be classified into three groups: Case/Pairwise Deletion, Parameter Estimation and Imputation [39]. The first approach discards the observations that contain missing values. Usually if the amount of missing data is not relevant in the study, that is, there are very few values that are unknown, we choose to discard them. In this way, all statistical analysis is carried out without considering them and only the available data are proceeded. In the case of identifying any variable made up entirely of missing data, it will be removed from the data set. Subsequently, the dimensions of the data set are reduced and the statistical properties change [40]. The treatment of lost data is outside the scope of this work, and thus, possible

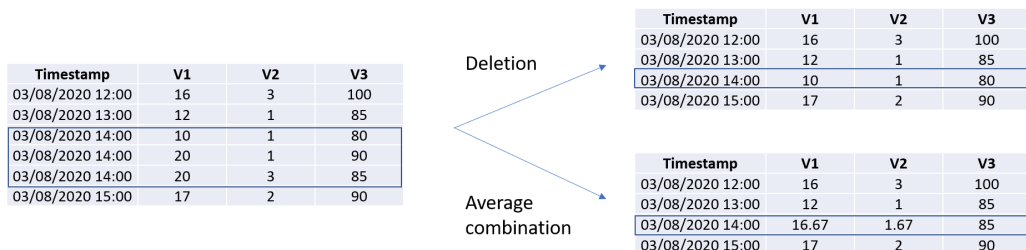


FIGURE 3. Two solutions proposed when timestamps are repeated.

solutions are mentioned without detailing the methodologies. Imputation replaces missing values with estimations using various methods. Different imputation methods can be applied to treat data when the number of missing values or their effects on the study are high. The process of imputing the missing data consists of calculating an estimation based on the available data and replacing them in the set. There are simple techniques and others that are more sophisticated, but the effectiveness of each of the methods is not known because it will depend on the nature of the series.

The simplest method is imputation based on the average of all available data or the average of the maximum and minimum values. Similarly, the median can be used to estimate all missing values. Another possibility is to use interpolation between the last value and the next value available in each gap. Depending on the case, we can use interpolation of a different degree, although the most common is linear interpolation. On the other hand, Machine Learning techniques can be used to estimate missing values. In these cases, the values available prior to data loss are used to estimate the best model. The predictions made with that model will be the estimates of missing data in the series. Finally, a variant of the KNN algorithm can be used to find similar patterns in the time series. This method called KNPTS finds the most similar subseries in the available history and estimates the future values with combinations of the values that follow each of the selected subseries. [41].

E. ACCURACY

1) PROBLEM IDENTIFICATION

Four metrics are defined to measure the accuracy of the data and they should be interpreted according to their nature. These metrics provide information about the degree of reproducibility of measured values that may or may not be close to real world values. These metrics are calculated for each numerical variable in the data set. The final value is the average of the results obtained for each variable.

First, the *Range* metric quantifies the values within the lower and upper bands which can be provided or simulated before executing this metric. Therefore, measuring the *Range* value requires expert knowledge of the problem and ignorance can significantly vary the tolerance level and

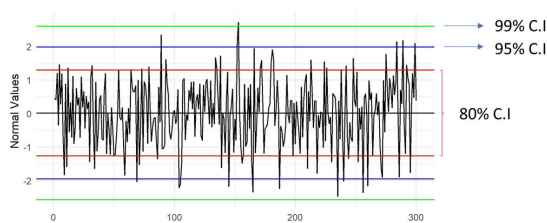


FIGURE 4. Example of simulated values following a Gaussian distribution and the three boundaries of the confidence intervals (CI) with confidence levels of 80%, 95% and 99%.

consequently the result of DQ. Let $y_{min} = (y_{min}^1, \dots, y_{min}^N)$ and $y_{max} = (y_{max}^1, \dots, y_{max}^N)$ be the sets of lower and upper values, respectively. The subset of Y contains values into these limits and is defined as follows:

$$Y_j^* = \{y_{ij} \in Y \mid y_{ij} \in [y_{min}^j, y_{max}^j]\}, \quad t = 1, \dots, T \quad (12)$$

So using that notation,

$$\begin{aligned} Range &= \frac{1}{N} \sum_{j=1}^M (Range_j) \\ &= \frac{1}{N} \sum_{j=1}^N \left(\frac{Y_j^*}{T} \right) \end{aligned} \quad (13)$$

The remaining *Accuracy* metrics are *Consistency*, *Typicality* and *Moderation*. These three assume a Gaussian distribution in variables and 80%, 95% and 99% confidence intervals (CI) are built using the data available for each numerical variable. *Consistency* is the proportion of values in an interval with a confidence level of 80% and its corresponding z-score is 1.28. The set of consistent values for variable j is expressed as follows

$$Y_j^C = \{y_{ij} \in Y \mid y_{ij} \in (\bar{y}_j - 1.28s_j, \bar{y}_j + 1.28s_j)\} \quad (14)$$

where \bar{y}_j is the average, and s_j is the standard deviation of the variable j calculated in a random selection of 30% of the points in the first part of the series.

So using that notation,

$$\begin{aligned} \text{Consistency} &= \frac{1}{N} \sum_{j=1}^M (\text{Consistency}_j) \\ &= \frac{1}{N} \sum_{j=1}^N \left(\alpha_C - \frac{Y_j^C}{T} \right), \quad t = 1, \dots, T \quad (15) \end{aligned}$$

where, $\alpha_C = 0.8$ is the level of confidence in that interval. Similarly, *Typicality* is calculated using a $\alpha_T = 0.95$ confidence level and a z-score of 1.96 and *Moderation* is calculated using a confidence level of $\alpha_M = 0.99$ and a z-score of 2.58. The corresponding typical values set and moderated values set are named Y_j^T and Y_j^M , respectively.

2) SOLUTION

The proposed methods to obtain high *Accuracy* are different in the *Range* metric than in the rest of the normality metrics. Although the methods to solve *Range* can be applied in *Consistency*, *Typicality* and *Moderation*, it must be considered that certain manipulations in the data could cause the distribution to change and they could no longer be considered Gaussian. The first solution is the simplest and consists of removing the values that are outside the boundaries. The remaining are different imputation techniques, that is, the estimation of the values identified by other data with more appropriate values. All the methods explained above to increase the value of the *Completeness* metric can also be applied with data out of range or outside the normality bands. In addition, the band values can be used to calculate other types of estimates. On the one hand, the values can be imputed by the mean value between the minimum and maximum allowed, that is, the mean of the upper and lower limits. On the other hand, it is possible to impute the values that exceed the upper limit using the maximum allowed, and the values that are below the lower limit using the minimum value allowed. All these methods, as we have commented, affect the distribution of the data. Therefore, an additional method is proposed to try to increase the *Consistency*, *Typicality* and *Moderation* values. A random period of initial data is used to calculate the theoretical mean (μ) and standard deviation (σ) of the data distribution. Once these statistics are calculated, random data that follow a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$ are generated. These values are used as estimates of the values outside of normality in each metric.

IV. IMPLEMENTATION OF DATA QUALITY METRICS

This section describes the complete flow from data acquisition to obtain the set with the desired quality. The process is illustrated in Figure 5.

The first step after accessing the data set is the identification of the Gaussian variables to assign weights to the *Consistency*, *Typicality* and *Moderation* metrics (A). Next, the existence of the Reference Data Set, Range Data Set, Maximum Time Difference and Unit of Time is checked (B). If they are unavailable, they are simulated. The third step is

the computation of DQ metrics using the *DQ* function in the complete set or by moving windows (C). If all the metrics reach the highest quality, it is concluded that the analysis set is correct. If the quality is not 1, a list of the metrics is returned in the order in which they should be treated. Next, using the *deepDQ* function, the problems with the first metric in the list are analysed in depth (D). Finally, the decision is made to rectify the data set. In that case, using the *handleDQ* function the metric is corrected until it reaches a value of 1 (D) and the metrics are recomputed in step C. If data are not modified in relation to this metric, they are removed from the list and if there were more items left in the list, it returns to step D using the next metric in the list. After the complete inspection of the list is done, the final modified data set is available.

A. NORMALITY CHECK

The first step in the process of calculating DQ is the search for variables that are expected to be normal in the data set. It is decided which of the available numerical variables follows a Gaussian distribution and which do not by means of the Shapiro-Wilks test [42]. This test considers the normality of the data as the null hypothesis, and a p-value lower than 0.05 is considered significant, leading to the rejection of the null hypothesis and the assumption of normality in the distribution.

A 30% random sample is taken from the beginning of the series (if possible in the first third of the time series) from each numerical variable and the Shapiro-Wilks test is evaluated for every sample. If the test allows us to decide that none of the available variables follow a Gaussian distribution, then the metrics *Consistency*, *Typicality* and *Moderation* are not calculated and their weights are set to zero. The remaining weights corresponding to the other metrics are recomputed as identically distributed. Therefore, 1/8 weights are assigned to each of the remaining eight metrics. Otherwise, some of the variables in the data set are considered to follow a Gaussian distribution and the normality metrics will have a weight of 1/11, the same as the rest of the variables.

B. CALCULATION OF MISSING PARAMETERS

It is important that the user knows the data to be analysed well and provides as much information as possible to obtain accurate results. Thus, it is ensured that the values of the metrics are in accordance with the quality standards expected from the data. Providing wrong values of initial parameters may lead to errors in the conclusions of the data quality analysis, for instance, assuming an incorrect frequency of the data or time units. When initial information is lacking, the system is prepared to simulate these values in the most realistic way to avoid erroneous conclusions.

1) REFERENCE DATA SET

The reference data set is made up of as many variables as the data set to be analysed has. Each of these variables must be

TABLE 1. Summary of problems and solutions for each of the proposed DQ metrics.

Dimension	Metric	Problem identification	Solutions
Conformity	Names	Wrong variable names	Copy names or formats from the reference data set
	Format	Different data formats	Delete variables with incorrect names or format
Uniqueness	Time Uniqueness	Repeated timestamps	Delete repeated observations Combine the repeated observations
Timeliness	Timeliness	Excessive waiting times between observations	Add missing observations
Accuracy	Range	Values out of range	Delete observations containing values out of range Imputation of values out of range
	Consistency	Values out of the 80% confidence interval	
	Typicality	Values out of the 95% confidence interval	
	Moderation	Values out of the 99% confidence interval	
Completeness	Global Completeness	There are missing values	Delete observations containing missing values Imputation of missing values
	Completeness by Observations	Some observations are lost	
	Completeness by Variables	Some variables are lost	

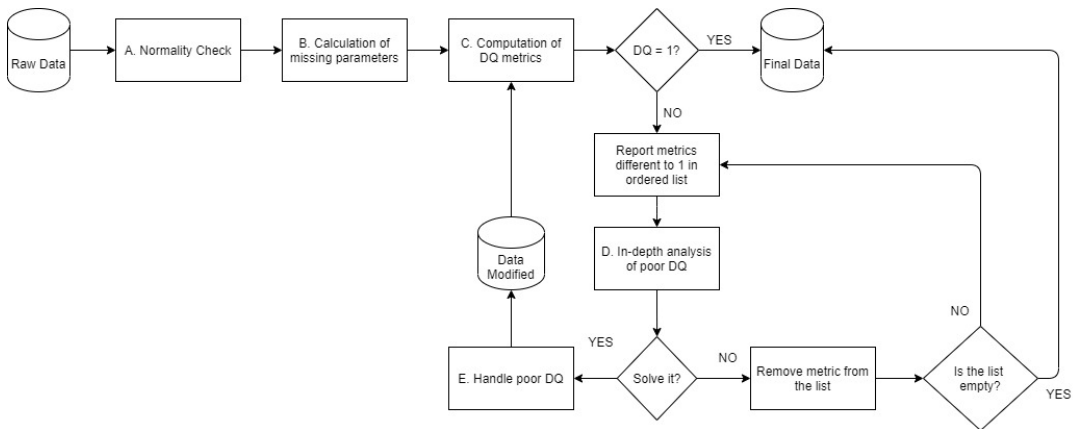


FIGURE 5. Flow for the detection, inspection and resolution of poor DQ problems.

correctly named. The content of each variable is the type of value expected from each variable.

If this reference set is not provided by the user when calculating the *Formats* and *Names* metrics, a set will be calculated by extracting the information from a random sample of 30% of the data located in the first third of the series.

2) RANGES DATA SET

The data set of ranges has the same number of variables as the data set to be analysed. The variables have the same name as the original data set and each contains two values. The first value of each variable contains the minimum value allowed in each case and the second is the maximum.

In the event that this information is not available at the time of computing the *Range* metric, a set of ranges will be calculated by extracting the minimum and maximum values of each variable in a random sample of 30 % of the available data located in the first third of the series.

3) MAXIMUM TIME DIFFERENCE AND UNITS

Observations are expected to be received at regular time intervals in time series data sets. For this reason, there is a maximum time difference allowed between the observations with their corresponding unit of time.

The value with the highest mode is extracted from a random sample of 30% of the available data located in the first third of the series if this information is not available at the time of calculating the *Timeliness* metric. In the same way, the unit of that time difference is extracted from that most common value in that random sample.

C. COMPUTATION OF DQ METRICS

This step can be performed in two ways depending on the interests of the user, as described in the following subsections. On the one hand, the value of the metrics can be obtained from the complete data set. On the other hand, there are three different ways to compute quality metrics by moving windows to inspect the evolution of DQ over time. This method of computing metrics is recommended when a large amount of data is available, because the information provided will be more accurate.

1) OVERALL DQ

The main objective in generalising the concept of DQ is to provide a general value that allows the comparison of the quality of different data sets. A value between 0 and 1 is obtained for each quality metric, in addition to a general value that is calculated from a combination of the remaining values.

2) MOVING WINDOWS

Calculation of the DQ metrics using moving windows provides information on the evolution of the metric values during the course of the temporal data. It is convenient that the windows are of the same size, that is, the scores are calculated with the same amount of data or that the windows start at the same point. This is forced so that the achieved values are comparable and bias is avoided. Three methods for calculating the quality of the data by moving windows are explained. On the one hand, it is decided whether the size of the window is fixed or changing. On the other hand, if the values of the series can belong to more than one window, that is, if the windows can overlap.

- Figure 6 shows the case in which the size of the windows is constant and does not overlap. In this option, the values to be analysed are completely different in each iteration of the DQ metrics.
- In the second case, the partition by windows of a constant size with overlap is shown. In this way, as it can be seen in Figure 7, in each iteration the first data of the window are deleted and new data are added at the end; however, some overlap is allowed in the windows. Therefore, some final values of one iteration are used to be the first values in the next computation of the DQ metrics.
- Finally, Figure 8 shows the case of a window of varying sizes and overlapping. In this case, the beginning of the interval remains fixed and more data are added to the window in each iteration.

Once the DQ value for each metric is known, they are ordered in a specific manner. This order places the perfect metrics at the bottom of a list and orders the rest so that their arrangement affects the rest as little as possible. In this way, if fixes are required for all metrics, the metrics relative to *Conformity* (*Names* and *Format*) will be dealt with first. These metrics could add or remove all variables and for that reason should be fixed first. The time metrics (*TimeUniqueness* and *Timeliness*) are then inspected. The reason for studying these metrics at this point is that the first could remove all observations from the data set and the second could add missing values to the time series. Next, the *Accuracy* metrics will be discussed, starting with *Range* and continuing with the three normality metrics (*Consistency*, *Typicality* and *Moderation*). Finally, the *Completeness* metrics, impute both the initial missing values and those that could have been added in previous steps.

Each time the data set is modified to raise quality in one of the dimensions, the quality metrics are recalculated. Thus, the list of metrics are modified. Once a list of metrics is obtained with values that are substantially good for the user, the new set can be saved and the modified data downloaded for possible analysis.

D. IN DEPTH ANALYSIS OF POOR DQ

The information obtained from an in-depth analysis of each metric is as follows

- *Names*: Name in the reference set and analysis set of variables with incorrect names and their positions in the original set.
- *Formats*: Formats in the reference set and analysis set of the malformed variables and their positions in the original set.
- *Time Uniqueness*: Repeated dates and frequencies they appear in the analysis data set.
- *Timeliness*: Instant in which a temporary wait longer than allowed begins and ends, in addition to the waiting time and the number of values that were expected to have been received in that period.
- *Range*: Name of the variables with out-of-range values accompanied by information about these values. This information can be in two forms: the value of the *Range* metric for each variable or the time positions in which these out-of-range values occur.
- *Consistency, Typicality and Moderation*: Name of the normal variables with more values outside the confidence intervals (CI) than allowed in each case. There are two options to show the information: the value of the corresponding metric in each of the variables or the time of all values outside the CI. Note that some of the values that will be shown outside the CI do not necessarily have to be incorrect because we assume that in every normal distribution, there will be a number of values outside the CI corresponding to the confidence level with which the interval has been built.
- *Completeness*: Name of variables with missing values with some useful information. Two options: The value of the *Completeness* metric by variables or the time in which data were lost for each of the variables.

E. HANDLE POOR DQ

One of the possible actions for the treatment of low data quality is the use of corrective functions for each of the metrics that do not reach the maximum quality score. These actions are mentioned in Section III. The options available for each DQ metric are as follows.

- *Names*: If any of the variables do not have the same tag or name as the variables given in the reference set, there are two options. First, defective variable are eliminated from the study set. The second option is the manual change of the name of the variable in which it fails, in the case that the expected name is known.
- *Formats*: There are several functions in the base R package that allow switching from one format to another. However, it must be borne in mind that this is not always possible. Therefore, apart from the typical format change options such as changing from character to numeric, it is possible to eliminate the variable from the study.
- *Time Uniqueness*: The first option available is to eliminate observations that contain repeated dates and leave only the first one. In the case of uncertainty about which of them provides more information, it is possible to

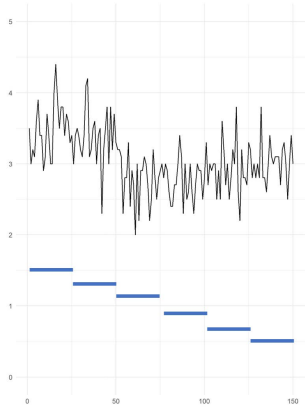


FIGURE 6. Constant windows without overlapping.

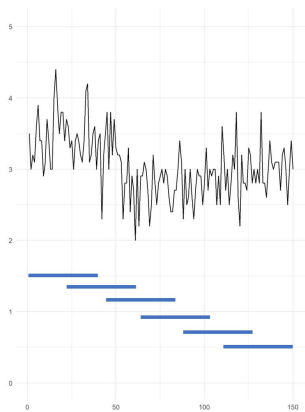


FIGURE 7. Constant overlapping windows.

combine the values of the rest of the variables by arithmetic mean, median, minimum or maximum.

- **Timeliness:** The low-value solution functions create the missing dates in the time variable. Once these values are generated, there are several options for completing the corresponding observations in the remaining variables. The four general options are using the minimum, the maximum, the arithmetic mean and the median. If none of these values adapts to the behaviour of the series, there is the option of not filling those gaps and leaving a null value in them. In that case, the *Completeness* metric changes its value, because artificial missing values are being introduced.
- **Range:** There are four simple methods that replace the values out of range by the average, the median, the minimum or the maximum value of that variable. Another possibility is to substitute the values by the mean of the last value in the range received and the next value. In addition, the KNPTS algorithm can be used

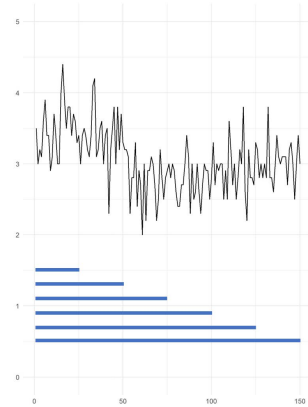


FIGURE 8. Non-constant windows with overlapping.

to estimate out of range values. Finally, there are two methods related to the values introduced in the Ranges Data Set. On the one hand, a function that replaces the values out of range with the average of the minimum and maximum allowed values for each variable. On the other hand, the values that exceed the values allowed are replaced by the maximum and the values that fell short are replaced by the minimum value allowed. Those options are implemented using the variables.

- **Consistency, Typicality and Moderation:** The options available are the same as in the *Range* case but it is not recommended to solve problems related to normality distribution because it can undermine the properties of the time series with a Gaussian distribution. Alternatively, the mean (μ) and standard deviation (σ) of the data in the first part of the time series are calculated and used to simulate random data that follow a Gaussian distribution $\mathcal{N}(\mu, \sigma)$. The last option is to do nothing with variables with values out of normality but to consider that the problem with their distribution exists.
- **Completeness:** There are four simple methods that replace the missing values with the average, the median, the minimum or the maximum of the available values in each variable. Another option is to substitute the missing values with the mean of the last value in the range received and the next value. Finally, the KNPTS algorithm can be used to estimate the missing values.

V. THE DQTS R PACKAGE

The work explained in the previous sections was implemented using the statistical software R. In R, the fundamental unit of shareable code is the package. A package bundles code, data, documentation, and tests together, and is easy to share with others. The `dqts` R package is available in GitHub in the following link for use by the entire R community.¹

¹<https://github.com/MeritxellGomez/dqts-R-package>

The R library is made up of four main functions explained below.

- The `DQ` function performs the three steps described in Section IV. First, it checks the normality of the variables and the availability of the necessary parameters in the input arguments. If unavailable, they are estimated. Finally, the values of the DQ metrics are calculated and this function allows two ways to do so, the overall one and by windows in three different ways.
- The `deepDQ` function takes the data, the name of the metric to inspect, and the parameters required for that metric as inputs. This function returns precise information regarding the data failures in the selected metric.
- The `handleDQ` function estimates solutions to faults found in the data for the metric introduced as an argument. It returns the data set with the necessary changes for that metric to achieve the highest quality score.
- The `plotDQ` function allows visualisation of the quality of the data. The output of the `DQ` function is introduced as an argument. In the case that quality has been calculated in the complete data set, the `plotDQ` function shows a bar graph where each bar indicates the numerical value of each of the metrics with magnitudes between 0 and 1. On the other hand, if the quality of the data has been computed by windows, a scatterplot is displayed with as many lines as metrics have been calculated and the time evolution of the metrics is shown.

VI. VALIDATION IN CASE STUDIES

This section presents an evaluation of the metrics developed by applying them to three data sets. To demonstrate the effectiveness of the quality metrics for different scenarios, it was decided to test the proposed method on a simulated data set created by the authors, on an open data set and on a real data set.

A. DETECTION AND SOLUTION IN SIMULATED DATA

1) DATA DESCRIPTION

In the first experiment, a multivariate time series was created with minute data and information on five random variables collected over one day. Three of these variables (G1, G2, and G3) follow a Gaussian distribution, another follows a chi-square distribution (V1) and the last was a binomial variable (V2) that takes values 0 or 1. In addition, the temporal variable (timestamp) collects minute values from “2021-01-01 00:00:00 UTC” to “2021-01-01 23:59:00 UTC”. In summary, six variables were available for the analysis of this data set.

Problems related to each of the DQ dimensions defined in table 1 were simulated. The table 2 contains detailed information regarding the simulations to be detected and fixed throughout this section.

2) DQ ANALYSIS

The multivariate time series after simulating quality errors was displayed in Figure 9. Next, the process of Figure 5 starts to examine and correct the quality of the data.

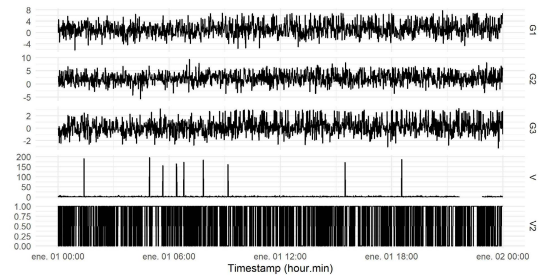


FIGURE 9. Visualisation of data distribution of all 5 variables of the simulated data set.

First, normality was evaluated and it was established that the variables G1, G2, and G3 follow a normal distribution, with p-values of 0.87, 0.15 and 0.59 in the Shapiro-Wilk test, respectively. This statement was possible because the three p-values were greater than 0.05, the null hypothesis of normality in their distributions can be considered as valid. For that reason, the weights for the *Consistency*, *Typicality*, and *Moderation* metrics were set the same as the other metrics. In this case, the combination of the metrics was balanced; therefore, the weights were all 1/11. Regarding the input parameters, the type of each variable was known, so the reference set was given as input in the DQ function. On the other hand, some fictitious ranges were established that were also entered as inputs in the function. Finally, we know the data acquisition frequency; therefore, the maximum value of time difference was set at 1 min.

In the first evaluation of the quality metrics in the complete set, the results of the metrics were obtained, as listed in Table 3. The order in which the results appear indicates the order in which problems should be addressed.

It can be seen that the fault related to *Names* was in the variable that was in position number 5, using an in-depth inspection of the problem. This variable was called V1 in the reference set and is now called V. As for *Formats*, V2 should be an integer, according to the reference set, but it was received as a character. These two problems were solved by renaming the variable V by V1 and changing the character values of V2 to integers.

Following the established order, the next metric that falls short of the highest quality score was *TimeUniqueness*. One timestamp was found more than once as shown in Table 4. This problem was solved by eliminating the observations that were repeated.

The *Timeliness* was discussed below. An in-depth inspection of the problems related to this metric indicates that temporary waits are due to a stop between 2021-01-01 03:15:00 and 2021-01-01 03:20:00. Four observations were excluded. The missing observations were generated using the `handleDQ` function, and the remaining values were imputed by the median of each variable.

None of the three normal variables (G1, G2, or G3) achieved the highest score for *Consistency*, *Typicality*

TABLE 2. Problems forced to appear in simulated data.

Metric	Variable	Description
Names	V1	renamed to "V"
Format	V2	Character type instead of numeric
Time Uniqueness	timestamp	"2021-01-01 00:09:00 UTC" is repeated 4 times
Timeliness	timestamp	time gap between "2021-01-01 03:15:00 UTC" and "2021-01-01 03:20:00 UTC"
Range	V1	10 values out of range in random position
Consistency	G1	10% of the correct values move outside the 80% of CI
Typicality	G2	10% of the correct values move outside the 95% of CI
Moderation	G3	10% of the correct values move outside the 99% of CI
Completeness	V1	5 % of NA in random position

TABLE 3. DQ metrics in overall evaluation.

Metric	Value	Interpretation
Names	0.8333	83.33% of variables with correct names
Format	0.8333	83.33% of variables with correct formats
Time Uniqueness	0.9979	99.79% of unique values in timestamp
Timeliness	0.9972	99.72% of correct waiting times between consecutive timestamps
Range	0.9988	99.88% of values within ranges
Consistency	0.9492	94.92% of values following Gaussian distribution according to CI with 80% of confidence level
Typicality	0.9556	95.56% of values following Gaussian distribution according to CI with 95% of confidence level
Moderation	0.9791	97.91% of values following Gaussian distribution according to CI with 99% of confidence level
Completeness	0.9915	99.15% of data received have a value
Completeness Observations	1	100% of observations have at least one value
Completeness Variables	1	100% of variables have at least one value

TABLE 4. Deep inspection of time uniqueness quality.

Repeated Date	Frequency
2021-01-01 00:09:00	4

TABLE 5. Deep inspection of timeliness quality.

Loss Start Date	Loss End Date	Waiting times	Missing values
2021-01-01 03:15:00	2021-01-01 03:20:00	5 mins	4

and *Moderation*. Note that these three metrics are related. If *Moderation* lowers its quality value, *Typicality* and *Consistency* also worsen. Suppose that the analyst needs those variables in the future. Therefore, they can not be deleted, and the best option for dealing with low quality is to simulate the values using the mean and standard deviation. The means of the first third of G1, G2 and G3 were 0.91, 1.91 and 0 and their standard deviations were 2.07, 1.97 and 1.04, respectively. The rest of the time series was imputed by random values following a $\mathcal{N}(0.91, 2.07^2)$, $\mathcal{N}(1.91, 1.97^2)$ and $\mathcal{N}(0, 1.04^2)$ distributions.

Finally, *Completeness* was inspected in depth, and it can be seen that the metric was 1 for all the variables except for V1 which was 0.9493. The KNPTS method was used to impute the missing values in V1.

Once the DQ analysis has been completed and the conflicts of the different variables resolved, the total quality of the final set was 1.

B. DETECTION AND SOLUTION IN OPEN DATA

1) DATA DESCRIPTION

The airline passenger data set contains monthly data of airline passengers in thousands from January 1949 to December 1960 [43]. AirPassenger is one of the most well-known

open data sets available in R. It is a univariate time series with 144 monthly values that does not follow a Gaussian distribution.

This section takes advantage of the availability of data to compare the precision of time series predictive models when they were trained with quality data and when they were trained with data that lack quality.

An ARIMA model was trained using the univariate time series of the original data for the first 11 years to predict the data for the previous year. The ARIMA model that best fits the data was an ARIMA(1,1,0)(0,1,0)[12] model. This model was chosen because of the clear trend and seasonality presented in the time series. The forecast for the next year of data was shown in Figure 10. The graph also shows the confidence bands of the intervals with a confidence levels of 80% and 95%. The RMSE was used to validate the precision of this forecast. We obtained an RMSE of 23.93. The results were used as benchmarks.

Next, *Completeness* and *Range* problems were forced into the training set for the first 11 years of data. In the first case, missing values were simulated in 12 values corresponding to 1955. In addition, an extremely high value (1500) was added in May 1957.

2) DQ ANALYSIS

The DQ flow in Figure 5 was executed in the set in which we simulated the problems. First, the variable that measures passengers does not follow a Gaussian distribution; therefore, the *Consistency*, *Typicality* and *Moderation* metrics will not be computed. In this case, the minimum value of the range that is allowed for the number of passengers is zero, because a negative number is meaningless. The maximum value was simulated to be 1000. The reference data set was randomly calculated from the first third part of the time series.

TABLE 6. DQ metrics and RMSE achieved in forecasting using original and simulated data.

Metric	Original Data	Simulated Data
Names	1	1
Formats	1	1
Time Uniqueness	1	1
Timeliness	1	1
Range	1	0.99
Completeness	1	0.95
DataQuality	1	0.99
RMSE	23.93	90.68

In the first computation of the DQ of the original data set, all metrics achieved the maximum score and therefore the total value of the quality of this set was 1. Therefore, the AirPassengers set does not originally present any problems with DQ. The quality metrics in the simulated set assign a value of 0.95 to *Completeness* and a value of 0.99 to *Range*. These two values indicate that there were two problems with DQ. The total value of the quality of this time series was 0.99, with null weights assigned to the three metrics related to normality.

The effect of poor data quality can be seen when the search process for the best time series model was repeated with data from the same time period. The seasonality was no longer captured and the best model was ARIMA (0,1,1). Those issues were reflected in the precision of future estimates. The ARIMA model cannot accurately predict the next values as shown in Figure 11. Furthermore, the confidence interval bands widened, highlighting the uncertainty associated with new predicted values. The RMSE obtained in this forecast was 90.68, which represents an increase of 378% compared to the forecast made with the original data that had full quality. This result suggests that if the quality of the data is not controlled, erroneous results can be obtained with algorithms that worked correctly.

Forecasts from ARIMA(0,1,1)

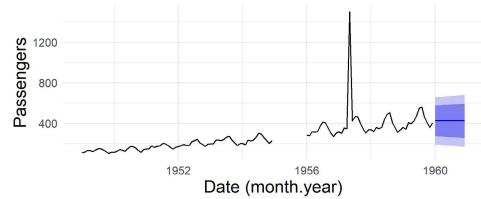


FIGURE 11. One year ahead forecasting training ARIMA model with simulated data with 0.95 in Completeness and 0.99 in Range.

Forecasts from ARIMA(1,0,2)(0,1,1)[12] with drift

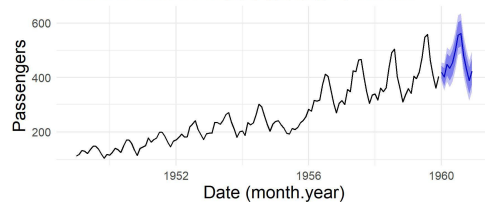


FIGURE 12. One day ahead forecasting using fixed data with highest quality.

the correction of *Completeness*. After these two corrections, the total data quality was achieved. The model that best fits these new training data was ARIMA (1,0,2)(0,1,1)[12]. The forecast data for the following year were shown in Figure 12. The improvement was remarkable. The confidence interval bands were adjusted to the predicted data and the RMSE error obtained for this forecast was 30.70. This value represents a 66% error reduction compared to the forecast made with the erroneous data. The difference between the RMSE obtained with the corrected data and original data was 6.77. In other words, the RMSE increases only 28%.

This simulation study shows, on the one hand, the importance of having quality data at the time of starting statistical analyses that give rise to predictions of future values. On the other hand, the advantages that the package of quality metrics presents to detect and solve quality problems quickly and effectively.

C. DETECTION AND SOLUTION IN REAL DATA

1) DATA DESCRIPTION

Monitoring of electrical power systems in industry has been on the rise in recent years. Forecasting electricity demand using Time Series techniques and Machine Learning is one of the tasks on the rise in the study of the production and consumption of electricity. The study of electricity consumption data began with the task of capturing data from high-frequency meters. These data were expected to be cyclical because of the activation behaviour of certain devices. However, they were not expected to follow a normal

Forecasts from ARIMA(1,1,0)(0,1,0)[12]

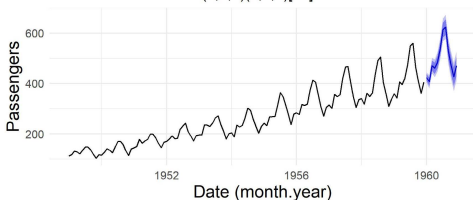


FIGURE 10. One year ahead forecasting training ARIMA model with original data.

Table 6 reflects the quality problems that appear in the new set and the impact that a decrease in DQ has on an increase in the prediction error.

The `handleDQ` function was used to solve problems with *Range* and *Completeness*. On the one hand, the correction of the value that was out of range was made using the mean method. On the other hand, the KNPTS method was used for

TABLE 7. Metrics that not achieve the highest value of quality in electricity consumption dataset.

Metric	Lowest Value	Highest Value
TimeUniqueness	0.9583	1
Timeliness	0.9600	1
Range	0.6667	1

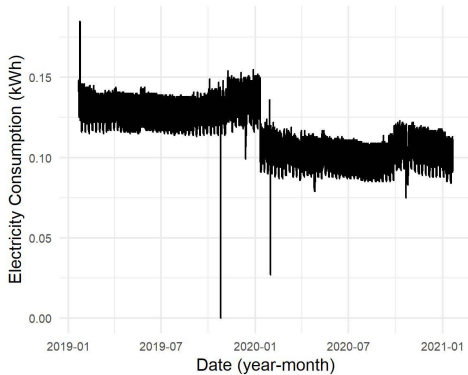


FIGURE 13. Distribution of the electricity consumption

distribution; therefore, the normality metrics in this case were weighted with a null weight.

In this case, data were taken from a meter of a sentry box in which an electric pump is installed that supplies water to nearby farms. The small construction site is located in the facilities of the Tekniker technology centre, in Eibar, Spain. The data set corresponds to a univariate time series capturing hourly electricity consumption data measured in kWh. A total of two years of data were captured from 21 January, 2019 at 00:00 a.m. to 21 January, 2021 at 11:00 p.m. Figure 13 shows the distribution of the data over time.

2) DQ ANALYSIS

The flow in Figure 5 was executed to compute the DQ in the time series of the electricity consumption data. Figure 13 shows the distribution of the data. In this case, the allowed ranges for the electricity consumption variable were not known, but the function that automatically generates them was used, taking a random sample of 30% of the data located in the first third of the series. Figure 14 shows the evolution of the quality metrics computed using constant 24-hour windows without overlap.

The value of the DQ in each moving window ranged from 0.958 to 1. The reason why the quality was not 1 in the entire set was that some metrics have detected quality problems at different points in the series. The system returns a list like that in Table 7 in which it can be seen the metrics that fail in the order in which they should be treated.

The deepDQ function was used in this step to check the metrics that did not reach the highest quality score.

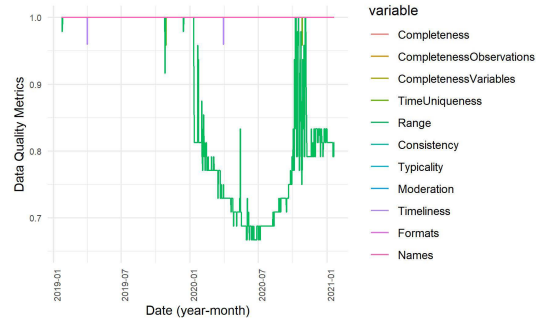


FIGURE 14. Evolution of the DQ metrics.

TABLE 8. Deep inspection of time uniqueness quality.

Repeated Date	Frequency
2019-07-27 02:00:00	2
2020-10-25 02:00:00	2

TABLE 9. Deep inspection of timeliness quality.

Loss Start Date	Loss End Date	Waiting times	Missing values
2019-03-31 01:00:00	2019-03-31 03:00:00	2 hours	1
2020-03-29 01:00:00	2020-03-29 03:00:00	2 hours	1

Table 8 lists the output of the *Time Uniqueness* inspection. Low values of that metric are due to repetitions on two different dates. The first on 2019-07-27 02:00:00 and the second on 2020-10-25 02:00:00. Both with a frequency of 2. The Deletion method was used to solve this problem.

Next, the *Timeliness* metric was inspected in depth and the information collected in Table 9 was obtained. Two waiting times of 2 hours were identified at two different points in the series. The first between the dates 2019-03-31 01:00:00 and 2019-03-31 03:00:00 and the second between the dates 2020-03-29 01:00:00 and 2020-03-29 03:00:00. Both involve a loss of one observation. Those two observations were generated. Note that this arrangement introduces two missing values in the series and causes the *Completeness* to decrease from 1 to 0.999.

Finally, *Range* was analysed and it was seen that in the first half of the series there were values out of range on the dates that were collected in Table 10. In the second half of the time series, none of the windows achieved the highest score in *Range*. This was due to a change in the data trend. We identified by the low value of *Range* for all iterations that the boundaries calculated in the first section of the series could no longer be applied. In that case, if the metrics were calculated daily after data acquisition, after collecting a significant number of followed *Range* values different from 1, we could make the decision to recompute the maximum and minimum values allowed. In the calculation we would use the consumption values for the year 2020, which is when a change in the distribution of the data was identified. The values identified

TABLE 10. Values out of range in 2019.

Date	Consumption (kWh)
2019-01-23 17:00:00	0.185
2020-10-25 01:00:00	0.027
2019-10-25 02:00:00	0.0
2019-10-25 03:00:00	0.0
2019-10-25 04:00:00	0.0
2019-10-25 05:00:00	0.004
2019-12-13 09:00:00	0.099

TABLE 11. Values out of range in 2020.

Date	Consumption (kWh)
2020-01-29 09:00:00	0.136
2020-01-30 14:00:00	0.027

out of range in the second half of the time series using the new boundaries were listed in Table 11.

VII. DECISION SUPPORT SYSTEM FOR DATA QUALITY MANAGEMENT AND IMPROVEMENT

The work presented in this article contributes an important advance toward the development and integration of Decision Support System (DSS) to improve the quality management of streaming time series data.

A system that supports the decision-making process to improve the quality of streaming time series data has been designed and implemented based on the methodology and the `dqts` R package. The system provides functionalities and mechanisms to assess and handle problems regarding the quality of the data. In addition, it provides support for the end user and helps to select the most appropriate alternatives, regardless of the characteristics of the end user (that does not require an analyst profile to use the module).

The *DQ-REMAIN (Data Quality REport Management and ImproveMent)* system, improves the data management, focuses on the DQ improvement, and consequently, the final exploitation through the KDD (Knowledge Discovery in Databases) process. The findings will be more accurate and will improve the competitiveness and capabilities in areas where these types of systems are integrated. Manufacturing and energy sectors, for example, could be potential users and beneficiaries of this type of systems.

The functionality of the system is described by 4 functional blocks that can be seen in Figure 15 and are described below.

- Database connectivity: it provides a connection to the data source where the information is located or acquired (data streaming). It provides access to the data, represented as “Raw data”.
- Engine: it calls the functions provided by the `dqts` module to generate the results for the data quality assessment.
- `dqts` module: it provides the functionalities associated with DQ assessment.
- User Interface: it informs throughout the visualisation about the status of the DQ based on the representation

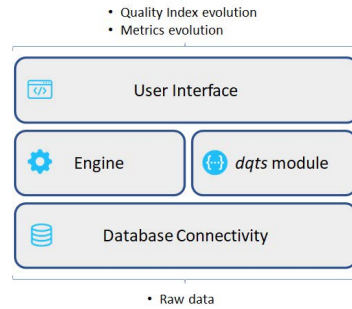


FIGURE 15. Functional blocks of DSS.

over time of the value obtained for the “Metrics” and also for the “Quality index”. In addition, it facilitates interaction between the module and the end user to solve quality problems.

The system provides a new functionality that, integrated into existing data management, monitoring and analysis platforms, allows monitoring and correcting the quality of the data over time. The quality of the data improves significantly and data do not lose value for further exploitation.

Figure 16 presents the architecture of the solution for deploying the *DQ-REMAIN* system in a manufacturing information management platform. The platform is responsible for acquiring and managing all information, such as condition monitoring data along with process data, from its assets (test benches, machines, industrial robots, etc.). The platform acquires the necessary information and stores it in different databases for later exploitation using advanced analytical techniques. This enables the user to generate new insights, obtain an asset health assessment, and be able to perform predictive or condition-based maintenance. However, this generation of new knowledge depends on both the amount and quality of data received. The deployment of the *DQ-REMAIN* system within the platform, periodically computes the quality of the data received and assists in solving data quality issues.

The *DQ-REMAIN* system provides a graphical interface through which the status of the quality of the data monitored from the different assets can be visualised. Figure 17 shows an example of the computation of the quality of data obtained from a test bench. After detecting that the quality of the data has decreased over a period of time, the list of metrics that need to be improved in order of priority is shown by the *DQ-REMAIN* system. In the example of Figure 18, there are some metrics that cause a decrease in the quality index value: *TimeUniqueness*, *Timeliness* and *Completeness*. The system offers alternatives to correct the problems that arise depending on the metric being affected. Users can then select different methods to solve the low quality of each metric. In this example, the system offers some alternatives, a list of the algorithms available to solve repetitions in time variable, create missing dates and impute missing values. The end user

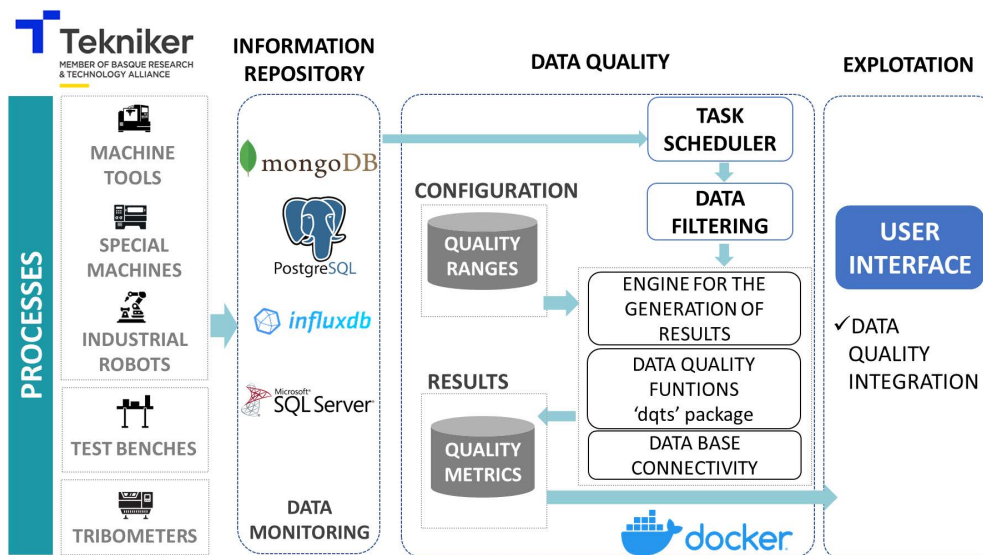


FIGURE 16. DQ-REMAIN integration into Manufacturing platform.

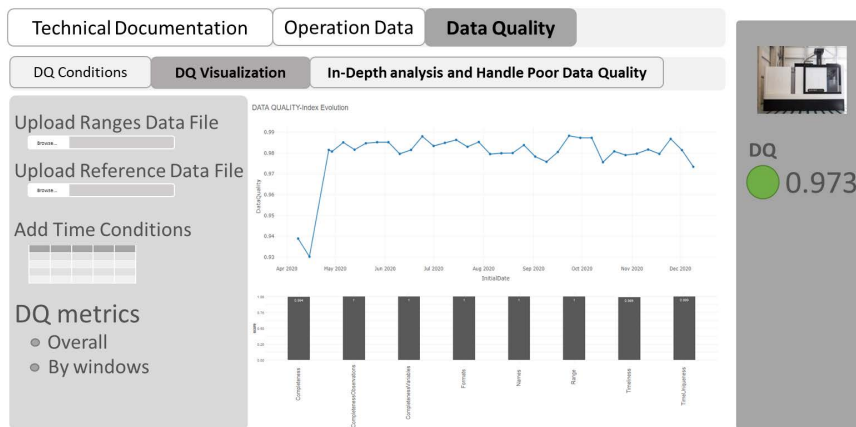


FIGURE 17. Visualisation of the computation of DQ metrics of data from a test bench.

can use the algorithm that best suits the situation and recalculates metrics. If the solution proposed by the system is good enough and the metrics improve, then it is possible to save new data in the database for future exploitation guaranteeing their highest quality.

There are many scenarios from industrial applications of IoT where the ingestion of data through sensors presents certain difficulties that affect the quality of the collected data and this type of decision support system is exploitable. Furthermore, as it is composed of functional blocks (library packages), the deployment and distribution of software for its integration into third-party solutions is

carried out quickly and easily. Only ad-hoc development will be necessary to connect with new data repositories and to integrate the visual exploitation in another type of interface.

At present, the DSS is being integrated and will be used in the facilities of Tekniker (a Technology Research Centre, www.tekniker.es) to monitor and correct quality problems in certain operating machines (mainly test benches and machining centres). The results of this improvement in the quality of the data will be exploited in the near future to make use of this good quality data, and implement a health assessment of the state of the machines. This is one more step towards Smart Factoring Systems for Industry 4.0.

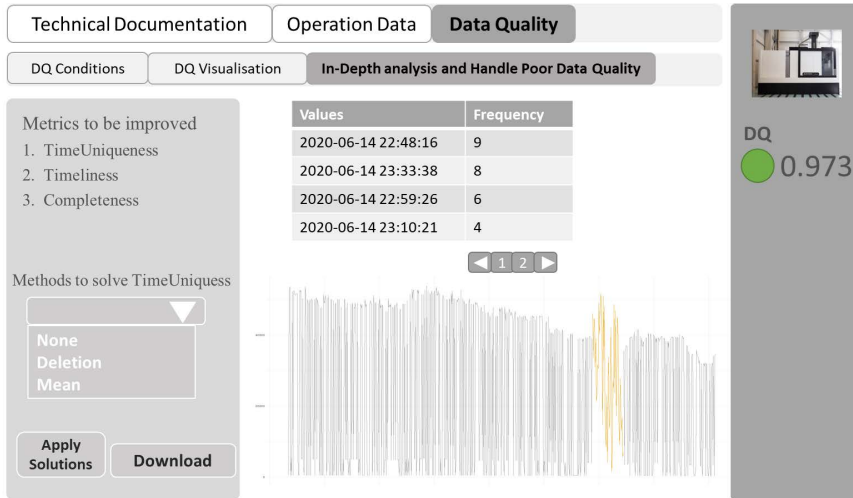


FIGURE 18. Interactive dashboard in which to choose low quality resolution methods and inspect the results.

VIII. CONCLUSION

In a world in which data plays a leading role in the evolution and fruitfulness of industries and companies, ensuring the quality of the data captured is a problem of great importance. Low-quality data greatly influences future results and conclusions reached from them. In this study, the need to define and develop a set of methodologies, procedures, and practical tools for the treatment of data quality focused on measuring and treating different aspects in time series data, the usual structure of data captured by IoT devices, has been identified. The technological novelty of this study addresses this gap through:

- The mathematical definitions of the metrics to quantify the quality of the data, beyond the existing theoretical formulation and the definition of the DQ indicator as a combination of them.
- Definition of the set of corrective functions, enhanced alternatives, to improve some aspects of quality (e.g., data imputation for missing values).
- Definition and representation of a complete methodology that describes the optimal workflow for a global process of detection, inspection, and resolution of potential problems regarding the quality of data in a time series.
- The implementation and packaging of all functions for their final use and exploitation.

This work has also been tested and validated in three data sets from different sources with different purposes to demonstrate the capacity of DQ metrics, compare the results from forecasting models model in both cases, address the quality of the data and without having done so and finally, to show the possibility of implementing the data quality treatment flow in a real case of data handled by an industrial sector.

Improving the DQ is an essential task that must be addressed early, before, and during the process of transferring

and storing data for its final exploitation. Doing this from the origin avoids many problems in the later phases. From the point of view of data analysis and the generation of detection and prediction models, having good quality data means drawing more precise and realistic conclusions. The models were developed from algorithms that learn based on the data provided. The better the data are, the more representative the models will be with respect to reality. The proposed research work presents two mechanisms to analyse and improve data quality in the early stages, once the data are collected and stored before the development of the model. First, the `dqts` package which implements all the required functionalities. Second, the decision support system, called *DQ-REMAIN*, which facilitates the management of the methodology proposed in this work for handling of the quality of data based on the `dqts` package.

REFERENCES

- [1] L. Atzori, A. Iera, and G. Morabito, "The Internet of Things: A survey," *Comput. Netw.*, vol. 54, no. 15, pp. 2787–2805, Oct. 2010.
- [2] A. Karkouch, H. Mousannif, H. Al Moatassime, and T. Noel, "Data quality in Internet of Things: A state-of-the-art survey," *J. Netw. Comput. Appl.*, vol. 73, pp. 57–81, Sep. 2016.
- [3] S. Suthaharan, *Machine Learning Models and Algorithms for Big Data Classification* (Integrated Series in Information Systems). New York, NY, USA: Springer, 2016.
- [4] L. Zhou, S. Pan, J. Wang, and A. V. Vasilakos, "Machine learning on big data: Opportunities and challenges," *Neurocomputing*, vol. 237, pp. 350–361, May 2017.
- [5] M. H. Frické, "Data-information-knowledge-wisdom (DIKW) pyramid, framework, continuum," in *Encyclopedia Big Data*. Cham, Switzerland: Springer, 2018.
- [6] S. Moore, "How to create a business case for data quality improvement," Gartner Inc., 2017. [Online]. Available: <https://www.gartner.com/en/documents/3872887>
- [7] T. C. Redman, "The impact of poor data quality on the typical enterprise," *Commun. ACM*, vol. 41, no. 2, pp. 79–82, Feb. 1998.
- [8] S. Moore, "How to stop data quality unferming your business," Gartner Inc., 2018.

- [9] B. Heinrich, D. Hristova, M. Klier, A. Schiller, and M. Szubartowicz, "Requirements for data quality metrics," *J. Data Inf. Qual.*, vol. 9, no. 2, pp. 1–32, Jan. 2018.
- [10] C. Cichy and S. Rass, "An overview of data quality frameworks," *IEEE Access*, vol. 7, pp. 24634–24648, 2019.
- [11] C. G. Jacobs, "Challenges to the quality of data-quality measures," *Food Chem.*, vol. 113, no. 3, pp. 754–758, Apr. 2009.
- [12] W. H. Woodall, "The statistical design of quality control charts," *Statistical*, vol. 34, no. 2, p. 155, 1985.
- [13] R. Y. Wang and D. M. Strong, "Beyond accuracy: What data quality means to data consumers," *J. Manage. Inf. Syst.*, vol. 12, no. 4, pp. 5–33, 1996.
- [14] Y. Wand and R. Y. Wang, "Anchoring data quality dimensions in ontological foundations," *Commun. ACM*, vol. 39, no. 11, pp. 86–95, Nov. 1996.
- [15] C. Batini and M. Scannapieco, *Data Quality: Concepts, Methodologies and Techniques*. Berlin, Germany: Springer, 2006.
- [16] P. Hodson, "Review: Data quality for the information age," *Comput. Bull.*, vol. 39, no. 6, p. 31, Dec. 1997.
- [17] L. L. Pipino, Y. W. Lee, and R. Y. Wang, "Data quality assessment," *Commun. ACM*, vol. 45, no. 4, pp. 211–218, 2002.
- [18] L. Orfanidis, P. D. Bamidis, and B. Eaglestone, "Data quality issues in electronic health records: An adaptation framework for the Greek health system," *Health Informat. J.*, vol. 10, no. 1, pp. 23–36, Mar. 2004.
- [19] N. Laranjeiro, S. N. Soydemir, and J. Bernardino, "A survey on data quality: Classifying poor data," in *Proc. IEEE 21st Pacific Rim Int. Symp. Dependable Comput. (PRDC)*, Nov. 2015, pp. 179–188.
- [20] A. Klein and W. Lehner, "Representing data quality in sensor data streaming environments," *J. Data Inf. Qual.*, vol. 1, no. 2, pp. 1–28, Sep. 2009.
- [21] M. Bovee, R. P. Srivastava, and B. Mak, "A conceptual framework and belief-function approach to assessing overall information quality," *Int. J. Intell. Syst.*, vol. 18, no. 1, pp. 51–74, Jan. 2003.
- [22] I. El Alaoui, Y. Gahi, and R. Messoussi, "Big data quality metrics for sentiment analysis approaches," in *Proc. Int. Conf. Big Data Eng.*, Jun. 2019, pp. 36–43.
- [23] L. Cai and Y. Zhu, "The challenges of data quality and data quality assessment in the big data era," *Data Sci. J.*, vol. 14, p. 2, May 2015.
- [24] C. C. G. Rodríguez and S. Servigne, "Managing sensor data uncertainty," *Int. J. Agricult. Environ. Inf. Syst.*, vol. 4, no. 1, pp. 35–54, Jan. 2013.
- [25] C. Batini, C. Cappiello, C. Francalanci, and A. Maurino, "Methodologies for data quality assessment and improvement," *ACM Comput. Surveys*, vol. 41, no. 3, pp. 1–52, Jul. 2009.
- [26] A. Maydanchik, *Data Quality Assessment (Data quality for practitioners)*. Basking Ridge, NJ, USA: Technics Publications, 2007.
- [27] R. Gitzel, S. Turring, and S. Maczey, "A data quality dashboard for reliability data," in *Proc. IEEE 17th Conf. Bus. Informat.*, Jul. 2015, pp. 90–97.
- [28] R. Gitzel, "Data quality in time series data: An experience report," in *Proc. CEUR Workshop*, 2016, pp. 41–49.
- [29] R. Gitzel, S. Subbiah, and C. Ganz, "A data quality dashboard for CMMS data," in *Proc. 7th Int. Conf. Oper. Res. Enterprise Syst.*, 2018, pp. 170–177.
- [30] A. Richter, C. Schmidt, M. Krüger, and S. Struckmann, "DataQieR: Assessment of data quality in epidemiological research," *J. Open Source Softw.*, vol. 6, no. 61, p. 3093, May 2021.
- [31] Y. Dong, Y. Kazachkova, M. Gou, L. Morgan, T. Wachsman, E. Gazit, and R. I. D. Birkler, "RawHummus: An R Shiny app for automated raw data quality control in metabolomics," *Bioinformatics*, vol. 38, no. 7, pp. 2072–2074, Mar. 2022.
- [32] N. Martin, *daqapo: Data Quality Assessment For Process-Oriented Data*, R package version 0.3.1, 2020. [Online]. Available: <https://cran.r-project.org/web/packages/daqapo/daqapo.pdf>
- [33] A. Jain, *StatMeasures: Easy Data Manipulation, Data Quality and Statistical Checks*, R package version 1.0, 2015.
- [34] C. Ryu, *dlookr: Tools for Data Diagnosis, Exploration, Transformation*, R package version 0.5.4, 2021.
- [35] E. Waring, M. Quinn, A. McNamara, E. A. De La Rubia, H. Zhu, and S. Ellis, *skimr: Compact and Flexible Summaries of Data*, R package version 2.1.3, 2021.
- [36] A. Hebbali, *xplorerr: Tools for Interactive Data Exploration*, R package version 0.1.2, 2021.
- [37] L. Ehrlinger, E. Ruzs, and W. Wöb, "A survey of data quality measurement and monitoring tools," *Frontiers Big Data*, vol. 5, p. 28, Mar. 2022.
- [38] R. Adhikari and R. Agrawal, "An introductory study on time series modeling and forecasting," Ph.D. thesis, Lambert Academic, Germany, 2013.
- [39] J. Luengo, S. García, and F. Herrera, "On choice best imputation methods for missing values considering three groups classification methods," *Knowledge Information Systems*, vol. 32, no. 1, pp. 77–108, 2012.
- [40] R. K. Elisavet, H. Spyros, and T. Ioannis, "Missing data in time series and imputation methods," M.Sc. dissertation, Univ. Aegean, Samos, Greece, Feb. 2017, p. 65.
- [41] M. Gomez-Omella, I. Esnaola-Gonzalez, and S. Ferreiro, "Short-term forecasting methodology for energy demand in residential buildings and the impact of the COVID-19 pandemic on forecasts," in *Artificial Intelligence XXXVII (Lecture Notes in Computer Science)*, M. Bramer and R. Ellis, Eds., vol. 12498. Cham, Switzerland: Springer, 2020, pp. 227–240.
- [42] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, nos. 3–4, p. 591, Dec. 1965.
- [43] M. Geurts, G. E. P. Box, and G. M. Jenkins, "Time series analysis: Forecasting and control," *J. Marketing Res.*, vol. 14, no. 2, p. 269, 1977.



MERITXELL GÓMEZ-OMELLA received the B.Sc. degree in mathematics from the Universitat Autònoma de Barcelona (UAB), in 2018, and the M.Sc. degree in modeling and mathematical, statistical and computing research from the University of the Basque Country (UPV/EHU), in 2019, with a master's thesis related to data quality and missing data imputation in collaboration with the Research Center Tekniker, where she is currently pursuing the Ph.D. degree in computer science engineering about analysis, modeling and forecasting of time series data in industrial environments. She is also a Data Scientist of the Intelligent Systems Unit, TEKNIKER, and she has been working in different national and European projects.



BASILIO SIERRA is currently a Full Professor with the Computer Sciences and Artificial Intelligence Department, University of the Basque Country (UPV/EHU). He is the Co-Director of the Robotics and Autonomous Systems Group, RSAIT. He is also a Researcher in the fields of robotics and machine learning, where he is working on the use of different paradigms to improve robot's behaviors. He works as well in multidisciplinary applications of machine learning paradigms, in agriculture, natural language processing, medicine, and so forth. He has published more than 50 journal articles, and several book chapters and conference papers.



SUSANA FERREIRO received the degree in computing science from the Technical School, University of the Basque Country, Spain, in 2005, and the Ph.D. degree (*cum laude*) in the area of probabilistic models for artificial intelligence and data mining, in 2012. Since 2005, she has been working with TEKNIKER in national and European projects in the application of the artificial intelligence for diagnosis and prognosis applied to different industrial sectors. She is also the Head of Predictive Analytics Research Line and a Senior Data Scientist of the Intelligent Systems Unit, TEKNIKER. She has several publications in conferences, journals with impact index and has collaborated in many books as well as in the supervision of theses. Her research interests include the technologies related to predictive analytics for data processing and analysis, including the descriptive and prescriptive analytics.

Optimizing Porosity Detection wire Laser Metal Deposition Processes using Data-driven Classification Techniques.

- Title: Optimizing Porosity Detection wire Laser Metal Deposition Processes using Data-driven Classification Techniques
- Authors: Meritxell Gómez-Omella, Jon Flores, Basilio Sierra, Susana Ferreiro, Nicolas Hascoët, Francisco Chinesta
- Journal: Submitted to Engineering Failure Analysis
- Year: Submitted in 2023
- Quartile: Q2
- DOI: -

Optimizing Porosity Detection wire Laser Metal Deposition Processes through Data-driven AI Classification Techniques

Meritxell Gómez-Omella^{a,b}, Jon Flores^b, Basilio Sierra^a, Susana Ferreiro^b,
Nicolas Hascoët^b, Francisco Chinesta^b

^a*University of the Basque Country, Address One, Donostia-San Sebastian, 20018, Spain*

^b*Tekniker, Basque Research and Technology Alliance (BRTA), C/ Iñaki Goenaga,
5, Eibar, 20600, Spain*

^c*PIMM Lab, Arts et Metiers Institute of Technology, 151 Boulevard de
Hopital, Paris, 75013, France*

Abstract

Additive manufacturing (AM) is an interesting solution for many companies that produce geometrically complex parts. This process consists of the deposition of material layer by layer following a sliced CAD geometry. It brings several benefits to manufacturing capabilities, such as design freedom, reduced material waste, and short-run customization. However, one of the current challenges faced by users of the process, mainly in wire laser metal deposition (wLMD), is to avoid defects in the manufactured part, especially the porosity. This defect is caused by the extreme conditions and metallurgical transformations of the process. And not only does it directly affect the mechanical performance of the parts, especially the fatigue properties, but it also means an increase in costs due to the inspection tasks to which the manufactured parts must be subjected. This work compares three operational solution approaches, product-centric, based on signal-based feature extraction and Topological Data Analysis together with statistical and Machine Learning (ML) techniques, for the early detection and prediction of porosity failure in a wLMD process. The different forecasting and validation strategies demonstrate the variety of conclusions that can be drawn with different objectives in the analysis of the monitored data in AM problems.

Keywords: Additive Manufacturing, Porosity Detection, Artificial Intelligence, Supervised Classification

PACS: 0000, 1111

Preprint submitted to Nuclear Physics B

May 29, 2023

1. Introduction

In contrast to conventional manufacturing techniques based on material subtraction, Additive Manufacturing (AM) consists of material deposition layer by layer following a sliced CAD geometry. This brings several advantages to manufacturing capabilities such as design freedom, reduction of waste material and customisation in short runs [1]. In line with industrial development, AM techniques focused on metallic materials have gained special interest from researchers and manufacturing sectors. In fact, multiple metal AM processes can be found in the literature with different degrees of development and which are classified based on the feedstock format and the energy source [2]. This study focuses on the direct deposition of material in wire format and a laser beam that melts both the added material and the substrate, called wire Laser Metal Deposition (wLMD). Compared to powder-based techniques, it offers a higher deposition rate and almost completely reduces waste material [3]. Using the laser as the source of energy provides greater precision and control over the thermally affected area, achieving less dilution and a more uniform weld bead [4]. Hence, wLMD achieves a balance between the advantages of AM, which makes it an appropriate technology for the manufacture of medium and large parts, obtaining near-net shape geometries.

However, due to the complexity of the material deposition process, it has not yet been mastered enough to enable the industrialisation of wLMD. The consistency of the conditions in the deposition process and in the generation of beads is defined by the interaction between the laser and both the fed material and the substrate. The process parameters have a major influence on this interaction, the most relevant being laser power, movement velocity and wire feed rate [5]. The complexity increases in the manufacturing of 3D geometries, where other factors such as working distance, CAM strategies and base layer conditions must be considered. The inaccurate and unstable definition of manufacturing conditions can lead to process failure [6], but even without failure, poor material deposition conditions lead to defects or pores due to a lack of bonding between beads or layers [7]. In metal AM, achieving full density is the primary objective, as the porosity significantly affects the mechanical performance of parts, especially fatigue properties

[8]. And particularly for the wLMD process this continues to be a challenge because the process produces variability in porosity size and distribution even when the parameters of the process remain unchanged.

During the last few years, the advancement in Computed Tomography (CT) technology has made it a powerful tool for the analysis of the internal integrity of AM metal parts [9]. CT analysis is a Non-Destructive Testing (NDT) method and provides information about the internal voids in the manufactured parts through X-rays. Nevertheless, there are some limitations in the quality of the images obtained related to the material and dimensions of the parts [10].

In addition, the use of this method is limited, on-site inspection for in-process defect detection is not feasible, and post-process inspection is expensive.

On the other hand, the analysis of the state of the art has shown so far that the number of works carried out around approaches and technologies aimed at Zero Defects Manufacturing (ZDM) has increased in recent years (2018-2020). In most of the works the approach based on the 'single stage process' predominates, as described in [11], while the number of works oriented to 'product-centric and defective parts' is lower. However, they are starting to increase in recent years due to the emergence Industry 4.0 phenomenon, which brings a wide range of technologies [12] that are considered key enabling technologies of ZMD including Internet of Things (IoT) and Artificial Intelligence (IA).

Regarding Artificial Intelligence technologies, data-driven techniques for automated data analysis and decision-making are remarkable. Currently, the efforts in additive manufacturing, and particularly in wLMD, are focused on the development of alternatives based on data and AI algorithms to create systems capable of detecting defective parts in real time a subsequently stopping the process, saving effort and economic costs. Therefore, in the field of Artificial Intelligence, Machine Learning (ML) techniques have been used from the different phases to improve the process and avoid defective parts.

Related to the design phase, some works propose the use of Convolutional Neural Networks (CNN) to train the database from which the mechanical properties are calculated using the Finite Element Method (FEM) [13] or Hierarchical Cluster Analysis (HCA) and Support Vector Machines (SVM) to obtain feature recommendations from the design phase [14]. These ML techniques have been used also to optimize operating parameters based on back-propagation algorithms such as multi-layer perceptron (MLP) [15] or

to perform the prediction of defects from a Random Forest (RF) [16] or SVM [17]. While from the point of view of the production phase, these ML techniques are mainly focused on process planning [18], quality control [19] and ensuring the security of data [20]. ML techniques are of particular interest and are increasingly used for quality control, aimed at monitoring the process in situ to identify defects such as cracks, deformations, lack of fusion or porosity, but where they remain a challenge. Research works are paying increasing attention and great efforts to monitor the processes during the printing of parts, which implies tedious and low-fidelity steps [21].

Concerning the prediction techniques used for porosity detection, these can be classified according to the strategy followed to build the model. There are some works related to the use of input variables captured during the manufacturing process, the material, the shape of the part and other considerations. The works usually focused on data-driven modelling, training the models through in-situ and post-manufacturing characterization. There are also some works that present solutions based on a visual detection technique. Regarding the relation between porosity and features extracted from sensors data, some studies have been done to demonstrate the correlation between the visualisation and the errors detected with microsections [22] that justify the approaches of pore classification works. An Artificial Neural Network-based framework that connects extracted characterisations and profiles of melt pool to the profiles of the pore is proposed in [23]. An SVM algorithm is trained with multiple images collected at each built layer using a high-resolution digital single-lens reflex (DSLR) camera [24]. RF and Early Stopping Neural Network (ESNN) algorithms are compared in [25] to classify pores based on Thermal Images. Similarly, in [26], the authors compare the effectiveness of some ML algorithms (Decision Trees (DT), k-Nearest Neighbours (KNN), Support Vector Machines (SVM) and Linear and Quadratic Discriminant Analysis (LDA and QDA)) to predict the porosity. Although this type of ML technique has been used and investigated for the detection of pores in the AM process in recent years, the most widely adopted are the optical-based techniques, focused on the classification of the images captured by cameras from optical signals. However, it is not a feasible solution for in-site monitoring during the manufacturing process to detect and predict pores because its deployment is not viable.

This research work proposes a solution approach based on Topological Data Analysis combined with statistical and ML algorithms, from the process parameters collected during wire Laser Metal Deposition monitoring, for the

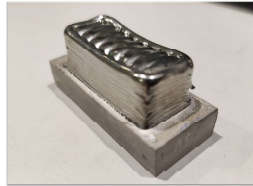


Figure 1: Part manufactured by a wire Laser Metal Deposition process.

early detection and prediction of the appearance of pores. For wLMD to perform optimally, it is necessary to achieve high accuracy and ensure 'zero defect' products. And this remains a major challenge.

The rest of the article has been structured as follows. In Section 2 the use case is presented and the objectives are exposed. The methodology used is detailed in Section 3 which includes balancing the data sets, the three proposed forecasting strategies to detect porosity and finally, the validation approaches. The results obtained are shown in Section 4. Finally, Section 5 contains conclusions and future work.

2. Experimental setup and data collection

The monitoring of AM processes provides large amounts of data containing high-frequency information on the status of both the manufacturing process and the built parts. In the experimental process, three 3D geometry parts (identified as 13, 16 and 18) were manufactured and the signals were acquired during the process for the subsequent obtaining of the data set.

These parts depicted in Figure 1 are formed by 12 rectangular layers with dimensions of 16 by 40 mm approximately. Each layer is manufactured using a predefined material deposition trajectory consisting of a perimeter strategy and a zigzag for the interior filling. The time spent building them ranges between 18.2 and 18.8 seconds.

A set of optimal process parameters has been used, and adjusted in a previous experimental process. The commands that control these parameters are constant throughout the manufacturing process, the main ones being laser power, wire feed speed and movement velocity. Regarding the trajectory, an optimal distance between beads that provides uniform growth is determined. Finally, from the experimental tests of the selected set of parameters, the theoretical growth of the layers is estimated.

Different types of signals are acquired at different stages of the process so that time-referenced and spatially-referenced signals can be distinguished. A preprocessing and data fusion step is necessary to create the most explainable data set to properly train supervised classification algorithms. Once the signals are acquired, a data processing and fusion method is implemented to generate the data set. The data fusion is based on the trajectory information, which enables independent signals of different typologies to be converted into a spatially and temporally aligned data set. Therefore, the variables extracted from the signal processing can be linked to a moment in the manufacturing process and to a location in the part geometry. This data preparation task is out of scope of this article and it is explained in detail in [27].

The high-frequency monitoring of the process generates a data set of approximately 19.000 observations per part containing information on the following features:

- The position over time is recorded, which defines the material deposition trajectory performed by the robot. The signal is composed of three univariate variables: X, Y, Z, and Time. Both variables serve as the reference to merging all acquired data, as it includes the relation between space and time information. From the position record, the *Vertical Movement* during material deposition is extracted. This variable describes the positioning inaccuracy of the robot in the vertical direction. It is measured in the Z axis with respect to the preset height value of each layer.
- During the material deposition process, two types of signals are acquired with reference to the processing time. On the one hand, the signals that control the laser power and the wire feed speed are recorded by the Programmable Logic Controllers (PLC). On the other hand, images of the area where the material is being added are acquired by coaxial monitoring of the melt pool. The homogeneity of the growth across the manufacturing process is obtained through geometric scanning. Since the manufacturing is carried out layer by layer, after each layer is deposited, the resulting surface geometry is scanned. The resulting 3D point cloud that represents the surface is spatially referenced to the trajectory and the manufactured part.

The *Overlap Factor* variable is calculated from the overlapping condi-

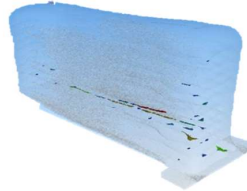


Figure 2: Scanned part provided by the CT.

tions between the beads that form each layer. In fact, the trajectory recorded describes the distance between adjacent beads, which affects the uniformity and effectiveness of the filling of the layers. A bigger gap between two adjacent beads means less overlap between them and therefore less material in that area. Thus, the *Overlap Factor* is calculated based on the distance of the beads to each other along each layer.

After manufacturing each layer, the surface geometry is scanned and compared with the theoretical growth, from which the geometric deviation is computed. Two variables are extracted when merging the information with the trajectory. On the one hand, if linked to the movements of the last layer deposited, the variable *Z Distortion* describes the resulting geometric distortion. On the other hand, if it is linked to the trajectory of the next layer, the information refers to the geometric conditions of the base surface on which the material is deposited, which is named *Base Distortion*. Notice that in the first layer, this value should be 0 because the material is deposited on a flat surface.

- After the completion of the manufacturing of each part, the porosity is measured offline. The 3D Computed Tomography (CT) shown in Figure 2 provides data on the location of the centre of each pore as well as its volume and shape, which is spatially referenced to the part geometry. This information is used to label the recorded data sets and distinguish the pore observations.

Based on the porosity information, each pore is simplified and visualised

by ellipsoids calculated from their corresponding length, width and height data. The trajectory and the ellipsoids are analysed together and the points of the trajectory that lie within the volume of the ellipsoids are located. Therefore, two labels are created regarding porosity. The *Inside Pore* variable takes the value 1 if the observation is within a pore and 0 if not.

3. Methodology

This work focuses on finding out if porosity can be anticipated or only estimated locally. To do this, the first step is the capture of data during the manufacturing process and subsequent labelling using the information obtained from the CT analysis. To examine pore memory, two strategies for feature extraction from the temporal series are employed: on one hand, a point-wise extraction on the signals, and on the other hand, Topology Data Analysis (TDA) was used to extract persistence images from time series. To estimate the failures locally, the raw data of the process was used as independent observations.

During the phase of creating and selecting the best predictive model, two key aspects are tested: classification strategy and validation strategy. Firstly, three different strategies are proposed for classifying observations to detect porosity: local classification, time series features classification and persistence images classification, which are explained in detail below. Secondly, three methods are suggested for validating the results by employing various train-test splits on the data sets: using the information from the parts separately, mixing them all, or using some parts to train a model that predicts pores in the remaining part.

3.1. Balance data sets

In each of those classification strategies, the available dataset consists of a different number of observations. The classes used to label the data to distinguish pores from the rest of the observations also represent a different percentage.

The difference between the number of observations in the minority and the majority class, becomes a problem providing non-realistic predictions or performance values. There are an extensive amount of techniques to handle

imbalanced data set and they can be classified in two main groups: under-sampling methods act minimising the number of observations of the majority class while oversampling increases the observations of the minority class.

Tomek Links is one of a modification of Condensed Neighbors Neighbors that matches data from each class with the minimum Euclidean distance between them. These "Tomek Links" can be used to locate all the closest neighbours of different classes, that is, to define the class boundary. The procedure is used to remove all those data in the majority class that are closest to the minority class [28].

SMOTE is one of the most popular oversampling techniques [29]. Unlike random oversampling that only duplicates some random examples from the minority class, SMOTE generates examples based on the distance of each data (usually using Euclidean distance) and the minority class nearest neighbours, so the generated examples are different from the original minority class.

SMOTE-Tomek Links combines the SMOTE to increase the minority class and Tomek links to remove observations from the majority class [30].

3.2. Forecasting Strategies

Each of the classification strategies leads to different conclusions, and the results of all three are compared to determine whether porosity is a manufacturing defect that can be anticipated by analyzing values prior to its occurrence or if, on the contrary, the pores need to be detected retrospectively after they have already happened. That is, when past features are related to the current failure, it can be concluded that there is a temporal relationship manifested in the values captured before the appearance of the porosity. On the other hand, if the classification is done observation by observation, it is understood that porosity in the parts is a local failure that cannot be anticipated but can be estimated with data-driven models that replace classical CT.

In each of the strategies, the results of four of the most commonly used Machine Learning (ML) models in this type of classification problem are compared: k-Nearest Neighbours (KNN), Support Vector Machines (SVM), Random Forest (RF), and Extreme Gradient Boosting (XGBoost). Each of these algorithms requires a different parameter tuning process, which also depends on the chosen strategy.

1. Local classification is performed in the entire data set to estimate the

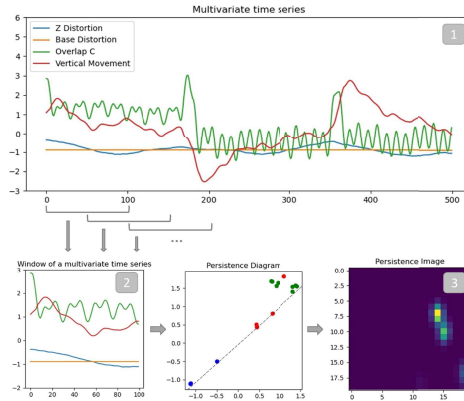


Figure 3: Creation of the persistence images from the multivariate time series.

pores in parts by classifying the captured data during the manufacturing observation by observation. Therefore, raw data shown in Figure 3, graph 1, is classified point by point using ML algorithms. In this case, the values of the 4 signals of *Z Distortion*, *Base Distortion*, *Overlap Factor* and *Vertical Movement* are used as inputs, and the presence of the pore is estimated as a binary classification output.

The data balancing in this case is achieved by randomly reducing 80% of the data from the majority class, followed by the application of SMOTETomek to equalise the observations of both classes. This number is chosen due to the large number of values without pores compared to the few values with pores, as mentioned before.

The model tuning is done through an exhaustive search for parameters that provide the highest recall. For each algorithm, an appropriate grid search is designed. For KNN, the optimal choice of k is tested from 1 to 15. In RF, the set of the number of estimators is $\{50, 200, 500\}$ with a maximum depth of $\{4, 6, 10\}$. The minimum number of samples required to be at a leaf node is tested with values of $\{5, 10, 20\}$, and the Gini criterion is used to evaluate efficiency. The regularisation parameter C of SVC is tested with values of $\{0.1, 1, 10, 100\}$. The chosen kernel is the exponential function, and its parameter is tested with values of $\{0.01, 0.1, 1\}$. Finally, for XGBoost, different numbers of estimators are tested $\{60, 160, 260, 360, 460, 560\}$ with maximum depths

Table 1: Percentage of pores in each of the three parts after rolling window processing.

Part	ID	N ^o Records	N ^o Pores	N ^o Non Pores	Percentage
1	13	339	23	316	6.78%
2	16	336	51	285	15.18%
3	18	338	5	333	1.48%

of $\{2, 4, 6, 8, 10\}$. The learning rate is tested for values $\{0.01, 0.1, 0.5\}$.

2. A rolling window of size 100 is applied in the multivariate time series skipping 50 values to avoid unnecessary overlaps. Each of the sub-series is labelled according to the presence or absence of pores, regardless of the quantity. Table 1 summarises the number of available sub-series for each part and indicates which of them contain pores.

The class balancing in this case is done considering the number of sub-series available in the data set, which are 336, 338, and 339 for each part, and the percentage of these sub-series that contain pores. The 40% of the observations without pores are randomly removed from the data set, and the remaining observations are balanced using SMOTE-Tomek to have an equal number of sub-series with and without pores. The model tuning is different in this approach because the number of training instances is not the same. Due to the varying sizes of the datasets in different classification strategies, the process of tuning and optimizing the models may need to be adjusted accordingly. The changes made compared to the local classification strategy are as follows: the depth of the RF is tested for $\{1, 3\}$, and the number of estimators is adjusted to $\{5, 10, 20\}$. For the XGBoost the number of estimators is tested for $\{10, 35, 60, 85, 110, 135, 160, 210, 235, 260\}$.

Then, from the set of sub-series, two different strategies of time series feature extraction are tested: statistical and persistence images.

- A time series signal-based feature extraction is done to create a set of inputs for ML algorithms from each window shown in Figure 3, graph 2. The mean, the standard deviation, the maximum, the minimum, the trend, the entropy and the curvature values are used as explanatory variables to predict porosity. The entropy is a measure of the complexity of the time series based on its regularity [31].

- The persistence images shown in Figure 3, graph 3, are created using Topological Data Analysis (TDA) for time series using `ripser` library for Python [32]. TDA is mainly motivated by the idea that topology and geometry provide a powerful approach to infer robust qualitative, and sometimes quantitative, information about the structure of data [33, 34]. Recently, TDA is being used in time series to study the structure and shape of the data [35].

The extraction of persistence diagrams from time series and the creation of persistence images from them are explained in detail in the following lines. A persistence diagram is a graphical representation of the evolution of topological features over time. It provides a way to visualise the lifespan of topological features, such as connected components or holes, as they appear and disappear in data. To construct a persistence diagram, TDA algorithms first transform data into a simplicial complex, a mathematical structure that encodes the topological features of data. This simplicial complex is then analysed using persistent homology, a mathematical tool that identifies the connected components, holes, and higher-dimensional voids that persist across different scales in the data.

The output of the persistent homology algorithm is a set of points in a two-dimensional plane, where each point corresponds to a topological feature and its position reflects its birth and death times. Specifically, each point has two coordinates: the birth time, which represents the scale at which the feature first appears in data, and the death time, which represents the scale at which the feature disappears or merges with another feature.

The persistence diagram can be visualised as a scatter plot, where each point represents a topological feature and is coloured by its dimensionality.

First, let $\{Y_t, t = 1, \dots, T\}$ a univariate time series (Figure 4a), there are different topological approaches to obtain a point cloud [36]. The number of points P in the point cloud varies according to the method determined.

Lower-star filtration or sublevel set filtration is a stable way of describing local minimums and maximums in a time series [37]. Therefore, P min-max pairs $(b^1, d^1), \dots, (b^P, d^P)$ have been con-

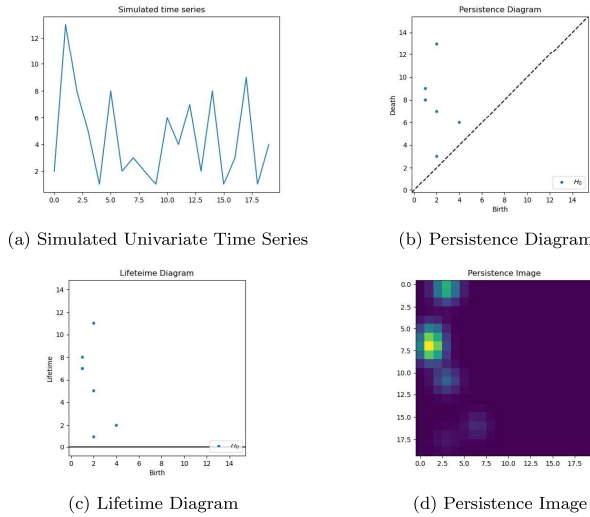


Figure 4: Process of creating persistence images of a univariate time series using TDA

structured where b refers to the minimum or birth and d refers to the maximum or death. This set of points represented in the plane defines the persistence diagram (Figure 4b). As d values are always greater than or equal to their b pairs, the points $(b^1, d^1), \dots, (b^P, d^P)$ are grouped in the upper part of the diagonal $b = d$ and they can be represented in the life diagram $(b^1, d^1 - b^1), \dots, (b^P, \dots, d^P - b^P)$ (Figure 4c). Distances are possible between those clouds of points but applying ML methods is still challenging. An alternative way is to associate to each point from the lifetime diagram a bivariate normal distribution, weighted and then, integrated into different patches on a square domain covering the support of the regularised lifetime diagram, leading to the so-called persistence images (Figure 4d) [38].

The persistence image of each of the available variables is extracted as an array of 20 by 20 pixels dimension. Then, data sets are written as a vector of length 400 and ML techniques to classify the vectorised pixels, using each pixel as an input to the algorithm. Some of these pixels have the same value for all per-

sistence images, so columns with near-zero variance are removed to reduce the size of the input data matrix. In total, information from 367 pixels is retained.

3.3. Validation Techniques

One of the recurring problems in the industry, when classification cases are proposed for fault detection, is the lack of repeated parts to experiment with in model training processes. For this reason, validating data-driven models requires realistic strategies that allow for the extraction of appropriate conclusions for generalisation. For each of the strategies explained below, three different validations are tested to evaluate the efficiency of the approaches.

- For each part, 2/3 of the data is used to train the models and the remaining 1/3 of the part is used to test the fitted model. In the case of classification using features extracted from the sub-series, part 18 is not considered in this validation approach due to having only 5 sub-series with pores.
- Data from all three parts are used together. 2/3 of the data is used to fit the model in training and 1/3 for validation.
- Three tests are done by training the model with the data of 2 complete parts and reserving an entire part for testing.

For each validation, a binary classification is provided. In binary classification, the confusion matrix is a 2x2 table representing the actual and the predicted values, as can be seen in Table 2. On the diagonal, well-classified values are shown: True Negative (TN) and True Positive (TP). At opposite vertices, the values are classified in the wrong class: False Negative (FN) are the values classified as 0 when they are actually 1 and False Positive (FP), otherwise.

Table 2: Confusion Matrix in a Binary Classification

		Predicted	
		0	1
Actual	0	True Negative (TN)	False Positive (FP)
	1	False Negative (FN)	True Positive (TP)

The Confusion Matrix of each classification is examined through *Accuracy* and *Recall* metrics, using the following equations.

$$Accuracy = \frac{TN + TP}{TN + FP + FN + TP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

On the one hand, *Accuracy* provides an overall value of the model's success in predicting new classes by the total number of correct predictions divided by the total number of predictions made. On the other hand, *Recall* calculates how many of the Positives the model captures by labelling it as TP. So, it is the number of positive values well classified divided by the sum of the correct positive and incorrect negative classifications. It is especially useful in faulty diagnosis because in those problems it is expected to penalise the non-identification of the failure. It is known that *Recall* shall be the metric used when there is a high cost associated with FN.

4. Results

The results of all the experiments are shown in Table 3. Classification errors are measured by accuracy and recall metrics and the mean of the results of the 5 different random train-test partitions are shown.

In general, in any of the strategies and algorithms, classifications are more accurate when using data from the same part to detect pores in the rest of the part. When mixing the data from all three parts and treating them equally, the accuracy tends to remain the same although the recall slightly decreases. However, when using the data from two parts to predict the third part, the recall drastically drops, providing estimations that cannot be considered reliable.

Regarding the forecasting strategies, local prediction achieves promising results in porosity detection. The results obtained with this forecasting strategy and validation approaches are similar for the four algorithms. For instance, the confusion matrices in Table 4 show classifications performed with KNN in a specific train-test partition. The number of neighbours used are $k = 5$, $k = 7$ and $k = 3$, respectively in parts 13, 16 and 18.

In the case of classifications performed on the features extracted from the temporal subseries of the signals, the difference in results is more noticeable. For instance, the recall obtained with SVM in the first validation strategy is

Table 3: Summary of the performance obtained by detecting porosity using different approaches

Validation Partition	Algorithm	Local Class.		TS features Class.		Image Class.	
		Accuracy	Recall	Accuracy	Recall	Accuracy	Recall
2/3 part to train and 1/3 part to test	KNN	0.954	0.855	0.827	0.665	0.714	0.398
	SVM	0.959	0.855	0.839	0.408	0.641	0.485
	RF	0.937	0.818	0.666	0.702	0.803	0.310
	XGBoost	0.964	0.804	0.730	0.755	0.811	0.299
2/3 total to train and 1/3 total to test	KNN	0.971	0.742	0.814	0.505	0.753	0.418
	SVM	0.956	0.762	0.870	0.264	0.697	0.437
	RF	0.914	0.728	0.620	0.766	0.795	0.400
	XGBoost	0.972	0.694	0.857	0.493	0.801	0.356
2 parts to train and 1 part to test	KNN	0.969	0.393	0.715	0.348	0.748	0.226
	SVM	0.966	0.034	0.871	0.106	0.733	0.295
	RF	0.903	0.267	0.631	0.397	0.813	0.251
	XGBoost	0.975	0.021	0.844	0.211	0.819	0.211

Table 4: Confusion matrices from local classification part by part using KNN algorithm.

Id Part		13		16		18	
		Predicted					
		0	1	0	1	0	1
Actual	0	6052	309	5544	762	6258	155
	1	4	34	5	46	1	13
Accuracy		0.951		0.879		0.976	
Recall		0.895		0.902		0.929	

Table 5: Confusion matrices from time series features classification part by part using XGBoost algorithm.

Id Part		13		16	
		Predicted			
		0	1	0	1
Actual	0	97	8	77	17
	1	1	7	4	14
Accuracy		0.951		0.813	
Recall		0.895		0.778	

0.408, the lowest of the four. However, with the same features and validation, XGBoost achieves a recall of 0.755. In the latter case, Table 5 displays the classification in one of the train-test splits. In this strategy, the classification remains binary, where 1 indicates the presence of some pore in that time subseries, without specifying the quantity or positions.

If all the extracted subseries from different parts are considered independent and used to predict porosities, in the second validation strategy, it can be seen that the RF achieves a good classification result with an average accuracy of 2 and an average recall of 9. Table 6 displays an example of this classification in the confusion matrix, combining the three parts in a specific train-test split.

Finally, it can be observed that with the extraction of topological features instead of basic features from the signals, both accuracy and recall decrease. This fact suggests that the presence of pores in the parts is related to the absolute values taken by the captured signals, rather than their shape.

Table 6: Confusion matrix from time series features classification using the three parts and the RF algorithm.

		Predicted	
		0	1
Actual	0	197	115
	1	6	20

5. Conclusions

Metal additive manufacturing technology has been used for years with the aim of developing parts for many industrial sectors. It is an advanced production method that allows complex parts to be created at a relatively low price compared to traditional techniques. However, due to the manufacturing process, the materials often have tiny pores inside them, which can grow when a load is applied, resulting in poor quality defective parts.

The research presented in this work, related to data-driven modelling applied to this type of AM applications shows the limitation due to the low experimental repeatability. However, the results obtained in that work are considered very promising as compared to those presented in the current literature.

The accuracy and recall results obtained by all the tested algorithms in the local classification are considered high, suggesting an approach for pore detection based on data that could lead to the replacement of the costly tomography techniques currently used for this purpose.

The results obtained by classifying the temporal subseries conclude that there is a way to classify pores using smaller datasets. In this case, the accuracy would decrease as the simplicity of the computation increases.

The poor performance of pore estimations when using information from two parts to predict a third part is due to the context in which these parts were manufactured. The conclusion that can be drawn in this case is twofold. On one hand, it is necessary to replicate experiments under the same conditions to contextualize the manufacturing process. On the other hand, the variables collected for this problem are incomplete as the algorithms should have some input regarding the manufacturing parameters that make the parts different.

There are some lines of activity and research to continue with the work in the future. The first one, as it is expected to have a larger number of

monitored and tomographed parts, is to train the classification models with data from several parts and then validate the performance of the fitted models using data from an entire part. At this point, it is important to emphasise the importance of data coming from parts manufactured under the same conditions, since any variation in the configuration of the input parameters could significantly modify the behaviour of some data, adding variability to the data set and avoiding models from capturing those behaviours. The second one is to extend the study presented in this work to other types of geometrically different parts to validate the proposed methodology. Finally, the third line of activity considers the possibility of expanding the set of methods and algorithms used to image pre-processing and classification tasks or exploring a distance-based approach from the persistence images.

List of Abbreviations

CT Computed Tomography

KNN k-Nearest Neighbours

ML Machine Learning

PLC Programmable Logic Controllers

RF Random Forest

SVM Support Vector Machines

TDA Topological Data Analysis

XGBoost Extreme Gradient Boosting

References

- [1] S. Ford, M. Despeisse, Additive manufacturing and sustainability : an exploratory study of the advantages and challenges, *Journal of Cleaner Production* 137 (2016) 1573–1587. doi:10.1016/j.jclepro.2016.04.150.
- [2] A. Vafadar, F. Guzzomi, A. Rassau, K. Hayward, Advances in metal additive manufacturing: A review of common processes, industrial applications, and current challenges, *Applied Sciences* 11 (3) (2021). doi:10.3390/app11031213.

- [3] D. Ding, Z. Pan, D. Cuiuri, H. Li, Wire-feed additive manufacturing of metal components : technologies , developments and future interests, *The International Journal of Advanced Manufacturing Technology* 81 (1) (2015) 465–481. doi:10.1007/s00170-015-7077-3.
- [4] C. Leyens, E. Beyer, 8 - innovations in laser cladding and direct laser metal deposition, in: J. Lawrence, D. Waugh (Eds.), *Laser Surface Engineering*, Woodhead Publishing Series in Electronic and Optical Materials, Woodhead Publishing, 2015, pp. 181–192. doi:https://doi.org/10.1016/B978-1-78242-074-3.00008-8.
- [5] M. Froend, S. Riekehr, N. Kashaev, B. Klusemann, J. Enz, Process development for wire-based laser metal deposition of 5087 aluminium alloy by using fibre laser, *Journal of Manufacturing Processes* 34 (2018) 721–732. doi:https://doi.org/10.1016/j.jmapro.2018.06.033.
- [6] M. Motta, A. G. Demir, B. Previtali, High-speed imaging and process characterization of coaxial laser metal wire deposition, *Additive Manufacturing* 22 (2018) 497–507. doi:https://doi.org/10.1016/j.addma.2018.05.043.
- [7] M. Akbari, R. Kovacevic, An investigation on mechanical and microstructural properties of 316lsi parts fabricated by a robotized laser/wire direct metal deposition system, *Additive Manufacturing* 23 (2018) 487–497. doi:https://doi.org/10.1016/j.addma.2018.08.031.
- [8] J. J. Lewandowski, M. Seifi, Metal additive manufacturing: A review of mechanical properties, *Annual Review of Materials Research* 46 (2016) 151–186.
- [9] A. du Plessis, I. Yadroitsava, I. Yadroitsev, Effects of defects on mechanical properties in metal additive manufacturing: A review focusing on x-ray tomography insights, *Materials Design* 187 (2020) 108385. doi:https://doi.org/10.1016/j.matdes.2019.108385.
- [10] M. R. Khosravani, T. Reinicke, On the Use of X-ray Computed Tomography in Assessment of 3D-Printed Components, *Journal of Nondestructive Evaluation* 39 (4) (2020). doi:10.1007/s10921-020-00721-1.
- [11] M. C. Daryl Powell, Maria Chiara Magnanini, O. Myklebusta, Advancing zero defect manufacturing: A state-of-the-art perspective

- and future research directions, *Computers in Industry* 136 (2022). doi:<https://doi.org/10.1016/j.compind.2021.103596>.
- [12] A. B. T. Zheng, M. Ardolino, M. Perona, The applications of industry 4.0 technologies in manufacturing context: a systematic literature review, *International Journal of Production Research* 59 (2021) 1922–1954. doi:[10.1080/00207543.2020.1824085](https://doi.org/10.1080/00207543.2020.1824085).
 - [13] G. X. Gu, C. T. Chen, D. J. Richmond, M. J. Buehler, Bioinspired hierarchical composite design using machine learning: Simulation, additive manufacturing, and experiment, *Materials Horizons* (2018). doi:[10.1039/c8mh00653a](https://doi.org/10.1039/c8mh00653a).
 - [14] X. Yao, S. K. Moon, G. Bi, A hybrid machine learning approach for additive manufacturing design feature recommendation, *Rapid Prototyping Journal* (2017). doi:[10.1108/RPJ-03-2016-0041](https://doi.org/10.1108/RPJ-03-2016-0041).
 - [15] F. Caiazzo, A. Caggiano, Laser direct metal deposition of 2024 al alloy: Trace geometry prediction via machine learning, *Materials* (2018). doi:[10.3390/ma11030444](https://doi.org/10.3390/ma11030444).
 - [16] J. Zur Jacobsmuhlen, S. Kleszczynski, G. Witt, D. Merhof, Detection of elevated regions in surface images from laser beam melting processes, in: *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*, 2015. doi:[10.1109/IECON.2015.7392275](https://doi.org/10.1109/IECON.2015.7392275).
 - [17] C. Gobert, E. W. Reutzel, J. Petrich, A. R. Nassar, S. Phoha, Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging., *Additive Manufacturing* (2018). doi:[10.1016/j.addma.2018.04.005](https://doi.org/10.1016/j.addma.2018.04.005).
 - [18] Y. Tang, G. Dong, Q. Zhou, Y. F. Zhao, Lattice Structure Design and Optimization with Additive Manufacturing Constraints, *IEEE Transactions on Automation Science and Engineering* (2018). doi:[10.1109/TASE.2017.2685643](https://doi.org/10.1109/TASE.2017.2685643).
 - [19] I. Baturynska, O. Semeniuta, K. Wang, Application of machine learning methods to improve dimensional accuracy in additive manufacturing, in: *Lecture Notes in Electrical Engineering*, 2019. doi:[10.1007/978-981-13-2375-1_31](https://doi.org/10.1007/978-981-13-2375-1_31).

- [20] M. A. Al Faruque, S. R. Chhetri, A. Canedo, J. Wan, Acoustic Side-Channel Attacks on Additive Manufacturing Systems, in: 2016 ACM/IEEE 7th International Conference on Cyber-Physical Systems, ICCPS 2016 - Proceedings, 2016. doi:10.1109/ICCPS.2016.7479068.
- [21] C. Wang, X. P. Tan, S. B. Tor, C. S. Lim, Machine learning in additive manufacturing: State-of-the-art and perspectives, Additive Manufacturing 36 (January) (2020) 101538. doi:10.1016/j.addma.2020.101538.
- [22] R. Bernhard, P. Neef, H. Wiche, C. Hoff, J. Hermsdorf, S. Kaieler, V. Wesling, Defect detection in additive manufacturing via a toolpath overlaid melt-pool-temperature tomography, Journal of Laser Applications 32 (2) (2020) 022055. arXiv:https://doi.org/10.2351/7.0000055, doi:10.2351/7.0000055.
- [23] R. Jafari-Marandi, M. Khanzadeh, W. Tian, B. Smith, L. Bian, From in-situ monitoring toward high-throughput process control: cost-driven decision-making framework for laser-based additive manufacturing, Journal of Manufacturing Systems 51 (2019) 29–41. doi:https://doi.org/10.1016/j.jmsy.2019.02.005.
- [24] Application of supervised machine learning for defect detection during metallic powder bed fusion additive manufacturing using high resolution imaging., Additive Manufacturing 21 (2018) 517–528. doi:10.1016/j.addma.2018.04.005.
- [25] M. Behnke, S. Guo, W. Luo, Comparison of early stopping neural network and random forest for in-situ quality prediction in laser based additive manufacturing, Procedia Manufacturing 53 (2021) 656–663, 49th SME North American Manufacturing Research Conference (NAMRC 49, 2021). doi:https://doi.org/10.1016/j.promfg.2021.06.065.
- [26] M. Khanzadeh, S. Chowdhury, M. Marufuzzaman, M. A. Tschopp, L. Bian, Porosity prediction: Supervised-learning of thermal history for direct laser deposition, Journal of Manufacturing Systems 47 (2018) 69–82. doi:https://doi.org/10.1016/j.jmsy.2018.04.001.
- [27] J. Flores Prado, I. Cabanes Axpe, I. Garmendia Saez de Heredia, O. Gonzalo de Francisco, E. Portillo Pérez, A multiple data fusion method based on the deposition toolpath in the additive manufacturing

with wire laser metal deposition, Submitted to Journal of Manufacturing Science and Engineering (2022).

- [28] Two modifications of `cnn`, IEEE Transactions on Systems, Man, and Cybernetics SMC-6 (11) (1976) 769–772. doi:10.1109/TSMC.1976.4309452.
- [29] K. W. Bowyer, N. V. Chawla, L. O. Hall, W. P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, CoRR abs/1106.1813 (2011). arXiv:1106.1813.
- [30] G. E. A. P. A. Batista, A. L. C. Bazzan, M. C. Monard, Balancing Training Data for Automated Annotation of Keywords : a Case Study.
- [31] M. Rostaghi, H. Azami, Dispersion entropy: A measure for time-series analysis, IEEE Signal Processing Letters 23 (5) (2016) 610–614.
- [32] C. Tralie, N. Saul, R. Bar-On, Ripser.py: A lean persistent homology library for python, The Journal of Open Source Software 3 (29) (2018) 925. doi:10.21105/joss.00925.
URL <https://doi.org/10.21105/joss.00925>
- [33] F. Chazal, H.-d. T. Data, A. Handbook, High-Dimensional Topological Data Analysis Frédéric Chazal To cite this version : HAL Id : hal-01316989 HIGH-DIMENSIONAL TOPOLOGICAL DATA ANALYSIS (2016).
- [34] F. Chazal, B. Michel, An Introduction to Topological Data Analysis: Fundamental and Practical Aspects for Data Scientists, Frontiers in Artificial Intelligence 4 (2021) 1–44. arXiv:1710.04019, doi:10.3389/frai.2021.667963.
- [35] N. Ravishanker, R. Chen, Topological Data Analysis (TDA) for Time Series (2019) 1–29 arXiv:1909.10604.
- [36] F. Takens, Detecting strange attractors in turbulence, Journal of Animal Ecology 84 (6) (2012) 388–400.
- [37] G. Damiand, E. Paluzo-Hidalgo, R. Slechta, R. Gonzalez-Diaz, Approximating lower-star persistence via 2D combinatorial map simplification, Pattern Recognition Letters (2020). doi:10.1016/j.patrec.2020.01.018.

- [38] R. Mezher, J. Arayro, N. Hascoet, F. Chinesta, Study of concentrated short fiber suspensions in flows, using topological data analysis, *Entropy* 23 (9) (2021) 1229.

Other publications

During this research, participation has been held in various research conferences. Below are the contributions made and the articles that have been published as a result of these conferences.

In October 2020, we attended the Sustainable Places 2020 event and presented the work *Short-Term Electric Demand Forecasting for the Residential Sector: Lessons Learned from the RESPOND H2020 Project*. This article was published in January 2021 in the Proceedings journal, belonging to the Proceedings of The 8th Annual International Sustainable Places Conference (SP2020) (112).

In December 2020, we participated in the 40th SGAI International Conference on Artificial Intelligence. The article entitled *Short-Term Forecasting Methodology for Energy Demand in Residential Buildings and the Impact of the COVID-19 Pandemic on Forecast* was awarded as the best application student paper. The article was published in the Artificial Intelligence XXXVII Proceedings (23).

In July 2021, we were invited to participate in the 6th European Conference of the Prognostics and Health Management Congress for being a member of the winning team of the data challenge of that year. The solution that we proposed to that diagnosis problem was the most efficient and our work was published in their Proceedings with the title *Divide, Propagate and Conquer: Splitting a Complex Diagnosis Problem for Early Detection of Faults in Manufacturing Production Line* (25).

In July 2022, we contributed to the 2022 European Conference on Computing in Construction with an article entitled *PRENERGET: A*

Framework for the Inclusion and Adaptation Strategies of Machine Learning Models for Energy Demand Forecasting in Buildings (113). We proposed a software framework aimed at addressing the deployment and exploitation of ML models to different energy-efficiency strategies in buildings. The system regularly evaluate and upgrade the deployed models to ensure the robustness of the overall solution. That framework was further tested with other data sets to validate its effectiveness and the work was again presented at the EESAP International Conference 2022. The publication was entitled *PRENERGET, framework for energy forecasting*.

The documents mentioned above are attached in the same order below.



Short-Term Electric Demand Forecasting for the Residential Sector: Lessons Learned from the RESPOND H2020 Project [†]

Meritxell Gómez-Omella ^{1,2,*} , Iker Esnaola-Gonzalez ¹ and Susana Ferreiro ¹

¹ TEKNIKER, Basque Research and Technology Alliance (BRTA), C/Iñaki Goenaga, 5, 20600 Eibar, Spain; iker.esnaola@tekniker.es (I.E.-G.); susana.ferreiro@tekniker.es (S.F.)

² Faculty of Informatics, University on the Basque Country (UPV/EHU), Paseo Manuel Lardizabal, 1, 20018 Donostia-San Sebastian, Spain

* Correspondence: meritxell.gomez@tekniker.es

[†] Presented at the Sustainable Places 2020, Online, 28–30 October 2020.

Published: 6 January 2021



Abstract: RESPOND proposes an Artificial Intelligent (AI) system to assist residential consumers that would like to make use of Demand Response (DR) and incorporate it into their energy management systems. The proposed system considers the forecast energy consumption based on the data acquired. This work compares the results obtained by different forecasting methods using the Root Mean Square Error (RMSE) as a measure of the forecast performance. The ARIMA, Linear Regression (LR), Support Vector Regression (SVR) and k-Nearest Neighbors (KNN) models are tested, and it is concluded that the results achieved with the KNN obtain a better fit. In addition to obtaining the lowest RMSE, KNN is the algorithm that spends less time in obtaining the forecasts.

Keywords: time series forecasting; k-nearest neighbor; electric demand; RESPOND project

1. Introduction

Buildings use more than 35% of global energy use but a significant amount can be saved if they are properly operated. Apart from the large energy consumption of buildings, peak energy demand certainly attracts lots of attention because of its negative impact on energy grid capital, operational cost and environmental pollution to name a few. This impact is a direct consequence of the carbon-intense generation plants that grid operators deploy to satisfy energy demand during these peak periods [1]. Demand Response (DR) activities including load shifting or peak shaving have a huge potential to match energy demand with energy supply side, thus avoiding these undesirable peaks [2]. The implementation of DR activities is especially promising in the residential sector, where the full capabilities of the DR are yet to be unlocked.

This is where the RESPOND H2020 project (<https://project-respond.eu>) originates, aiming to bring DR programs to neighborhoods across Europe. More specifically, RESPOND aims to deploy an interoperable energy automation, monitoring and control solution to deliver DR programs at a dwelling, building and district level to neighborhoods across Europe. To do so, RESPOND proposes an Artificial Intelligent (AI) system to assist residential consumers that would like to make use of DR and incorporate it into their energy management systems [3]. The proposed system considers the forecast energy consumption and production based on the data acquired by the deployed IoT equipment and looks for modifications that would mitigate potential instabilities in the energy supply network by applying optimal energy use and load shifting. This article focuses on the energy consumption forecasting part of the AI system, where the effectiveness of different models of time series and machine learning has been evaluated.

The rest of the article is structured as follows. The data available to make the study is introduced in Section 2. Then, the theoretical principles of the forecasting methods are explained in Section 3. In Section 4, the results obtained are summarized. Finally, conclusions of the lesson learned are written in Section 5.

2. Data Availability

The available electrical consumption data (measured in Wh) is obtained from different houses located in three different places: Madrid (Spain), Aarhus (Denmark) and the Aran Islands (Ireland).

Some indicators are calculated to assess the quality of the data available. The explanation of the data quality metrics used is out of the scope of this paper. There are some indicators that take values between 0 or 1 depending on the quality of the data in some respects. A value of 0 is considered the worst possible quality and a value of 1 represents the perfect quality for that indicator. In general, the quality metrics obtain values close to 1 except the Timeliness indicator. When a sensor fails for whatever reason, it stops sending data, including the time value. Due to failures in sensors, waiting times occur in the time variable. This fact is reflected in the low values of Timeliness indicators.

Even though all the data collection processes began on the same date, on 1 January 2019 it can be observed that not all the pilot sites have the same data availability. The analysis was performed on 6 March 2020. It is decided to eliminate from the study the houses with more than 30% of missing data. Following this criterion, data are available from 10 houses in Madrid, 11 in Aarhus and 8 in the Aran Islands.

3. Methodology

Traditionally, the energy demand forecasting has been addressed via data-driven algorithms due to their high performance. Therefore, RESPOND's Energy Demand Forecasting Service has targeted these algorithms with views to having the best performance possible.

In what follows, a short description of some well-known forecasting methods are presented. Auto Regressive Integrated Moving Average (ARIMA) are models designed to make forecasting in time series and past values of the response variables are used to estimate the future values. Linear Regression (LR), Support Vector Regression (SVR) and k-Nearest Neighbors (KNN) are machine learning algorithms that use explanatory variables to estimate the values of the response variable. First, we decide that the explanatory input variables in the predictive models were extracted from the time variable. On the one hand, this agreement provides simplicity to the models and allows the results to be explained. On the other hand, continuity in time allows the imputation of the missing data in case of sensor failure. These variables, also called features, were *day*, *sinMonth*, *cosMonth*, *sinHour*, *cosHour*, *Season*, *weekday* and *workingDay*. Before creating models, we identified some outlier values. These are values that excessively exceed the typical values for electrical consumption. After observing the behavior of the consumption data for the different houses, we concluded that a common pattern would lack precision. Finally, we decided to remove values greater than 3000 Wh. These values are considered meaningless and possibly caused by a failure in the data collection method.

The Root Mean Square Error (RMSE) is used to measure the standard deviation of the prediction errors. Therefore, the forecasts with the lowest RMSE are considered the best fit.

The calculations have been executed with the statistical software R. The *caret* and *forecast* packages have implemented the necessary functions to carry out the predictions, perform cross validation and automatically search for the optimal values of the necessary parameters in each algorithm.

3.1. ARIMA

In the process of finding the best predictive model, we started with Autoregressive Integrated Moving Average $ARIMA(p,d,q)$ models. Those models are fitted to time series data to predict future points where data show evidence of non-stationarity. Time series can be transformed into stationary by differentiation d times. Once the series is stationary, we used the classic explanatory methods to

choose the orders p and q based on the comparison of *Akaike Information Criterion* (AIC) and *Bayesian Information Criterion* (BIC).

3.2. Linear Regression

Linear regression (LR) attempts to model the relationship between two variables by fitting a linear equation to observed data [4]. A linear regression line has an equation of the form $Y = \beta_0 + \beta_1 X$, where X is the explanatory variable and Y is the dependent variable. Multiple Linear Regression (MLR) uses more than one explanatory variable to fit the response variable.

3.3. Support Vector Regression (SVR)

SVR uses the same principles as Support Vector Machine (SVM) but it is used in a regression method, so we can use SVR for working with continuous values. SVR allows the definition of the width of the band around the error in our model and the discovery of an appropriate hyperplane to fit the data. The objective function of SVR is to minimize the coefficients, not the squared error. The error term is instead handled in the constraints, where we set the absolute error less than or equal to a specified margin, called the maximum error, ϵ (epsilon) (<https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2>) [5].

3.4. K-Nearest Neighbors

K-nearest neighbors algorithm (KNN) is a supervised machine learning algorithm that can be used to solve regression models. First, the distance between the explanatory variables of the point x and the rest of observations should be calculated. Therefore, each point has a distance value associated. The k nearest neighbors to the point x are the observations with the lower distance. These k observations are used to compute the value of the predictor Y . The value of Y in some point is the average of the dependent variable of its k nearest neighbors [6].

4. Results and Discussion

This section summarizes the results obtained from the forecasts using the different methods mentioned above. The RMSE shown is the mean of the errors obtained in the 29 available houses.

4.1. ARIMA

Due to high amount of data, the search for the optimal p and q was neither simple nor satisfactory. It has multiple functions for the treatment of time series in R. Specifically, we used a method that finds the best Seasonal Autoregressive Integrated Moving Average (SARIMA) model. The idea is that SARIMA models are $ARIMA(p, d, q)$ models whose residues are $ARIMA(P, D, Q)$. The RMSE value achieved using ARIMA model was 1540.07 Wh whereas the RMSE was 1294.81 Wh using SARIMA model.

4.2. Linear Regression

Linear relationship between dependent variables and independent variable was found. Although the RMSE was significantly lower than in the results obtained with ARIMA, the Coefficient of determination (R^2) was less than 0.3 in all fitted models. The best RMSE value was 540.22 Wh and it was obtained when train the model with all the time related features. Furthermore, the predictions seem to repeat the same pattern every day.

4.3. Support Vector Regression (SVR)

Electricity consumption took negative values in some cases using SVR and this makes no sense. Electric consumption can never be less than 0 Wh. Although RMSE obtained was lower than the

previous, the method was rejected because this problem could not be controlled. Furthermore, SVR is computationally expensive and takes too much time and resources to forecast.

4.4. K-Nearest Neighbors

The best fit with the KNN algorithm was obtained using all the mentioned exploratory variables. In all cases the RMSE takes values between 156 and 439 Wh. Furthermore, the k value indicating the number of neighbors to be used was tested from 1 to 10. All forecasts were obtained using less than 5 neighbors.

5. Conclusions

Data quality is important to ensure that forecasting models work properly and take advantage of the useful information provided by historical data. The low RMSE obtained with the KNN shows that it is an optimal algorithm with which to make the forecast of electricity demand. Depending on the data availability, performance of forecasters varies. Therefore, predictive models are periodically re-trained as they are expected to improve their performance as a bigger historical data size is available.

Author Contributions: Data Curation, Formal Analysis and Conceptualization, M.G.-O.; Investigation and Writing M.G.-O. and I.E.-G., Supervision S.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by RESPOND (integrated demand REsponse Solution towards energy POSitive Neighbourhoods) project grant number 768619.

Acknowledgments: This work is partly supported by the RESPOND (integrated demand REsponse Solution towards energy POSitive Neighbourhoods) project, which has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement no. 768619.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Collins, L.D.; Middleton, R.H. Distributed demand peak reduction with non-cooperative players and minimal communication. *IEEE Trans. Smart Grid* **2017**, *10*, 153–162.
2. Albadi, M.H.; El-Saadany, E.F. Demand response in electricity markets: An overview. In Proceedings of the 2007 IEEE Power Engineering Society General Meeting, Tampa, FL, USA, 24–28 June 2007; pp. 1–5.
3. Esnaola-Gonzalez, I.; Diez, F.J.; Pujic, D.; Jelic, M.; Tomasevic, N. An artificial intelligent system for demand response in neighbourhoods. In Proceedings of the Workshop on Artificial Intelligence in Power and Energy Systems (AIPES 2020), Santiago de Compostela, Spain, 4 September 2020. doi:10.13140/RG.2.2.30279.32163.
4. Aalen, O.O. A linear regression model for the analysis of life times. *Stat. Med.* **1989**, doi:10.1002/sim.4780080803.
5. Drucker, H.; Surges, C.J.; Kaufman, L.; Smola, A.; Vapnik, V. Support vector regression machines. *Adv. Neural Inf. Process. Syst.* **1996**, *9*, 155–161.
6. Keller, J.M.; Gray, M.R. A Fuzzy K-Nearest Neighbor Algorithm. *IEEE Trans. Syst. Man Cybern.* **1985**, doi:10.1109/TSMC.1985.6313426.

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



© 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).



Short-Term Forecasting Methodology for Energy Demand in Residential Buildings and the Impact of the COVID-19 Pandemic on Forecasts

Meritxell Gomez-Omella^{1,2} , Iker Esnaola-Gonzalez¹ ,
and Susana Ferreiro¹

¹ TEKNIKER, Basque Research and Technology Alliance (BRTA),
Iñaki Goenaga 5, 20600 Eibar, Spain

{meritxell.gomez, iker.esnaola, susana.ferreiro}@tekniker.es

² Faculty of Informatics, University on the Basque Country (UPV/EHU),
Paseo Manuel Lardizabal, 1, 20018 Donostia-San Sebastian, Spain

Abstract. Demand Response (DR) can contribute towards the energy efficiency in buildings, which is one of the major concerns among governments, scientists, and researchers. DR programs rely on the anticipation to electric demand peaks, for which the development of short-term electric demand forecasting models may be valuable. This article presents two different variants of the KNN algorithm to predict short-term electric demand for apartments located in Madrid (Spain). On the one hand, the use of an approach based on the estimation of a Machine Learning model (KNFTS) is studied. In this method, time-related and date-related features are used as exploratory variables. On the other hand, a method based on the recognition of similar patterns in the time series (KNPTS) is analysed. The Edit Distance for Real Sequences (EDR), Root Mean Square Error (RMSE) and Dynamic Time Warping (DTW) are used to measure the accuracy of forecasts for both approaches. The experiments demonstrate that the KNPTS has a higher accuracy over the KNFTS when predicting the short-term electric demand. Furthermore, the models' adaptation to unusual situations is showcased in this article. The impact of the COVID-19 pandemic derived in a worldwide electric demand drop due to the lockdown and other confinement measures, and the retraining method proposed for the KNPTS model has been demonstrated to be valid, as it improves the forecasting accuracy.

Keywords: Time series forecasting · K-nearest neighbours · COVID-19 · Demand Response

1 Introduction

The energy consumed by buildings has dramatically increased over the last decade due to diverse causes including the population growth, an increase in

the time spent indoors or the rise of demand for building functions and indoor quality [1]. As a matter of fact, ensuring the energy efficiency of the built environment is one of the major concerns among governments, scientists and researchers since the building sector consumes more than 35% of the global energy [2].

Apart from the total energy consumed, another challenge is the management of peak energy demands. The energy during these periods is supplied by the Peaking Power Plants, which are carbon-intense generation plants with a negative impact on energy grid capital, operational cost and environmental aspects. In this regard, DSM (Demand Side Management) actions such as load curtailment (i.e. a reduction of electricity usage) or reallocation (i.e. a shift of energy usage to other off-peak periods) have the potential to minimise the undesirable peak periods, and consequently, to reduce the use of the Peaking Power Plants. In combination with DSM activities, the Demand Response (DR) can influence customer's use of electricity in ways that will produce desired changes in the utility's load shape, that is, changes in the time pattern and magnitude of a utility's load [3]. DR can be understood as the set of technologies or programs that concentrate on shifting energy use [4], and although traditionally the DR programs' implementation has been limited to industrial buildings due to their high energy demand, residential buildings are specially promising due to their potential to reduce demand peaks [5]. The RESPOND H2020 project¹ aims to bring DR programs to the residential sector in order to reduce these undesirable energy demand peaks. To do so, it is necessary to identify these demand peaks ahead of time. This is why, the RESPOND Energy Demand Forecasting service is developed, which allows the forecasting of electric demand for houses participating in the project.

In this article, the development of the forecasting models that are the main base of the Energy Demand Forecasting service is addressed. And subsequently, the degradation of the models during the COVID-19 pandemic in Spain is analysed and an approach to mitigate this situation is proposed.

The rest of the article is organised as follows. Section 2 presents two different methods to multi-step ahead forecasting in time series. All the steps are described including the forecasting strategies, the forecasting methods, and the performance measurement metrics. Next, Sect. 3 describes the context in which this study is carried out. The data set is presented, and the implementation and deployment of the methodology are explained. As a result, in Sect. 4 the results are discussed, and it is shown how to forecast electric demand during the COVID-19 crisis. Finally, Sect. 5 introduces the conclusions and future work.

2 Short-Term Forecasting Methodology for Time Series

A Time Series is a collection of values $Y = \{y_t\}$ obtained over time, often at regular intervals, where $t = 1, \dots, T$ represents the time elapsed. A discrete time series is the one that the index set contains discrete points of time. The

¹ <http://project-respond.eu/>.

consecutive observations can usually be recorded at equally spaced time intervals such as hourly, daily, weekly, monthly or yearly time separations [6].

The interest in the analysis of time series lies in the estimation of the behaviour of the historical data to forecast future values. Time series forecasting is a method to estimate future values in a temporal sequence. Given a univariate time series $\{y_t\}$ where $t = 1, \dots, T$ the interest of this work is focused on the forecasting of future values $\{y_{T+h}\}$ with $h \in \mathbb{N}$. A single-step ahead forecasting is made when $h = 1$ and only the next value in the series is estimated. Multi-step ahead forecasting is used to forecast more than one value, that is $h > 1$.

2.1 Strategies for Multi-Step Ahead Time Series Forecasting

A multi-step ahead time series forecasting consists of estimating the next h values $\{y_{T+1}, \dots, y_{T+h}\}$ using the previous values $\{y_1, \dots, y_T\}$. In order to forecast more than one value $h > 1$ in time series using Machine Learning (ML) algorithms, five strategies can be chosen called *Recursive*, *Direct*, *DirRec*, *MIMO* and *DIRMO* strategies. This work is focused on *Multi-Input Multi-Output (MIMO) Strategy*. The remaining strategies are left out of this section and a detailed explanation of them can be found at [7].

The *MIMO* strategy executes one prediction after learning a single multiple-output model. It consists of a target vector of length equal to the number of points to be forecast h and a feature vector containing the previous values. The same model is used for all the horizons so the computational time needed is less than necessary for strategies that include the *Direct* approach (*Direct*, *DirRec* and *DIRMO* strategies). In addition, the accumulation of errors that is obtained using the recursive strategy does not happen. The forecasts are returned in one-step by a multiple-output model.

2.2 KNN Algorithm

The methodology presented in this work is based on the KNN algorithm, which is a non-parametric method used for classification and regression problems in ML. The general idea is to examine the distance between the independent variables, then choose the K nearest observations and finally use a combination of their response values to estimate the next value of the output variable. In order to forecast future values using a ML algorithm, some decisions have to be made including the multi-step ahead strategy and the selection of input variables. Particularly for the KNN, other decisions are made: the distance function to evaluate the similarity between instances, the value of K nearest neighbours and the way output points are combined [8]. Different distance functions can be used to define the similarity between instances, although the most commonly used is the Euclidean distance [9]. A cross-validation can be implemented to evaluate different values of K , looking for the one that minimises the error. This error is a metric chosen to measure the difference between the estimated and actual values. Once the best parameter K is chosen, the future value is estimated as a combination of the target values of each K nearest neighbours. The most

common combination is the arithmetic mean, which assigns the same weight to all values. Alternatively, the distance weighted average can be used, where some data points contribute more than others to the estimated value. Assuming that d_k is the distance between the pattern and the k -th nearest neighbour, $\omega_k = 1/d_k^2$ is defined to be its weight. In order that the neighbors' weights add up to 1, they are each divided by the sum of all the weights.

In this article, two ways of making time series forecasting based on the KNN algorithm are proposed. On the one hand, an approach based on the estimation of a ML model where the features extracted from the time variable are used as exploratory variables. On the other hand, a method based on the recognition of similar patterns in the time series.

K-Nearest Features for Time Series (KNFTS). An important step prior to training any ML model is the choice of n exploratory variables of T values X_t^i , $i = 1, \dots, n$, $t = 1, \dots, T$. These variables are introduced as input into the model and serve as an aid to estimate the value of the response variable Y . Input variables are also called features in the field of ML, and feature engineering is the task of creating and using them to improve the performance of ML algorithms [10].

Exploratory variables X_t^i are extracted from the time variable when the data is a univariate time series. So the input variables X_t^i are date and time related features. Below is a description of some features X_t^i than are used to calculate the distances between observations.

- Date-related features are numerical values having information about day, month and year.
- Time-related features are extracted for the time stamp and are numerical variables like hour, minute and second.
- Weekday is a variable that takes values between 0 and 6 providing information of the day of the week from Monday to Sunday.

The nearest neighbours are obtained by comparing the Euclidean distance between X_t^i variables in each observation. Some of these variables have a cyclical meaning that is not reflected in the calculation of distances. For example, the Euclidean distance between January and December (represented by 0 and 11 respectively) is 11 but they are two consecutive months. To avoid this fact, a trigonometric transformation v is done in this kind of variables². First, the periodicity of the variable is found (in this case $P = 12$). Then, the Eq. 1 is applied to each value X_t^i , $t = 1, \dots, T$. Two variables are obtained from each of the transformations made (one from the sine and the other from the cosine).

$$v : [0, P] \longrightarrow [-1, 1]$$

$$X_t^i \rightarrow v(X_t^i) = \left(\sin \left(\frac{2\pi X_t^i}{P} \right), \cos \left(\frac{2\pi X_t^i}{P} \right) \right) \quad (1)$$

² <https://www.avanwyk.com/encoding-cyclical-features-for-deep-learning/>.

The Euclidean distance between $(X_{T+1}^1, \dots, X_{T+1}^n)$ and (X_t^1, \dots, X_t^n) , for all $t = 1, \dots, T$ is calculated to choose the K nearest neighbours. The next value Y_{T+1} is obtained as a weighted combination of the target values of the K nearest neighbour.

$$Y_{T+1} = \frac{\sum_{k=1}^K \omega_k Y^k}{\sum_{k=1}^K \omega_k} \tag{2}$$

where Y^k is the value of the target variable corresponding on the k nearest neighbour. This argument applies to all h horizon values to be forecast. The general idea of this method is that in two similar moments of time in terms of date and time, the value of the response variable should be similar.

K-Nearest Patterns in Time Series (KNPTS). The nearest neighbours are defined as the most similar subsets of data in the time series using this KNN approach. Given the window size $m \in \mathbb{N}$, a reference pattern consisting of the last values of the output variable (y_{T-m}, \dots, y_T) is used in the training process. This pattern is compared with all the other subsets of length m in the series, that is, $Y^j = (y_j, \dots, y_{j+m})$, for all $j = 1, \dots, T - m - h$ by calculating the Euclidean distance. Therefore, it is determined that the K subsets with the lowest Euclidean distance are the K nearest neighbours [11].

Figure 1 represents an example of nearest-neighbour multiple-step forecasting using MIMO strategy. It represents a simple case in which $m = 5$ previous values are used to find the $K = 1$ nearest neighbour. Then, the most similar subset is used to forecast the $h = 3$ future values of the time series.

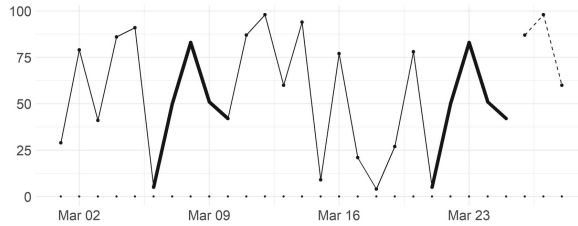


Fig. 1. Nearest-neighbour three-step-ahead forecasts. A window of length five is selected and one neighbour is used to estimate the next three values.

The next value Y_{T+1} is calculated by the weighted average of the next values of each k nearest neighbour.

$$Y_{T+1} = \frac{\sum_{k=1}^K \omega_k Y_{k+m+1}^k}{\sum_{k=1}^K \omega_k} \tag{3}$$

This argument applies to all h horizon values to be forecast. The general idea of this method is that, in two different periods of time with similar values, the following value should be similar as well.

2.3 Evaluation

In the context of this article, the following three metrics have been considered to assess the accuracy of the forecasting models, as they are representative for measure the similarity between time series: Root Mean Square Error (RMSE), Dynamic Time Warping (DTW) and Edit Distance for Real Sequences (EDR) [12].

- (i) **Root Mean Square Error (RMSE)** The RMSE measures the error by calculating the difference between the current and predicted values and it has been widely used for the evaluation of the goodness of fit in regression models. The RMSE value is always non-negative and the lower the RMSE, the more similar the time series are.
- (ii) **Dynamic Time Warping (DTW)** The DTW algorithm is used to compare the similarity or calculate the distance between two time series and minimises the effects of shifting and distortion in time [?]. A dynamic programming approach is used to align the series and allows to compare series of different length [13]. The lower the DTW value, the more similar the time series are.
- (iii) **Edit Distance for Real Sequences** Edit Distance (ED) was initially used to calculate the similarity between two strings. It quantifies the basic edit operations (insert, delete and replace) needed to transform one string into the other. In order to define matching between numerical series, different adaptations are proposed in [14]. Given a positive threshold ϵ , the distance between two points is reduced to 0 or 1 as below.

$$d_{\epsilon}(\hat{y}_t, y_t) = \begin{cases} 0, & d(\hat{y}_t, y_t) \leq \epsilon \\ 1, & d(\hat{y}_t, y_t) > \epsilon \end{cases} \quad (4)$$

The EDR metric is the number of edit operations needed to transform a numerical series into an other, taking into account the Eq. 4. The lower the EDR value, the more similar the time series are.

3 The RESPOND Energy Demand Forecasting Service

The RESPOND project aims to bring DR programs to neighbourhoods across Europe, and in particular, to dispatch real-time optimal DR strategies to dwellers towards the achievement of demand peak reduction. RESPOND has developed an Artificial Intelligence (AI) system to detect potential energy conservation opportunities while ensuring the occupants' required comfort levels [15]. One of the components of this AI system is the RESPOND Energy Demand Forecasting service, which allows the short-term forecasting (i.e. hourly for the upcoming 24 h) of electric demand for the houses participating in the project. In total, over 40 houses are participating in the RESPOND project, including participants from Aarhus (Denmark), the Aran Islands (Ireland) and Madrid (Spain).

Unlike the regularity within commercial buildings, electric consumption in the residential sector may vary significantly from house to house. As a matter of

fact, this high electricity usage variance derives from the users' lifestyle, occupancy behaviour, building characteristics and calendar information [16]. Figure 2 shows the differences in the data distributions in some apartments in Madrid.

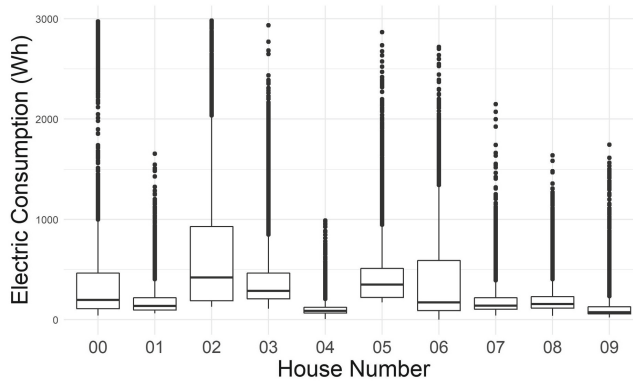


Fig. 2. The electric consumption data distribution in Madrid apartments.

3.1 The Methodology Implementation

In this section, the methodology explained in Sect. 2 is implemented. The rationale behind choosing the KNN algorithm over other methods is based on the results of a previous experimentation, which is out of scope of this article.

The strategy chosen for the multi-step ahead time series forecasting was *MIMO*, based on the comparison made in the article [7].

The *hour*, *day* and *month* are used as features for the KNFTS model training process. Four new features are created from the *hour* and *month* (called *sin.Hour*, *cos.Hour*, *sin.Month* and *cos.Month*), by calculating the sine and cosine (trigonometric transformations explained in the Sect. 2.2), to have more discriminatory power than in the original space. And apart from these, new features are added (*Season* and *Workingday*) taking into account the behaviour of electric consumption data. A snippet of this data can be seen in Table 1.

Table 1. A snippet of the electric consumption data set for a participant house.

Timestamp	EC	sin.Hour	cos.Hour	Day	sin.Month	cos.Month	Season	Weekday	Workingday
2019-10-10 15:00:00	114.8796	-0.7071	-0.7071	10	-0.8660	0.5000	Fall	Thursday	Yes
2019-10-10 16:00:00	156.0299	-0.8660	-0.5	10	-0.8660	0.5	Fall	Thursday	Yes
2019-10-10 17:00:00	350.0857	-0.9659	-0.2588	10	-0.8660	0.5	Fall	Thursday	Yes
...
2019-10-12 00:00:00	98.7279	0	1	12	-0.8660	0.5000	Fall	Saturday	No

In the training process, to calculate good estimates of the error rate of the models, a 10-fold cross validation is applied and repeated 5 times. For both methods, the K value is tested from 1 to 10 to find the optimal value. And finally, EDR, RMSE and DTW error measures are calculated. The EDR is considered the metric that best evaluates the error, whose objective is to obtain a forecast that adapts to reality. The RMSE compares point by point and penalises a prediction of a value at a slightly deviated point of time. Although DTW considers the deviation over time, it strictly compares two similar values. EDR, instead, allows to evaluate the similarity considering the elongation in time and the similarity of close values by setting a threshold value. A threshold of $\epsilon = 30$ Wh is established in the Formula 4 after observing the common behaviour of the series of electricity consumption available. Since the EDR is the number of edit operations necessary to convert one series into another, the model with the lowest EDR is chosen. The maximum EDR between any two series is infinity unless restrictions are added. In this problem the maximum EDR value is considered as the length of the series. This number comes from changing all values in one series to values from the other. However, all three measurements are calculated in this study. The model with the lower EDR is chosen as the one that provides the more accurate forecasts. If there is more than one model with the same EDR result, the DTW and RMSE values are compared. RMSE and DTW values are obtained from the calculation of distances between consumption values and are not useful to compare between houses since, as previously mentioned, each one presents a different variability in the data. For this reason, the scale on which it is measured is important. In order to compare more efficiently the values obtained from the different apartments, these two error measures are normalised by the average as follows.

$$NRMSE = \frac{RMSE}{\bar{y}} \quad NDTW = \frac{DTW}{\bar{y}} \quad (5)$$

where \bar{y} is the average of the historical consumption $\{y_1, \dots, y_T\}$.

3.2 The Service Deployment

Based on the proposed methodology, two models (based on KNFTS and KNPTS methods) has been developed for each of the 40 houses participating in RESPOND project. In total, 80 models that forecast the hourly electric consumption for the upcoming 24h. These models have been developed in R programming language. The KNFTS models have been developed with the functions within the caret package³. However, the functions used to develop the KNPTS have been implemented by the authors for this work. As for the functions to evaluate the similarity between estimated and actual values in terms of DTW and EDR, they have been taken from the TSdist package [12].

The developed predictive models have been exported in the form of *.rds* files and deployed in an R Server, where they are currently automatically executed

³ <http://topepo.github.io/caret/index.html>.

using periodical tasks executed by a *cron daemon* process. Furthermore, both the predictions made by the models and the actual values, subsequently obtained, are stored in a database for further exploitation (data visualisation and data analysis purposes).

4 Evaluation and Results Discussion

In this section, the accuracy of both forecasting methods proposed in Sect. 2 to estimate a day ahead forecast are evaluated and discussed. The results collected below are focused on the houses located in Madrid. Table 2 shows the average of the performance measures evaluated on ten different random days. The data prior to those dates is used to train the model and then, a 24 h ahead forecasting is done.

As explained above, the performance measure that provides more information in our case is the EDR. The value of the EDR is lower for the KNPTS rather than for the KNFTS method in all the houses, and the same happens with the NRMSE and NDTW for most of the houses.

Table 2. Comparative results of forecast accuracy with KNFTS and KNPTS

	EDR		NRMSE		NDTW	
	KNFTS	KNPTS	KNFTS	KNPTS	KNFTS	KNPTS
House 00	19.3	18	0.8812	1.0010	12.4415	15.0408
House 01	14.8	14.3	0.8772	0.7343	13.0310	9.6848
House 02	19	17.6	0.6403	0.6533	9.8858	9.2594
House 03	16.6	14.4	0.5109	0.3383	7.3718	5.7123
House 04	10.1	9.1	0.5983	0.5995	7.7291	7.6235
House 05	18.4	17.1	0.7383	0.6793	8.8291	8.3404
House 06	18.3	15.6	1.0956	0.9926	17.7200	14.9810
House 07	12.6	12.4	0.6840	0.6762	9.5653	9.3508
House 08	15.2	13.5	0.6915	0.6643	10.3706	8.6867
House 09	10.6	8.5	0.7408	0.7057	9.2065	8.3519

In House 02, the NRMSE value is 0.013 higher for the KNPTS. The average electric consumption in that house is 634.92 Wh, which means that a mean deviation of 8.25 Wh happens in each of the predicted points. This value is not considered high enough to conclude that the forecaster's accuracy overall is worse in the KNPTS compared with the KNFTS. Furthermore, the RMSE is a performance measurement that compares a point by point performance, that is, it compares each real consumption value with its corresponding forecast at the same instant. For this reasons, the KNPTS method is identified as better approach considering the best results in EDR and NDTW. A similar situation

happened in House 04, where the KNPTS has a higher NRMSE, but it is too low (0.001, which means a deviation of 0.13 Wh in each point) to consider that it has a worse performance than the KNFTS.

As for House 00, the NRMSE and NDTW values are higher in the KNPTS compared with the KNFTS. The variation of 0.1198 in the NRMSE represents a mean deviation of 47.58 Wh. The difference in NDTW is 2.5993, equivalent to 1032 Wh of total distance between forecasts in 24 h (43.02 Wh per hour on average). The average of the electric consumption in House 00 is 397.20 Wh and the standard deviation is 507.8 Wh so, the mentioned mean deviation is not considered high enough either.

In the rest of the houses, the KNPTS obtains a considerable improvement for the three performance measures. In particular, House 06 gets an average improvement of 2.7 edition changes in EDR. Given two series of 24 points each, the maximum value for the EDR metric is 24 (the number of values to be forecast). An improvement of 2.7 changes represents an increase in EDR of 11.25%. Figure 3a shows a 24 h ahead forecast in House 06 and Fig. 3b shows the comparison of the three error measures for this forecast. This evaluation demonstrated

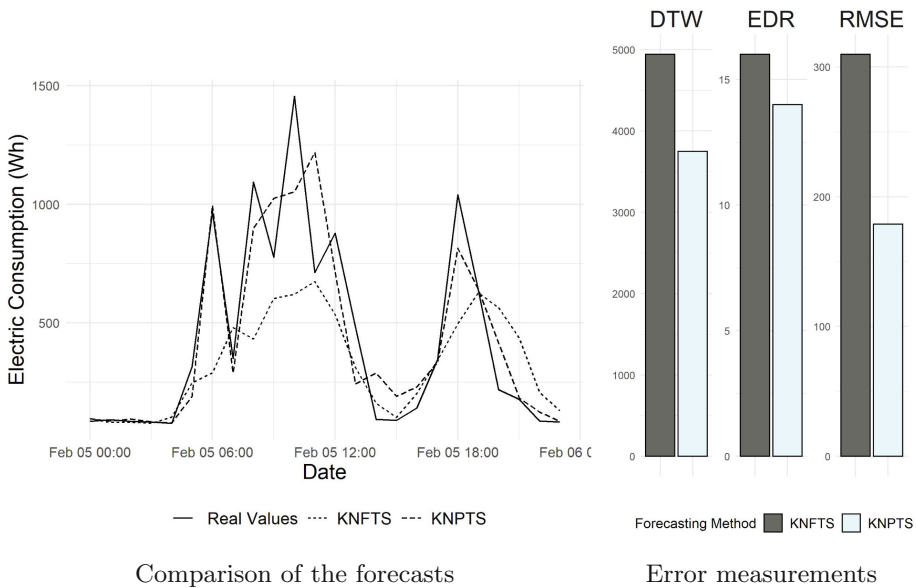


Fig. 3. Evaluation of one-day ahead forecasting with KNFTS and KNPTS.

that the forecasts and the real values are more similar in the results obtained with the KNPTS method. This suggests that KNPTS method is more accurate to forecast the electrical consumption of the Madrid apartments. Then, KNPTS method is implemented for the future models of the RESPOND Electric Demand Forecasting service.

4.1 COVID-19 Impact on Electric Demand Forecasting

According to the International Energy Agency's latest report⁴, the impact of the COVID-19 pandemic derived in a worldwide electric demand drop due to the lockdowns and other confinement measures, although this demand is steadily recovering now that measures are softened. In the case of Spain, according to Cornwall Insight during the first four weeks of the state of alarm, power demand fell by 16.7%. However, this fall is attributable mainly to industry and commerce, since according to the Spanish Consumer and User Organisation, there has been an increase of 28% in household bills in Spain⁵.

The average daily consumption in the Madrid apartments participating in RESPOND has stabilised around a lower value since the beginning of the lockdown. Thus, as of March 14, the electric consumption data distribution changed and the data variability has decreased. This fact is evidence of the change in the distribution of the data in such a way that the electric consumption among days is more similar, without ceasing to appear the characteristic peaks of electricity consumption. This is definitely an indicator that the obligation to remain confined due to the state of alarm has altered users' daily habits. Therefore, forecasting models trained on normal human behaviour data are finding that the normality has changed and they are no longer working as expected. And the RESPOND Energy Demand Forecasting service is no exception.

Under normal conditions, forecasting models' performance degrade over time due to a change in the environment that violates the models assumptions [17]. This fact is known as concept drift, and the most common method to deal with it is the retraining of the model, which consists in re-running the process that generated the previous model on the new set of data available. However, the unusual change of daily habits derived by the COVID-19 pandemic, the typical retraining strategies are not sufficient.

The correlations between the features and their distributions are commonly examined to identify the concept drift. In the case of the KNPTS, there are no features, but the change in the accuracy of the predictions is given by a change in the output variable. Furthermore, the reason for this change is known (i.e. the confinement measures) and therefore, action can be taken to control it. The data preprocessing task prior to the training of any forecasting model includes the selection of the appropriate data and features to improve the precision of the estimation. Understanding the importance of the distance between instances in the KNN methodology, a necessary step before training the forecast model for estimating electric consumption during the lockdown days, is the selection of the data generated during the lockdown. This way, the model is more effective since it finds the nearest neighbours within a data set where the variability is less. This restriction prevents estimates to be based on the data from the previous year, as it is known that the behaviour of the electric consumption is not the same.

⁴ <https://www.iea.org/reports/covid-19-impact-on-electricity>.

⁵ <https://www.ocu.org/vivienda-y-energia/gas-luz/noticias/aumento-consumo-electrico-confinamiento>.

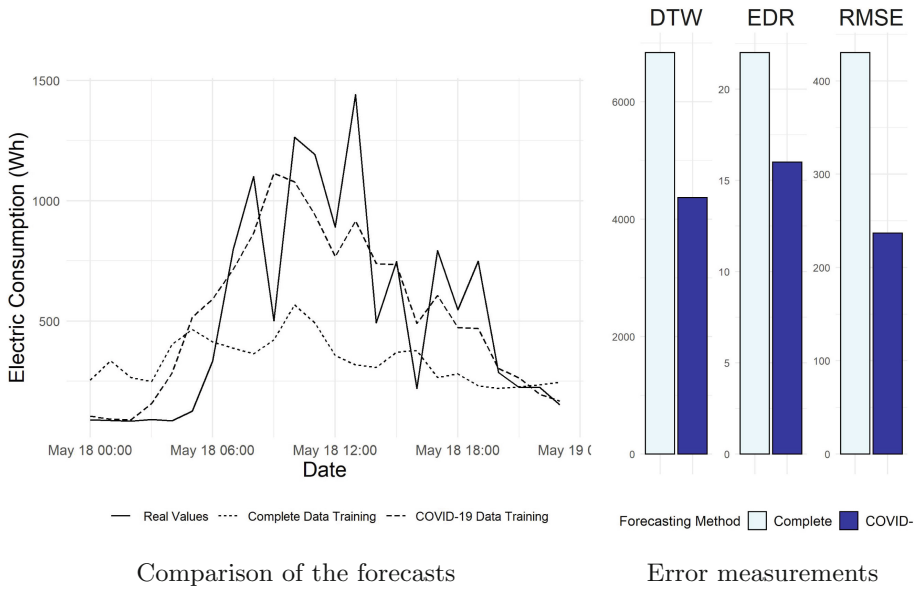


Fig. 4. Improving the accuracy of electric consumption forecasting in the COVID crisis.

The model trained using only lockdown data gives better results than using all the data available. Figure 4 shows an example of the comparison of the one-day ahead forecast using the complete historical data available and using only the lockdown data after March 14. These results were obtained for House 06.

An improvement of the estimation of the peak values is observed when only the lockdown data is used. The model trained with complete data gets a EDR value of 22, while the model developed for estimating the electric consumption of days prior to March 14 got a EDR value of 16. In order to reduce the EDR value for forecasts after March 14, the model is trained only with lockdown data. This new model gets an EDR value of 16, which represents an improvement of 25%. A decrease of 193.53 Wh per hour on average in RMSE and a decrease of 2466.38 Wh in total distance between both forecasts in DTW (102.77 Wh per hour) can be observed.

The improvement derived from the retraining approach followed for the COVID-19 can be observed in all the houses in Madrid during the lockdown period. The average EDR reduction is 6.67%.

5 Conclusions

In order to dispatch real-time optimal DR strategies, a reliable and accurate forecast of short-term electricity demand is needed. In this work, two different methods based on the KNN algorithm have been developed and validated: the KNFTS based on the similarity of time-related and date-related features, and the KNPTS based on the similarity of the sequences of the response variable. Both

approaches are identified as adequate for the problem of forecasting electricity consumption after previous experimentation with other ML algorithms (beyond the scope of this article). The extraction of date and time-related features and the use of past values of the output variable to forecast future consumption values was satisfactory. It suitably considers the relation between instances in the time series.

The accuracy of the forecast has been tested in 10 different apartments located in Madrid, and their performance has been evaluated based on the EDR, NDTW and NRMSE error measures. The EDR value has been the most effective measure to assess the accuracy of multi-step ahead forecasting in time series, although it is not commonly used in previous works. It provides a more realistic measure of the number of changes to be made and therefore, it does not penalize error in time. The results have showed that the predictions are more accurate using KNPTS model. The EDR metric gets lower values for all the cases. The average obtained is 1.44 edit operation and it represents an improvement of the 6%. As for the NDTW, it improved on the 90% of the houses, and the NRMSE in the 70%.

The COVID-19 pandemic changed users' electricity consumption habits, which led to the degradation of the performance of forecast models. This is understood as the concept drift problem and the models had to be retrained. The results have showed that the retraining of the models using only data from the lockdown period, provides an average improvement of 6.67% in terms of EDR.

5.1 Future Work

The improvement in the prediction of peak values has been observed using KNPTS method instead of KNFTS. However, the accuracy to predict these values remains a challenge for time series forecasting models. A pattern classification model in the series could be useful in dealing with sharp peaks. It consists of a step prior to prediction and allow the choice of trained models with similar data. Another solution that might be appropriate would be the combination of both models, assigning more weight to the delayed values of the output variable.

Acknowledgments. This work is partly supported by the RESPOND (integrated demand REsponse Solution towards energy POSitive Neighbourhoods) and the REACT (Renewable Energy for self-sustAinable island CommuniTies) projects, which have received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 768619 and no. 824395 respectively.

References

1. Cao, X., Dai, X., Liu, J.: Building energy-consumption status worldwide and the state-of-the-art technologies for zero-energy buildings during the past decade. *Energy Build.* **128**, 198–213 (2016)

2. Global Alliance for Buildings and Construction, International Energy Agency and the United Nations Environment Programme. 2019 global status report for buildings and construction: Towards a zero-emission, efficient and resilient buildings and construction sector. Technical report (2019)
3. Contreras, J., Asensio, M., de Quevedo, P.M., Muñoz-Delgado, G., Montoya-Bueno, S.: Demand response modeling (chap. 4). In: Contreras, J., Asensio, M., de Quevedo, P.M., Muñoz-Delgado, G., Montoya-Bueno, S. (eds.) Joint RES and Distribution Network Expansion Planning Under a Demand Response Framework, pp. 33–40. Academic Press (2016)
4. Warren, P.: A review of demand-side management policy in the UK. *Renew. Sustain. Energy Rev.* **29**, 941–951 (2014)
5. Bartusch, C., Alvehag, K.: Further exploring the potential of residential demand response programs in electricity distribution. *Appl. Energy* **125**, 39–59 (2014)
6. Adhikari, R., Agrawal, R.K.: An introductory study on time series modeling and forecasting. Ph.D. thesis (2013)
7. Ben Taieb, S., Bontempi, G., Atiya, A.F., Sorjamaa, A.: A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Syst. Appl.* **39**, 7067–7083 (2012)
8. Martínez, F., Frias, M.P., Pérez, M.D., Rivera, A.J.: A methodology for applying k-nearest neighbor to time series forecasting. *Artif. Intell. Rev.* **52**, 2019–2037 (2019). <https://doi.org/10.1007/s10462-017-9593-z>
9. Abu Alfeilat, H.A., et al.: Effects of distance measure choice on k-nearest neighbor classifier performance: a review. *Big Data* **7**, 221–248 (2019)
10. Zheng, A., Casari, A.: Feature Engineering for Machine Learning. O'Reilly Media, Inc., Sebastopol (2018)
11. Bontempi, G., Ben Taieb, S., Le Borgne, Y.-A.: Machine learning strategies for time series forecasting. In: Aufaure, M.-A., Zimányi, E. (eds.) eBISS 2012. LNBIP, vol. 138, pp. 62–77. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36318-4_3
12. Mori, U., Mendiburu, A., Lozano, J.A.: Distance measures for time series in R: the TSdist package. *R J.* **8**, 451 (2016)
13. Berndt, D., Clifford, J.: Using dynamic time warping to find patterns in time series. In: Workshop on Knowledge Knowledge Discovery in Databases (1994)
14. Chen, L., Özsu, M.T., Oria, V.: Robust and fast similarity search for moving object trajectories. In: Proceedings of the ACM SIGMOD International Conference on Management of Data (2005)
15. Esnaola-Gonzalez, I., Diez, F.J., Pujic, D., Jelic, M., Tomasevic, N.: An artificial intelligent system for demand response in neighbourhoods. In: AIPES - The Workshop on Artificial Intelligence in Power and Energy Systems (Accepted, to be published)
16. Lusic, P., Khalilpour, K.R., Andrew, L., Liebman, A.: Short-term residential load forecasting: impact of calendar effects and forecast granularity. *Appl. Energy* **205**, 654–669 (2017)
17. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. *Mach. Learn.* **23**(1), 69–101 (1996). <https://doi.org/10.1023/A:1018046501280>

Divide, Propagate and Conquer: Splitting a Complex Diagnosis Problem for Early Detection of Faults in a Manufacturing Production Line

Kerman López de Calle - Etxabe¹, Meritxell Gómez - Omella² and Eider Garate - Perez³

^{1,2,3} *Tekniker, Basque Research and Technology Alliance (BRTA), Iñaki Goeanaga, 5, 20600, Eibar, Spain*
{kerman.lopezdecalle, meritxell.gomez, eider.garate}@tekniker.es

² *Faculty of Informatics, University of the Basque Country (UPV/EHU), Manuel Lardizabal Pasealekua, 20018, Donostia, Spain*

³ *Faculty of Science and Technology, University of the Basque Country (UPV/EHU), Barrio Sarriena s/n, 48940, Leioa, Spain*

ABSTRACT

This work elaborates the procedure followed by the HIRUTEK team to solve the data challenge proposed by the PHM 2021 organisation. This challenge deals with a manufacturing line that continuously tests fuses and suffers from several malfunctions. The solution addresses the diagnosis of the faults; the efficiency of the diagnosis; the identification of the signals related to each fault type; and, the identification of different operation settings that occur during the non-faulty conditions. This problem presents some difficulties that are common to machine fault diagnosis or manufacturing line monitoring; such as the class imbalance; the high amount of missing values; multicollinearity and high dimensionality; and, experimental noise. Additionally, the evaluation criteria presents further challenges such as the consideration of chronology and the detection of operation states (also referred in the literature as context awareness). The consideration of all these factors turns this exercise in a very representative and challenging problem. The solution here proposed, that obtained the highest score in the contest, relies on the combination of decision tree algorithms and a propagation system. The trees provide observation-wise diagnoses while the propagation system deals with chronology by adding a Kalman style filter that updates the probabilities, resulting in a more reliable result.

1. PROBLEM DESCRIPTION

1.1. Condition monitoring in manufacturing lines

The pursue of a more competitive manufacturing has led the production equipment to be more flexible, sustainable and re-

Kerman López de Calle - Etxabe et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

quire of less human supervision for its operation. Recent advances as the inclusion of sensors in the line have allowed the use of operator inputs together with the sensor measurements to determine the state of the manufacturing process (Stavropoulos, Chantzis, Doukas, Papacharalampopoulos, & Chrissolouris, 2013). This state or condition detection that is provided by decision systems allows line operators to take corrective actions which, in turn, improves the responsiveness in terms of the machine downtime reduction. However, as the decisions depend on the data received, providing reliable data is critical in order to optimize the manufacturing system (Assad et al., 2021)

Considering the previous, exploring decision-making algorithms for manufacturing systems at testbed level is of great interest, as it provides a better insight of the kind of problems that can be faced at industrial level and fosters the adoption of reliable decision-making algorithms in real industrial setups.

1.2. Experimental rig

The problem proposed by the 6th European Conference of Prognostics and Health Management Society 2021 (PHM) resembles a typical component of a large-scale quality-control pipeline of a production line. The experimental bed, courtesy of the Swiss Centre for Electronics and Microtechnology (CSEM), generates data similar to a real-world industrial manufacturing line.

The line consists of a 4-axis SCARA-robot with a vacuum gripper that picks up fuses from a feeder to a fuse-test-bench. On the test-bench, fuse current conductivity is measured, and, if the conductivity is appropriate, heating is applied to the fuse while a thermal camera measures during the heating process. Once the fuse is tested, it is sent back to the feeder with two conveyor belts.

1.3. Data

The testbed had different sensors installed that captured 50 signals measuring different physical properties of the system, such as pressure, vacuum, humidity, etc. Those signals were continuously measured during the experiments. However, some statistical descriptors were computed every 10 seconds instead of storing raw measurements. Those signal features were: counts (*Cnt*), frequency (*Freq*), maximum (*Max*), minimum (*Min*), standard deviation (*Std*), *Trend* and *value*, which were not computed for every signal.

During each experiment, lasting between 1 and 3 hours, disturbances were applied to some of them, simulating 8 different system failure conditions, labelled 2, 3, 4, 5, 7, 9, 11 and 12. With a total of 9 different classes, including the healthy class, labelled 0, in which no disturbances were introduced.

During the challenge the data was provided in two stages. Firstly, 50 healthy experiments and 4 experiments with each of the labels 2, 3, 5, 7 and 9 were released. Algorithm development began with those data. Later, another 20 data sets without failures and 3 of each of the 4, 11 and 12 faults were made available.

1.4. Evaluation

The data challenge organising committee proposed the following 4 criteria in order to assess the goodness of each approach to the problem.

- Identification and classification of faults: Determining which experiments are faulty and identifying which type of fault it is. That is, a fault diagnosis system that provides the class when given an experiment. This is tested by providing unseen experiments to the solution and checking how many of them are classified correctly.
- Root cause analysis: The solution had to provide a ranking of the signals in descending order of importance that could be causing each of the 8 failures. The appropriate signal should be among the top four signals, assigning a proportional score to the position of that signal in the ranking.
- Prediction in the shortest time: Algorithms are forced to consider the chronology of the experiments when providing the class. The solution is run twice to validate this assumption. In each run, the time required to reach a definitive diagnosis is provided. Later, this time is used to cut the experiment, repeat the process and ensure, this way, that the diagnostic system is robust (it returns the same diagnostic in the second run).
- System operation parameter identification: Apparently, the test rig can operate with two different operation parameter configurations that provide different sensors readings but are not causing faulty operation of the line. The solution should be able to identify the point where taken

in one or the other condition, without having any labelling related to the different conditions as they are all labelled as healthy experiments.

2. SOLUTION

2.1. Evaluation Related Inference

- The fourth evaluation criterion implied the existence of two data groups within the same label 0 (healthy). Hence, developing algorithms without considering this fact would lead to a worse diagnosis capability. For that reason, instead of considering this aspect as an optional bonus, it was decided to start tackling this point first.
- The challenge had an interesting system that validated the time allegedly spent by the algorithm to reach a solution. For that validation, the experiment was cut following the indicated time and it was fed back to the algorithm, ensuring that the class provided by the algorithm in the second attempt coincided with the one previously obtained. This evaluation system was designed to avoid data leaking from one run to the next, as the algorithm could not know whether it was being ran in the first (with the full experiment) or the second (with the reduced experiment) time. However, it was not perfect, as during the first run the algorithm could see the full experiment.

2.2. Issues

On the top of the previous observations, the first exploratory analyses unveiled some additional aspects that needed to be considered to properly tackle the problem:

1. Missing Values: Data values that are not properly stored or are missing can have a significant impact on the analysis and further conclusions. The complete data set contained 10% of missing values not identically distributed by the variables.
2. Class imbalance: Class imbalance occurs in classification predictive modelling when there is an unequal distribution of classes in the training set. Typically this kind of problem hinders the obtaining of reliable diagnosis algorithms since the traditional models and performance metrics (such as accuracy) assume a balanced class distribution. In a first analysis of the available data, clear imbalance was identified as there were only 4 experiments for each 2, 3, 5, 7 and 9 classes; 3 experiments for classes 4, 11 and 12; whereas there were 70 experiments belonging to class 0 (healthy).
3. Multicollinearity: Multicollinearity occurs typically in highly sensed environments where the same signal is described with various statistical descriptors. As different descriptors belonged to the same signal, and some signals were measuring similar effects, many of the resulting variables were highly correlated, which can be very harmful, particularly in linear models.

4. High dimensionality: A total of 248 variables were obtained in each test, with a number of observations ranging from 357 to 1081 per experiment. Comparatively, this amount of variables was high, making us aware of potential downsides of employing distance based techniques (due to the curse of dimensionality) or a high chance of facing overfitting if the algorithms used noisy signals when no relevant signal was found.
5. Bias in experiments/Experimental noise: In relation to the previous point, an additional source of noise in condition monitoring is the experimental noise. For any reason, exact replications of experiments in the same testbeds lead to have different signal values. In that regard, considering each observation as purely independent (as in most machine learning problems) is risky, as the inertia the systems have tends to be used by the algorithms to be capable of detecting the experiment in contrast to generalising the class value. For that reason, using standard randomly created train/test splits needed to be avoided.
6. Chronology in diagnosis: Considering chronology can be beneficial and detrimental at the same time. On the one hand, considering chronology could ease the identification of some faults that were not present on the complete signal due to their intermittent behaviour. On the other, considering chronology needed of tools that were not of-the-shelf, hence requiring to build ad-hoc designed algorithms to benefit from it.

2.3. Algorithm Development

Bearing in mind all the previously presented issues, an algorithm was developed by combining the techniques explained throughout this section. On the one hand, the statistical software R was used for the exploratory analyses. The libraries used to preprocess the data were `dplyr` (Wickham, François, Henry, & Müller, 2019), `imputeTS` (Moritz & Bartz-Beielstein, 2017) and `cluster` (Maechler, Rousseeuw, Struyf, Hubert, & Hornik, 2021). On the other hand, the development of the final algorithm was programmed in Python. `sklearn` library was used to train and validate the models and also to perform Principal component Analysis (Pedregosa et al., 2011) and `imblearn` to manage the imbalanced data (Lemaître, Nogueira, & Aridas, 2017).

2.3.1. Missing Value Handling

After a completeness analysis was carried out variable-wise, a relationship was identified between the appearance of missing values in some features. In many cases, when the feature *Cnt* (Counter) of a variable took a value of 0, the *Freq* feature is also 0 and the rest of the features (*Max*, *Min*, *Std*, *Trend*, *value*) did not have any value. For that reason, it was decided to eliminate from the study those variables that contained more than 80% of missing values in some tests. For the remaining variables, in each test the missing values

were imputed using the LOCF (Last Observation Carried Forward) (Barnes, Lindborg, & Seaman, 2006) method followed by a backward fill. This way the amount of information brought from "the future" was minimised as most of the values were imputed by the first forward imputation pass.

2.3.2. Validation of the Performance in Classifications

The effectiveness of the classification models was validated by comparing the results of metrics extracted from the confusion matrix. It was decided to use Recall because this metric penalises false positives (Ting, 2010). Therefore, Recall values close to 1 were desired, minimising the cases in which failed experiments were classified as healthy.

2.3.3. Identification of the Two System Parameter Configurations

For the identification of the two parameters sets under the healthy cases CLARA clustering algorithm was used forcing the algorithm to identify two clusters (Kaufman & Rousseeuw, 1986). This algorithm applies the Partition Around Medoids (PAM) in different samples of the dataset to obtain an optimal set of medoids. It was implemented using Manhattan distance for 50 samples with 500 observations each. The high dimensionality of the problem hindered the obtaining clear clusters, as, inside the same experiment, CLARA assigned very different proportions of class values. This was assumed to be incorrect, because according to the problem statement, each experiment could only contain a single set of configuration parameters.

Therefore, the experiments that were clustered clearly (with all the observations belonging to either cluster 0 or 1) were taken and a supervised decision tree was used to identify which rules were critical in their identification. This same model was applied to the rest of experiments (the ambiguous ones in the clustering) and the supervised model proved to be a perfect cluster classifier with only a single feature.

As a consequence, considering the inherent difficulties of training a model that had sub-classes inside a class, an additional label was created. From this point on, experiments with 0 label were split into two different classes according to the results of the decision tree.

2.3.4. Root Cause Analysis

The detection of important signals related to each fault was carried out with a PCA (Principal Component Analysis), as it is well suited for high dimensionality and multicollinearity scenarios. From the whole dataset (with all the available experiments), subsets containing the observations of the healthy class and the observations of each faulty class were created. In each subset, PCA was applied separately for the features coming from each signal, thus, as many PCAs as the number

of signals were computed. For each PCA, the number of components to be kept was optimised according to f1-score and a Classification Tree, in that way, the information of signal (from three to seven features) was condensed in a reduced set of principal components. This procedure was done for each data subset containing the observations of the healthy class and the observations of each faulty class. Once the signals were reduced with PCA, the decreasing impurity of a Classification Tree was used to identify the signals which might have caused the faults. The approach here was validated using LeaveOneGroupOut cross validation for each data subset with the healthy observations and faulty observations, leaving out a different faulty experiment in each iteration. Note that the results obtained in this process were used for the sole purpose of answering the root cause analysis task of the challenge. The knowledge gathered at this stage was used only as a notion of which features could be of interest for the diagnosis algorithm, as using the same signals was expected.

2.3.5. Diagnosis models

Fault identification process was split into two layers. In the first layer, algorithms were developed to provide the probability of a single observation (data from 10 second window) of belonging to the different classes. In the second layer or the propagation, the observation-wise probabilities were used by another algorithm to identify the underlying signal, the true class of the experiment, that was obtained by considering the chronological information.

The following key aspects were considered during the development of the diagnosis algorithms:

- In order to avoid experimental noise, instead of carrying out random training/testing partitions, LeaveOneGroupOut cross validation paradigms were favoured, as they did not leak data from the same experiment to the testing set resulting in a more reliable estimation of the error. Data from one experiment of each class was left out of the training phase and all of them were used to validate at each iteration.
- Imbalance was tackled by combining the SMOTE (Synthetic Minority Oversampling Technique) method (Chawla, Bowyer, Hall, & Kegelmeyer, 2002) followed by the Tomek Links method for undersampling (Tomek, 1976). This way the number of observations belonging to each of the classes was equalized.
- Tree-like algorithms were preferred due to their simplicity (explainability) and capability to handle multiconlinearity. Additionally, seeking for robustness in the development of the trees, the tree depth was set to 5 to avoid overfitting and make the tree use only those signals that were significant. The minimum number of observations per leaf was set at 450. This decision was made to force the trees to contain more than one complete experiment

on the final leaf, since the shortest experiment contains 360 observations. These values were established in order to generalise the solution as much as possible. However, other values were tested for these parameters, obtaining similar results. *Gini* impurity function was used for the measure of the quality of a split.

As some faults required of less effort than others to be diagnosed, several diagnosis models were stacked. Each time some faults were identified as similar and difficult to distinguish from each other, another model was created for those specific faults trying additional and more complex approaches.

2.3.6. Propagation of probabilities

The diagnosis models explained in the previous section provided a observation-wise or instant-wise class probability array as shown in Figure 1 Left). In order to improve the overall accuracy of the model, a Kalman filter like algorithm was used (Kalman, 1960). Starting from a scenario of equally probable class state, the algorithm kept updating the states (probability of having a certain class fault) with each new observation (a vector of probabilities provided by the diagnosis model). This way, the algorithm filtered the intermittence of the diagnosis layer by providing clearer trends as in the example of Figure 1 Right).

2.3.7. Feature engineering

As detecting some classes was non trivial from the raw data, feature engineering was used to create more meaningful features that could help to disambiguate. This feature engineering consisted on the use of certain thresholds to detect the amount of data surpassing this value from the total number of measurements at that time, that is, creating a ratio. This ratio variables were done in an on-line trend without violating any time series constraints. The meaningful features detected were *VacuumValveClosedvStd* for class 5 and *DurationPickToPickvStd* for class 7. If V is the ordered set containing chronologically ordered observations of *VacuumValveClosedvStd*, and D the ordered set containing chronologically ordered observations of *DurationPickToPickvStd*, the new variables *ratioV* and *ratioD* are defined for i -th observation as follow:

$$\begin{aligned} ratioV_i &= \frac{|N_{V,i}|}{i}, \\ ratioD_i &= \frac{|M_{D,i}|}{i}, \end{aligned} \quad (1)$$

where the sets $N_{V,i}$ and $M_{D,i}$ are defined by,

$$\begin{aligned} N_{V,i} &= \{v \in \{v_k\}_{k=1}^i \subseteq V \mid v > 0.2\}, \\ M_{D,i} &= \{d \in \{d_k\}_{k=1}^i \subseteq D \mid 0.26 < d < 0.45\}. \end{aligned} \quad (2)$$

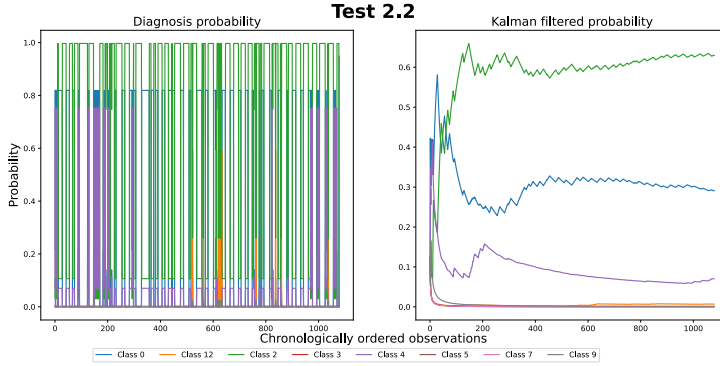


Figure 1. Diagnosis and propagated diagnosis of test 2 of class 2. Left) Diagnosis probabilities as provided by the diagnosis model. Right) Kalman filtered diagnosis probabilities.

Table 1. Glimpse of feature creation on test 5.0

Observation	V	ratioV
⋮	⋮	⋮
2	0.0498	0/2 = 0
3	0.2519	1/3 = 0.3333
4	0.0946	1/4 = 0.25
5	0.0944	1/5 = 0.2
6	0.2045	2/6 = 0.3333

Table 2. Glimpse of ratio feature creation for test 7.0

Observation	D	ratioD
⋮	⋮	⋮
11	0.2135	0/11 = 0
12	0.4209	1/12 = 0.0833
13	0.0169	1/13 = 0.0769
14	0.4373	2/14 = 0.1429
15	0.0290	2/15 = 0.1333

Examples of the creation of these new features for tests 5.0 and 7.0 are shown in Table 1 and Table 2, respectively.

2.3.8. Shortest answer time

There are several aspects to consider in order to determine the shortest required time:

Firstly, as some imputation was carried out, it was necessary to consider that within the elapsed time, at least a single non-empty observation of the employed variables should exist. This instant was named T_{NA} .

Secondly, the minimum time necessary to classify the experiment in the two system parameter classifications (SPC1 and SPC2) needed to be computed, which was named as T_O .

Finally, for the diagnosis, as mentioned in Section 2.1, the evaluation method could not avoid the algorithm having a full picture of the experiment on the first run. However, there was no direct way to transfer information regarding the run to the algorithm. Consequently, it was decided to assume that, regardless of the run, the most probable real class was identified at the end of the experiment. By doing so, the cutting point/instant became the first point of time where the most probable class was the same as the most probable class at the end of the input test. This assumption ensured consistent results and allowed the algorithm to benefit from the full picture it had on the first run, this time was called T_D . Note that, as many diagnostic models are present in the algorithm, T_D represents the longest time required by the diagnostic algorithms of the current test. Therefore, $T_D = \max(T_{D'}, T_{D''})$ and $T_{D''} = 0$ in case the class was different to 0, 5 and 7.

In summary, the minimum number of time windows required by the algorithm to classify the input experiment, was calculated as $T_c = \max\{T_{NA}, T_O, T_D\}$.

3. RESULTS

The final algorithm consisted of a combination of sub-models which were assembled as shown in the diagram in Figure 2.

The divide, propagate, and conquer strategy allowed to split the initial big problem (classification of 9 classes) into smaller classification problems, in which robustness of the final algorithm was always sought. One of the major worries during the development of the algorithm was incurring in overfitting, as the classes were really imbalanced and not considering that could lead to bad results on the testing. At the same time, due to the small amount of the faulty class tests, it was not possible to totally exclude some data to validate later our methods. Efforts were made to withdraw strong rules that fitted well the data and did not provide false positives nor overfit the

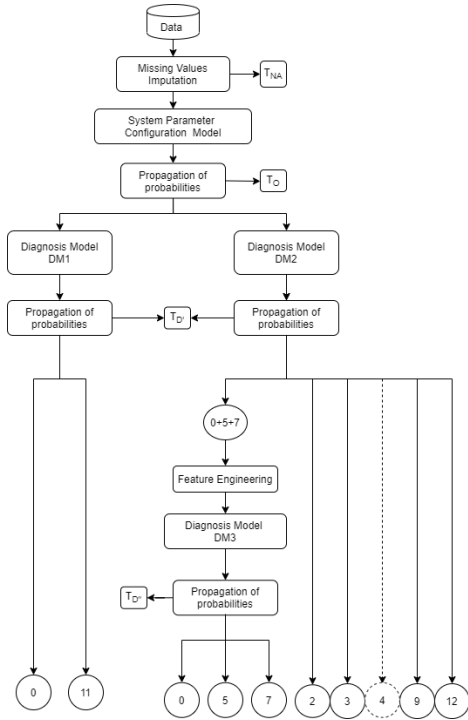


Figure 2. Execution Flow of the Classification Algorithm.

data and provide overoptimistic results.

Initially, regarding the identification of operating conditions in healthy tests, a single rule was required for their correct identification as Figure 3 shows. This rule was capable of detecting the different operating conditions without ambiguity, except for a couple of outliers that appeared on the beginning of some tests.

Due to the peculiarities of the dataset, more than a single diagnosis models were required to perform a full identification of the faults. The first decision tree (DM1) separated the healthy experiments that were done with the first parameter configuration and class 11 failures. The remaining healthy cases and the 8 faulty classes were disambiguated by another decision tree (DM2) since those experiments were carried out with the second parameter configuration. In addition, due to the difficulties to discern between classes 0, 5 and 7, an additional tree (DM3), with extra features (ratio variables for VacuumValveClosed and DurationPickToPick) was needed. Using this new approach, a diagnostic model was achieved that correctly classified the two faults 5 and 7, obtaining an average Recall in the training step of 0.9825 for class 5 and 0.9806 for class 7. In the validation step of the DM3 model,



Figure 3. Boxplots of healthy tests, red line represents the operation configuration decision rule.

a mean value for Recall was obtained in the corresponding iterations to leave out each of the experiments of 0.9625 for class 5 and 0.94875 for class 7. Those results were considered successful and the DM3 diagnosis model was added to the final algorithm.

Finally, regarding fault class 4, differences between the three available tests of this type were observed. Some class 4 tests had the same behaviour as some healthy tests whereas the remaining one had a similar behaviour as other healthy classes, as shown in Figure 4. Hence, depending on the tests which were selected for training and testing, the models were very different without giving a chance to obtain robust solution for the diagnosis of this class. Thus, although the mean value of Recall metric in the training phase was 0.714, the diagnosis model was not able to correctly classify the testing experiments and the Recall value in the validation was 0 in all iterations. Furthermore, the most important signals were different in the three tests. For these reasons, it was assumed that the testing set would have a similar distribution of the cardinality as the one on the training, and the final algorithm labelled 4 class predictions as healthy. Additionally, to compare the results obtained for the diagnosis of faulty classes 5 and 7 with the results obtained with faulty class 4, LeaveOneGroupOut cross validation is used once again. Figure 5 shows the confusion matrix of one of these iterations.

All in all, the final algorithm only required 10 features for the diagnosis and propagation, which are shown in Table 3 .

Regarding the Root-cause-analysis, the signals that were identified as most important in relation to each fault class are shown in Table 4. Finally, the propagation thought the Kalman algorithm provided very interesting results. In the tests where the class value was not clearly predominant according to the diagnosis algorithm, the filter allowed to visualise a clearer trend, as the Figure 1 shows, which improved the final diagnosis.

The solution dealt properly with the identification of the tests

Table 3. Set of features used by the final algorithm.

Features	
SmartMotorPositionErrorvMin	VacuumValveClosedvStd
DurationPickToPickvStd	DurationRobotFromFeederToTestBenchvalue
SharpnessImagevalue	NumberFuseDetectedvMin
TotalCpuLoadNormalizedvStd	SmartMotorSpeedvStd
Temperaturevalue	DurationRobotFromTestBenchToFeedervalue

Table 4. Feature Importance for each class

Class	Features
Class 2	FeederAction2
	Humidity
	NumberFuseDetected
Class 3	SharpnessImage
Class 5	VacuumValveClosed
Class 7	DurationPickToPick
Class 9	SmartMotorSpeed
Class 11	SmartMotorPositionError
	DurationRobotFromFeederToTestBench
	DurationRobotFromTestBenchToFeeder
	DurationTestBenchClosed
Class 12	DurationRobotFromFeederToTestBench
	DurationRobotFromTestBenchToFeeder

that were provided for the training, allowing a totally accurate diagnosis of faults except for class 4 which was predicted as 0 on purpose (assuming the testing set would follow a similar distribution on the number of tests per class). Additionally, the time required to determine the class was most of the times very short, since 2 observations were enough to detect the class in many of the tests. In average 18 observations were required for all the tests in the train and only in 5 cases more than 100 observations were necessary to classify the experiment correctly. Finally, the features related to each class (or root cause analysis) were visually validated.

In the context of the challenge, the solution has been proved to be robust. It has obtained the highest Final score, obtaining

the highest scores amongst the participants for accuracy and required time, a full score in clustering and a more modest score in root cause identification.

These results demonstrate that the algorithm has been able to generalise well, in sight of the accuracy score; and, that the policy developed for minimising the time required has also behaved correctly according to the timing score. In addition, the clustering rule has shown very robust results.

Regarding the root-cause-analysis, the main differences between our solution and the official solution are found in fault classes 7 and 11. For fault class 7, the important features according to the official solution are FusePicked and VacuumFusePicked, instead of DurationPickToPick, which is our choice. In a scenario with highly colinear features, it is probable that all the mentioned features could be valid. Furthermore, for faulty class 11, the official solution has only considered the features corresponding to the separation between the two operation configurations, but features that separate the faulty class 11 with its operation’s healthy class are also considered in this work.

4. FINAL REMARKS

The solution presented in this paper must be considered in the context of a challenge, with limited access to relevant information and, at the same time, a considerably short time to develop a proposal. This approach is based on the need to split a big problem into smaller and simpler parts that could be solved independently from the each other. This was achieved by: employing decision trees and similar structures that allowed to identify certain classes at a time and leaving less uncertainty to the next layer of the algorithm; a separate feature selection methodology that helped in the development of diagnosis algorithms; the propagation of the observation-wise diagnosis of the algorithms; and, a backwards identification of the required computation time.

Overall, the solution has yielded satisfactory results, obtaining the highest score in the competition and showing great generalisation capability by only failing to detect the class 4 tests, as expected by design. This has been achieved by stacking a rater simple algorithm: The decision tree, which has the added advantage of being understandable. Additional interesting findings are the appropriateness of feature engineering approaches, that have helped in the diagnosis of difficult

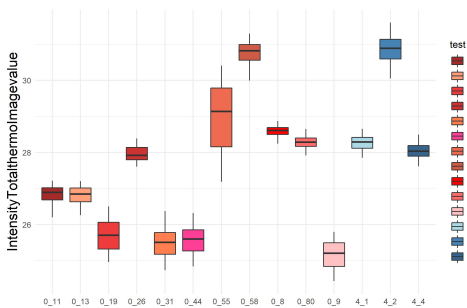
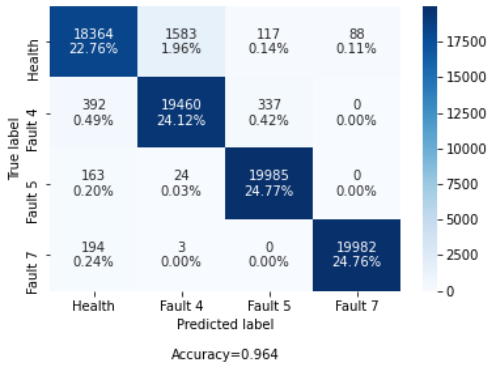
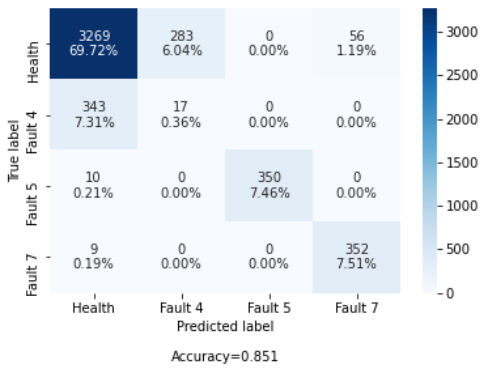


Figure 4. IntensityTotalThermoImagevalue boxplots per test for class 0 and 4.



(a) Train Confusion Matrix



(b) Test Confusion Matrix

Figure 5. Confusion Matrix of an iteration of LeaveOne-GroupOut Classes 0,4,5 & 7. Test used for validation: 10 random tests of class 0, and test 4.1, 5.4 and 7.5 for other classes.

classes; the propagation with Kalman filter, that allows the identification of trends in noisy tests; and the strategy used for the detection of required time, which helps the solution to make very precise adjustment of the required time.

This approach has, however, some limitations. Firstly, regarding the strategy used to detect the shortest time required for the classification, it needs to be mentioned that this approach fits well inside the challenge context, but it is not applicable in a real scenario, as there is no first and second run for each test. Secondly, it has not being possible to detect all the faults, which makes this solution incomplete.

Team HIRUTEK is aware of the limitations of this work, but, at the same time, some interesting techniques are presented and could be further studied. In that sense, we consider of great interest working on the identification of fault 4, which has been left aside in this work, but that we believe that could

be tackled by using frequency or other time/frequency techniques. Also, it might be interesting to study the missing values and their correct imputation, as they could be a source of interesting information even if they have been overlooked in this work. Finally, the use of Kalman filters to visualise trends of probabilities could be further extended and studied including more realistic strategies to identify at which point is the class probability stable enough to provide a label for the test.

ACKNOWLEDGMENT

This work is partly supported by the project 3KIA (KK-2020 / 00049), financed by the SPRI-Basque Government through the ELKARTEK program and AI-PROFICIENT, which has received funding from the European Union’s Horizon 2020 research and innovation program under grant agreement No 957391.

REFERENCES

Assad, F., Konstantinov, S., Nureldin, H., Waseem, M., Rushforth, E., Ahmad, B., & Harrison, R. (2021). Maintenance and digital health control in smart manufacturing based on condition monitoring. *Procedia CIRP*, 97, 142–147. doi: 10.1016/j.procir.2020.05.216

Barnes, S. A., Lindborg, S. R., & Seaman, J. W. (2006). Multiple imputation techniques in small sample clinical trials. *Statistics in Medicine*. doi: 10.1002/sim.2231

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*. doi: 10.1613/jair.953

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering, Transactions of the ASME*. doi: 10.1115/1.3662552

Kaufman, L., & Rousseeuw, P. J. (1986). CLUSTERING LARGE DATA SETS. In *Pattern recognition in practice*. doi: 10.1016/b978-0-444-87877-9.50039-x

Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17), 1-5. Retrieved from <http://jmlr.org/papers/v18/16-365>

Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., & Hornik, K. (2021). cluster: Cluster analysis basics and extensions [Computer software manual]. Retrieved from <https://CRAN.R-project.org/package=cluster>

Moritz, S., & Bartz-Beielstein, T. (2017). imputeTS: Time series missing value imputation in R. *R Journal*. doi: 10.32614/rj-2017-009

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V.,

- Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Stavropoulos, P., Chantzis, D., Doukas, C., Papacharalampopoulos, A., & Chryssoulouris, G. (2013). Monitoring and Control of Manufacturing Processes: A Review. *Procedia CIRP*, 8, 421–425. doi: 10.1016/j.procir.2013.06.127
- Ting, K. M. (2010). Precision and recall. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of machine learning* (pp. 781–781). Boston, MA: Springer US. Retrieved from https://doi.org/10.1007/978-0-387-30164-8_652 doi: 10.1007/978-0-387-30164-8_652
- Tomek, I. (1976). Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11), 769-772. doi: 10.1109/TSMC.1976.4309452
- Wickham, H., François, R., Henry, L., & Müller, K. (2019). *dplyr: A Grammar of Data Manipulation. R package version.*

BIOGRAPHIES



Kerman López de Calle - Etxabe obtained his bachelor in Renewable Energies Engineering from the University of the Basque Country (UPV/EHU) in 2016. In 2016-2017 he obtained his master's in Computational Engineering and Intelligent Systems from the same university, after he developed his master's thesis in collaboration with the Research Center Tekniker, where he carries out his research cur-

rently. From 2017 to 2020 he pursued a PhD in Information Engineering related to the development of condition monitoring algorithms. To date, he has published various works in high impact journals and has also contributed to diverse conferences in the field of condition monitoring and data analysis.



Meritxell Gómez - Omella got her BSc in Mathematics from the Universitat Autònoma de Barcelona (UAB) in 2018. In 2019 she obtained her MSc degree in Modeling and Mathematical Research, Statistical and Computing from the University of the Basque Country (UPV/EHU), with a master's thesis related to data quality and missing data imputation in collaboration with the Research Centre Tekniker. Since September 2019 she is pursuing a PhD in Time Series Analysis and Forecasting. Currently, she is a Junior Data Science of the Intelligent Systems Unit in Tekniker and she has been working in different national and European projects.



Eider Garate - Perez acquired her Bachelor in Mathematics from the University of the Basque Country (UPV/EHU) in 2020. Presently, she is studying a MSc degree in Modeling and Mathematical Research, Statistical and Computing from the same university, and is doing her master's thesis related to missing values imputation methods in time series in collaboration with the Research Center Tekniker.

PRENERGET: A FRAMEWORK FOR THE INCLUSION AND ADAPTATION STRATEGIES OF MACHINE LEARNING MODELS FOR ENERGY DEMAND FORECASTING IN BUILDINGS

Iker Esnaola-Gonzalez¹, Meritxell Gomez-Omella¹, Susana Ferreiro¹, Alvaro García¹, Francisco Javier Díez¹
¹TEKNIKER, Basque Research and Technology Alliance (BRTA), Spain

Abstract

Machine Learning (ML) models are key enablers for the implementation of different energy-efficiency strategies in buildings. There are a variety of frameworks that facilitate the development of ML models, but it is necessary to move into a different environment for their deployment and exploitation. Furthermore, their performance tends to degrade over time. Consequently, they need to be regularly evaluated and upgraded to ensure the robustness of the overall solution. The seamless exploitation, adaptation and evolution of ML models is still an open issue nowadays, and in this article, a software framework called PRENERGET aimed at addressing this issue is presented. The main contributions of PRENERGET are, on the one hand, the facilitation of the exploitation of ML models to make forecasts related to energy efficiency, and on the other, the maintenance and, if possible, the improvement of the forecasting performance over time.

Introduction

Nowadays, the building sector's energy consumption accounts for almost a third of the total energy consumption and its share of emissions has risen to almost 30% (International Energy Agency 2021). As a matter of fact, the energy consumed in the building sector is responsible for nearly 3 Gt of direct CO₂ emissions. Space heating, cooking and other daily activities account for the majority of global CO₂ emissions today in the buildings sector (International Energy Agency 2021), and the demand side management (DSM) and demand response (DR) programs have emerged in an effort to minimise these figures (Warren 2014, Albadi & El-Saadany 2007). However, the implementation of DR programs is not straightforward (Esnaola-Gonzalez et al. 2021), and being able to accurately forecast the energy demand plays a crucial role.

Artificial intelligence (AI) systems, and more precisely, Machine Learning (ML) based models have showcased their prominence in creating accurate predictions (Gómez-Omella et al. 2021, Tascikaraoglu & Sanandaji 2016, Lussis et al. 2017). There are a variety of environments and frameworks that facilitate the development of ML models, but when it comes to their deployment and exploitation, it is necessary to move into a different environment. Certainly, the value offered by the models is more often than not limited by the use of inappropriate application logic. Furthermore, under normal conditions, ML-based forecasting models performance degrade over time due to a change in the environment that violates the models assumptions (Widmer & Kubat 1996). Consequently, the deployed models need to be regularly evaluated and up-

graded to ensure the robustness of the solution they are part of (Žliobaitė et al. 2016).

The seamless exploitation, adaptation and evolution of ML-based models is still an open issue nowadays, and in this article, a software framework called PRENERGET aimed at addressing this issue is presented. Namely, the main contributions of the developed framework are:

- To facilitate the exploitation of ML models to make forecasts related to energy efficiency.
- To maintain and, if possible, improve the forecasting performance over time.

The rest of the article is structured as follows. First, a summary of previous works found in the literature is presented in the Related Work Section. The development and deployment of PRENERGET is detailed in The Framework Section. Later, in the PRENERGET On the Loop Section, PRENERGET is validated in a real use case and the results obtained are shown. Finally, the conclusions obtained are summarised and future work is presented.

Related work

The AI is currently experiencing an upsurge that can be attributed to advances in computing and the increasing availability of data, and it has been useful for solving problems of different nature, including forecasting problems. In the field of energy efficiency in buildings, being able to accurately forecast future situations is key to ensure optimal decision-making. For example, forecasting the energy to be consumed and the energy to be produced by renewable systems installed in a building can contribute to maximising their energetic efficiency by implementing load curtailment (i.e., a reduction of electricity usage) or reallocation (i.e., a shift of energy usage to other off-peak periods) (Esnaola-Gonzalez et al. 2021).

Regarding the forecasting of energy coming from renewable sources, different approaches and mechanisms can be found in the literature, including ML algorithms. As a matter of fact, they have been proven to perform well when forecasting energy to be produced by photovoltaic panels (Ahmed et al. 2020, VanDeventer et al. 2019), solar thermal collectors (Unterberger et al. 2021) or even wind farms (Juban et al. 2007). Likewise, for the forecasting of buildings' energy consumption, different ML algorithms have been demonstrated to be valid. In (Zhang et al. 2018), a support vector regression modelling approach has been used for forecasting households electricity consumption, both to daily and hourly data granularity. But other algorithms such as regression trees and neural networks have

also been used in day-ahead load forecasting for residential customers problems (Luis et al. 2017).

There are numerous software applications, tools and IDEs (integrated development environment) such as R Studio, Weka or Rapidminer that offer many possibilities for creating and training ML models. However, they do not offer functionalities that go beyond their own environments, and in order to deploy the developed models, the integration with other software is necessary. This concept has been extended to the cloud, which offers many advantages when it comes to the deployment of applications, both in terms of ease and scalability. In this regard, products such as Google Vertex AI and Amazon SageMaker offer sets of tools that enable deploying models in their own cloud infrastructure. The problem arises when users want to deploy these models on their own infrastructure or integrate them with applications that use their infrastructure.

In addition, the rapidly changing environment where we live in leads to the degradation of forecasting models' performance. This fact can be evident when abrupt changes occur such as during the COVID-19 pandemic-related confinement, curfew and mobility restriction measures (Gomez-Omella et al. 2020), but there are also other subtle changes that may equally affect the performance. This is known as the concept drift problem, which means that the statistical properties of the target variable the model is trying to forecast, change over time in unforeseen ways (Lu et al. 2018). In the face of such changes, forecasting models need to be adapted (Gama et al. 2014). Understanding the effect of the drift on the ML performance and defining the most appropriate adaptation strategies to make them more robust is one of the first steps to be considered. As a matter of fact, depending on the type of change, different adaptation mechanisms may be implemented. The work presented by (Celik & Vanschoren 2021) proposes different adaptation strategies that start from an initial model trained at least once with an initial batch of data. The strategy to follow will not always be the same and it will depend on many factors (e.g., the nature of the data, the application domain, unexpected events, etc.). This strategy will be determined by the data analysis task who is in charge of understanding the behaviour of the derivative of the model, detecting it, and even anticipating it.

The monitoring of the forecasting models' performance on the one hand, and the implementation mechanisms of the model adaptation on the other, are repetitive tasks that have a cost in terms of personnel dedication that, in many cases, may not offer improvements and therefore added value. Automating the models' performance evaluation and adaptation as well as the deployment of the adapted models alleviates workers from this tedious task and achieves better results by potential possible manual errors. Furthermore, this automation will also facilitate the selection of strategies to be followed when adapting the models.

Google also offers the open-source product TensorFlow which allows integration into on premise infrastructure.

This suite includes the Pusher model deployment tool as well as other tools for model evaluation and validation such as Evaluator and InfraValidator. In this case, the developer is limited to use these libraries to generate the models and will not be able to use other languages such as R. On the other hand, the automation of the updates will have to be implemented in programming code using the previous tools.

Clipper (D. Crankshaw 2017) has been developed at a higher level of abstraction so as not to depend on certain frameworks to build the model. This framework is modular and allows invoking models developed in Apache Spark, Scikit-Learn, Caffe or TensorFlow. Clipper facilitates the integration of models through a unified REST interface but lacks model updating capabilities. Data and Learning Hub for science (DLHub) (Zhuozhao Li 2021) is another framework for the development of ML models. One of its pillars is the use of a standard model invocation via the previously developed "funcX" (R. Chard 2019, 2020), function and other one the use of Docker based containers as the current work. It is an excellent environment with good model characterisation capabilities and good performance, but it is oriented to research environments and not to production environments where resources have different restrictions. Muthusamy et al. (2018) shows a layer that encapsulates the ML models to provide a microservice interface that can be exploited in business applications. In this case, the model developer must be able to implement the defined interface. This interface offers functionalities to detect data drift and KPIs on accuracy and therefore assesses the need for retraining, but the update of the models is left out of the scope of the work.

Data scientists are experts in creating the ML models that solve the aforementioned prediction problems but the deployment of these models into production for their exploitation is not as straightforward as it might be thought. As a matter of fact, when it comes to exploiting the models, they are confronted with different execution environments than those used for analysis. These environments may work with technologies that are beyond the scope of knowledge of data analysts. And here is where programmer analysts come into play. They are specialists in creating software to be exploited in a given environment, but their knowledge of ML is often insufficient. Therefore, collaboration between the two types of profiles is not straightforward due to the different nature of the environments in which each works.

A framework to unify the interfaces between both of them can significantly improve the interaction necessary for the joint development of the software needed to not only exploit the forecasting models but to also ensure their performance over time. To the extent of knowledge of authors, existing approaches do not cover these requirements (Simmhan et al. 2013, Choi et al. 2016). Therefore, the definition and implementation of the necessary infrastructure, mechanisms, channels, interfaces, and workflows to facilitate the automation, deployment, and execution

of ML models under the same software architecture to streamline the process is a necessity.

The Framework

In this section, the design and the development of PRENERGET is presented. PRENERGET is a software framework for the deployment of ML models and its automatic adaptation to potential changes. The software framework offers a pipeline and a set of interfaces to incorporate data from different sources; the invocation and execution of multiple ML models; the mechanisms for monitoring and estimating the error generated by the ML models; and the required elements for the evaluation of the model and its future adaptability. PRENERGET provides a modular architecture with all the advantages that this involves and includes consistency in development, reduced development time, and flexibility.

Development

PRENERGET provides the data flows and workflows that maps out the flow of information, the definition of the pipelines for the interconnection of the different blocks or functionalities and the workflow engines that orchestrate the execution of tasks and exchanges of data between them. It is based on the interoperability offered by a set of REST API services, used to design, and integrate application software on different platforms, and a set of standard interfaces with an execution environment based on the R programming language that allows the development of advanced ML functionalities. This makes it easily integrated with other software either locally or remotely.

PRENERGET includes the following functionalities:

- **R Script execution:** The set of programs and functions implemented in R programming language. They are in charge of training, executing and evaluating the ML models. They also include the functions to implement the automatic adaptation strategy or model adjustment. These functions can change, vary or even be replaced by others at any time without affecting the rest of the infrastructure because PRENERGET provides a modular architecture.
- **Task Scheduling:** The set of Web services implemented in Java that encompass a set of tasks that can be executed both periodically and on demand. Most of these calls use a client to communicate with an R server, where the scripts are executed.
- **Data management:** The PRENERGET architecture uses two types of databases. The first type, called 'external databases', is used to acquire the data and to develop the initial ML model its future re-adaptation. Being a modular and flexible architecture, it can work for any type of database (e.g., relational, time series...). All it takes is to change the connector to the database and the architecture will continue maintaining the rest of the functionalities. The second type,

called 'internal database', is a database that includes the information to manage the tasks, and metadata about the models. This database allows the traceability monitoring and the correct understanding and interpretation of the models.

Figure 1 shows an example of the data flow between the R Scripts and the Web service that handles task scheduling and data management. In the flow, the components that are executed in the Web service are coloured blue and those that are carried out in the R scripts are coloured yellow. The components with grey bands represent the main scripts or classes while the solid components indicate actions that are carried out within them.

Deployment

The functionalities described in the previous section are provided by 3 software components. These have been encapsulated in Docker (Merkel 2014) containers to ease deployment, as shown in Figure 2.

- **R server:** Will contain a running instance of R engine, along with the scripts and models to be called. As R does not need compiling to be executed, scripts can be updated or added during run-time, without the need of stopping the software.
- **Apache Tomcat:** The task scheduling and data management functionalities will be provided by a Web service developed in Java, hosted in a Apache Tomcat server. By publishing the functions in a Web based API interface testing and integration with other software becomes easier.
- **Internal Database:** This component stores the data used by the Web service: models, variables, data sources and scheduled tasks. This data is for internal usage only; data used to feed the model such as historical knowledge is stored in databases external. They are used by the software but are not considered part of the software solution.

The specific actions of the scheduled tasks change between different types but they follow a similar approach. After retrieving information of the involved variables from the database, the task will prepare the call to the script, retrieving data from the associated data sources and setting up the parameters. The R engine will be called to execute a specific script with the given parameters. After the execution has finished, it will read the results and use them, storing them, notifying other software or using them as parameters of other scripts.

PRENERGET on the loop

For the automation of PRENERGET, the period of execution of the previously explained flow (Figure 1) is decided. In each of these executions, a different training data set will be chosen from an external database to develop a model.

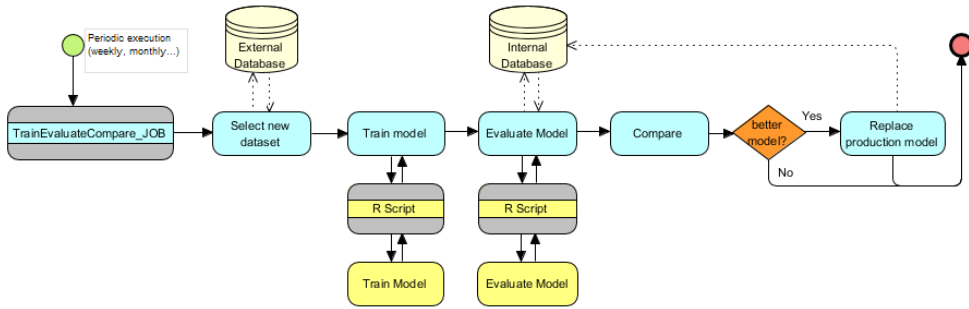


Figure 1: Data flow for model adaptation and updated

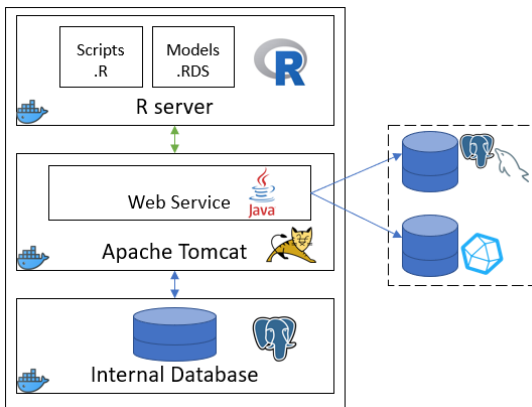


Figure 2: Software components architecture (with data sources)

Since the data is stored as time series over time, it is decided to use moving windows to update the values of the training phase. In this way, in each model fit, the oldest historical data is eliminated and the most recent is added to the training set. Afterwards, the training phase is started by calling an R script in charge of adjusting the model parameters using different techniques that allow generalisation, such as the commonly used k-fold cross validation. In the next phase, the evaluation of the model is done by calling another R script that obtained the errors or deviations of the training phase. These errors are calculated comparing the forecast values with the actual values registered available from the historical data. The update of the model in production will be based on the improvement of these error values that can be calculated using different metrics such as the Root Mean Squared Error (RMSE) or the Mean Absolute Error (MAE) among others. That is, in the last phase, the errors from the new and the old models are compared and if the new model improves the performance of the deployed model, it will be used to replace such a deployed model.

Being modular and configurable, PRENERGET allows

programming different strategies, both for the testing and evaluation of the models, and for their update and adaptation to changes, in an agile and simple way.

Use cases

The feasibility of the PRENERGET framework has been tested in two real-world energy efficiency scenarios. On the one hand, in a neighbourhood in Madrid (Spain), and on the other, in a research centre in Eibar (Spain). For the sake of simplicity, this section will focus only on the latter. However, it is worth mentioning that the PRENERGET framework is designed to be applicable to other energy-related use cases.

The main objective of the forecast task was to provide accurate electric demand forecasts for the next day, so that adequate energy-saving strategies could be implemented in advance. Such forecast task has been performed using ML strategies based on the historical records containing hourly electric consumption data measured in kWh. So, before running the data flow of Figure 1, the forecasting algorithm, input variables, training method, and error metrics to be used should be set.

A previous comparative study of the results obtained from different ML algorithms concluded that the K-Nearest Neighbours (KNN) was the most efficient algorithm to provide further values in the problem at hand. The KNN was compared with an ARIMA model, a Linear Regression and a Support Vector Regression, obtaining better results both in forecast errors and in computational time. As a matter of fact, this method gives an accurate forecast due to the seasonality and the repetitions in the daily electric demand (Gómez-Omella et al. 2020).

After analysing different date and time related variables, the hour, the day, the month, the season of the year, the day of the week and a binary variable indicating whether the day is a working day or not, were the selected variables as they provided the highest correlations with the output variable. These support variables are needed to be used as inputs for the KNN algorithm, as the available data was a univariate time series containing the electricity consumed and a time index. As the KNN is an algorithm

that calculates distances between instances in order to decide the most similar neighbours, trigonometric transformations were made in cyclical variables. In other words, the hours and the month are modified to the sine and the cosine of their values to force their values to be equidistant, as explained in detail in Gomez-Omella et al. (2020).

Then, a 5-fold cross-validation technique was set to decide the optimal number of 'k' neighbours and the model was then fitted using a subset containing the 70% of the entire data available. This is a classic ML model training strategy that reserves 30% of the data to validate the decisions made in training task.

Finally, regarding the evaluation of the model accuracy, the RMSE was the metric chosen to compare the performance of the models. This value quantifies the mean error made in a forecast by averaging the squared errors made in all the estimated future points once their real values are known. The RMSE unit of measure is the same as the original data, making it intuitive to interpret.

Once the characteristics of the process have been configured for the specific problem, a first version of the KNN model was developed and deployed, and an automated task was programmed to execute the data flow from Figure 1 every day at 00:00h, that is, once every 24 hours.

In each iteration, the first phase consists in retrieving set of data to train the model. This set of data consists of historical energy consumption registries of the last 365 days stored in an external database. It is worth mentioning that, in this phase, a 24-hours rolling window has been set, so that in each iteration, the previous iteration's data set's first day is removed, and the last day is added to conform the new data set to train the model. The second phase consists in training the ML model. To do so, an R script is called which contains the functions to carry out the cross validation process and choose the optimal number of neighbours k to be used. Once the model is trained, in the next phase, it is evaluated by obtaining its RMSE. By calling an R script, the RMSE of the training phase is obtained and the score is stored in an internal database so that it can be retrieved at any given time. Furthermore, in this phase, the currently deployed model's RMSE is obtained in order to, in the next phase, compare it with the newest model's RMSE. As it can be seen in Figure 1, the model version update occurs in case the new model's RMSE is lower than the deployed model's RMSE. Every time a new model is created in each iteration, it is tagged with a different version code.

It starts with version 0.0 and then, version 0.1 is created. In case that version 0.1 was better than 0.0 and the current model needs to be replaced, version 0.1 will be automatically renamed to 1.0. Therefore, the model version coding convention is as follows. The models created in each iteration are labelled as minor versions x.1, x.2, x.3,... and the deployed ones as major versions 1.0, 2.0, 3.0,... If a given created model performs better than the deployed one, it is renamed as a major version. This approach allows to control the evolution of the model performance and to alert in

case the values deviate from the acceptable boundaries.

Results

The process was initialised developing a model with a training set containing hourly electric consumption from 2020-03-30 to 2021-03-29, and an RMSE of 22.16 kWh. This model corresponds to the first version being deployed, so it is labelled as version 0.0. Once deployed, the first 24 hourly values forecast were corresponding to the next day, that is, 2021-03-30.

In the next iteration, the model 0.1 was trained with data from 2020-03-31 to 2021-03-30. Then, the RMSE obtained from the training of the deployed model 0.0 and the training of the new model 0.1 was compared. However, the new model 0.1's RMSE was higher, so the model 0.0 was not replaced and it continued to be active.

In the next iteration, a new model 0.2 was trained using data from 2020-04-01 to 2021-03-31 and the RMSE obtained was 21.58 kWh, less than that provided by the then-deployed model 0.0. For that reason, the model 0.2 is renamed as model 1.0 and it replaced the deployed model 0.0.

These iterations are repeated every 24 hours and the information of the models developed and deployed in the aforementioned scenario are summarised in Table 1, where the different updates of the versions of the model can be seen. Notice that the models that which are not deployed, are not included in that table.

In the first update, the RMSE of the model decreased from 22.16 kWh to 21.58 kWh in two iterations, that is forecast performance improved on average 0.58 kWh per day. Then, ten days later, the RMSE improve 0.30 kWh and that difference in error decreases in subsequent iterations. The rate improvement of the estimations depends on the data and the execution time, although the error is expected to reach a stable state. The process is still running and the model is continually being updated in order to provide estimates of further electric demand as accurate as possible.

It is expected that the better the model fits in forecasting historical data, the more accurate future forecasts will be. This can be different in case an unexpected sharp change in data statistical properties changed. That can be seen in Figure 3, where forecast values obtained with the different model versions are shown (to facilitate the visualisation, it was decided not to show the forecasts of version 0.0). As it can be observed, the estimated values and the actual values are increasingly similar as the version of the model is updated.

Conclusions

AI systems, and more precisely, ML-based models are key enablers for the implementation of different energy-efficiency strategies in buildings. However, their seamless exploitation, adaptation and evolution when they are deployed into production is still an open issue nowadays. In this article, a software framework called PRENERGET

Table 1: Snippet of the version update report of the deployed models

Version	Initial Training Date	Final Training Date	RMSE (kWh)	First Forecast Date
0.0	2020-03-30	2021-03-29	22.16	2021-03-30
1.0	2020-04-01	2021-03-31	21.58	2021-04-01
2.0	2020-04-11	2021-04-10	21.28	2021-04-11
3.0	2020-04-12	2021-04-11	21.25	2021-04-12
4.0	2020-04-14	2021-04-13	20.93	2021-04-14

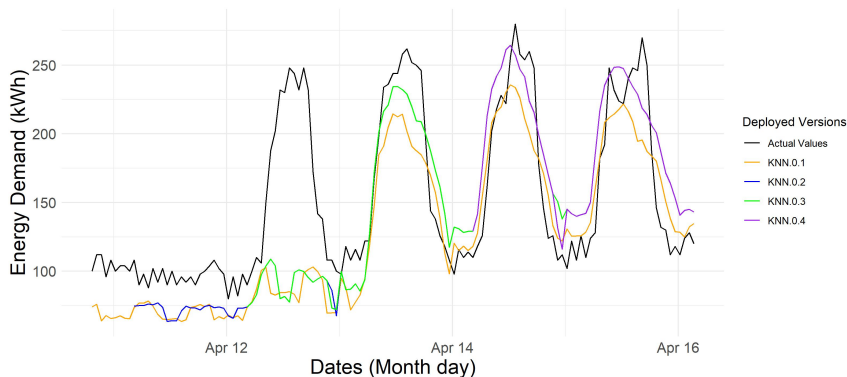


Figure 3: Evolution of the electric demand forecasts from the different versions of the deployed models

has been presented, aimed at facilitating the exploitation of ML models, and maintaining and if possible, improving their performance over time.

PRENERGET’s modular architecture facilitates the deployment of forecasting models. Analysts can concentrate on developing the models in their environment based on R, or another programming language. People closer to systems management can implement them in a simple way as they only have to link the databases used and program when the invocations to the models will be made to obtain the desired predictions. In this way, it will be very easy to have several forecasters of different variables such as energy, temperature or occupancy applied to different facilities such as rooms, machines or entire buildings.

ML models are unable to keep pace in today’s fast-changing world and their performance tends to degrade with time. Dealing with this issue and ensuring that deployed ML models provide operational results requires from an intensive effort of data scientists and ML experts. Even worse, in environments where many models are deployed, this can end up being an insurmountable barrier that hinders the successful deployment of an energy efficiency system. PRENERGET reduces costs, time and errors derived from human intervention in ML model performance maintenance and improvement tasks by automating it. Results show that in, a rather limited period, the performance of models can be improved up to 6%.

Future Work

After evaluating the work done, two possible points for future improvement are identified. On the one hand, in some cases, missing data were identified after the execution of

the forecasts. Furthermore, when the connection was re-established and the values were captured again, the first value registered was the accumulated value of all the missing values. Therefore, the values provided by the systems were not realistic due to the number of values that were not correctly received. A function that identifies this kind of failures and impute the missing values with the most suitable method before the execution of the forecaster is left to further research. It is expected that the results obtained after this modification provide more accurate results. On the other hand, the implementation of a degradation control system of the final model is to be implemented. As mentioned, the internal model error is expected to stabilise over time and updates to production models will become less frequent. In these cases, the evolution of the errors in the predictions could be evaluated to avoid the concept drift of the deployed model.

Acknowledgements

This work is partly supported by the project 3KIA (KK-2020/00049), funded by the SPRI-Basque Government through the ELKARTEK program and the REACT project which has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement no. 824395.

References

- Ahmed, R., Sreeram, V., Mishra, Y. & Arif, M. (2020), 'A review and evaluation of the state-of-the-art in pv solar power forecasting: Techniques and optimization', *Renewable and Sustainable Energy Reviews* **124**, 109792.
- Albadi, M. H. & El-Saadany, E. F. (2007), Demand response in electricity markets: An overview, in '2007 IEEE power engineering society general meeting', IEEE, pp. 1–5.
- Celik, B. & Vanschoren, J. (2021), 'Adaptation strategies for automated machine learning on evolving data', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **43**(9).
URL: <https://ieeexplore.ieee.org/document/9366792>
- Choi, T.-M., Chan, H. K. & Yue, X. (2016), 'Recent development in big data analytics for business operations and risk management', *IEEE transactions on cybernetics* **47**(1), 81–92.
- D. Crankshaw, X. Wang, G. Z. M. J. F. J. E. G. I. S. (2017), Clipper: A low-latency online prediction serving system, in '14th USENIX Symposium on Networked Systems Design and Implementation (NSDI)', pp. 613–627.
- Esnaola-Gonzalez, I., Jelić, M., Pujić, D., Díez, F. & Tomasevic, N. (2021), 'An AI-Powered System for Residential Demand Response', *Electronics* **10**.
- Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M. & Bouchachia, A. (2014), 'A survey on concept drift adaptation', *ACM computing surveys (CSUR)* **46**(4), 1–37.
- Gomez-Omella, M., Esnaola-Gonzalez, I. & Ferreira, S. (2020), Short-term forecasting methodology for energy demand in residential buildings and the impact of the covid-19 pandemic on forecasts, in 'Proceedings of 40th SGAI International Conference on Artificial Intelligence', Vol. 12498, pp. 227–240.
- Gómez-Omella, M., Esnaola-Gonzalez, I. & Ferreira, S. (2020), 'Short-term electric demand forecasting for the residential sector: Lessons learned from the respond h2020 project', *Proceedings* **65**(1).
URL: <https://www.mdpi.com/2504-3900/65/1/24>
- Gómez-Omella, M., Esnaola-Gonzalez, I., Ferreira, S. & Sierra, B. (2021), 'k-nearest patterns for electrical demand forecasting in residential and small commercial buildings', *Energy and Buildings* **253**, 111396.
- International Energy Agency (2021), 'World energy outlook 2021'.
URL: <https://www.iea.org/reports/world-energy-outlook-2021>
- Juban, J., Siebert, N. & Kariniotakis, G. N. (2007), Probabilistic short-term wind power forecasting for the optimal management of wind generation, in '2007 IEEE Lausanne Power Tech', IEEE, pp. 683–688.
- Lu, J., Liu, A., Dong, F., Gu, F., Gama, J. & Zhang, G. (2018), 'Learning under concept drift: A review', *IEEE Transactions on Knowledge and Data Engineering* **31**(12), 2346–2363.
- Lusis, P., Khalilpour, K. R., Andrew, L. & Liebman, A. (2017), 'Short-term residential load forecasting: Impact of calendar effects and forecast granularity', *Applied Energy* **205**, 654–669.
- Merkel, D. (2014), 'Docker: lightweight linux containers for consistent development and deployment', *Linux journal* **2014**(239), 2.
- Muthusamy, V., Slominski, A. & Isahagian, V. (2018), Towards enterprise-ready ai deployments minimizing the risk of consuming ai models in business applications, pp. 108–109.
- R. Chard, T. J. Skluzacek, Z. L. Y. B. A. W. B. B. S. T. I. F. K. C. (2019), 'High performance function as a service for science', *Serverless supercomputing*.
- R. Chard, Y. Babuji, Z. L. T. S. A. W. B. B. I. F. K. C. (2020), funcx: A federated function serving fabric for science, in 'Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing'.
- Simhan, Y., Aman, S., Kumbhare, A., Liu, R., Stevens, S., Zhou, Q. & Prasanna, V. (2013), 'Cloud-based software platform for big data analytics in smart grids', *Computing in Science & Engineering* **15**(4), 38–47.
- Tascikaraoglu, A. & Sanandaji, B. M. (2016), 'Short-term residential electric load forecasting: A compressive spatio-temporal approach', *Energy and Buildings* **111**, 380–392.
- Unterberger, V., Lichtenegger, K., Kaisermayer, V., Gölles, M. & Horn, M. (2021), 'An adaptive short-term forecasting method for the energy yield of flat-plate solar collector systems', *Applied Energy* **293**, 116891.
- VanDeventer, W., Jamei, E., Thirunavukkarasu, G. S., Seyedmahmoudian, M., Soon, T. K., Horan, B., Mekhilef, S. & Stojcevski, A. (2019), 'Short-term pv power forecasting using hybrid gasvm technique', *Renewable energy* **140**, 367–379.
- Warren, P. (2014), 'A review of demand-side management policy in the uk', *Renewable and Sustainable Energy Reviews* **29**, 941–951.
- Widmer, G. & Kubat, M. (1996), 'Learning in the presence of concept drift and hidden contexts', *Machine learning* **23**(1), 69–101.

Zhang, X. M., Grolinger, K., Capretz, M. A. M. & Seewald, L. (2018), Forecasting residential energy consumption: Single household perspective, in '2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)', pp. 110–117.

Zhuozhao Li, Ryan Chard, L. W. K. C. T. J. S. Y. B. A. W. S. T. B. B. M. J. F. I. F. (2021), 'Dlhub: Simplifying publication, discovery, and use of machine learning models in science', *Journal of Parallel and Distributed Computing* **147**, 64–76.

Žliobaitė, I., Pechenizkiy, M. & Gama, J. (2016), 'An overview of concept drift applications', *Big data analysis: new algorithms for a new society* pp. 91–114.

13th International Conference on Energy Efficiency and Sustainability in Architecture and Urbanism (EESAP 13)

Universidad del País Vasco/Euskal Herriko Unibertsitatea
Donostia-San Sebastián, Spain
5-6 october 2022
www.eesap.eu

PRENERGET, framework for energy forecasting PRENERGET, entorno para la predicción energética

Meritxell Gomez-Omella¹, Susana Ferreiro¹, Alvaro Garcia¹, **Francisco Javier Díez**¹

¹TEKNIKER, Basque Research and Technology Alliance (BRTA), Iñaki Goenaga 5, Eibar, 20600, Gipuzkoa, Spain
+34 673935792 franciso.diez@tekniker.es

Key Words: (energy, forecast, machine learning, artificial intelligence, framework)

Abstract

For the optimisation of energy management, it is crucial to be able to make decisions in advance. For this decision making it is necessary to have reliable predictions. In a building, there can be different types of predictions related to energy management; demand, production, temperature, price, occupancy, etc. Machine learning algorithms are a good technology to make these predictions, but they must be adapted to each variable and context, so many algorithms are needed running in parallel. In this paper a framework that allows to facilitate the execution of these predictions is presented. The boundary conditions on which these algorithms are based change over time and the predictions become less reliable. The presented framework allows to adapt to these changes in order to maintain the reliability of the predictions.

Introduction

PRENERGET is a framework for programming and executing machine learning algorithms to make predictions related to the energy efficiency of buildings. The recent energy crisis shows the need to increase renewable energy production. This implies a greater importance of active demand management in order to synchronise demand with production. Algorithms to perform this optimisation need to make predictions of different variables such as energy demand in different systems [1]. Machine learning (ML) based models have proven their importance in making accurate predictions [2][3][4]. Although there are a variety of environments that facilitate the development of ML models, when it comes to their deployment and exploitation it is necessary to move to a

different approach. Data analysts are used to working in modelling environments whereas deployment environments are used by systems analysts and programmers.

The value provided by models is hindered in many cases by inadequate application logic. Additionally, under normal conditions the accuracy of ML-based forecasting models [5]. degrades over time due to changes in the environment that modifies the original assumptions of the models. Consequently, deployed models need to be regularly evaluated and updated to ensure the robustness of the solution they are part of [6].

The seamless exploitation, adaptation and evolution of ML-based models remains an open question today. In particular, the main contributions of the developed framework are:

- Facilitate the exploitation of ML models for energy efficiency related predictions.
- Maintain and, if possible, improve the accuracy of the predictions over time.

The rest of the article is structured as follows. Firstly, the motivation for the development of the environment is presented, followed by a summary of previous work found in the literature in the related work section. The theoretical and practical approach to the solution is shown in the strategy section for the control and continuous improvement of ML models. This is followed by the use cases that have been used to validate the environment. The final part shows the results obtained and the conclusions and ends by outlining the lines for future work.

Motivation

Today, the energy consumption of the building sector accounts for almost one third of total energy consumption and its share of emissions has risen to almost 30% [7]. In fact, energy consumed in the building sector is responsible for almost 3 Gt of direct CO₂ emissions. Space heating, cooking and other everyday activities account for the largest share of overall CO₂ emissions in the buildings sector, and demand side management (DSM) and demand response (DR) programmes have emerged in an effort to minimise these figures [8].

One of the ways to improve the energy efficiency of a building is to automate the operation to achieve energy management optimisation based on active demand side management. This optimisation is not simple because it depends on many factors such as the characteristics of the equipment, functionalities of the systems, activity, meteorology, building envelope, etc. In addition, in many cases there is an inertia that means that the actions taken do not have an immediate effect. For all these reasons, it is necessary to make predictions that allow us to take decisions in advance so that the effect of the actions carried out is applied at the right time, neither before nor after.

As we have said, there are many factors that influence energy management, some of which are static and others dynamic. The a priori dynamic factors are all predictable. This means that there are many variables that can be predicted. Some of these variables are energy consumed, energy produced, power, temperature, humidity, occupancy or luminosity. In addition, many of these variables must be aggregated or disaggregated by space, system and equipment. Consequently, a good optimisation in a large infrastructure requires many predictors running in parallel at time intervals that can vary from 15 minutes to a day.

We are then faced with the need to be able to run several different predictors in a stable and scheduled manner so that the optimisation of energy management can be automated. At this point, integration with other systems must be taken into account. In some cases, a classical integration via synchronised read and write to a database may be sufficient. For this it is necessary to synchronise the execution of the predictor in advance of its use, considering the time needed for the execution. In many cases, the prediction cannot be done much in advance either, as the results would be worse. The ideal synchronisation occurs when the optimisation algorithm is able to invoke the prediction algorithm when it needs the data. In this case it is not necessary to schedule the prediction, but it is necessary to define an interface that allows communication between the two modules. Service-oriented architectures (SOA) and web service interfaces have greatly facilitated the interoperability and integration of software systems.

PRENERGET aims to build a development environment that facilitates the generation, execution and maintenance of predictors of all these variables based on machine learning algorithms. For this, it is necessary to have easy access to the historical data of the monitored variables with different time horizons, as these periods will be very relevant in the results of the algorithms. It is also necessary to be able to make different schedules for the execution of the predictors given that depending on the variable and objective the intervals between predictions may be greater or lesser. Another aspect to consider is the integration of the results. PRENERGET stores the results in a normalised database in such a way that both the latest and previous predicted values are available to other software systems. In addition, PRENERGET incorporates a REST web service interface that allows the asynchronous invocation of any predictor in real time. The results of the execution, in addition to being recorded in the database, are returned in the call so that they can be used by the invoker directly.

After the generation and controlled execution of the predictors, they must be kept operational with good accuracy. It is normal for machine learning-based predictors to lose accuracy due to changes in the trend of any of the variables on which they are based. This has been particularly evident in the recent COVID 19 crisis in which there have been abrupt changes in normality that have significantly affected predictions made using pre-COVID data. Maintaining the accuracy of predictors generally involves periodic retraining and evaluation to determine whether the new predictor data is better than the old predictor data. This maintenance is tedious for the analyst and time-consuming and does not provide value. PRENERGET has automated this operation so that the prediction models are updated periodically without the need for manual retraining and evaluation.

Finally, it should be noted that another of PRENERGET's motivations is that it is a scalable system as well as improvable and updatable. To this end, in addition to modularity and interoperability, the advantages of each programming language/environment have been taken into account. Thus, a Java-based environment is used for programmatic tasks and an R-based environment for analytical tasks.

Related work

In recent years, AI has seen a strong advance that can be largely attributed to advances in modern computing and the increasing availability of data. The implementation of technology in buildings has led to the concept of

'Smart Building' where intelligent buildings collect, process and analyse data to efficiently manage energy resources and other supplies. Buildings can, for example, forecast the energy consumed and generated by the systems and based on this maximise energy efficiency through load restrictions or reallocation as explained in [1].

In the literature review, we can find numerous works aimed at forecasting energy from renewable or non-depletable sources that provide different approaches and mechanisms, among which the use of ML algorithms stands out. Several works such as those presented by [9], [10] and [11] show the validity of these algorithms for forecasting the energy produced in photovoltaic panels, solar thermal collectors and wind farms, among others. The approach and algorithms used differ from one author to another: for example, [12] shows a regression model using support vectors to forecast hourly and daily electricity consumption in households, while [4] uses regression trees and neural networks.

On the other hand, it is not only necessary to develop and integrate ML models but also to be able to perform functional monitoring of them to ensure that they are still functional and the results they provide are good. This is known as the concept drift problem, which means that the statistical properties of the target variable that the model is trying to predict change over time in unforeseen ways [13]. As explained in the work by [14], a changing environment can lead to degradation in the performance of forecasting models. And in the face of such changes, forecasting models must adapt [15]. It is necessary to understand the effect of drift on ML performance and to define the most appropriate adaptation strategies to make them more robust. Indeed, depending on the type of change, different adaptation mechanisms can be implemented. The work presented by [16] proposes different adaptation strategies starting from an initial model trained at least once with an initial batch of data. However, the strategy to follow will not always be the same and will depend on many factors (e.g. the nature of the data, the application domain, unforeseen events, etc.).

In this respect, there are different approaches found in the state of the art. Google for example offers the open-source product TensorFlow that allows integration into local infrastructure. This suite includes the model implementation tool Pusher, as well as other tools for model evaluation and validation, such as Evaluator and InfraValidator. In this case, the developer is limited to using these libraries to generate the models and will not be able to use other programming languages (e.g. Python, R...). On the other hand, the automation of updates must be implemented in the programming code using the tools mentioned above. The infrastructure presented by Clipper [17] has been developed at a higher level of abstraction in order not to rely on certain frameworks to build the model. This framework is modular and allows you to invoke models developed in Apache Spark, Scikit-Learn, Caffe or TensorFlow. It facilitates model integration through a unified REST interface but lacks model update capabilities. Data and Learning Hub for Science (DLHub) [18] is another framework for developing ML models. One of its pillars is the use of a standard model invocation through the "funcX" function [19] and the use of Docker-based containers. It is an excellent environment with good model characterisation capabilities and good performance, but it is oriented towards research environments rather than production environments where resources have different limitations. [20] shows a layer that encapsulates ML models to provide a microservice

interface that can be exploited in commercial applications. In this case, the model developer must be able to implement the defined interface. This interface provides functionalities to detect data drift and accuracy KPIs and therefore assesses the need for retraining, but updating the models is outside the scope of the work.

To the authors' knowledge, existing approaches do not cover these requirements [21] and [22]. Therefore, defining and implementing the necessary infrastructure, mechanisms, channels, interfaces and workflows to facilitate the automation, implementation and execution of ML models under the same software architecture to streamline the process is still a necessity today.

Strategy for monitoring and continuous improvement of ML models

PRENERGET helps to deploy ML models and to perform functional monitoring of ML models in a flexible, adaptable and reconfigurable way.

Figure 1 shows the flow of tasks configured from the selected strategy for functional monitoring of the models and its adaptation for the use case of energy consumption prediction in a building block (the use case is shown in the following section).

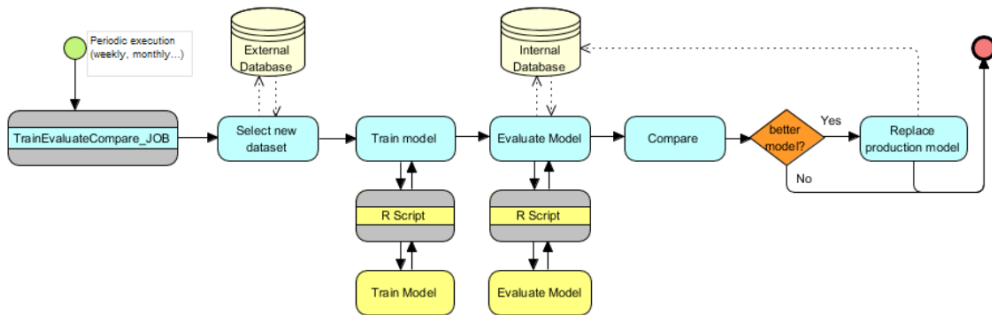


Figure 1 Data flow for model adaptation and updating.

We start from a time series database (TBS) in where the generated consumption data and the ML model are stored. The model has been initially trained with a 1-year dataset (hourly consumption) and has a starting accuracy adjustment (calculated in its validation phase). The strategy selected for drift detection and model adaptation is based on periodic scheduled monitoring of the error generated by the model during its life cycle and the use of moving time windows for updating the data used by the model in the retraining phase.

The 'Model Training' phase is executed daily, which trains the model with the most recent annual data obtained from the TBS from an R script. The algorithm searches and adjusts the hyperparameters of the model to maximise its accuracy. It returns as a result the error metric established and obtained by the new model.

Subsequently, the 'Model Evaluation' phase computes the error or drift obtained by the current running model using an R script, which currently compares the predicted values with the actual historical data values obtained and returns as a result the established error metric.

The drift detection of the model in production and its update by the new generated version is based on the improvement of these calculated error values by using error metrics that give us an idea of how good or bad our prediction model is (e.g. RMSE, MAE, etc.). It is in this last phase 'Comparison' where the errors of the new model and the previous one in production are compared, based on this whether the one in operation continues in use or is replaced.

As it is modular and configurable, PRENERGET allows different strategies to be programmed, both for the testing and evaluation of the models, as well as for updating and adapting them to changes, in a simple and agile way.

Use cases

The PRENERGET service has been tested in 4 scenarios in a technology research facility in Eibar (Spain). The different rooms chosen for the validation of the system have different energy consumption characteristics and the results obtained vary from one to another. The objective of the predictive task in the PRENERGET system is to provide hourly predictions of the power consumption expected for the next day using ML models trained with historical data. These data consist of univariate time series whose data are pairs of time records and records of electricity consumption measured in kWh.

In this case, through a previous comparative study, it was decided that the algorithm to be used was the k-nearest neighbours (KNN). The efficiency of KNN was found to be higher than that of ARIMA, linear regression and Support Vector Regression.

In order to choose the best explanatory variables in relation to the electricity consumption of the rooms, a set of variables was generated based on date and time. The contribution of each variable to the prediction was analysed and the following variables were chosen: time, day, month, season of the year, day of the week and a binary variable indicating whether the day is a working day or a public holiday, based on the local calendar. We observe that the numerical variables such as the time and the month have a cyclical behaviour. Since the KNN is based on the distances between instances, it is convenient to transform these variables to avoid erroneous conclusions. Therefore, the three variables were evaluated in the following function, where P is the period of each one (P=24 in the case of the hour, P = 12 in the case of the month) and obtaining two new variables corresponding to the sine and cosine of the initial values.

$$v : [0, P] \rightarrow [-1, 1]$$

$$X_t^i \rightarrow v(X_t^i) = \left(\sin \left(\frac{2\pi X_t^i}{P} \right), \cos \left(\frac{2\pi X_t^i}{P} \right) \right)$$

To pick the number of neighbours to be used, a cross-validation of 5 iterations on 70% of the data is used. This process is called training and is used to fit the model to the historical data. The process consists of dividing the

training dataset into 5 equal parts and using 4 of them to fit a model by testing various values of k. Each of them is used to make predictions for the model. With each of them, predictions are made on the reserved set. By calculating the prediction errors in each of the tests, the k that gives the smallest average error is chosen. This choice is different for each dataset and once k is chosen a prediction is made on the remaining 30% of the data to evaluate the effectiveness of the decision taken.

Both in the training process and in the final testing, the Root Mean Squared Error (RMSE) was the selected metric. This is a way of calculating the prediction error in the same units of measurement as the original data and is widely used as it is one of the most intuitive to interpret. Given a set of real values $\{y_t, t = 1, \dots, T\}$ and a set of values predicted by an ML model $\{\hat{y}_t, t = 1, \dots, T\}$, the RMSE error is calculated as

$$RMSE(y_t, \hat{y}_t) = \frac{1}{T} \sqrt{(y_t - \hat{y}_t)^2}$$

Once all the above decisions were made, the KNN model was trained with the latest year of historical data and a periodic task was scheduled to automatically run through the flow of Figure 1 every day at the same time, i.e. every 24 hours. Each time one of the iterations is run, the horizon of the training set is moved one day forward, so the oldest day is removed and data from the most recent day is added. Then, a model is trained with the newly updated data and the training error, i.e. the RMSE that has been used for the choice of the best k in the model fitting phase, is obtained. The update of the deployed model occurs in case the RMSE of the new model is lower than the one obtained in the current model fit. Each model update is labelled with an incremental numbering that marks the evolution of the model version.

Results

Some of the results obtained in the PRENERGET run for the 4 rooms mentioned above are discussed below. For all cases, the initial model labelled 0.0 was trained with historical data from 1 April 2020 to 31 March 2021. Consequently, the first prediction was made for the 24-hour consumption corresponding to 1 April 2021.

Table 1 shows the updated versions of the electricity consumption prediction model for the CGBT2 consumption point. Up to 17 August 2021, 10 more accurate versions of the predictive model were deployed. In the first two columns are the start and end dates of the data period used for training the model. The RMSE in the third column corresponds to the mean error obtained with the number of neighbours chosen in the cross-validation. In this case, the error is reduced from 16.481 kWh to 15.5 kWh, an improvement of 6%. The last column is the incremental label of the model version.

Table 1: Evolution of the RMSE of the training of the different deployed versions of the predictive model of electricity consumption for the CGBT2 consumption point.

Initial date	End date	RMSE	Version
2020-04-01	2021-03-31	16.481	0.0
2020-04-02	2021-04-01	16.430	0.1
2020-04-03	2021-04-02	16.311	0.2
2020-04-09	2021-04-08	16.093	0.3
2020-04-10	2021-04-09	15.976	0.4

2020-04-13	2021-04-12	15.959	0.5
2020-04-20	2021-04-19	15.951	0.6
2020-04-29	2021-04-28	15.871	0.7
2020-05-02	2021-05-01	15.768	0.8
2020-05-03	2021-05-02	15.655	0.9
2020-08-18	2021-08-17	15.500	1.0

The graph in Figure 2 shows the model updates from version 6 to version 9. In black the actual values collected are shown and the different colours are used to differentiate the predictions obtained with each of the running versions.

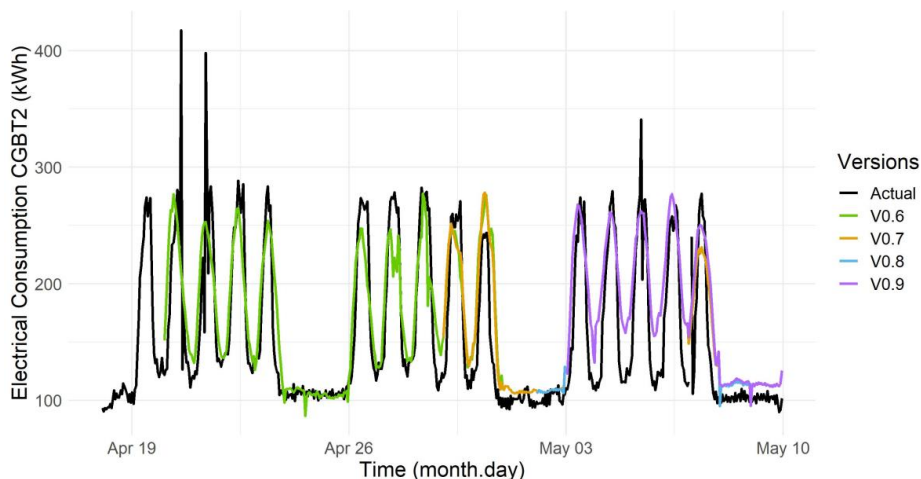


Figure 2: Evolution of predictions obtained by versions 6, 7, 8 and 9 of the deployed model.

The upgrade to version 10 that occurs on 17 August to start predicting values on 18 August is shown in Figure 3.

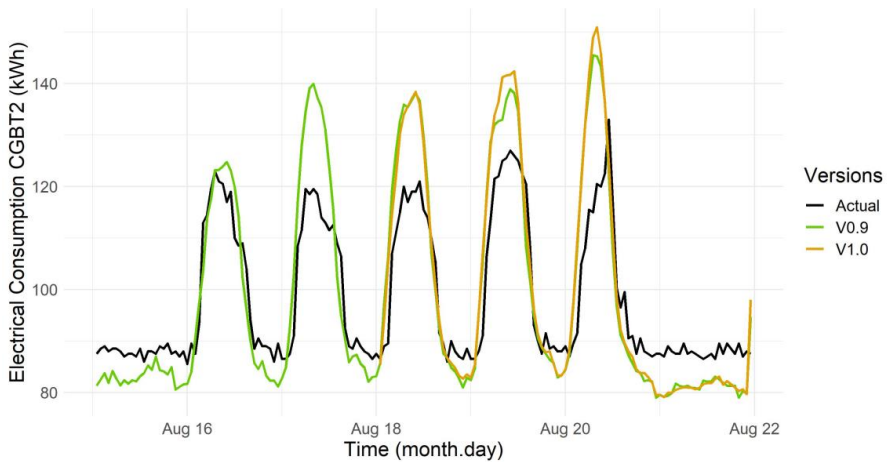


Figure 3: Evolution of the predictions obtained by versions 9 and 10 of the deployed models.

The model update tables for the other 3 monitored variables are shown below.

Table 2 shows the update of the model deployed for the consumption point CGBT1. In this case the process was stabilised at version 12 reached on 5 June 2021, having reduced the RMSE of the model from 49.804 to 14.791. This improvement represents a 70% reduction in training error.

Table 2: Evolution of the RMSE of the training of the different deployed versions of the predictive model of electricity consumption for the CGBT1 consumption point.

Initial date	End date	RMSE	Version
2020-04-01	2021-03-31	49.804	0.0
2020-04-02	2021-04-01	48.802	0.1
2020-04-07	2021-04-06	48.633	0.2
2020-04-08	2021-04-07	47.924	0.3
2020-04-10	2021-04-09	47.803	0.4
2020-04-11	2021-04-10	46.237	0.5
2020-04-17	2021-04-16	46.043	0.6
2020-05-23	2021-05-22	43.065	0.7
2020-05-28	2021-05-27	38.118	0.8
2020-06-03	2021-06-02	38.060	0.9
2020-06-04	2021-06-03	15.169	1.0
2020-06-05	2021-06-04	14.939	1.1
2020-06-06	2021-06-05	14.791	1.2

The update of the model deployed for the clean room is shown in Table 3. In this room the consumptions are lower and more controlled. This is reflected in the slower improvement of the RMSE. The model was updated to version 8 on 20 June, achieving a 30% improvement in the RMSE.

Table 3: Evolution of the RMSE of the training of the different deployed versions of the predictive model of electricity consumption for the clean room.

Initial date	End date	RMSE	Version
2020-04-01	2021-03-31	3.016	0.0
2020-04-16	2021-04-15	2.972	0.1
2020-05-28	2021-05-27	2.843	0.2
2020-05-29	2021-05-28	2.811	0.3
2020-06-04	2021-06-03	2.141	0.4
2020-06-05	2021-06-04	2.129	0.5
2020-06-09	2021-06-08	2.121	0.6
2020-06-12	2021-06-11	2.115	0.7
2020-06-21	2021-06-20	2.103	0.8

Finally, Table 4 shows the update of the model deployed for the consumption point under cover. In this case, the final version was version 13 reached on 18 June, having lowered the training error from 17,511 to 5,456. This improvement represents a 69% decrease in the internal RMSE of the model.

Table 4: Evolution of the RMSE of the training of the different deployed versions of the predictive model of electricity consumption for the consumption point under cover.

Initial date	End date	RMSE	Version
2020-04-01	2021-03-31	17.511	0.0
2020-04-03	2021-04-02	17.192	0.1
2020-04-07	2021-04-06	17.183	0.2
2020-04-10	2021-04-09	16.263	0.3
2020-05-23	2021-05-22	13.969	0.4
2020-05-24	2021-05-23	13.522	0.5
2020-05-26	2021-05-25	13.424	0.6
2020-05-28	2021-05-27	12.383	0.7
2020-06-04	2021-06-03	5.548	0.8
2020-06-05	2021-06-04	5.502	0.9
2020-06-06	2021-06-05	5.480	1.0
2020-06-07	2021-06-06	5.467	1.1
2020-06-13	2021-06-12	5.459	1.2
2020-06-19	2021-06-18	5.456	1.3

Conclusions

ML models based on historical electricity consumption data are key for demand forecasting and for different energy efficiency strategies. PRENERGET is a system that contributes to the control and updating of the models developed for these tasks in an automatic, simple and efficient way. In this way, it is possible to maintain and even improve the error of the ML models that periodically provide estimates of future electricity demand values. The modular architecture of PRENERGET allows analysts to test different algorithms and evaluate them with different training and error measurement strategies. In a simple way, the different versions of the models are stored together with the information of each model that has been used during the course of the iterative tasks. PRENERGET is cost-saving, especially in terms of time, as its structure provides a simple way to change the prediction approach avoiding the effort of analysis from the beginning.

In addition, model errors decrease over time, so having this automatic model re-training tool improves model resilience by automatically adapting to changes in the environment. In the examples analysed in this work, it has been shown that in about 3 months the RMSE of the models has been reduced by up to 70% in some cases.

Future work

PRENERGET is a system for periodically evaluating the effectiveness of electricity demand forecasting models based on historical consumption. The update of the models is given when a new adjustment with more recent training data obtains a lower error in the cross-validation. As a further evaluation, a module to evaluate model degradation by assessing the prediction errors obtained in the past is proposed as future work. That is to say, the error committed in the prediction of the past day would be collected by calculating a metric that compares the predicted values with the real ones and a Concept Drift detection system would be implemented. This method would detect deviations in the error distribution, which, by definition, should be normal.

Bibliography

- [1] Esnaola-Gonzalez, I., Jelić, M., Pujić, D., Díez, F. & Tomasevic, N. (2021), 'An AI-Powered System for Residential Demand Response', *Electronics* 10.
- [2] Gómez-Omella, M., Esnaola-Gonzalez, I., Ferreiro, S. & Sierra, B. (2021), 'k-nearest patterns for electrical demand forecasting in residential and small commercial buildings', *Energy and Buildings* 253, 111396.
- [3] Tascikaraoglu, A. & Sanandaji, B. M. (2016), 'Short term residential electric load forecasting: A compressive spatio-temporal approach', *Energy and Buildings* 111, 380–392.
- [4] Lusi, P., Khalilpour, K. R., Andrew, L. & Liebman, A. (2017), 'Short-term residential load forecasting: Impact of calendar effects and forecast granularity', *Applied Energy* 205, 654–669.
- [5] Widmer, G. & Kubat, M. (1996), 'Learning in the presence of concept drift and hidden contexts', *Machine learning* 23(1), 69–101.
- [6] Žilobait'e, I., Pechenizkiy, M. & Gama, J. (2016), 'An overview of concept drift applications', *Big data analysis: new algorithms for a new society* pp. 91–114.
- [7] International Energy Agency (2021), 'World energy outlook 2021'. URL: <https://www.iea.org/reports/world-energy-outlook-2021>
- [8] Warren, P. (2014), 'A review of demand-side management policy in the uk', *Renewable and Sustainable Energy Reviews* 29, 941–951.
- [9] Ahmed, R., Sreeram, V., Mishra, Y. & Arif, M. (2020), 'A review and evaluation of the state-of-the-art in pv solar power forecasting: Techniques and optimization', *Renewable and Sustainable Energy Reviews* 124.
- [10] Unterberger, V., Lichtenegger, K., Kaisermayer, V., Gölls, M. & Horn, M. (2021), 'An adaptive short-term forecasting method for the energy yield of flat-plate solar collector systems', *Applied Energy* 293.
- [11] Juban, J., Siebert, N. & Kariniotakis, G. N. (2007), 'Probabilistic short-term wind power forecasting for the optimal management of wind generation', in '2007 IEEE Lausanne Power Tech', IEEE, pp. 683–688.
- [12] Zhang, X. M., Grolinger, K., Capretz, M. A. M. & Seewald, L. (2018), 'Forecasting residential energy consumption: Single household perspective', in '2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)', pp. 110–117.
- [13] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J. & Zhang, G. (2018), 'Learning under concept drift: A review', *IEEE Transactions on Knowledge and Data Engineering* 31(12), 2346–2363.
- [14] Gomez-Omella, M., Esnaola-Gonzalez, I. & Ferreiro, S. (2020), 'Short-term forecasting methodology for energy demand in residential buildings and the impact of the covid-19 pandemic on forecasts', in 'Proceedings of 40th SGAI International Conference on Artificial Intelligence', Vol. 12498, pp. 227–240.
- [15] Gama, J., Žilobait'e, I., Bifet, A., Pechenizkiy, M. & Bouchachia, A. (2014), 'A survey on concept drift adaptation', *ACM computing surveys (CSUR)* 46(4), 1–37.
- [16] Celik, B. & Vanschoren, J. (2021), 'Adaptation strategies for automated machine learning on evolving data', *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43(9). URL: <https://ieeexplore.ieee.org/document/9366792>
- [17] D. Crankshaw, X. Wang, G. Z. M. J. F. J. E. G. I. S. (2017), 'Clipper: A low-latency online prediction serving system', in '14th USENIX Symposium on Networked Systems Design and Implementation (NSDI)', pp. 613–627.
- [18] Zhuozhao Li, Ryan Chard, L. W. K. C. T. J. S. Y. B. A. W. S. T. B. B. M. J. F. I. F. (2021), 'DIhub: Simplifying publication, discovery, and use of machine learning models in science', *Journal of Parallel and Distributed Computing* 147, 64–76.
- [19] R. Chard, Y. Babuji, Z. L. T. S. A. W. B. B. I. F. K. C. (2020), 'funcx: A federated function serving fabric for science', in 'Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing'.
- [20] Muthusamy, V., Slominski, A. & Isahagian, V. (2018), 'Towards enterprise-ready ai deployments minimizing the risk of consuming ai models in business applications', pp. 108–109.
- [21] Simmhan, Y., Aman, S., Kumbhare, A., Liu, R., Stevens, S., Zhou, Q. & Prasanna, V. (2013), 'Cloud-based software platform for big data analytics in smart grids', *Computing in Science & Engineering* 15(4), 38–47.
- [22] Choi, T.-M., Chan, H. K. & Yue, X. (2016), 'Recent development in big data analytics for business operations and risk management', *IEEE transactions on cybernetics* 47(1), 81–92.

Implementation of dqts R package

The dqts R package is available in GitHub and the `install_github` function of the `devtools` package can be used to install it.

```
1 install.packages("devtools")
2 devtools::install_github("MeritxellGomez/dqts-R-package")
```

Once installed, three simulated data sets are also available in the package to test the functions and become familiar with their use: `weatherdf` is a data frame with daily data from a multivariate time series of temperature and humidity; `salesets` is a univariate time series with monthly sales data of a product; and finally, `dfnorm` is a data frame with daily data of a Gaussian variable.

The use of the package is detailed below using the `weatherdf` data set as an example. As can be seen in the sample data in Table A.1, the data frame contains 3 columns with the information of the day in `yyyy-mm-dd` format, the temperature in degrees Celsius and the relative humidity in percentage. In total, there are 100 observations. Also available is the file of allowed ranges shown in Table A.2.

The four main functions can be used for DQ calculation, visualisation, inspection, and future improvement using `DQ` function, `plotDQ` function, `deepDQ` function and `handleDQ` function respectively.

- The `DQ` function calculates DQ metric values in two alternative ways: in the whole time series or by moving windows.

Tab. A.1.: Sample of the data contained in the `weatherdf` set of the `dqts` R package.

Date	Temperature	Humidity
2020-07-07	24.4	88
2020-07-08	28.9	85
2020-07-09	24.3	88
2020-07-10	25.5	90
2020-07-11	55.0	85

Tab. A.2.: Ranges values allowed for `weatherdf` data set of the `dqts` R package.

Date	Temperature	Humidity
2020-07-07	0	50
2020-10-23	40	100

This function accepts two possible R formats as the main input of the data to be evaluated: data frames, which are two-dimensional data structures and can contain data of different types; and time series, a special format for containing time-ordered information. The other input parameters contain relevant information for the correct calculation of the metrics:

- `var_time_name` is a string that indicates the name of the time variable to be evaluated, which is used as the temporal index of the data set.
- `maxdif` is a numerical value that indicates the frequency acquisition of the time series, that is, the maximum difference allowed between timestamps.
- `units` contains the information about time units, e.g. hours or days.
- `ranges` provides the minimum and maximum values expected for each of the variables.

- `dataref` is an extract of time series data with the standard used to check the name and format of the variables.
- `windows` is a logical parameter indicating whether DQ metrics are calculated by moving windows or not; and in case of being positive:
 - * `initialWindows` fixes the size of the first subseries.
 - * `skip` is the number of observations between the beginnings of two consecutive window subseries.
 - * `fixedWindow` is a logical value to determine if the window size remains constant or not.
- `weighted` is a vector of length 11 to assign weights to each DQ metric which is fixed by default to assign equal importance to all metrics. The users can use their knowledge to modify the weight of each metric in the final quality value.

Regarding data inputs, some of them, such as ranges can be estimated by the function using a random sample drawn from the data set in case they are not provided by the user. Other inputs, like `windows`, have a default value which is set to calculate the metrics on the full set of the time series. However, other information like `var_time_name` depends on the evaluated time series and can only be provided by the user, thus if that information is not available, the function throws an error message and does not allow the calculations to continue.

With the following code instructions, the total data quality of the `weatherdf` data set is calculated. Note that the default fixed value in the `windows` argument is `FALSE`, so the metrics are automatically executed using all available information.

Tab. A.3.: Metric scores for the complete execution of DQ function in weatherdf data set.

Initial Date	2020-07-07
Final Date	2020-10-23
Completeness	0.98
Completeness by Observations	1
Completeness by Variables	1
Time Uniqueness	0.99
Range	0.9932
Consistency	0
Moderation	0
Typicality	0
Timeliness	0.9009
Formats	1
Names	1
Data Quality	0.9830

```
1 dq_complete <- DQ(data = weatherdf ,  
2 var_time_name = 'date', maxdif = 1,  
3 units = 'days', ranges = weatherRange)
```

The `dq_complete` data frame is composed of a single row and 14 columns with the initial and final dates, the value of the 11 quality metrics, and the average total value of the data quality calculated as an arithmetic mean of the metrics. Table A.3 displays the obtained result flipped to facilitate visualisation.

Problems can be seen in the *Completeness* as its value is 0.98, which means that 2% of the values in the series are missing. 0.7% of the data are out of the established ranges because the value of its metric is 0.993. Furthermore, 8.9% of the observations have not been received at expected time instants, so *Timeliness* = 0.901. Finally, *TimeUniqueness* = 0.99 indicates that there are repetitions in 1 out of 100 time entries.

On the other hand, the value of quality metrics can be computed by rolling windows by setting `windows = TRUE` and specifying the size of the windows. In the following example, the metrics are executed by windows of size 10, and a leap of 3 values is made between consecutive windows.

```
1 dq_windows <- DQ(data = weatherdf ,
2   var_time_name = 'date', maxdif = 1,
3   units = 'days', ranges = weatherRange ,
4   windows = TRUE, initialWindow = 10, skip = 3)
```

As the original data set consists of 100 observations, this type of execution performs 23 calculations of the metrics, obtaining the `dq_windows` data set of 23 observations for the 14 columns mentioned in the previous case. In this case, for each row of the data set, the initial and final dates indicate the position of the window in which the metric calculations are performed.

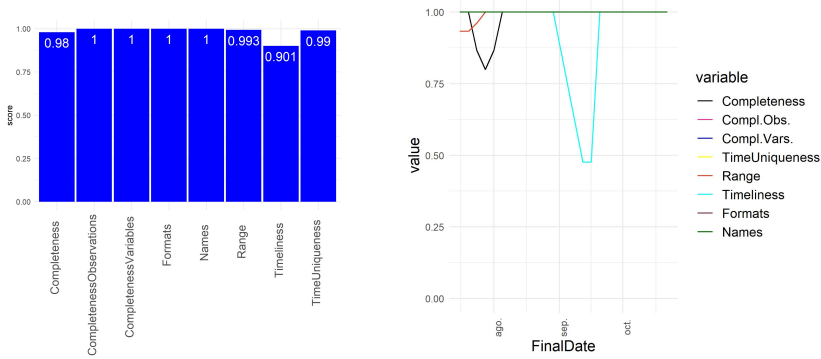
- The `plotDQ` function allows the visualisation of the quality metric scores and receives as input the output of `DQ` function. There are two types of graphs depending on whether the `DQ` function has been run per window or on the complete data set. Figure A.1 shows the results of executing this function on both `DQ` information obtained from the previous function.

On the left, the visualisation of the performance of the quality metrics in a complete time series is obtained by the following code:

```
1 dqplot_complete <- dqplot(dq_complete ,
2   normal = FALSE)
```

On the other hand, the graph on the right shows the evolution of the quality metrics over time and it is obtained by the following code:

```
1 dqplot_windows <- dqplot(dq_complete ,
2   normal = FALSE)
```



(a) Complete Data Quality

(b) Moving Windows Data Quality

Fig. A.1.: The two modes of execution of the data quality visualisation in an example without normality metrics

This shows that the *Completeness* drops at the beginning of the series, before August, and later stabilises again at the maximum value. It can also be seen that the *Timeliness* value drops below 0.5 in mid-September and that the *Range* failures are due to the start of the time series.

- The deepDQ function provides details about the failures in the indicated metric. Therefore, this function receives as input the data and the name of the metric to be evaluated. In addition to the parameters mentioned in the previous function concerning temporality, range and reference set, this function has two different ways of execution. If the logical parameter *position* is negative, the information given is the value of the metric indicated by *variable*, whereas the alternative option is to indicate the positions where the metric is failing in each of the variables. For instance, both possibilities are shown to inspect problems in *Completeness*. In the first case, it can be observed that the failure of *Completeness* is due to the humidity variable obtaining a score of 0.94. In the second case, the exact dates on which the humidity variable has missing data can be seen.

```

1  deepDQ(data = weatherdf, metric = 'Completeness',
2  var_time_name = 'date', position = FALSE)
3
4      date temperature    humidity
5      1.00         1.00         0.94
6
7
8  deepDQ(data = weatherdf, metric = 'Completeness',
9  var_time_name = 'date', position = TRUE)
10
11  $date
12  Date of length 0
13
14  $temperature
15  Date of length 0
16
17  $humidity
18  [1] "2020-07-21" "2020-07-22" "2020-07-23"
19  "2020-07-24" "2020-07-25" "2020-07-26"

```

- The `handleDQ` function estimates solutions to faults and returns a data set with the necessary changes for that metric to achieve the highest quality score. This function also receives the classic temporality, range and reference parameters. In addition, it allows the selection of the correction procedure from a list of possible methods by entering the name in the `method` argument.

The *Conformity* dimension can be solved by two methods both for *Format* and *Name* related problems. The first approach is to remove from the dataset those variables that present errors. The second case is to force the relevant changes so that the dataset complies with the characteristics specified in the reference set.

For the metrics related to the Gaussian variables (*Consistency*, *Typicality* and *Moderation*) the method is to consider the mean and standard deviation of a sample of the data and simulate

values that meet the confidence intervals with this, the data that cause the low quality of these three metrics is replaced.

A similar procedure can be applied to solve quality problems in the rest of the metrics. The first step is to identify the observations that cause low quality. In the case of a low value in *TimeUniqueness*, observations with repeated time values are identified. When there is a low *Timeliness* value, missing data is found by observing the temporality of the data. To resolve the low-range values, it is necessary to find the data that fall outside the established ranges. Finally, to solve for *Completeness*, missing values are identified.

Once the data to be processed have been identified, techniques such as deletion or imputation can be used. Deletion consists of removing the identified values that cause problems in quality. Imputation is the technique of replacing data with estimated values, typically used to fulfil missing values. Different methods can be used to deal with missing values. The most common approach is to use a single value such as an average, median, maximum, or minimum value to replace anomalous data. Nevertheless, other methods are available in the *dqts* package. Among them, the user can choose the interpolation between the values before and after the loss of data by means of linear or polynomial interpolation and with the Last Observation Carried Forward (LOCF) or Next Observation Carried Backward (NOCB) methods where the missing values are replaced with the last or the next observed value, respectively. In addition, by having the allowed ranges for each variable, data can be imputed with the minimum, maximum or average of these ranges. The values to be replaced can also be estimated using the outcome of executing the KNPTS algorithm.

Continuing with the previous example, it has been seen in Table A.3 that the metrics that do not reach the total score are

Time Uniqueness, Timeliness, Range, and Completeness, in the order in which they will be addressed. The chosen methods are the elimination of observations with repeated timestamps, the creation of missing timestamps by filling temperature and humidity with missing values, the replacement of out-of-range values with missing values, and finally the imputation of the original missing values and those created in the other two fixes through a mean. These steps are performed in the following way using pipelines of the `dplyr`¹ package:

```
1  handleDQ(data = weatherdf, metric = 'TimeUniqueness',
2    var_time_name = 'date', method = 'deletion') %>%
3
4    handleDQ(data = ., metric = 'Timeliness',
5      var_time_name = 'date', maxdif = 1,
6      units = 'days', method = 'missing') %>%
7
8    handleDQ(data = ., metric = 'Range',
9      var_time_name = 'date', ranges = weatherRange,
10     method = 'missing') %>%
11
12   handleDQ(data = ., metric = 'Completeness',
13     var_time_name = 'date', method = 'mean')
```

After these actions, the dataset with the highest data quality score can be saved and used in future analytical activities.

The execution flow to follow for the correct handling of the package is summarised in the diagram in Figure A.2. The process starts with the detection of Gaussian variables in step *A* and the calculation of missing parameters in case of need in step *B*. Then, in step *C*, the corresponding 11 metrics are computed with the DQ function and the quality of each of them is reported. In case the quality of the data does not reach the maximum score, the non-perfect metrics can be inspected with `deepDQ` in step *D*. Once the problems are understood, the user

¹<https://dplyr.tidyverse.org>, <https://github.com/tidyverse/dplyr>

can decide whether to improve the value of a certain metric in step *E* with the correction function `handleDQ`. One by one metric scores can be improved until they reach the desired quality.

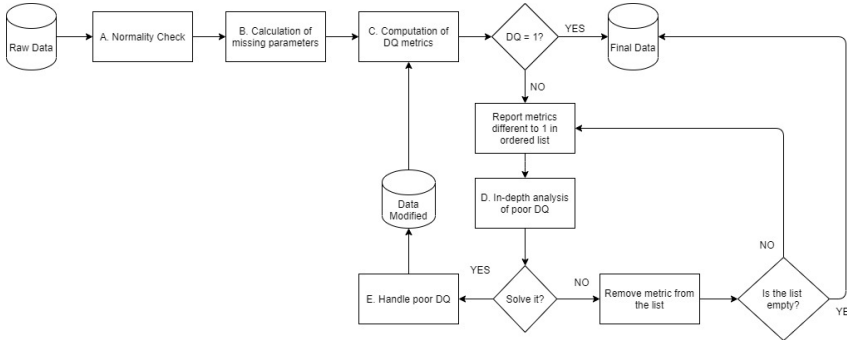


Fig. A.2.: Flow for the detection, inspection and resolution of poor DQ problems

This flow represents the iterative logic of the functioning of the `dqts` package functions for inspecting and improving data quality in time series. Thanks to the distribution of functions and the package structure, it is possible to expand or customize its capabilities by creating new metrics in the DQ function to evaluate new scenarios. Additionally, in the `handleDQ` function, other resolution methods can be added to enhance data quality.

Implementation of KNPTS R package

The KNPTS R package is available in GitHub and the `install_github` function of the `devtools` package can be used to install it.

```
1 install.packages("devtools")
2 devtools::install_github("MeritxellGomez/knpts-R-package")
```

The main function, `predictKNPTS`, is used to obtain the next H values of a univariate time series, obtained from the combination of K different intervals of the historical series.

The number of K intervals used for the prediction can be chosen either manually or through cross-validation by providing the maximum value to be tested in the process with the `kmax` argument, which is fixed as 10 by default. The internal iteration is responsible for selecting the optimal value of K that provides the most accurate estimation. In cross-validation, the number of partitions done in the training phase can also be chosen by the argument `repeats`, the value of which is, by default, 3.

Different time series similarity measures can be used to decide the most similar patterns. In this way, similar intervals can be chosen because they minimise the Euclidean distance or the Dynamic Time Warping (DTW) value, which are discussed in more detail in Section ???. This decision can be taken by using the input argument `distance`.

Predictions can be obtained following two different strategies explained in detail in Section 2.3.5: the *MIMO* strategy, where H values

are obtained all at once using the same patterns; or the *Recursive* strategy which provides the values one at a time, using K different patterns in each of the H predictions and entering the newly estimated value at each step. The user can decide the strategy used by introducing it as input in the main function.

The consecutive values of the selected K intervals used to make forecasts of the series can be combined using an arithmetic mean in which all values have the same weight or an average weighted according to the distance to the pattern. That decision is introduced by the boolean `pond` argument.

An example of a KNPTS run using the R package is shown with the data from the electricity consumption data mentioned use case described in Section 2.3.1. The following code is used to execute a one-week-ahead forecast using the KNPTS algorithm in two different ways: by following the MIMO strategy on the one hand, and recursive strategy on the other hand. In both cases, the length of the reference pattern is tested for one, two and three weeks, but the best results are obtained for the one-week window ($W = 7$). Furthermore, a cross-validation repeated 3 times is run to test the optimal number of neighbours to be used for forecasting between 1 and 5. The measure used to calculate the similarity between windows in the data history is the Euclidean distance and the accuracy measure to evaluate the predictions obtained is the RMSE.

The R code used for both strategies is shown below:

```
1 m_7_7_T <- predictKNPTS(ts = ts_train, w = 7, h = 7,
2   kmax = 5, dist = 'Euclidean', repeats = 3,
3   pond = TRUE, strategy = 'mimo', metric = 'rmse')
4
5 r_7_7_T <- predictKNPTS(ts = ts_train, w = 7, h = 7,
6   kmax = 5, dist = 'Euclidean', repeats = 3,
7   pond = TRUE, strategy = 'recursive', metric = 'rmse')
```

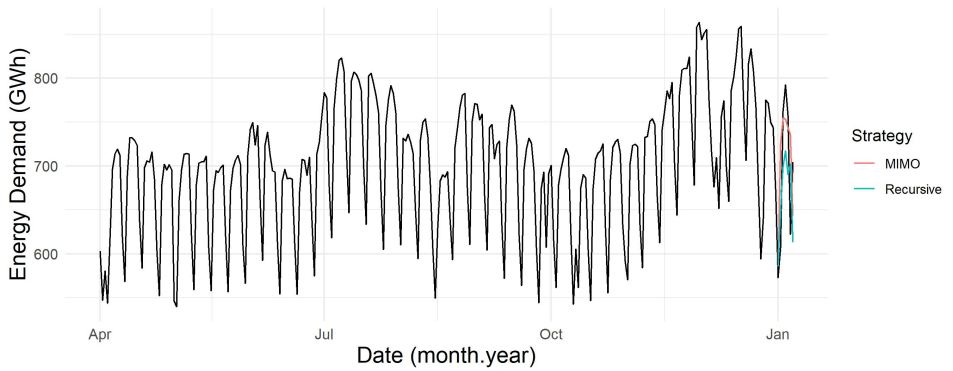


Fig. B.1.: One week ahead forecasting using KNPTS with MIMO and Recursive strategies.

The forecasting results are shown in Figure B.1, in which $K = 4$ neighbours and the estimation values have been combined by means of an average weighted by neighbours' distance to the reference pattern. Similar to the SARIMA model, the estimation of the seven points corresponding to the following week's forecast is sufficiently accurate as the periodicity of the series indicates that there are weekly patterns in electricity consumption.

As noted earlier, the problem arises with forecasts done more than one week ahead. The execution of ten-weeks-ahead forecasts is done with a pattern of $W = 21$ instead of $W = 7$, as in the previous case, since the most accurate results are obtained using three weeks as the window size. In Figure B.2, the forecasting results show that the *Recursive* strategy is more suitable for long-term predictions. As can be seen, providing such a large number of estimations in one execution causes a loss of accuracy and error accumulation.

Nevertheless, the results are more accurate if the forecast for the next 7 days is run periodically. In the graph in Figure B.3 the result of running the *MIMO* strategy without using estimated values demonstrates the accuracy of not using the estimates as inputs in time series, a strategy that clearly avoids the accumulation of error

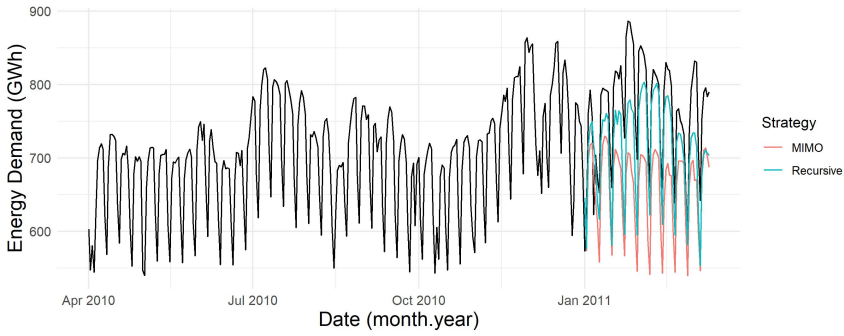


Fig. B.2.: Ten weeks ahead forecasting using KNPTS with MIMO and Recursive strategies.

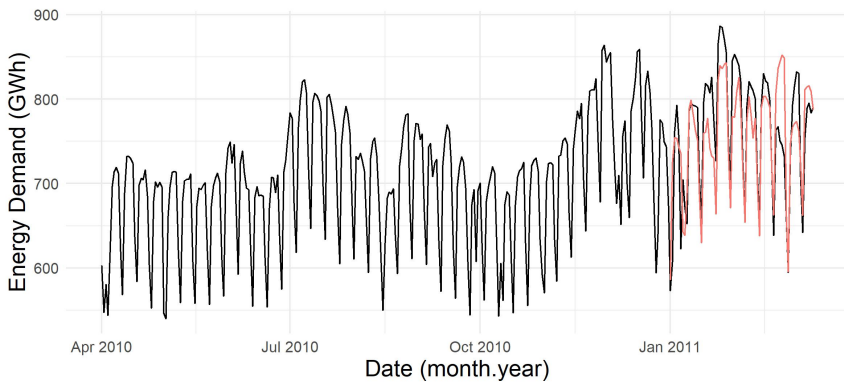


Fig. B.3.: Ten weeks ahead forecasting using KNPTS with ten different executions using MIMO strategy.

in the long run. Also, a periodic re-training to obtain the predictions generally gives a more accurate result. Therefore, it is advisable to use such an approach when the deployment method of the developed solution and the computational cost of the system allows it.

Colophon

This thesis was typeset with $\text{\LaTeX}2_{\epsilon}$. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at <http://cleanthesis.der-ric.de/>.

Declaration

You can put your declaration here, to declare that you have completed your work solely and only with the help of the references you mentioned.

Eibar, May 31, 2023

Meritzell Gómez Omella

