Master in Computational Engineering and Intelligent Systems

Master Thesis

2016-2017

Occupancy Estimation and People Flow Prediction in Smart Environments

Author

Unai Saralegui Vallejo

Advisors

Javier Muguerza

Olatz Arbelaitz

Department of Computer Architecture and Technology,
Faculty of Informatics, UPV/EHU

Miguel Ángel Antón

Smart Buildings & Cities, Sustainable Construction Division,
TECNALIA Research & Innovation

If there is effort, there is always accomplishment.
—Jigorō Kanō

Nire alboan zaudetenoi,
eskerrik asko zuen babes eta laguntzagatik.

Abstract

Recent advances in the field of the Internet of Things paradigm and the ability gained to install connected sensors, and obtaining data through them has opened various research paths. Smart buildings and cities are gaining incremental attention in the recent years with both public and private investments. In this work two of the problems existing in the area are analysed and some fixes are proposed for such smart environments. Firstly, a binary occupation detection method based in indoor quality monitoring data is proposed, while the use of a regression model for predicting the amount of people expected in one place at a specific moment is shown. Machine learning based classification and regression techniques have been used in order to build the models, obtaining overall acceptable results for both problems.

Laburpena

Azken urteotan *Internet of Things* eremuan eta konektatutako sentsoreetan eman diren aurrerapenak direla eta, bertatik datuak eskuratzea asko erraztu da hainbat ikerketa lerro bultzatuz. Eraikin eta hiri adimendunak geroz eta atentzio gehiago ari dira jasotzen, bai inbertsio publiko zein pribatuekin. Lan honetan lerro horretan dauden bi arazo aztertzen dira halako ingurune adimenduetarako konponbide batzuk proposatuz. Batetik, gela bateko okupazioa edo honen eza detektatzeko metodo bat proposatzen da inguruko aire kalitatea neurtzeko sentsoreen datuez baliatuz, eta horrez gain, erregresio teknikak erabili dira leku eta une jakin batean esperotako pertsona kopurua aurreikusteko. Horretarako ikasketa automatikoko tekniketan oinarritutako klasifikazio eta erregresio teknikak erabili dira ereduak sortzeko, orokorrean emaitza onargarriak lortuz.

Resumen

Los recientes avances en el área del *Internet of Things* y de la habilidad para colocar sensores conectados han reforzado varios ámbitos de investigación. Los edificios y ciudades inteligentes están obteniendo mayor atención con financiación tanto pública como privada. En este trabajo se analizan dos de los problemas existentes en el ámbito son analizados y se proponen algunas soluciones para dichos entornos inteligentes. Por un lado, se presenta un método de detección de ocupación binaria en una sala, mientras que por otro lado, se propone la aplicación de modelos de regresión para predecir el número de personas esperado en un lugar e instante concretos. Con el fin de crear ambos modelos, se han utilizado técnicas de aprendizaje automático para clasificación y regresión, consiguiendo en general unos resultados aceptables.

Contents

Chapter 1

Introduction

Smart, probably the most fashionable word in the last years. We are hopefully in the initial stages of the artificial intelligence revolution and the field has received increasing attention in the media because of some important advances such as the use of deep learning techniques for image recognition. In this current context, the word smart has become the second name of a bunch of products in the market (smart-phones, smart-fridges, smart-TV, etc.). But, what is smartness? What does it stand for in today's market?

According to the Cambridge Dictionary the word *smart* has at least two definitions that are valid for this consideration. The first of those definitions stands for the first thing that comes to our mind when thinking about smartness, the one developed by human beings. The second one refers to machines or other tools that make use of computers to work in an independent way. Now, the thought of every machine working by itself in an efficient way is still an utopia, but current products are able to perform some tasks automatically more efficiently than a human is able to do, that is what smart stands for in today's market.

Not only do electronic products receive the smart name, but also entire buildings or cities are aimed to become *smart*. Smart building and cities are two paradigms inside a bigger framework named Ambient Intelligence (AmI), which also takes names as pervasive computing or ubiquitous computing. Ambient Intelligence refers to environments that are sensitive and responsive to the presence of people [1]. In such an environment devices work in concert to support people in carrying everyday life activities in a natural and easy way, such process is performed by using information hidden in the network of devices available [2].

The mentioned intelligent environments base their functionality in the information of the near environment. So both hardware and software are needed to develop such a place. In that sense, the advances in the Internet of Things (IoT) field should be an important push for the Ambient Intelligence paradigm. Sensors are getting smaller and cheaper so lots of them can be put together at the same place for measuring as much information as possible.

The ambient intelligence field does not only stand for providing services to humans, it can also be valid for developing more efficient solutions. The network of connected sensors can measure lots of information which could be valid for developing efficient HVAC (heating, ventilation and air conditioning) systems control, for both energy efficiency and personal comfort for example. In this topic, deploying an IoT sensor network and analysing the data measured alongside BIM (Building Information Modelling) models of the buildings, which are files that support decision making regarding a building, should gain attention in the following years as they make it possible to simulate the performance of the changes that are applied. These technologies working together with artificial intelligence algorithms could revolutionize the construction industry in the future by allowing to design more efficient solutions [3].

The ambient intelligence paradigm hopes to develop solutions to help people in their everyday life, but there is a problem that should be addressed before of that, and that is detecting human presence. Three major methods have been tried for this particular purpose: video-image, infrared sensing and multi-sensor (temperature, humidity, CO_2, etc.) detection methods. A multi-sensor based detection method is proposed in this thesis; the importance of the proposed models are not the techniques used, as similar models have already been presented in the literature, but this study is relevant as the data have been gathered in a real building, while the majority of studies of this kind have been performed in closed laboratory like environments. As it will be explained, some considerations need to be taken into account when trying to apply these sort of techniques in a real building.

Being able to detect human presence could be used to count people, which could lead to generate predictive models forecasting the expected crowdedness or to determine changes in order to have a more efficient environment. This could be used to program events at the best moment, or to predict the expected amount of people in a building that could help in scheduling the HVAC functioning. In this work a predictive model for forecasting the amount of people expected in a particular place has been obtained.

Recent advances in the IoT paradigm and in the artificial intelligence field have opened the way for implementing different smart solutions that made able to provide better services while the cost can also be reduced. That is why this research area has gained attention not only in the scientific field, but also for public and private investors.

To get a better understanding of the work carried out, the structure of the document is as follows. In the following section the problems are described, the state of the art is presented and theoretical aspects about the supervised learning algorithms used in this thesis are introduced. In the third section the occupancy detection framework is proposed and the results obtained are discussed. Then, in section four, the people flow predictive model is presented and discussed. Finally, the general conclusions of the work are presented in section five.

Chapter 2

Problem Statement and State of the Art

In this section the analysed problems are introduced with the state of the art for each one, current research paths and possible limitations of the proposed methods are also mentioned. In the last section of this chapter some supervised learning algorithms used in this work are introduced and explained.

2.1 Context of the problems

As mentioned in the introduction, two related problems have been analysed. In the one hand, the problem of detecting people by using indoor climate monitoring infrastructure is analysed, while on the other hand, predicting the amount of people in one space based on some criteria is studied.

These two problems are grouped in the Ambient Intelligence (AmI) research field, and they address two relevant topics in the area. Ambient Intelligence is the area in which smart building and cities are being designed and developed. We can consider a smart building or city as a new way of providing services more efficiently by using such degree of smartness [4][5]. In the smart building and cities (SBC) area various research paths are gaining increasing attention, especially with the advances in the Internet of Things (IoT) paradigm and the Big Data analysis [6][7]. Some hot topics in this research field include city security, surveillance, providing more efficient public services, event scheduling, etc. [8][9][10][11].

In most fields of study in the ambient intelligence area it is a necessity to determine the occupancy of a specific place. Various approaches exist already to determine occupancy in a place, but each of the methods has its own pros and cons [12][13]. In the approach presented here, a privacy respectful method is proposed by using indoor climate monitoring technology. Such technology is privacy respectful as it does not record any image or person's ID. In contrast, the level of accuracy is inferior compared to other methods explained later, but it can be enough for some applications e.g. Heating Ventilation and Air Conditioning (HVAC) systems control.

Image-Video based detection is the most accurate way for obtaining the amount of people in a room and it has widely been studied in the literature [14][15][16]. However, issues regarding privacy and computational cost limit the usability of this kind of technology in certain areas. Another approach presented to measure occupancy in a room is the one based in PIR sensors. These type of sensors do not have any problem regarding privacy, but they show problems when no movement is detected [1]. Despite its limitations, some researchers have proposed to use them for determining presence [17][18][19].

In the work presented in this thesis the estimation of occupancy is determined by using an already existing indoor climate monitoring, in which the CO_2 concentration has shown to be a valid indicative [20][21][22][23][24]. This approach has also some limitations, such as the impossibility of detecting rapid changes, or the air spread through adjoining areas in a building [25][26]. Nonetheless, developing a method with those resources is aimed in the scientific field.

For pursuing the goal of detecting presence in a place with indoor climate monitoring, an elderly caring institution's building was monitored by the Smart Buildings and Cities group at Tecnalia Research & Innovation; the monitoring was primarily designed for HVAC control, but the data was made available to carry out this study. In this sense, it is a necessity to understand the data available. Because of that, different techniques need to be applied in order to detect abnormalities or to discover relations between variables that could lead to gain greater knowledge [27][28]. It is necessary to understand the data and perform whatever transformations that may be needed in order to clean, modify or edit the original set of values. In a real case scenario data is not clean and such cleaning is an important part of any project as it could lead to much better results [29].

As it will be later explained five rooms of the mentioned elderly caring institution's building have been monitored measuring temperature, relative humidity and CO_2 concentrations in 10 minute intervals. After cleaning and preprocessing the data, some visualizations have been performed to infer some relations and interesting phenomena. Finally, some supervised learning based models have been trained to determine if a room is occupied or not by using the three measured values.

Monitoring occupancy and amount of people in a room is vital for prediction of occupancies and people movements through a building or a city. Predicting how many people will be attending an event, or how many people is expected at the rooms of a building can be beneficial to manage HVAC systems or services [30][31][32][33][34]. It is especially remarkable the use of predictive models for predicting the amounts of tourists travelling from one country to other. Studies of this kind have been widely analysed since the 1970s till nowadays [35][36][37]. In the tourism area the first studies and predictive models were based on autoregressive models such as ARIMA [36]. Recently, because of the advances in the artificial intelligence field, more advanced approaches based on supported vector machines and artificial neural networks have been proposed [38][39][40][41][42][43][44].

In this field of study the first idea was to develop a predictive method based on data taken in the city of Ávila, under the *Smart Heritage City*, SHCity, project[1], partially funded by the European Regional Development Fund (ERDF)[2]. However, the data did not arrive on time, so the study was made over a publicly available dataset at Kaggle. The name of the dataset is “Crowdedness at the Campus Gym”[3], it is a dataset containing information from a Campus Gym, monitoring the number of people in the Gym, temperature data and other variables. This dataset was selected and used because its data is valid to work with the problem of predicting the expected amount of people in a place.

The problem with this dataset is that as the data was not directly taken by us, there were some details we did not know. Taking that limitation into account, the aim was to build the most accurate model possible and to become familiar with the techniques used in this type of problems, which will be helpful for future work.

The mentioned dataset consists in more than a year of monitoring, with data been taken about every ten minutes. As it will be later shown, the class of the variables available at the dataset was not rich, so the dataset was extended by adding additional information scraped from the web. After merging the original data with the scraped one, some visualization were performed to infer visual relations, while the relevant data was used to build some models to determine the expected amount of people based in the available data. As it will be explained, the results are generally good achieving lower error values than the currently reported ones.

2.2 Supervised Learning algorithms

As it has been mentioned, both problems analysed in this thesis use supervised learning algorithms to obtain a classifier in one case and a regressor in other. As it will be later seen, more than one method was used in each of the cases to build the models. In that way the best sort of methods for each case could be determined. In this section the most relevant of those models are introduced.

2.2.1 Artificial Neural Networks

Artificial neural networks (ANN) base their functionality in replicating the behaviour of human brain neurons, see figure 2.1. Each neuron receives some inputs by the dendrites and returns an output through the axon. That is the way a single artificial neuron, or perceptron, works, determining the output by the activation function. Such neurons are usually netted in a layer and the patterns are learned or modelled

[1]Smart Heritage City
[2]European Regional Development Fund (ERDF)
[3]Crowdedness at the Campus Gym

by adjusting the weights between the outputs and inputs of the neurons, which allows modelling non-linear problems [45][46]. Despite being a simple idea, the results obtained with this methodology are astonishing and their popularity is increasing due to the recent interest in deep learning [47].

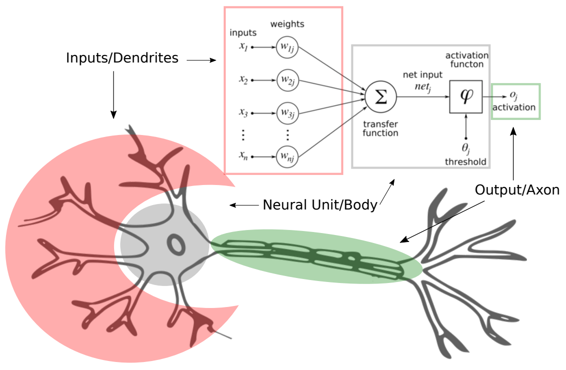


Figure 2.1: *A human neuron compared to an artificial neural unit.*

Figure 2.1 shows the relation between the human neurons and the perceptron. As it can be seen, the perceptron starts in some inputs, which are usually weighted; the inputs are passed to the transfer function, normally the summation of the inputs, whose output is passed to the activation function that returns the output. The activation function can take various forms, but it usually is either a logistic function (see equation 2.1) or a hyperbolic tangent (see equation 2.2).

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.1}$$

$$f(x) = tanh(x) = \frac{2}{1+e^{-2x}} - 1 \tag{2.2}$$

The neural units are usually organized in a bigger net, named neural nets, which are a structure organized by layers: an input and an output layers, and one or more hidden layers between them, see figure 2.2. Each of the layers is formed by one or more artificial neural units, and the outputs of a given layer are weighted and become inputs for the next layer. The learning process happens by adjusting the weights of the inputs of each unit, $\omega_{i,j}^{(k)}$, to provide the desired output. Such process can be performed by different optimization processes, being backpropagation, an optimization process that adjusts the weights by backpropagating the errors [48], one of the most used methods.

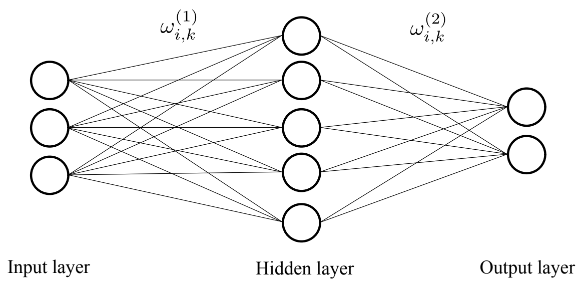


Figure 2.2: *Example of a neural net with a single hidden layer.*

2.2.2 Support Vector Machines

Support vector network (SVN) is a supervised learning technique for both classification and regression. The most differential characteristic of this method is that even if most algorithms try to reduce the input space to a lower dimensional one, the SVN maps the input feature space to a much higher dimensional one. The idea is to map the input space to a higher dimensional one where the separation of the different groups can be performed by a linear method, this simple idea is shown in figure 2.3. By using a linear decision surface as a separator for the classes the network acquires high generalization ability [49].

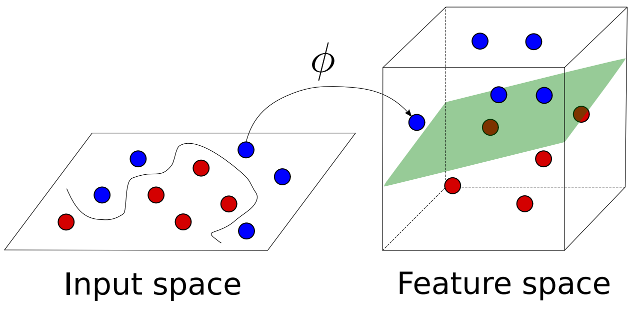


Figure 2.3: *Linear separation of classes in a higher dimensional space.*

The SVN algorithm tries to find an optimal hyperplane for separating efficiently the classes, such optimal hyperplane is defined as the linear decision function with maximal margin between the vectors of the two classes (an example of that hyperplane is shown in figure 2.3). Only a small amount of the training data, the *support vectors* are used for this task [49].

Support vector machine (SVM) is quite a satisfying algorithm from a theoretical point of view. In one hand it is based on some simple ideas and provides a clear intuition of what learning from examples is about. On the other hand, it can lead to high performances in practical applications [50].

Someone might think that even if the final separation is a simple linear method, the calculations required to map the inputs to a higher dimensional space must be computationally hard, but that limitation is overcome by using the so called *kernel trick*. The kernel trick enables to operate in a high-dimensional implicit feature space without ever computing the coordinates of the data in that space, but rather by simply computing the inner products between the images of all pairs of data in the feature space. This operation is often computationally cheaper than the explicit computation of the coordinates. Such kernel can be a linear one, but it is usually either polynomial (see equation 2.3), sigmoid (see equation 2.4) or radial (see equation 2.5) [51].

$$k(x,x') = (\langle x,x'\rangle + c)^p \tag{2.3}$$

$$k(x,x') = tanh(\kappa(x \cdot x') + \Theta) \tag{2.4}$$

$$k(x,x') = exp\left(-\frac{||x-x'||^2}{2\sigma^2}\right) \tag{2.5}$$

2.2.3 Decision Tree

A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules. So, the decision tree algorithm can be explained as a model that breaks down the data by making decisions by asking a series of questions. An example of a simple binary decision tree is shown in figure 2.4.

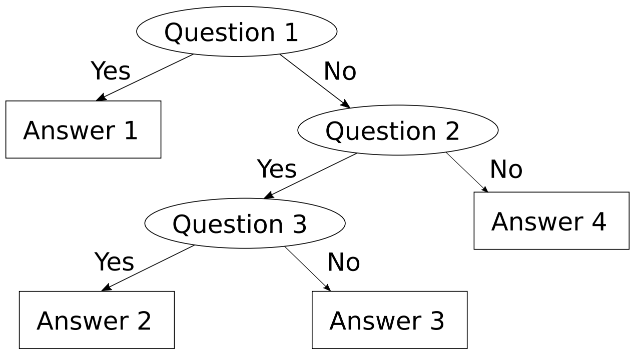


Figure 2.4: *Structure of a simple decision tree.*

One of the most important features of the decision tree algorithm is their capability to break down a complex decision-making process into a collection of simpler decisions, thus providing a solution that is often easier to interpret [52]. Thus, decision tree classifiers are attractive models according to the interpretability of the results due to their simplicity. Such simplicity haves its weaknesses because in complex problems the generated decision tree usually results in a very deep tree, which can easily lead to overfitting. The overfitting problem is tried to be avoided by pruning the tree by defining a limit for the maximum depth of the tree [53]. Of course defining such maximum limit is a problem as small limits lead to underfitting while great limits lead to overfitting problems.

2.2.4 Random Forest

Decision-tree classifiers are attractive because of their many advantages: the idea is intuitively appealing, training is often straight-forward and, best of all, classification is extremely fast. The problem with decision tree classifiers is that there is a fundamental limitation on the complexity of them; they should not be grown too complex to avoid overfitting the training data. As the complexity of the tree is limited, the results obtained with them are also limited; however, they are fast and easy to understand. The essence of the random forest method is to build multiple trees in randomly selected subspaces of the feature space [54]. The output of each single tree is taken into account, as the final output of the random forest model is a majority vote for classification tasks and averaging in regression problems. A visual representation of the algorithm can be seen in figure 2.5.

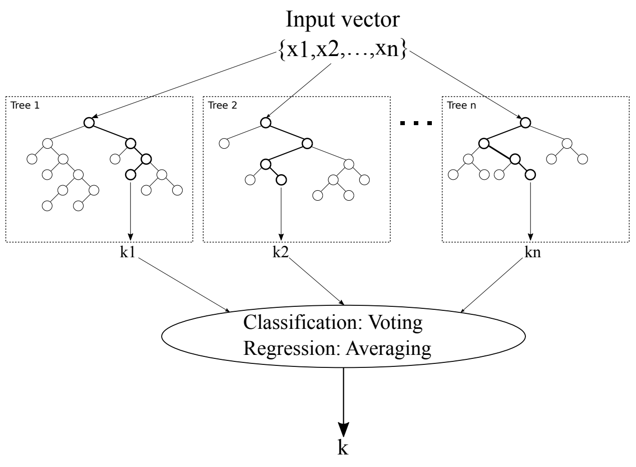


Figure 2.5: *Architecture of the random forest model.*

Making use of various classifiers to build a more robust one overcomes the bias problem, as the use of multiple classifiers can often compensate for the bias of a single classifier [55]. The process for creating the different trees uses the same data for all the trees, being the feature space what changes; given a feature space of m dimensions, there are 2^m subspaces in which a tree can be built. The vast number of subspaces in high dimensional feature spaces provides more choices than can be used in practice. Thus, each tree generalizes its classification in a different way.

2.2.5 Boosting Methods

Boosting methods are related with the previously introduced random forests in the way that both methods use various simple classifiers, generally simple decision trees, to produce a more robust one. What changes is the way of learning those simple classifiers; while the random forest algorithm uses different subspaces of the feature space, the boosting method bases its functionality on weighting the data samples regarding the output of each of the simple classifiers, in a way in which the misclassified entries are given a higher weight. Boosting is capable of both bias and variance reduction, and thus differs fundamentally from bagging, which is only able to reduce variance [56].

The basis of this algorithm remains in merging the output of various simple classifiers to build a robust model. In this case the models are built sequentially as the classification performed by the previous model is used to assign the weights of each of the entries [56]. In that way the new classifier tries to classify correctly the entries wrongly classified by the previous model, a simple example of this method can be seen in figure 2.6. It can be seen how the crosses that have been misclassified in the first step are given greater weight and the second classifier is able to classify them correctly, but it fails with three minus signs which are given a greater weight for the third step, and are correctly classified in that step. Finally, the three classifiers are weighted to create a final robust model which in this easy case is able to classify all the entries correctly.

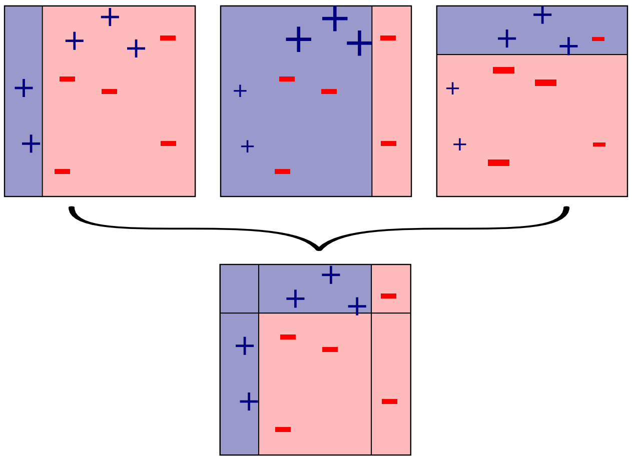


Figure 2.6: *Simple example of the boosting technique.*

In the last couple of years an implementation of the boosting method, XGBoost (extreme gradient boosting), has received great attention as it has been the method used to win various Kaggle[4], an Alphabet (Google) owned web-page where various data science and machine learning competitions are hosted, competitions. In fact, among the 29 challenge winning solutions published at Kaggle during 2015, 17 of them used XGBoost (the second most popular method, deep neural nets, was used in 11 solutions) [57].

[4]Kaggle, The Home of Data Science & Machine Learning

Chapter 3

Determining the Occupancy of a Room

Determining the occupancy of a room is not a trivial task, as it has already been mentioned in chapter 2. Various methods have been applied for this particular task, but all those methods fail in one way or other. In this section a methodology for determining binary occupancy in a room is studied and proposed; such method will use air quality measurement sensors, as they are both cheap and non-intrusive. The goal is to use an already existing sensor network, or at least an easily installable one, to gain additional information that could help in providing efficient and better services.

While research on this topic has already been carried out using this kind of sensors, they have all been experimented in laboratory like closed rooms, where almost everything is under control. In this work a model is proposed based on data of a real building where some residents were living while the data was being recollected. For this particular purpose an elderly caring institution was monitored for one year, the plan of the building is shown in figure 3.1 with the monitored rooms marked with a red dot.

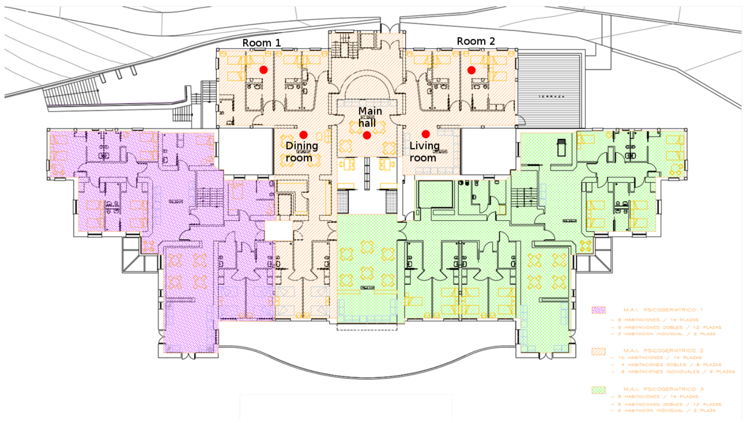


Figure 3.1: *Plan of the monitored building (the monitored rooms are marked with a red dot).*

As it can be seen in figure 3.1 some of the monitored rooms are side by side what implies that some non-desired effects, such as the air transfer between the rooms that will be later shown, can appear. In the following sections the data acquisition, preprocessing and visualization will be shown and discussed, finally creating some classification models to determine presence. In order to be able to perform all those operations the R project's language was used [58].

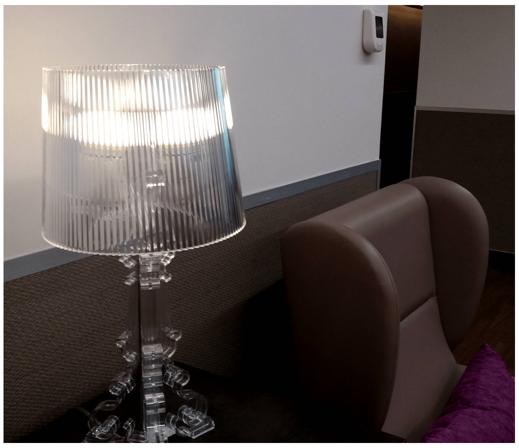
3.1 Data acquisition

The analysis shown in this chapter takes advantage of some data acquired in an elderly caring institution's building, whose plan is shown on figure 3.1. The building was monitored by a variety of sensors including temperature (°C), relative humidity (%RH) and CO_2 (ppm) concentrations, which were placed in five different rooms: the main hall, the living room, the dining room and two bedrooms. The monitoring was primarily performed for HVAC (heating, ventilation and air conditioning) controlling purposes. Even if the acquisition was not primarily intended for the analysis that is shown in this chapter, it will be clear how it is possible to use an already existing infrastructure as the one described to gain additional information about the house usage.

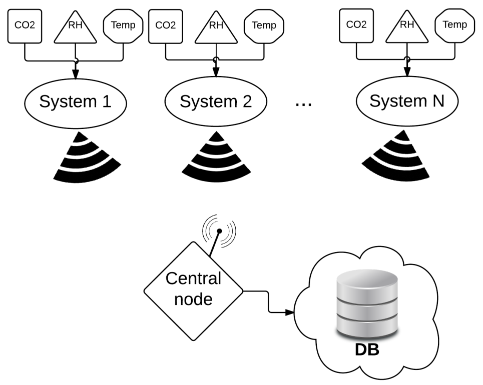
As mentioned, five different sensor systems were deployed in five different rooms, an example can be seen in figure 3.2a, each of those systems consist of temperature, relative humidity and CO_2 concentration sensors. If a sensing equipment of this type is to be deployed in a house, it is important to perform it making as few changes as possible, because of that the deployed sensor systems are powered by ambient light by means of an energy harvesting solar cell. In prolonged periods of low light, security backup batteries provide power to the sensors. As the network consists of a variety of sensors, the communications between the sensor systems and a central node, responsible for uploading the data to the cloud, is performed wirelessly using the open EnOcean® standard protocol (ISO/IEC 14543-3-10) for wireless communications [59] to avoid installing a network of cables, forming a centralized model [60]. An schematic version of the deployed infrastructure can be seen in figure 3.2b.

The monitoring period lasted for an entire year, the obtained dataset consisting of the measurements of the deployed sensors in ten minute intervals during the monitored year. It has to be mentioned that thirteen residents were normally living in the building while the data was being collected, with the monitored rooms being empty in some cases. Due to the condition of the residents, their routine is well defined and maintained through the year, the benefits of this are that having the knowledge of the time table, provided by the institution, it is easier to know where the residents are located at a given hour, so abnormalities are easier to detect. In general, the bedrooms are used during the night (20 p.m to 9 a.m.) and are empty during the day. The residents remain in the living room during the day, except at breakfast, lunch and dinner times when they congregate in the dining room.

Performing the monitoring in an elderly caring institution is somehow related with one of the possible applications of the occupancy monitoring. If it becomes possible to monitor occupancy in a non-intrusive way, the system could be used



(a) *Sensor system located in one of the rooms (up-right corner).*



(b) *Schematic of the data acquisition modules.*

Figure 3.2: *Deployed sensor infrastructure.*

to monitor the daily patterns of an elderly which lives alone; that way it could be possible to detect changes in the everyday patterns that could be an indicative of dementia, which is linked with various diseases.

3.2 Data preprocessing and analysis

Real world problems always require some effort to do before being able to build models with clean data. In that sense, data is usually noisy, it can contain non desired or missing values. Also, in most cases the structure of the data is not the best for performing data analysis or training models for any purpose. The case shown in this chapter is not an exception, and some work has been required to do before performing any operation.

As it has already been mentioned, the sensor systems send the data to a central node for later uploading it to a database in the cloud, such wireless communication has shown to have some problems, as the communications have sometimes got lost. Such lost in the communication system has resulted in some non-desired values. In those cases in which the communication is lost the central node uploads the last received value, so in periods in which the signal has got lost constant values are saved.

In order to remove those totally constant values the standard deviation of all the variables was checked. It could be observed how the metric *sd/mean* was greater than 0.004 in all cases except in those days in which the values were totally constant. As such phenomena only happens in the first two or three days, those values were removed from the dataset using the *sd/mean* criteria.

Another common issue is that of non-desired values. Those could be result of an incorrect measurement of a sensor, a fail in the communication, or also produced by other phenomenon different to the normal expected use. In the obtained dataset some values of such kind where recorded. The number of entries with non-desired values was small, mainly some high temperature recordings of above 50 degrees Celsius, that obviously shows something strange happened. As the number of entries with non-desired data was small those values were removed.

With this preprocessing, the dataset was considered to be clean of non-desired values. The next step was to visualize the data itself in order to get an insight into what was measured and to be able to find patterns that could lead to build more robust models.

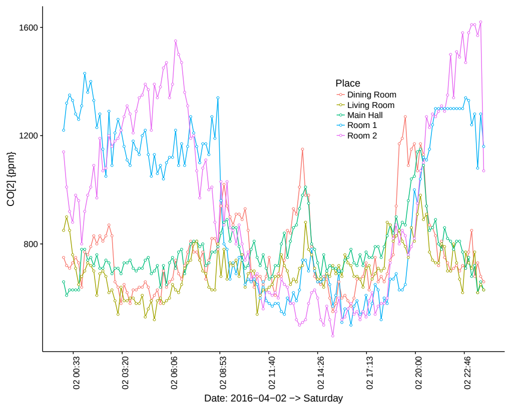
3.3 Data visualization

The dataset obtained after an entire year of monitoring is quite extensive, so visualizing the numbers directly is not helpful at all. Making some visualizations was of great help in order to analyse what was on the data. In this section some visualizations are shown, which help to have an idea of the data itself, and at the same time they are helpful as they provided information about some non-desired phenomena later described.

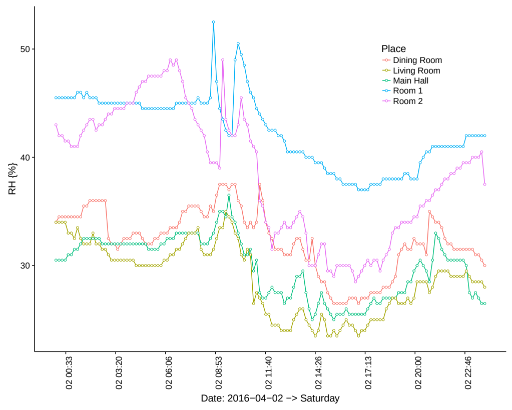
Firstly, the dataset was plotted separating the values by day, each of the plots having one line per monitored room. The aim with these visualizations was to get a first look at the data, with the hope that some differences related to the knowledge about the time table would appear. Some of those plots are shown as example in figure 3.3.

The plots in the figure 3.3 show the data for the 2nd of April 2016 as it is quite representative of the different phenomena that can be found in the data. Figure 3.3a shows the CO_2 measurements during the day in each of the monitored rooms. If we observe the evolution in the rooms 1 and 2, which are bedrooms, during the first hours of the day, different patterns are found; while the CO_2 concentration continually increases in room 2, the concentration remains more stable, and it sometimes decreases, in room 1 what makes suspect that the door was opened by the institution's staff, and because of that the air could flow to the adjoining rooms. The personal staff were asked if they occasionally open some bedroom doors at night, the answer was affirmative implying that the previous assumptions could be right, but that phenomenon should also be monitored to be completely sure, this will be proposed for future work.

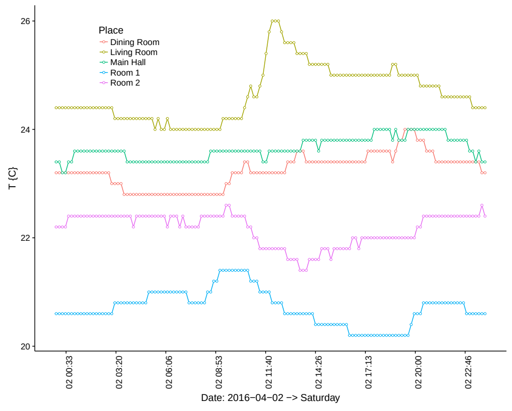
Another pattern that could be implied from the daily visualizations is the air flow between the adjoining rooms. The visualizations made possible to observe how the CO_2 concentration in the bedrooms increased when the residents were in a bigger adjoining area. As an example, it was seen how the CO_2 concentration increased in room 1 at lunch and dinner times. This idea of the air flow is thought to be relevant,



(a) CO_2



(b) Relative Humidity



(c) Temperature

Figure 3.3: *Measured data the 2nd of April, 2016.*

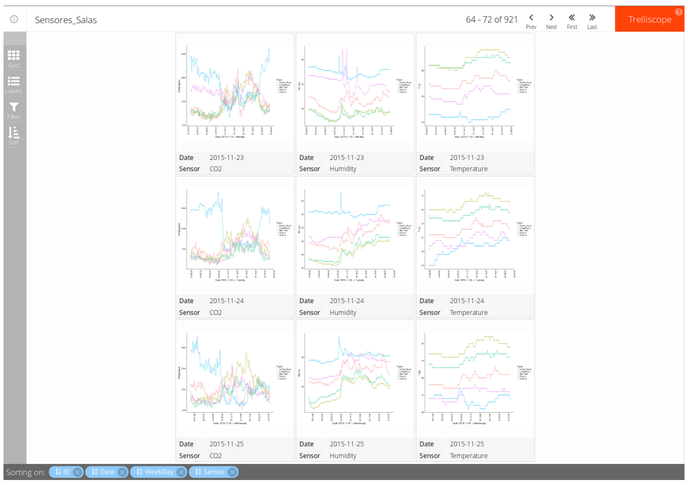
but it should be more precisely measured to determine it correctly.

The opening of the windows in the morning in the rooms, for ventilating purpose, could also be seen in the daily visualizations. That effect is clearly seen in the relative humidity values, where the curves show a clear peak around 9 o'clock, the moment when the caregivers open the windows. In the case of the temperature, the curves do not show great changes, but the variations seem to be somehow related with the human presence at least in the rooms 1 and 2, as the peaks are more or less similar to the ones observed in the CO_2 concentration evolution.

In order to show the plots in a comfortable way the *trelliscopejs*[1] library was used in R. This package allows organizing the visualizations, filtering the days to visualize etc. It is really comfortable working with it especially for time series style data. An example of the trelliscope layout is shown in figure 3.4. With this kind of visualization abnormal days were searched observing how most days show similar patterns.

Another interesting kind of visualization performed was that of a plot containing a box-plot for each hour period and month of the year. These visualizations allowed

[1]trelliscopejs R Package.



Sensores_Salas

64 - 72 of 921

Date 2015-11-23
Sensor CO2

Date 2015-11-23
Sensor Humidity

Date 2015-11-23
Sensor Temperature

Date 2015-11-24
Sensor CO2

Date 2015-11-24
Sensor Humidity

Date 2015-11-24
Sensor Temperature

Date 2015-11-25
Sensor CO2

Date 2015-11-25
Sensor Humidity

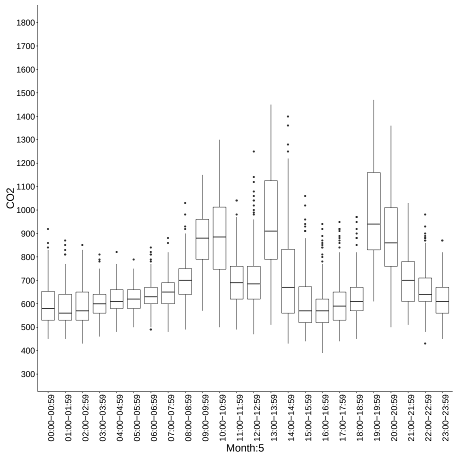
Date 2015-11-25
Sensor Temperature

Sorting on: ID Date WeekDay Sensor

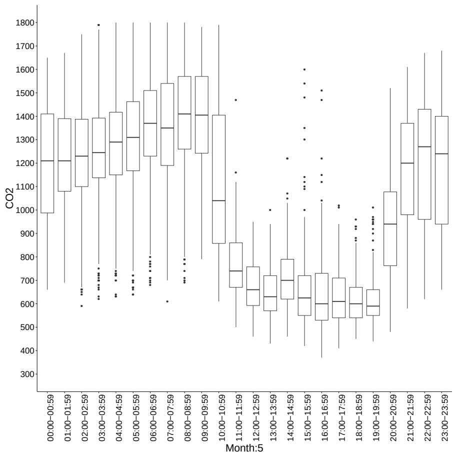
Figure 3.4: *Visualization of the data with trelliscopejs.*

checking if the previously mentioned patterns, which show a correspondence with the time table of the institutions, were available in the data. A plot was obtained for each month of the year in each room to be able to detect if the patterns change during the year.

Figures 3.5a and 3.5b are shown as an example of the figures obtained when performing the visualization of the data; each room shows similar behaviour during the year, the peaks in the CO_2 plots correspond in time with the time table so it is clear that those values are a good indicative of presence. Figure 3.5a shows the evolution of the CO_2 concentration in the dining room while figure 3.5b shows the evolution in room 2, the other variables are not shown as the plots do not provide any interesting information. The patterns in both rooms are completely different, the dining room shows three clear peaks corresponding to breakfast, lunch and dinner times, while room 2 is expected to be used only at nights and that is what can be seen in figure 3.5b where the CO_2 concentration shows a continuous increase during the night while the values are small during the day which agrees with the expected behaviour as the bedrooms are usually only used at nights.



(a) *Box-plot of measurements per hour in the dining room during May.*



(b) *Box-plot of measurements per hour in room 2 during May.*

Figure 3.5: *Patterns in the CO_2 values.*

An idea of the evolution of the variables and some of the patterns that appear on the data has been shown by now. In order to have a more global idea of the data and the relations between them, in figure 3.6 a plot is shown with the correlations between the variables, a histogram for each variable and the projections between the variables. As it can be observed there is no linear relationship between any of them.

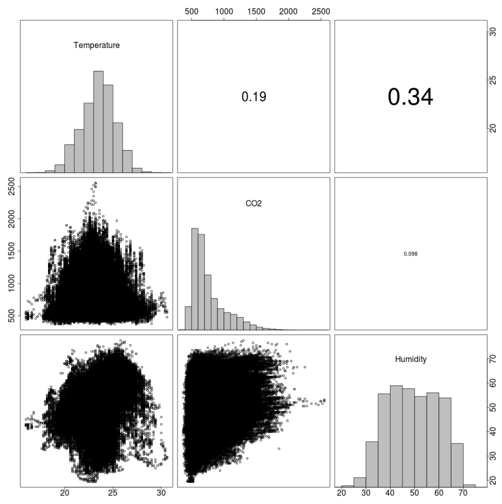


Figure 3.6: *Custom pairs plot showing correlations between variables, histogram of variables and the projections between them.*

3.4 Model building

After preprocessing the data, a clean dataset was obtained. Such dataset was used to train some models based in the previously described supervised learning techniques in order to determine the occupancy of a room based in the temperature, humidity and temperature data. The models were built using the R language and the *caret* package [61]. The caret package provides with lots of methods that can be used out of the box. In this particular case the problem was a classification problem as the aim was to determine binary presence in a room. Such binary presence was labelled by the elderly caring institution's staff, the labelling was not performed directly, instead the institution's staff noted down any incidence that happened, such as a resident going out of his/her room during the night or when one of them goes to the bedroom for having a nap. The binary presence was obtained after crossing the timetable and the incidences.

Even if the original data base contained information about five different rooms, taking into account that the data visualizations showed similar behaviour in all the rooms, the models were built only for the dining room. Such decision was taken because it is the room that seems most difficult to model as it shows the most rapid changes, and if it is possible to determine the binary occupancy there, it should be possible to perform a similar operation for the rest of the rooms.

In order to equalize the weight of the variables, which is necessary for some models, for example artificial neural networks (ANN), the independent variables (temperature, CO_2 level and relative humidity) were mapped to a [0,1] range. Taking into account the nature of the dinning room, it is only occupied at specific moments of the day, remaining empty excepting breakfast, lunch and dinner times. Such class imbalance problem was avoided by limiting the hour intervals, in that sense the database was reduced to maintain only a couple of hours during lunch and dinner (13:00 to 14:00 and 19:00 to 20:00), and another two hours out of that interval, that way the dataset used to train the models contains a very similar number of both classes.

As it has already been mentioned, the designed system should be able to determine if there is presence in a room based in the locally available data about the indoor air quality. With that design in mind three input values were passed to the models as input to determine the binary presence as output, the flow diagram of the system is available in figure 3.7.

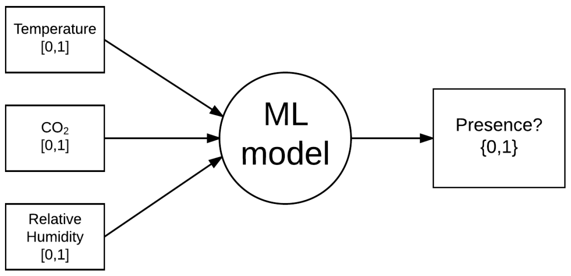


Figure 3.7: *Flow diagram of the presence detection system.*

For such particular purpose a bunch of classifiers was assembled and trained. The models that were used to build the classifiers include classical classification trees, the gradient boosting method, the multiclass Adaboost with bagging, the C5.0 classification tree, support vector machines, quadratic discriminant analysis, and neural networks with one principal component analysis step. A Linux based server, with 8 cores running in parallel and 8 GB of memory available, was used to train and test the models.

Those models usually require fixing the values of a sort of hyper-parameters. Using correct values of such hyper-parameters can sometimes improve the results obtained significantly. For such purpose a grid search was performed to obtain the values which result in obtaining higher values of accuracy. All this process was performed by using a 10-fold cross validation.

3.5 Results and discussion

A bunch of classifiers was assembled and trained in order to estimate the occupancy of a room based on three variables: CO_2 concentration, temperature and relative humidity. As an example of the capability of the models that were trained, figure 3.8 shows the performance of different models, by means of accuracy and Cohen's kappa coefficient, which relates the obtained and expected accuracy of qualitative (categorical) items when validated against a dataset that was not previously seen during the training phase. It can be seen that, in the case of the dining-room, the model that performs best is the one based in support vector machines. In general, it can be said that it performs noticeably better than the classical lineal models, ctree2 in the case of trees or the C5.0 rules. Multiple classifier methods, such as random forest (RF), the extreme gradient boosting (xgbtree) and Adaboost with bagging (Adabag) also obtain remarkable accuracy values.

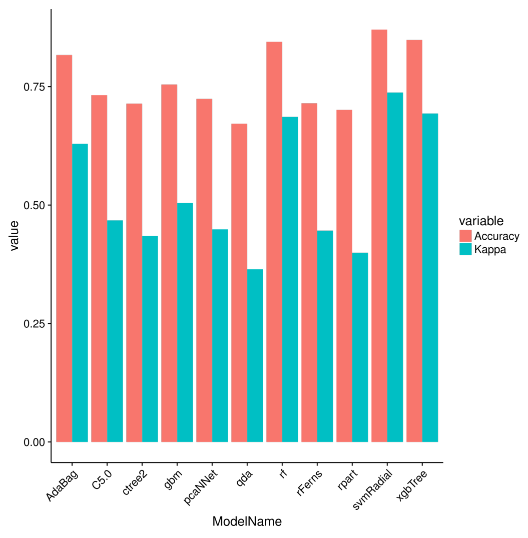


Figure 3.8: *Accuracy and Kappa values obtained with the models.*

In order to better understand the predictions of the models, figure 3.9 shows the ground truth in the dining room, the predictions done by the SVM classifier and the differences between them for a given day. Some errors do appear in the predictions, as the one after dinner in this case, but in general the predictions are acceptable. Even if the models are not completely accurate, it can be seen how the predictions make sense, showing that binary occupancy can be determined by using indoor air quality data.

In general, the best models achieve accuracy values around 80%, which are quite good considering that only 3 variables are used to make the predictions, even more, some of the phenomena visualized have not been considered in this first approach. The results show that it is possible to estimate binary occupancy with the data used.

This study and the results shown are a first step towards using indoor climate data to determine occupancy. It has been valid to test such hypothesis in a real use building, but at the same time some limitations were observed. Those limitations include the air flow between adjoining rooms and not being able to measure rapid changes. Taking those limitations into account further research will focus in applying modelling techniques to add phenomena like the air flow, window opening, etc. On the other side, the fast changes limitation will be fixed by adding PIR sensors,

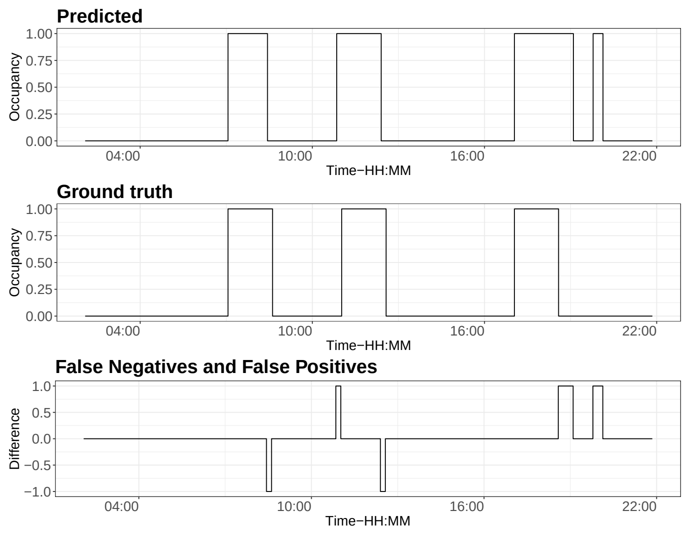


Figure 3.9: *Predictions, ground truth and differences with the SVM model for the dining room.*

thus, we hope that a combination of both, climate sensing and PIR sensors will provide better and more robust results.

Chapter 4

Predicting the Expected Amount of People

In this section the creation of a model to determine the expected amount of people at a specific place and time is shown. The main goal was to obtain a regressor, as accurate as possible, to predict the amount of people expected in a place. If the prediction system is good enough it could lead to develop efficient strategies to provide better services, or it could also be a tool to test if programming an event is a good idea knowing the expected or forecasted conditions. For this purpose, a dataset named "Crowdedness at the Campus Gym", publicly available at Kaggle, was used. The dataset consists of some values measured in a campus gym during a complete academic year. The variables available at the dataset are listed in the following section.

4.1 Data preprocessing and analysis

As the original uploader of the dataset mentions, the aim with this data is "to be able to predict how crowded the gym will be in the future". The dataset contains one measurement every 10 minutes more or less. Each entry in the dataset consists of different measurements that are listed below with their original name in the dataset:

- **number_people.** Amount of people in the gym.
- **date.** Date and hour of the measurement (format: YYYY-MM-DD hh:mm:ss).
- **timestamp.** Timestamp of the moment the data was taken (number of seconds elapsed from midnight).
- **day_of_week.** An integer representing the day, 0 for Monday and 6 for Sunday.
- **is_weekend.** A Boolean value representing if it is weekend (1) or no (0).
- **is_holiday.** A Boolean value representing if it is federal holiday (1) or no (0).

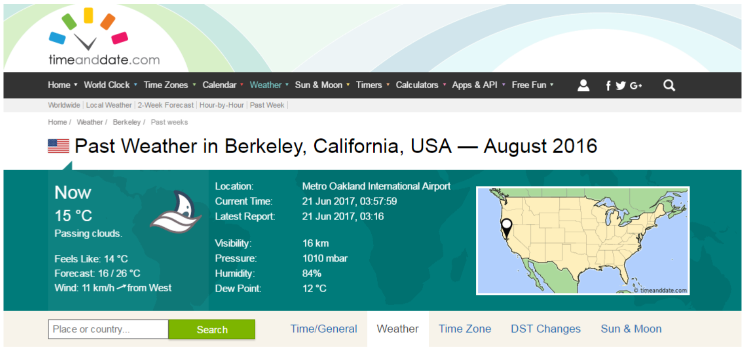
- **temperature.** Temperature at the moment the data was taken in Fahrenheit degrees.
- **is_start_of_semester.** A Boolean value representing if it is the start of the semester (1) or no (0).
- **is_during_semester.** A Boolean value representing if the day is during the semester (1) or no (0).
- **month.** Month at the moment the data was taken, 1 for January and 12 for December.
- **hour.** Hour at the moment the data was taken, 0-23.

As it can be seen, most variables are related to the date and time and no values about the weather are given apart from the temperature. Moreover, some values are irrelevant as they can be obtained from other variables, that is the case of the last two variables, such information is already available in the **date** variable. It can be said that the dataset itself is not rich, as the number of variables is small and some of them can be inferred from others.

To avoid the problem of lack of information in the problem, it was decided to extend the dataset by adding information that could a priori have relevance. It is reasonable to think that the weather conditions or the traffic nearby could affect to the amount of people going to the gym. So, adding that information can lead to obtain more accurate results.

In order to extend the dataset, the location of the gym must be known, but that information was not given in the dataset's information page. However, by searching through the discussion panel at Kaggle such information was given after a user asked for it. The dataset maintainer shared that the gym is located in the University of California Berkeley's campus. Once the location of the gym was obtained, the web was browsed to find information regarding the traffic conditions in the nearby, but it turned out to be impossible to obtain that information. After failing in the search of the traffic conditions, the weather information was located by using the timeanddate.com web-page (see figure 4.1). Such service provides information about Berkeley (California, USA) for every day contained in the original dataset with an entry for each hour (see figure 4.2), because of that web scraping techniques were used to get the data. Web scraping techniques refer to the tools used for extracting data from websites. While this can be done manually by a software user, the term typically refers to automated processes. It is a form of copying, in which specific data is gathered and copied from the web for later retrieval or analysis [62][63]. The process for obtaining the data is explained in appendix A.

As previously mentioned, the original dataset contains an entry every 10 minutes, but the scraped data only provides an entry per hour. Due to that, a strategy had to be chosen in order to mix both datasets. One possible strategy was to take only one value per hour in the original dataset, but that would have led to a huge


timeanddate.com
Home • World Clock • Time Zones • Calendar • Weather • Sun & Moon • Timers • Calculators • Apps & API • Free Fun •
Worldwide | Local Weather | 2-Week Forecast | Hour-by-Hour | Past Week |
Home / Weather / Berkeley / Past weeks
Place or country...
Search
Time/General
Weather
Time Zone
DST Changes
Sun & Moon
© timeanddate.com

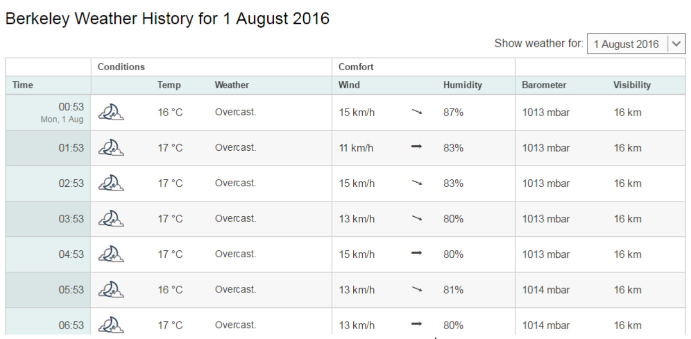
Past Weather in Berkeley, California, USA — August 2016

Now
15 °C
Passing clouds.

Feels Like: 14 °C
Forecast: 16 / 26 °C
Wind: 11 km/h ↗ from West

Location:	Metro Oakland International Airport
Current Time:	21 Jun 2017, 03:57:59
Latest Report:	21 Jun 2017, 03:16
Visibility:	16 km
Pressure:	1010 mbar
Humidity:	84%
Dew Point:	12 °C

Figure 4.1: *Overview of the timeanddate.com webpage.*


Berkeley Weather History for 1 August 2016

Show weather for: 1 August 2016

	Conditions			Comfort				
Time		Temp	Weather	Wind		Humidity	Barometer	Visibility
00:53 Mon, 1 Aug		16 °C	Overcast.	15 km/h	↘	87%	1013 mbar	16 km
01:53		17 °C	Overcast.	11 km/h	→	83%	1013 mbar	16 km
02:53		17 °C	Overcast.	15 km/h	↘	83%	1013 mbar	16 km
03:53		17 °C	Overcast.	13 km/h	↘	80%	1013 mbar	16 km
04:53		17 °C	Overcast.	15 km/h	→	80%	1013 mbar	16 km
05:53		16 °C	Overcast.	13 km/h	↘	81%	1014 mbar	16 km
06:53		17 °C	Overcast.	13 km/h	→	80%	1014 mbar	16 km

Figure 4.2: *Data available for the 1st of August, 2016.*

reduction in the number of instances available, lots of information would have been lost. Another strategy, and the chosen one, was to maintain the original dataset and expand it with the closest weather values, in this way more information was maintained, but consecutive instances will have the same weather values.

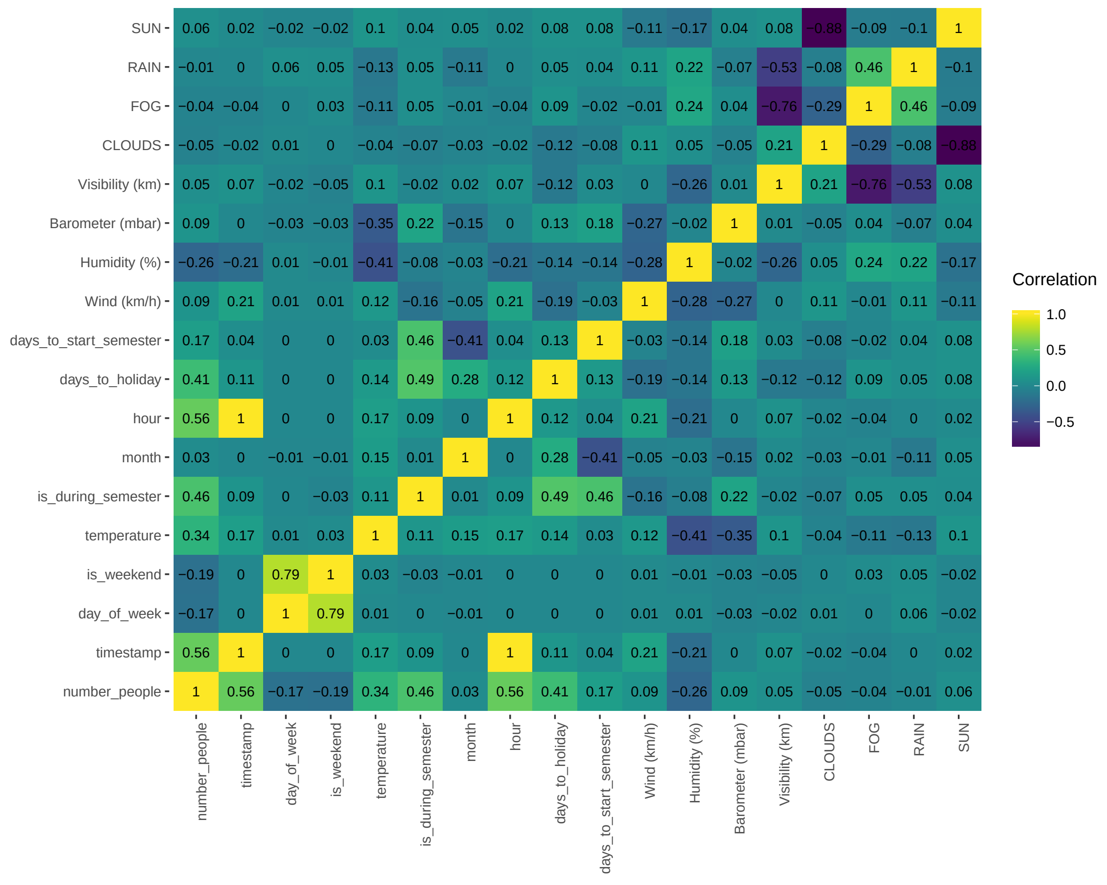
Once the data extending was carried out, the complete dataset was prepared to start with the data analysis, and finally training some models. It has to be mentioned that although temperature values were scraped, they were not taken into account, the original temperature value was maintained instead, because the difference between both values was in general lower than one degree.

The *Weather* variable obtained by scraping the web was a character string, as it can be seen in figure 4.2, so it could not be maintained in that way for building the models. By analysing the possible appearances, it was observed that they can be classified in five main groups: cloudy, rainy, foggy, thunder storming and sunny. A binary variable was created per class. The number of appearances of thunderstorm was small compared to the size of the dataset, just ten cases, and with the intention of having strong variables, those entries with thunderstorm were considered as rainy.

After obtaining the new dataset the next step was to analyse the data. For such purpose, correlation analysis, outlier finding and simple visualizations were done. In the same way as done in the previous problem the data analysis and the training of the models were performed using the R language [58].

First of all, the integrity of the dataset was checked ensuring that there were no missing values. The existence of outliers was also checked by using the Tukey's range test, with the outliers being defined as those that fall out of the borders defined by the range $[Q_1 - 1.5(Q_3 - Q_1)), Q_3 + 1.5(Q_3 - Q_1)]$ [64]. In this case no data entry could be defined as an outlier using this metric because all the points fell inside the indicated range.

In order to test the relation between the variables, and with the aim of not repeating the same information, a Pearson's correlation test was performed, the results are displayed in the figure 4.3. It can be seen that the variables that show highest correlation with the number of people are the *temperature*, the *is_during_semester*, and specially timestamp and hour variables. The last two were expected to show similar correlation values with the dependant variable, number_people, as they contain similar information; the correlation between them is 1. In order to avoid building models that use the same information twice it was decided to remove the *hour* variable, as *timestamp* is more accurate.

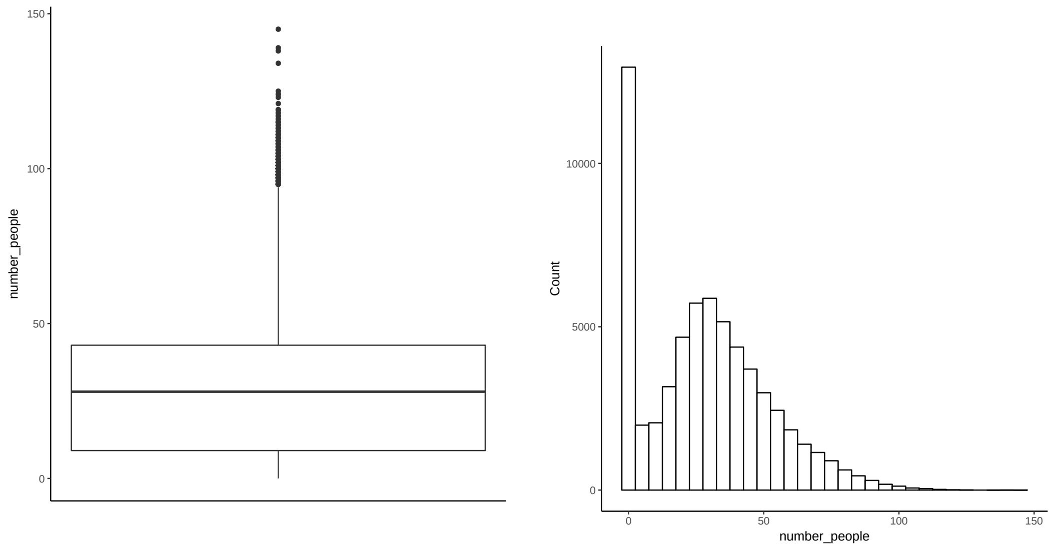


	number_people	timestamp	day_of_week	is_weekend	temperature	is_during_semester	month	hour	days_to_holiday	days_to_start_semester	Wind (km/h)	Humidity (%)	Barometer (mbar)	Visibility (km)	CLOUDS	FOG	RAIN	SUN
SUN	0.06	0.02	−0.02	−0.02	0.1	0.04	0.05	0.02	0.08	0.08	−0.11	−0.17	0.04	0.08	−0.88	−0.09	−0.1	1
RAIN	−0.01	0	0.06	0.05	−0.13	0.05	−0.11	0	0.05	0.04	0.11	0.22	−0.07	−0.53	−0.08	0.46	1	−0.1
FOG	−0.04	−0.04	0	0.03	−0.11	0.05	−0.01	−0.04	0.09	−0.02	−0.01	0.24	0.04	−0.76	−0.29	1	0.46	−0.09
CLOUDS	−0.05	−0.02	0.01	0	−0.04	−0.07	−0.03	−0.02	−0.12	−0.08	0.11	0.05	−0.05	0.21	1	−0.29	−0.08	−0.88
Visibility (km)	0.05	0.07	−0.02	−0.05	0.1	−0.02	0.02	0.07	−0.12	0.03	0	−0.26	0.01	1	0.21	−0.76	−0.53	0.08
Barometer (mbar)	0.09	0	−0.03	−0.03	−0.35	0.22	−0.15	0	0.13	0.18	−0.27	−0.02	1	0.01	−0.05	0.04	−0.07	0.04
Humidity (%)	−0.26	−0.21	0.01	−0.01	−0.41	−0.08	−0.03	−0.21	−0.14	−0.14	−0.28	1	−0.02	−0.26	0.05	0.24	0.22	−0.17
Wind (km/h)	0.09	0.21	0.01	0.01	0.12	−0.16	−0.05	0.21	−0.19	−0.03	1	−0.28	−0.27	0	0.11	−0.01	0.11	−0.11
days_to_start_semester	0.17	0.04	0	0	0.03	0.46	−0.41	0.04	0.13	1	−0.03	−0.14	0.18	0.03	−0.08	−0.02	0.04	0.08
days_to_holiday	0.41	0.11	0	0	0.14	0.49	0.28	0.12	1	0.13	−0.19	−0.14	0.13	−0.12	−0.12	0.09	0.05	0.08
hour	0.56	1	0	0	0.17	0.09	0	1	0.12	0.04	0.21	−0.21	0	0.07	−0.02	−0.04	0	0.02
month	0.03	0	−0.01	−0.01	0.15	0.01	1	0	0.28	−0.41	−0.05	−0.03	−0.15	0.02	−0.03	−0.01	−0.11	0.05
is_during_semester	0.46	0.09	0	−0.03	0.11	1	0.01	0.09	0.49	0.46	−0.16	−0.08	0.22	−0.02	−0.07	0.05	0.05	0.04
temperature	0.34	0.17	0.01	0.03	1	0.11	0.15	0.17	0.14	0.03	0.12	−0.41	−0.35	0.1	−0.04	−0.11	−0.13	0.1
is_weekend	−0.19	0	0.79	1	0.03	−0.03	−0.01	0	0	0	0.01	−0.01	−0.03	−0.05	0	0.03	0.05	−0.02
day_of_week	−0.17	0	1	0.79	0.01	0	−0.01	0	0	0	0.01	0.01	−0.03	−0.02	0.01	0	0.06	−0.02
timestamp	0.56	1	0	0	0.17	0.09	0	1	0.11	0.04	0.21	−0.21	0	0.07	−0.02	−0.04	0	0.02
number_people	1	0.56	−0.17	−0.19	0.34	0.46	0.03	0.56	0.41	0.17	0.09	−0.26	0.09	0.05	−0.05	−0.04	−0.01	0.06

Figure 4.3: *Correlations between the variables in the dataset.*

4.2 Data visualization

The aim of this problem is to be able to predict the number of people at a certain moment of a day under some circumstances. Because of that, it is important to visualise the evolution of the number of people during the time, but first of all, some key features of this variable should be analysed. Firstly, it has to be mentioned that the crowdedness values go from 0 to 145. A boxplot of the number of people in the gym is shown in the plot in 4.4a. It can be seen that most of the points are located in the [0-50] range, but that is greatly influenced by the amount of zeroes in the dataset. The plot in 4.4b shows the count of appearances for the values classified in five people bins ([0-5), [5-10), etc.), it is clear that the number of appearances of values in the [0-5] people range is far greater than the rest of values. In fact, it could be said that the number of people variable almost follows a Gaussian distribution if the [0-5) range is not taken into account.



(a) *Boxplot of the number of people in the gym.*

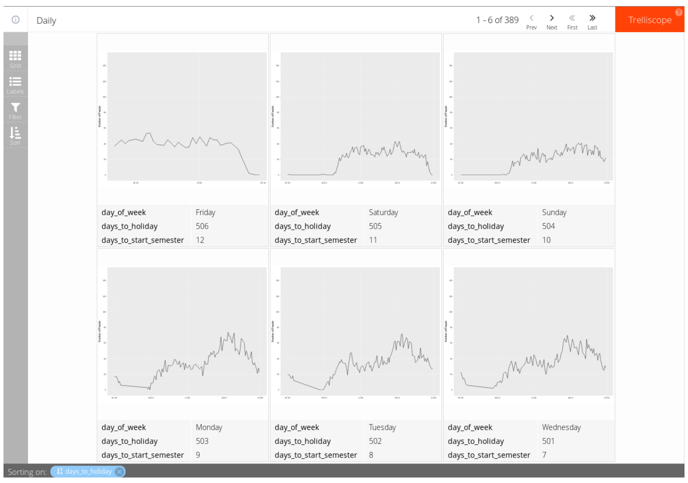
(b) *Count of crowdedness in five people groups ([0-5), [5-10), etc.).*

Figure 4.4: *Visualization of the crowdedeness distribution.*

Visualizing the data is a useful process when analysing it to have an idea of the trends of the variables used, or to visualize some kind of pattern. Different types of visualizations were carried out, such as a day by day visualization, a week by week visualization of the gym attendants, and a boxplot with the occupancy values per hour for every week day.

Daily visualizations can give an idea of how the crowdedness in the gym evolves during a day. This type of visualization made possible to find irregular or especial days in which something strange has happened. In this case the visualizations were also performed with the *trelliscopejs* package in R. The obtained trelliscope is shown in figure 4.5.

The visualization in figure 4.5 allowed observing the evolution in the different days, but it was difficult to extract any useful information from it. Therefore, the data was analysed in a way which could show if different week days show different patterns. Some possible questions could be: Do working days and weekend days



day_of_week	Friday
days_to_holiday	506
days_to_start_semester	12

day_of_week	Saturday
days_to_holiday	505
days_to_start_semester	11

day_of_week	Sunday
days_to_holiday	504
days_to_start_semester	10

day_of_week	Monday
days_to_holiday	503
days_to_start_semester	9

day_of_week	Tuesday
days_to_holiday	502
days_to_start_semester	8

day_of_week	Wednesday
days_to_holiday	501
days_to_start_semester	7

Figure 4.5: *Visualization of the data with trelliscopejs.*

show similar patterns? Are there any differences between Monday and Friday? etc. With the aim of giving an answer to those and more questions other types of plots were obtained. Such plots show each weekday (Monday-Sunday) separated, and each day is divided in hourly ranges, with a boxplot being done for each hour gap for each day. With this visualization it should be possible to observe if different patterns appear for different days.

An example of the mentioned visualizations is shown in figure 4.6. By analysing the seven week days, it seemed that they can be classified in three different bins, an example of each of those bins is displayed in the figure. The first bin contains the days between Monday and Thursday, the second bin is reserved to Fridays and the third contains the weekend days. General days in the first bin and Fridays show similar patterns, but there are differences. For example, days in the first bin show two high peaks, the second one always bigger, but there is a valley between those peaks. On Fridays such valley is not so clear and the second peak is slightly larger, this could be an evidence of the phenomena known as “workout before party” (it is common students going to the gym little before going to party). The weekends show special patterns; first, there are no peaks, the evolution is more stable, and second, the starting and ending times are completely different, while in common work days, Friday inclusive, the activity in the gym ends at 2 o’clock in the morning and starts at 6 o’clock in the morning, during weekends this changes and the activity starts at 8 o’clock in the morning and lasts until midnight.

Because of the phenomena explained and the graphics available in figure 4.6 it seems that the day of the week is important as they are the timestamps. In order to take advantage of the knowledge obtained with the data visualization, the days could be separated in the mentioned bins instead of maintaining the original (0-6) definition, gaining generalization with this change.

It has already been seen that the timestamps are important and the day of the week also seems to be also a valid factor. By growing in such scale it should be considered analysing if the month is also an important factor or not. For such purpose

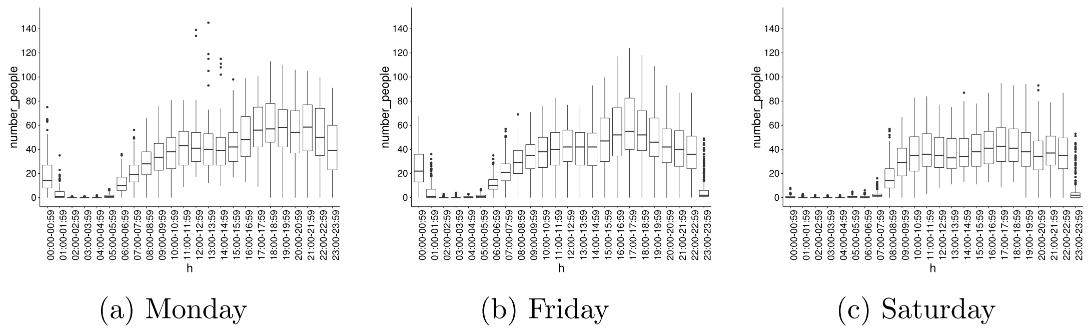


Figure 4.6: *Boxplot per hour for different week days.*

graphical visualizations were again used. In this case a calendar style visualization was used, viewing the mean and maximum occupation for each day. The visualizations are shown in figures 4.7 and 4.8.

If figures 4.7 and 4.8 are analysed, it can be seen that the dataset contains a complete academic year, 2015-2016. It is both remarkable and expected the lower occupation of the gym during Christmas period. The patterns on both figures are quite similar and they offer similar information, in fact it can be seen how the first months are the ones that show highest occupations, while the occupation levels are clearly inferior in the last months, this is probably influenced by the final exams and summer holidays. So, it is clear that the month variable also provides some useful information.

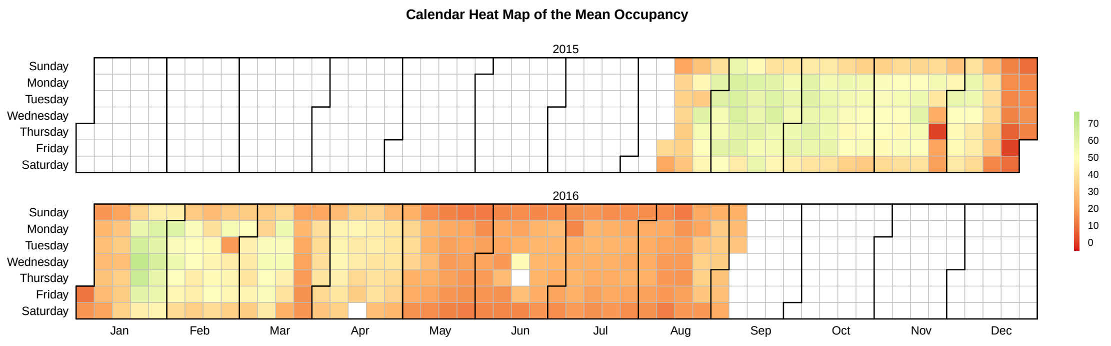


Figure 4.7: *Evolution of the mean crowdedness during the years.*

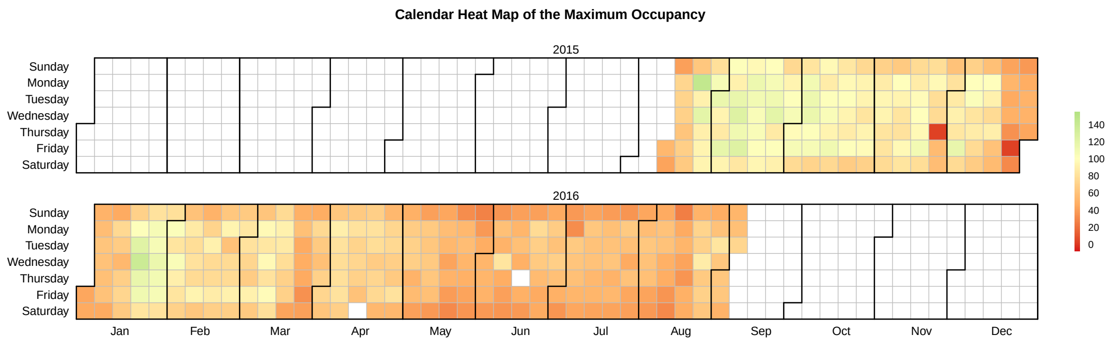


Figure 4.8: *Evolution of the maximum crowdedness during the years.*

4.3 Model building

Usually when training supervised learning models two partitions are created in the dataset, one for training and other for testing. In this case however, due to the nature of the data, the learning and testing of the models was not done only with a train and a test partition, but some data (the last 1000 points) were saved for being able to visualize the evolution of the predictions alongside the real evolution. So, the learning and testing processes were performed by using three data partitions, a typical combination of train and test set, and an extra test set containing the last 1000 points.

In order to divide the dataset, the first thing done was to reserve the last 1000 points as mentioned before. The training and testing sets were randomly obtained from the remaining data, reserving 75% for training and the rest for testing. The learning of the models was carried out by performing a 10-fold cross validation and the test set was used to check the results obtained by the models in not previously seen data. In order to equilibrate the weight of the variables, they were all normalized to a [0,1] range, excepting the variable to predict that was maintained in its original scale.

The learning of the models was performed in a Linux based server using 8 cores working in parallel and 8 GB of memory available. For such purpose the R project's language [58] was used with the *caret* package [61]. The models were trained using different methods: support vector machines, gradient boosting, artificial neural networks and random forest.

Some of the models mentioned have some hyper-parameters that need to be fixed, and the selection of those hyper-parameters is critical as changes in those values can lead to more or less accurate results. R's caret package allows to find the best values in a specified range by performing a grid search, which avoids manually finding the values, but at the same time requires greater computational effort as the possible values are checked one by one, this is especially hard if the defined grid is big. The models have been trained in order to minimize the root mean squared error (RMSE) value.

Different models were trained by using different datasets to compare the results. Three different datasets were used: original data, preprocessed data with weather values, and preprocessed data with weather values and days classified in bins.

4.4 Results and discussion

In this section the results obtained, with the models built based on different Machine Learning techniques, are analysed showing the pros and cons of using each of the models trained. In this sense, and as a brief summary, the results obtained are generally good and the best results obtained are better than the ones currently

(September 11, 2017) reported at Kaggle.

As it was mentioned before, the methods used to train were support vector machines, gradient boosting, artificial neural networks and random forest. Each model has its advantages and disadvantages in this particular case, but generally speaking random forest and gradient boosting (XGBoost implementation [57]) are the ones able to achieve the best results.

The attention was focused mainly in two metrics, root mean squared error (RMSE) and the coefficient of determination or R^2. With the root mean squared error an idea of the error committed is obtained, while the R^2 metric gives information about the proportion of the variance in the dependent variable, the number of people, that is predictable from the independent variables.

In order to understand the results obtained, and to be able to compare the obtained values with the ones reported at Kaggle a model was trained for the original data with the gradient boosting method. Such model scored a RMSE value of 6.5 people. The value itself is not bad, but taking a look into the results for one day as example, it can be seen how the predictions made show some weaknesses in some cases. An example of the predictions made with such model are shown at figure 4.9.

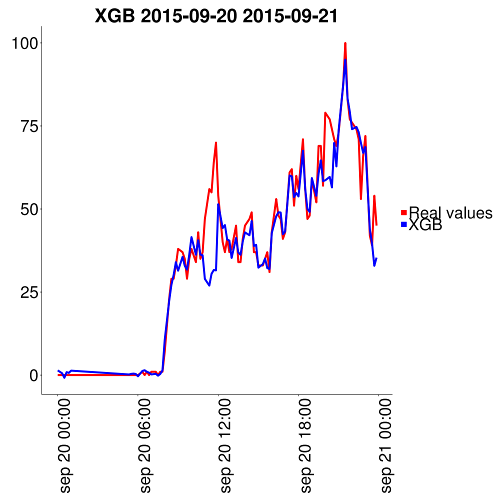


Figure 4.9: *Real evolution and XGB predictions for the 20th of September, 2017.*

Figure 4.9 how the prediction is in general accurate but there are some points in which the predictions are not good. This behaviour is shown in most of the days. Figure 4.10 shows the predictions for the last 1000 points saved for testing. The figure shows that the regression with this data is more or less accurate, but there are some values that do not make sense as a negative value the 3rd of September; it is also notorious the failure in the last peak in the last values. So, this model shows a reasonable behaviour but it has some weaknesses that we hope to be overpassed by adding more relevant data as explained in the previous sections.

This study has focused the attention in adding variables that could have influ-

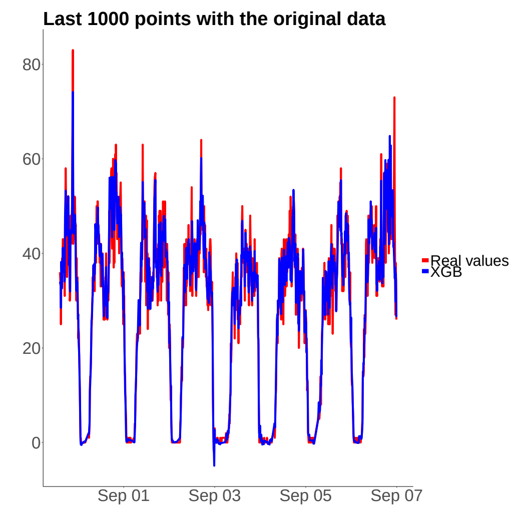


Figure 4.10: *Real evolution and XGB predictions for the last 1000 points in the dataset.*

ence in the amount of people expected at a gym, and to do so some weather related data were obtained. The RMSE metric shows that the preprocessing done and the added variables allow to improve the results. Figure 4.11 shows the RMSE and R^2 values achieved by the trained models, when predicting not previously seen data, in which it can be seen how the root mean squared error is inferior compared to the model trained with the original data, and it can also be seen how the models trained without classifying the days in bins achieve in general lower error values, even if they are quite similar. The plot also shows that the random forest and gradient boosting methods are the ones that obtain better results, the difference between them is negligible.

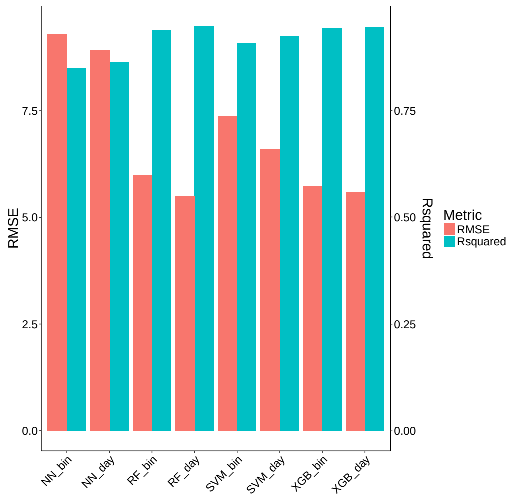
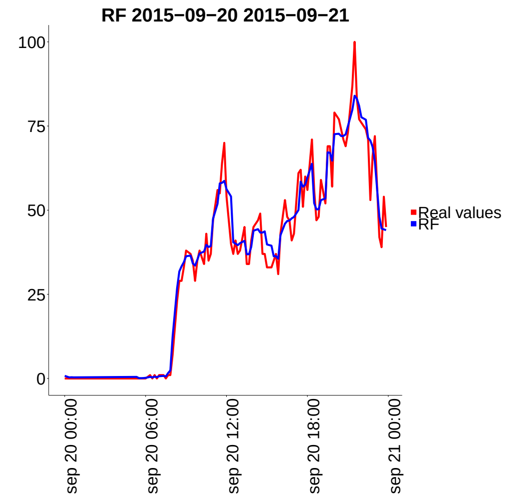


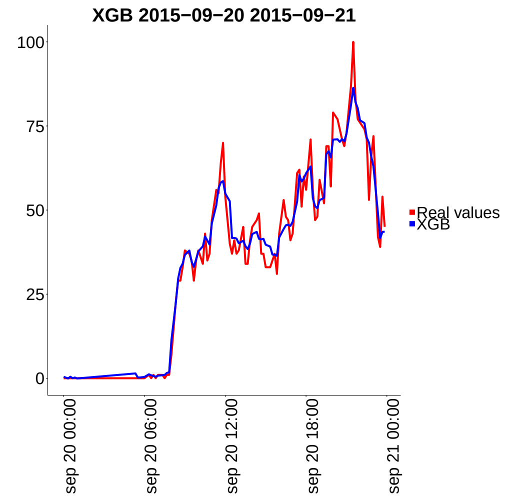
Figure 4.11: *Values for the RMSE and R^2 metrics achieved by the trained models.*

Even if the error rates are smaller in the case in which additional weather variables have been added, the difference in the RMSE value is not big. So, in order to decide the improvements of the models that take such extra information into account different predictions were made obtaining the predictions of the models for a series of days, and also for the last 1000 points saved in previous steps. As the results obtained with the models that maintain the days in its original form and the ones in which the days were classified in bins are very similar, only the ones with the original day variable are shown, because the error rates are slightly inferior.

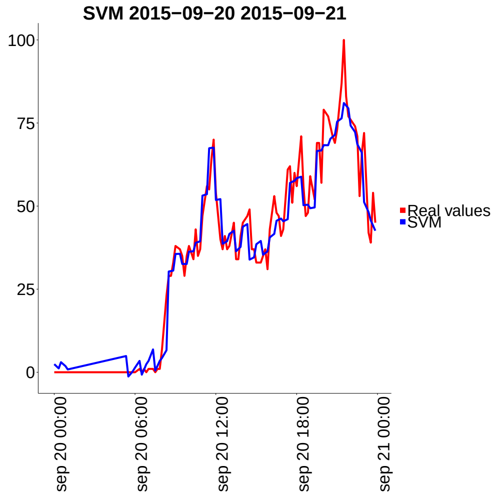
Firstly, plots in figure 4.12 show the predictions made by the models for a specific day, the same day that was tested and shown in figure 4.9. In general, the predictions with the random forest (RF, plot 4.12a) and the gradient boosting method (XGB, plot 4.12b) are accurate, perhaps they are a bit less accurate than the predictions made with the original dataset in figure 4.9, but they do not show imortant mistakes as the previous model did. According to the predictions made by the support vector machines (SVM, plot 4.12c) and the neural network (NN, plot 4.12d) the results are poorer, especially in the case of the neural network. SVM shows overall good results in most cases but there are some points in which it fails notoriously.



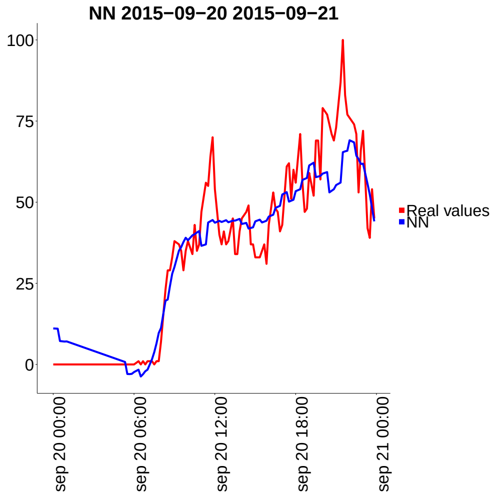
(a) RF



(b) XGB



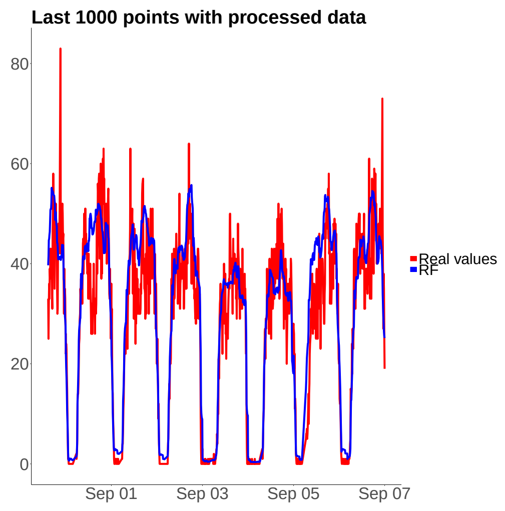
(c) SVM



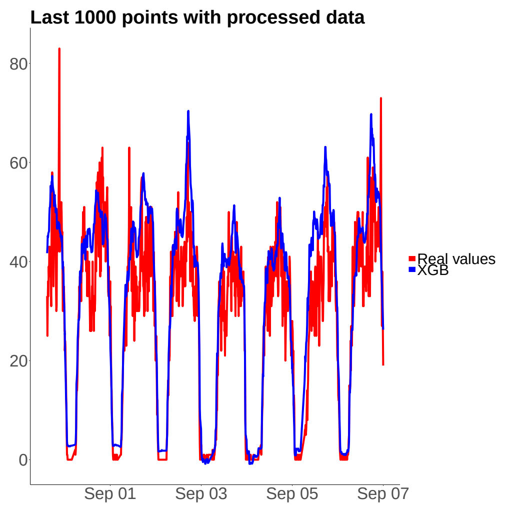
(d) NN

Figure 4.12: *Real data in red and predictions made by each of the models in blue for Sunday, 20th of September 2015.*

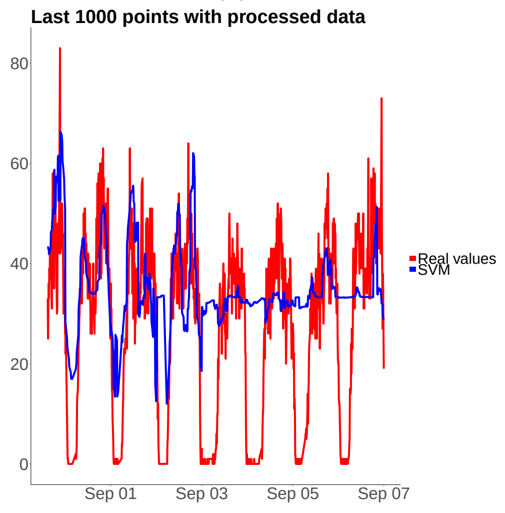
Finally, the predictions made by the models for the last 1000 points, not previously seen by the models, are shown in the plots in figure 4.13. As it can be seen, the support vector machine model and the neural network are not able to model the data correctly and they show non desired behaviours. As mentioned before the SVM model performs good predictions for some cases but there are other ones in which it fails notoriously as it is the case of the last points shown in plot 4.13c. As it was also shown in figure 4.12d the trained neural network has weaknesses the most important one being that it returns negative values that do not make sense. In contrast, random forest and gradient boosting models show more accurate predictions, there are not so many failures, even if they are not able to respond correctly in the higher points that where correctly predicted by the model trained with the original data. One of the positive sides of these two models is that they do not return negative values in any of the cases predicted, and the predictions are softer.



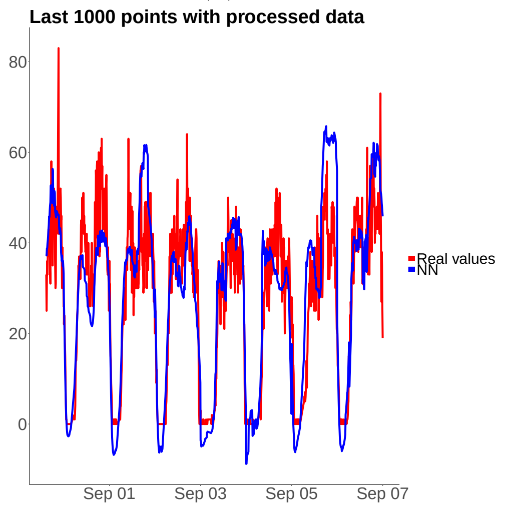
(a) RF



(b) XGB



(c) SVM



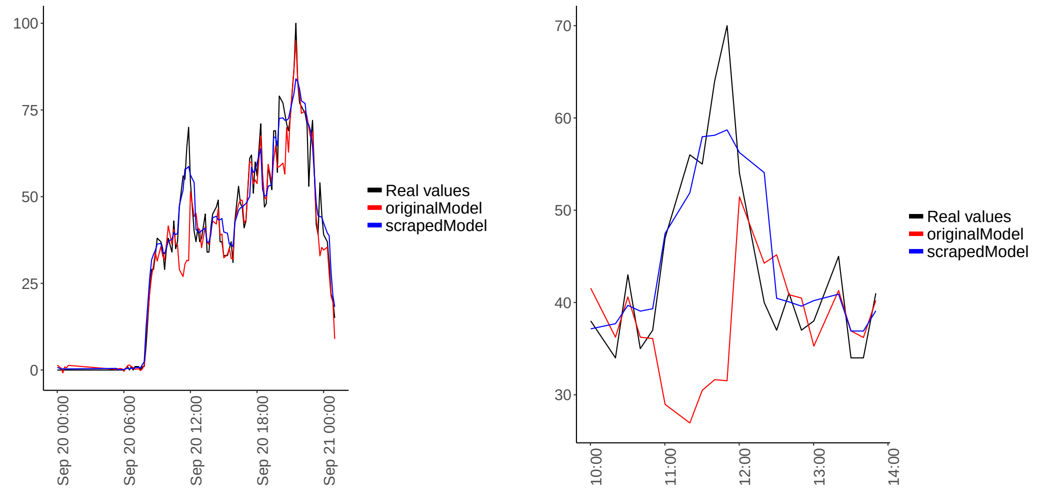
(d) NN

Figure 4.13: *Real data in red and predictions made by each of the models in blue.*

4.4.1 How have the models improved with the added variables?

Previously was mentioned that the root mean squared errors (RMSE) were similar for the models trained with the original data and the ones trained with the original data plus the scraped weather data. In fact, the RMSE values were about 6.5 for the gradient boosting model trained with the original data and the lower RMSE value after adding the scraped variables was obtained by the random forest algorithm, RMSE value of 5.5 people. Taking into account that the range of the number of people variable is [0,145] it does not seem that the improvement is big.

The improvement obtained by adding the weather variables cannot be directly seen in the raw numbers (even if there is an improvement of one person), the improvement is better seen when observing the predictions made by each of the models. Figure 4.9 shows the predictions made by the model trained with the original data, it can be seen how the model is quite accurate in general, but it has some stages in which the predictions fail considerably. If the predictions of the random forest model trained with both the original data and the weather values for the same day are visualized (figure 4.12a), it can be seen that this second model does not fail in the points in which the first model showed some weaknesses. The random forest model is able to model correctly those points, but at the same time the rest of the predictions are less accurate, that is why the RMSE value does not improve too much. The predictions made by the two mentioned models are shown together in figure 4.14a.



(a) *Real values and predictions for the 20th of September 2015.* (b) *Plot on the left hand side zoomed at the greatest error.*

Figure 4.14: *Comparison between the predictions with the different models.*

As mentioned, the improvement in the root mean squared error obtained by adding the weather values is not huge, but the overall behaviour of the models is better as the weaknesses of the initial model are clearly overpassed as it can be seen in figure 4.14b. The difference is more clearly seen in figure 4.15 in which the absolute errors committed by each of the classifiers that are being tested are shown. It can be seen that the maximum errors committed by the models are slightly larger in the case of the model trained with the original data, while the small errors are greater in the case of the models trained after adding the weather values.

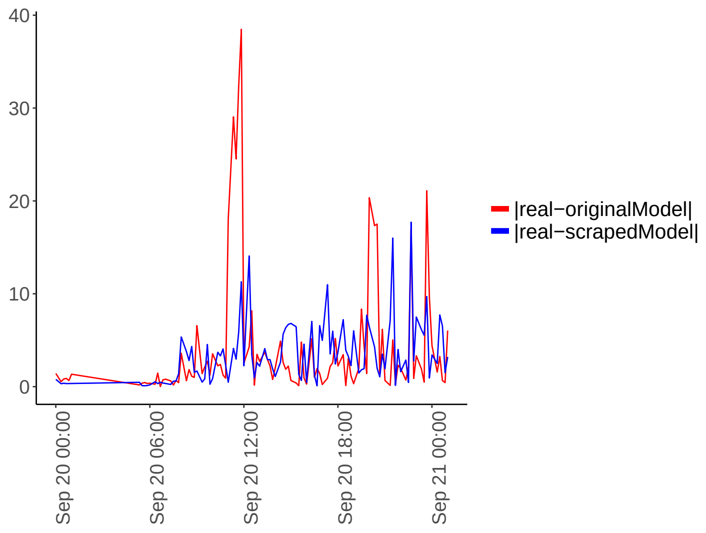


Figure 4.15: *Absolute errors committed by the model trained with the original data and the one trained after adding the weather data.*

Chapter 5

Conclusions

This Thesis addresses two of the problems existing in the Ambient Intelligence field. Due to the nature of the data, both classification and regression techniques were applied. The visualization of the data was extensive providing interesting insights in the case of the occupancy determination problem.

This Master Thesis has been a tool for myself to learn. That is in my opinion the most important thing, the results obtained are secondary for me. I have learned to take a raw dataset and obtain information from it. I have learned how to use graphical visualizations to understand the data, to observe relations, propose changes and test the hypothesis. That way, this thesis has helped me to learn lots of skills and tools that will be in my toolbox from now on.

In the following sections the conclusions for each of the problems worked in this thesis are mentioned.

5.1 Determining the occupancy of a room

Regarding the occupancy determination problem, interesting conclusions have been obtained. Non desired effects such as the window opening and the, thought to be, door opening have been found in the data. Even if a more exhaustive study is required for ensuring that those phenomena occur because of the explained reasons, it has been made clear that determining occupancy of a room in a building is not as straightforward as doing it in a laboratory like environment. Even if several problems have been encountered in the data, the results obtained are relatively good taking the limitations into account.

The results obtained made clear that the occupancy of a room can be determined with the data used, but some feature engineering should be done in order to take into account the visualised phenomena. We hope that those changes will bring more accurate models. The proposed binary detection method is a first step in a research path that is aimed to end in being able to measure the amount of people in a room with these non-intrusive data. For such a purpose the combination of the used sensors with a PIR system will be studied in the future.

As already mentioned, determining the amount of people in a room can be used to create models that can predict the expected amount of people, but we think that the proposed binary occupation determining methodology could be used to develop an ambient assisted living (AAL), a framework that stands for systems that make use of information and communication technologies to develop applications and services for the elderly, application to find changes in the everyday patterns of the elderly inside the house that could be indicative of dementia or other problems. Such research path will be considered in future work.

The work performed in this topic was presented in the 21st International Conference on Circuits, Systems, Communications and Computers (CSCC 2017) held in Crete Island, Greece, with the paper *An IoT-based system that aids learning from human behavior: A potential application for the care of the elderly* (to be published) [25], and an extended version was also published in the WSEAS Transactions on Environment and Development journal titled *Taking advantage of an existing indoor climate monitorization for measuring occupancy* [26].

5.2 Predicting the expected amount of people

According to the second problem, first mention that it has been a pity that the desired data did not arrive on time. Despite of it, the work was carried out with a Kaggle dataset, that way the required techniques were understood, so the analysis of the data that was going to be used will be made with a greater knowledge of the topic. The process with this problem was not straightforward, different regression approaches were tested to create the models, but finally a typical supervised learning configuration was used.

This problem had some issues, because the majority of the provided data where similar, because of that the dataset was extended by using web scraping techniques. The process for scraping the data was entertaining and it helped to obtain better results. Those results are better than the ones that have been reported at Kaggle and it has been shown how the improvement was mainly because of the added weather data. Maybe the models could be more accurate by adding previous amount of people values as input, but that was not desired as it would have made impossible to make predictions in a short-medium term.

Overall, the root mean squared error (RMSE) metric used to test the reliability of the models showed that the results obtained were quite good. But, by taking a deeper insight into the predictions, it could be seen that there were some cases in which the predictions were not very accurate, even if the range was generally good. That suggests that maybe it could be better to use bins in the value to the predict, for example 10 people groups (e.g. [0-10],[10-20], etc.) could be used. That way the problem will not be a classical regression problem, but a classification one instead.

Resolving this problem has shown me how the classification and regression techniques show different behaviours, even if the methodology and the techniques used are similar. I have been able to see how a single metric does not provide all the information about the results of the models, but it is interesting to have a greater insight into the results to determine possible failures and weaknesses of the models.

Bibliography

[1] Juan Carlos Augusto and Paul McCullagh. Ambient intelligence: Concepts and applications. *Computer Science and Information Systems*, 4(1):1–27, 2007.

[2] Hideyuki Nakashima, Hamid Aghajan, and Juan Carlos Augusto. *Handbook of Ambient Intelligence and Smart Environments*. Springer-Verlag New York Inc, 2009.

[3] Sébastien Lucas. Practical applications of Artificial intelligence in Architecture, 2017.

[4] Annalisa Cocchia. Smart and digital city: A systematic literature review. In *Smart city*, pages 13–43. Springer, 2014.

[5] Robert G Hollands. Will the real smart city please stand up? Intelligent, progressive or entrepreneurial? *City*, 12(3):303–320, 2008.

[6] Hatem Ben Sta. Quality and the efficiency of data in "Smart-Cities". *Future Generation Computer Systems*, 2016.

[7] Rob Kitchin. The real-time city? Big data and smart urbanism. *GeoJournal*, 79(1):1–14, 2014.

[8] B Bowerman, J Braverman, J Taylor, H Todosow, and U Von Wimmersperg. The vision of a smart city. In *2nd International Life Extension Technology Workshop, Paris*, volume 28, 2000.

[9] Kehua Su, Jie Li, and Hongbo Fu. Smart city and the applications. In *Electronics, Communications and Control (ICECC), 2011 International Conference on*, pages 1028–1031. IEEE, 2011.

[10] Swarnava Dey, Ankur Chakraborty, Soumitra Naskar, and Prateep Misra. Smart city surveillance: Leveraging benefits of cloud data stores. In *Local Computer Networks Workshops (LCN Workshops), 2012 IEEE 37th Conference on*, pages 868–876. IEEE, 2012.

[11] Paolo Neirotti, Alberto De Marco, Anna Corinna Cagliano, Giulio Mangano, and Francesco Scorrano. Current trends in Smart City initiatives: Some stylised facts. *Cities*, 38:25–36, 2014.

[12] Kemal Akkaya, Ismail Guvenc, Ramazan Aygun, Nezih Pala, and Abdullah Kadri. IoT-based occupancy monitoring techniques for energy-efficient smart buildings. In *Wireless Communications and Networking Conference Workshops (WCNCW), 2015 IEEE*, pages 58–63. IEEE, 2015.

[13] Timilehin Labeodan, Wim Zeiler, Gert Boxem, and Yang Zhao. Occupancy measurement in commercial office buildings for demand-driven control applications—A survey and detection system evaluation. *Energy and Buildings*, 93:303–314, 2015.

[14] Yannick Benezeth, Hélene Laurent, Bruno Emile, and Christophe Rosenberger. Towards a sensor for detecting human presence and characterizing activity. *Energy and Buildings*, 43(2):305–314, 2011.

[15] Varick L Erickson, Stefan Achleitner, and Alberto E Cerpa. POEM: Power-efficient occupancy-based energy management system. In *Proceedings of the 12th international conference on Information processing in sensor networks*, pages 203–216. ACM, 2013.

[16] Giovanni Diraco, Alessandro Leone, and Pietro Siciliano. People occupancy detection and profiling with 3D depth sensors for building energy management. *Energy and Buildings*, 92:246–266, 2015.

[17] Kazuhiko Hashimoto, Nobuyuki Yoshiike, and Katsuya Morinaka. Human occupancy detection method and system for implementing the same, 1997.

[18] X Guo, D K Tiller, G P Henze, and C E Waters. The performance of occupancy-based lighting control systems: A review. *Lighting Research & Technology*, 42(4):415–431, 2010.

[19] Alka R Kaushik and B G Celler. Characterization of PIR detector for monitoring occupancy patterns and functional health status of elderly people living alone at home. *Technology and Health Care*, 15(4):273–288, 2007.

[20] Ming Jin, Nikolaos Bekiaris-Liberis, Kevin Weekly, Costas Spanos, and Alexandre Bayen. Sensing by proxy: Occupancy detection based on indoor CO2 concentration. *UBICOMM 2015*, page 14, 2015.

[21] Qian Huang and Chen Mao. Occupancy estimation in smart building using hybrid CO2/light wireless sensor network. *Journal of Applied Sciences and Arts*, 1(2):5, 2017.

[22] Zhenghua Chen, Qingchang Zhu, Mustafa Masood, and Chai Soh Yeng. Environmental Sensors based Occupancy Estimation in Buildings via IHMM-MLR. *IEEE Transactions on Industrial Informatics*, 2017.

[23] Dominic Wörner, Thomas von Bomhard, Marc Röschlin, and Felix Wortmann. Look twice: Uncover hidden information in room climate sensor data. In *Internet of Things (IOT), 2014 International Conference on the*, pages 25–30. IEEE, 2014.

[24] Z. Yang, N. Li, B. Becerik-Gerber, and M. Orosz. A systematic approach to occupancy modeling in ambient sensor-rich buildings. *Simulation*, 90(8):960–977, 2014.

[25] Unai Saralegui, Miguel Ángel Antón, and Joaquín Ordieres-Meré. An IoT-based system that aids learning from human behavior: A potential application for the care of the elderly. *MATEC Web Conf.*, 2017.

[26] Unai Saralegui, Miguel Ángel Antón, and Joaquín Ordieres-Meré. Taking advantage of an existing indoor climate monitorization for measuring occupancy. *WSEAS Transactions on Environment and Development*, 13:327–334, 2017.

[27] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, 17(3):37, 1996.

[28] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78, 2012.

[29] Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.

[30] Varick L Erickson and Alberto E Cerpa. Occupancy based demand response HVAC control strategy. In *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*, pages 7–12. ACM, 2010.

[31] Varick L Erickson, Miguel Á Carreira-Perpiñán, and Alberto E Cerpa. OBSERVE: Occupancy-based system for efficient reduction of HVAC energy. In *Information Processing in Sensor Networks (IPSN), 2011 10th International Conference on*, pages 258–269. IEEE, 2011.

[32] Nan Li, Gulben Calis, and Burcin Becerik-Gerber. Measuring and monitoring occupancy with an RFID based system for demand-driven HVAC operations. *Automation in construction*, 24:89–99, 2012.

[33] Yuvraj Agarwal, Bharathan Balaji, Rajesh Gupta, Jacob Lyles, Michael Wei, and Thomas Weng. Occupancy-driven energy management for smart building automation. In *Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building*, pages 1–6. ACM, 2010.

[34] Kamin Whitehouse, Juhi Ranjan, Jiakang Lu, Tamim Sookoor, Mehdi Saadat, Carrie Meinberg Burke, Galen Staengl, Anselmo Canfora, and Hossein Haj-Hariri. Towards occupancy-driven heating and cooling. *IEEE Design & Test of Computers*, 29(4):17–25, 2012.

[35] Pauline J Sheldon and Turgut Var. Tourism forecasting: a review of empirical research. *Journal of Forecasting*, 4(2):183–195, 1985.

[36] Stephen F Witt and Christine A Witt. Forecasting tourism demand: A review of empirical research. *International Journal of forecasting*, 11(3):447–475, 1995.

[37] Haiyan Song and Gang Li. Tourism demand modelling and forecasting—A review of recent research. *Tourism management*, 29(2):203–220, 2008.

[38] Rob Law and Norman Au. A neural network model to forecast Japanese demand for travel to Hong Kong. *Tourism Management*, 20(1):89–97, 1999.

[39] Vincent Cho. A comparison of three different approaches to tourist arrival forecasting. *Tourism management*, 24(3):323–330, 2003.

[40] Ping-Feng Pai and Wei-Chiang Hong. An improved neural network model in forecasting arrivals. *Annals of Tourism Research*, 32(4):1138–1141, 2005.

[41] CJSC Burger, M Dohnal, M Kathrada, and R Law. A practitioners guide to time-series methods for tourism demand forecasting—a case study of Durban, South Africa. *Tourism management*, 22(4):403–409, 2001.

[42] Alfonso Palmer, Juan Jose Montano, and Albert Sesé. Designing an artificial neural network for forecasting tourism time series. *Tourism Management*, 27(5):781–790, 2006.

[43] Rong Chen, Chang-Yong Liang, Wei-Chiang Hong, and Dong-Xiao Gu. Forecasting holiday daily tourist flow based on seasonal support vector regression with adaptive genetic algorithm. *Applied Soft Computing*, 26:435–443, 2015.

[44] Xin Xu, Rob Law, Wei Chen, and Lin Tang. Forecasting tourism demand by extracting fuzzy Takagi–Sugeno rules from trained SVMs. *CAAI Transactions on Intelligence Technology*, 1(1):30–42, 2016.

[45] Donald F Specht. A general regression neural network. *IEEE transactions on neural networks*, 2(6):568–576, 1991.

[46] Simon Haykin and Neural Network. A comprehensive foundation. *Neural Networks*, 2(2004):41, 2004.

[47] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[48] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[49] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, sep 1995.

[50] Marti A Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[51] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[52] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21(3):660–674, 1991.

[53] Sebastian Raschka. *Python machine learning*. Packt Publishing Ltd, 2015.

[54] Tin Kam Ho. Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278–282. IEEE, 1995.

[55] Tin Kam Ho. Theory of Multiple Classifier Systems and Its Application to Visual Word Recognition. 1992.

[56] Jerome Friedman, Trevor Hastie, Robert Tibshirani, and Others. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.

[57] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794. ACM, 2016.

[58] R Core Team. R: A Language and Environment for Statistical Computing, 2016.

[59] ISO/IEC Standard 14543-3-10. International standard ISO/IEC 14543-3-10: Wireless Short-Packet (WSP) protocol optimized for energy harvesting—architecture and lower layer protocols, 2012.

[60] Diane J Cook, Juan C Augusto, and Vikramaditya R Jakkula. Ambient intelligence: Technologies, applications, and opportunities. *Pervasive and Mobile Computing*, 5(4):277–298, 2009.

[61] Max Kuhn. Contributions from Jed Wing, Steve Weston, Andre Williams, Chris Keefer, Allan Engelhardt, Tony Cooper, Zachary Mayer, Brenton Kenkel, the R Core Team, Michael Benesty, Reynald Lescarbeau, Andrew Ziem, Luca Scrucca, Yuan Tang, and Can Candan. caret: Classification and Regression Training, 2016.

[62] Eloisa Vargiu and Mirko Urru. Exploiting web scraping in a collaborative filtering-based approach to web advertising. *Artificial Intelligence Research*, 2(1):44, 2012.

[63] Ryan Mitchell. *Web Scraping with Python: Collecting Data from the Modern Web*. " O'Reilly Media, Inc.", 2015.

[64] John W Tukey. Exploratory data analysis. 1977.

Appendix A

Web Scraping for Obtaining Weather Data

In this appendix the process used to scrap the weather data for extending the dataset in the Gym problem, shown in chapter 4, is explained. As shown in such chapter a web-page containing information about the weather data was found, but the data could not be directly downloaded. Taking advantage of the possibility to access to the data, via a web browser, web scraping techniques were used to obtain it. These sorts of techniques refer to the tools used for extracting data from websites typically in an automated way. For this purpose a script, especifically designed for the time-anddate.com web-page, was developed by using the Python language.

The web page itself gives information about the temperature, the weather, the wind speed and direction, the relative humidity level, the atmospheric pressure and the visibility. The page contains a table for each day with values taken every hour (see Figure 4.2).

The first step when performing a web scraping is to analyse the html of the page it is aimed to scrape from. It is necessary to retrieve some key names and types that will help filtering the desired values from the rest of the html. The way to proceed in this case was to take one day as example and to obtain the values for that day. Once that was achieved the code could be reused for the rest of the days, in that way getting the scraping done automatically.

For performing the scraping two python libraries were used: **webkit_server**[1] for performing the web related tasks such as opening web pages, getting html text, executing commands in the web console, etc. while filtering and extracting the desired information was performed by using a popular web scraping python library named **BeautifulSoup**[2]. These libraries provide tools and functions to perform the scraping more easily while providing the freedom necessary to make whatever changes needed.

[1]Webkit-server module: Python bindings for the webkit-server

[2]Beautiful Soap: Python package for parsing HTML and XML documents

Once the scraping for one specific day was made, the commands used were structured inside a function in the proposed approach. Such function named *html2DF* receives the html of the page and returns a data frame with the weather data structured by using the **pandas**[3] library. Another problem found during the scraping was that each day does not have a specific url; instead, when clicking on the desired day a javascript command is executed which dynamically reloads the data in the same page. In order to simulate that clicking automatically the commands required to execute such javascript commands were filtered from the html. In that way when trying to obtain another day's data the command is executed by using the *eval_script* function of the webkit_server library.

With all these steps done the web scraping was performed with two for loops, one for the years and another one for the months. For this purpose the structure of the url name was taken as an advantage, as the required month and year appear clearly on it. As an example the url for August 2016 is `https://www.timeanddate.com/weather/usa/berkeley/historic?month=8&year=2016`. So by changing the year and month values in the url address and executing the necessary javascript commands the data for all the days in the required range were available to obtain.

After scraping the data with the explained procedure, the complete dataset with the values scraped was saved into a csv file. The code used to perform the explained web scraping is available on GitHub.

[3] Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language