Facultad de Informática / Informatika Fakultatea

Detección automática de objetos de interés en imágenes aéreas.

Alumna: Begoña San Martín Alarcia

Director: Abdelmalik Moujahid

Proyecto Fin de Carrera, junio 2014

ÍNDICE

ÍNDICE DE CONTENIDOS

CAPÍTULO 0: INTRODUCCIÓN

En este capítulo se realiza una introducción al proyecto para situar así al lector. En ella se destacan los antecedentes y la organización de la memoria.

0.1 INTRODUCCIÓN

El objetivo principal de este proyecto es la detección automática de objetos de interés en imágenes aéreas de zonas urbanas mediante el uso de descriptores característicos. Los descriptores considerados para esta tarea han sido los histogramas color y los descriptores LBP (Local Binary Pattern), así como la concatenación de ambos dando lugar a descriptores híbridos. El trabajo realizado consta de tres fases:

1. *Etiquetado de las diferentes regiones que componen la imagen:* Partiendo de imágenes ya segmentadas en regiones homogéneas, y conociendo la verdad del terreno, se ha asignado una etiqueta (o label) a cada una de las regiones segmentadas. Si para una región determinada el porcentaje de pixeles correspondientes a la región de interés es mayor al 90%, ésta región es etiquetada como "región de interés". Si éste porcentaje es menor a 1%, la región es etiquetada como "background". En caso contrario, la región es etiquetada como "otros".

2. *Caracterización de las regiones mediante el uso de descriptores:* Se generan todos los histogramas de cada región que compone la imagen utilizando los descriptores descritos anteriormente.

3. *Identificación de las regiones de interés:* Para separar los objetos de interés del fondo de la imagen (o background), se ha llevado a cabo una clasificación supervisada del conjunto de descriptores que caracterizan las diferentes regiones de la imagen segmentada. Los conjuntos de entrenamiento y testeo han sido

obtenidos mediante Validación cruzada (6-fold CrossValidation), utilizando 6 imágenes aéreas. Los histogramas identificando las regiones que pertenecen a la clase “otros” han sido descartados en la fase del aprendizaje, mientras que en la fase de testeo se han considerado todos los histogramas.

0.2ANTECEDENTES

La detección automática de objetos en imágenes consiste en la asignación de una etiqueta a un objeto en base a su contenido. Es un campo que atrae el interés de muchos investigadores y que tiene numerosas aplicaciones en áreas como la recuperación de imágenes basada en contenido, la seguridad y la video-vigilancia, el diagnóstico a partir de imágenes médicas, la visión y navegación de robots, etc.

En las últimas décadas las técnicas de visión por ordenador asisten cada vez más al proceso global de los sistemas de formación geográfica, la gestión de riesgo o la monitorización de regiones urbanas. Por ejemplo, la extracción automática de objetos como edificios, rutas en imágenes aéreas es un campo que ha suscitado el interés de muchas investigaciones. Es bien conocido que la segmentación de edificios en imágenes aéreas es una tarea difícil.

Este problema es generalmente considerado cuando se trata de procesar imágenes aéreas de alto nivel con el fin de extraer información simbólica o numérica. En este contexto, varios métodos han sido propuestos. Una de las técnicas más utilizadas es la segmentación semiautomática que requiere la intervención del usuario para extraer los objetos de interés a partir de una imagen.

Por otra parte, partiendo del paradigma del aprendizaje automático (Maching learning), es deseable la intervención del usuario en la fase de entrenamiento, pero en la fase de testeo la detección y reconocimiento son totalmente automáticos. En este proyecto, proponemos una herramienta que permite detectar los tejados de edificios en imágenes aéreas.

0.3 ORGANIZACIÓN DE LA MEMORIA

La presente memoria está compuesta por una serie de capítulos que se describen a continuación:

- *Capítulo 0:* Es una introducción al proyecto para situar así al lector. En ella se destacan los antecedentes y la organización de la memoria.
- *Capítulo 1:* Detalla el documento de objetivos del proyecto. Se compone por una descripción tanto del proyecto como de los objetivos y una planificación teniendo en cuenta los recursos disponibles, la estimación de costes y un plan de contingencia de riesgos.
- *Capítulo 2:* Describe algunos conceptos de minería de datos, detallando el proceso de extracción de conocimiento, así como los clasificadores utilizados.
- *Capítulo 3:* Introduce los descriptores basados en histogramas color y LBP utilizados para caracterizar las diferentes regiones de una imagen.
- *Capítulo 4:* Describe la detección de regiones de interés y el proceso de detección.
- *Capítulo 5:* Recoge los resultados obtenidos y análisis de los mismos.
- *Capítulo 6:* Detalla de la herramienta informática para la detección automática de los tejados.

Las conclusiones acerca de este proyecto, las referencias bibliográficas y los anexos correspondientes se recogen al final de esta memoria.

CAPÍTULO 1: DOCUMENTO OBJETIVOS DEL PROYECTO

En este capítulo se detalla el documento de objetivos del proyecto. Se compone por una descripción tanto del proyecto como de los objetivos y una planificación teniendo en cuenta los recursos disponibles, la estimación de costes y un plan de contingencia de riesgos.

1.1 DESCRIPCIÓN DEL PROYECTO

El principal objetivo de este proyecto es la detección automática de objetos de interés en imágenes aéreas mediante el uso de descriptores característicos. Concretamente, se trata de identificar de manera automática los tejados de edificios en imágenes.

Los descriptores utilizados para caracterizar las regiones segmentadas de la imagen se basan en histogramas color y descriptores LBP (Local Binary Pattern). Pero, también, se ha considerado la concatenación de estos descriptores dando lugar a descriptores híbridos. Para realizar la clasificación y detección de los objetos de interés, es decir, los tejados, han sido utilizadas herramientas de clasificación supervisada.

Finalmente las diferentes fases que componen este proceso de detección se recogen en una interfaz de usuario desarrollada en Matlab.

1.1.1 OBJETIVOS DEL PROYECTO

Los objetivos personales de la realización de este proyecto son los siguientes:

- Superar de manera exitosa el reto de enfrentarse a un proyecto fin de carrera.
- Poner en práctica los conocimientos adquiridos durante la carrera y consolidarlos.
- Investigar nuevas herramientas para la elaboración del proyecto.

En cuanto a los objetivos de desarrollo:

- Puesta en marcha de la interfaz Matlab que identifica de manera automática los tejados de edificios en imágenes.
- Realizar una interfaz lo más intuitiva posible y de fácil manejo para así minimizar el tiempo de adaptación por parte del usuario a la misma.

1.1.2 ANÁLISIS Y TOMA DE DECISIONES

Modelo de proceso del ciclo de vida: definiciones de alto nivel de las fases por las que transcurren los proyectos de desarrollo de software. No son guías concretas ni detalladas, son una muestra de las dependencias existentes entre las fases.

A la hora de realizar el desarrollo del proyecto se ha elegido seguir un modelo de proceso de ciclo de vida en cascada. Este modelo se define principalmente porque las distintas fases de desarrollo se realizan de una manera secuencial, es decir, cada fase comienza en el punto en el que termina la anterior.

De esta manera, se permite una sencilla evaluación del grado de avance del proyecto facilitando la propia gestión del tiempo. Se debe asumir que desde un principio los requisitos del proyecto serán estables y conocidos.

Este modelo está compuesto por las siguientes etapas:

- *Requisitos:* en esta etapa se analizan las condiciones finales del software que debe cumplir. Es importante decidir lo que se requiere del sistema ya que será aquello a seguir en las siguientes etapas.

- *Diseño:* se descompone y organiza el sistema en actividades que puedan realizarse de manera paralela.

- *Construcción:* en esta etapa se implementan las funciones y algoritmos necesarios para cumplir con los requerimientos previamente marcados en la primera etapa.

- *Pruebas:* se comprueba el correcto funcionamiento de la aplicación antes de ser entregado al usuario final.

- *Mantenimiento:* ofrecer un mantenimiento al usuario final el cual puede necesitar ampliaciones o modificaciones en la aplicación desarrollada.

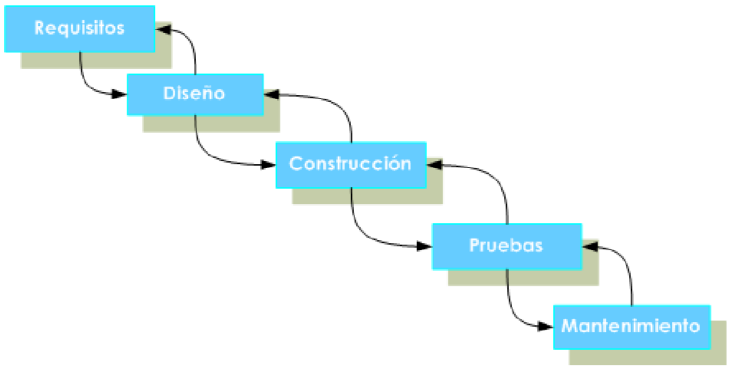

ILUSTRACIÓN 1: MODELO CICLO DE VIDA EN CASCADA

1.2 PLANIFICACIÓN

1.2.1 ESTRUCTURA DE DESCOMPOSICIÓN DEL TRABAJO

La descomposición del trabajo de este proyecto se compone por el tiempo invertido previamente para la planificación y aprendizaje de las tecnologías a utilizar, y estimación del esfuerzo necesario para desarrollar la aplicación.

Fase 0: Planificación.

- *Análisis de Requisitos:* se realiza la recogida de los requisitos necesarios para la finalización correcta del proyecto.
- *Plan de Gestión:* se elabora el plan para el desarrollo del proyecto, estimando todas las actividades y costes.

Fase 1: Aprendizaje de nuevas tecnologías.

- *Aprendizaje de Matlab*: es una herramienta de software matemática con lenguaje propio de programación con la cual se ha realizado la manipulación de imágenes, implementación de algoritmos y diseño de la interfaz.
- *Aprendizaje de Weka:* es una herramienta de software de minería de datos que permite el análisis de conjuntos de información mediante un aprendizaje automático.

Estimación del esfuerzo:

El desarrollo del proyecto consta de diversas fases las cuales se describen a continuación:

Fase2: Etiquetado de las diferentes regiones que componen la imagen.

- *Desarrollo de la interfaz Matlab:* se realiza un prototipo de la interfaz en la cual se introducen las funcionalidades a lo largo de las fases que componen el proyecto.
- *Asignación de labels:* se asignan etiquetas a las regiones segmentadas que componen la imagen. Se han considerado 3 clases, clase 1 correspondiente a la región de interés; la clase 0 indicando el fondo de la imagen (background) y la clase -1 refereriéndose a regiones que comparten tantos pixeles de la región de interés como del background.

Fase 3:Caracterización de las regiones mediante el uso de descriptores.

- *Generación de los histogramas:* se generan los histogramas de cada región que compone la imagen utilizando el descriptor histograma color o la versión híbrida

Fase 4: Identificación de las regiones de interés.

- *Se generan los conjuntos de entrenamiento y testeo:* mediante el uso de Cross Validation, eliminando aquellos histogramas que pertenecen a la clase -1.

Fase 5: Clasificación.

- *Clasificación inteligente de regiones mediante el uso de clasificadores como Knn o SVM.*
- *Asignación de clases:* se asigna a cada región de la imagen el valor de la nueva clase obtenida.

Fase 6: Pruebas y documentación.

- *Realizar pruebas:* se realizan comprobaciones del correcto funcionamiento y se corrigen los posibles errores.
- *Recopilación de documentación:* se buscan fuentes de información serias de donde basar los conocimientos explicados en la memoria para el entendimiento del proyecto.
- *Elaboración de la memoria.*
- *Elaboración de la presentación para la defensa.*

1.2.2ESTIMACIÓN DE COSTES

Se ha realizado una estimación de tiempos en base a antecedentes y estimación de la carga de trabajo. Partiendo de que para la correcta finalización del proyecto serán necesarias alrededor de 350 horas para la planificación, aprendizaje, desarrollo y pruebas. Si la carga de trabajo se realiza de una manera lineal, con una dedicación diaria de6 horas, el total de días necesarios son aproximadamente 60.

	ACTIVIDAD	TIEMPO ESTIMADO
FASE 0	Análisis requisitos	10H
	Plan de gestión	10H
FASE 1	Aprendizaje Matlab	20H
	Aprendizaje Weka	10H
FASE 2	Desarrollar la interfaz Matlab	20H
	Asignar una clase a cada región de la imagen	15H
	Asignar etiquetas a las regiones de la imagen	20H
FASE 3	Generar histogramas de cada región	20H
FASE 4	Análisis datos	15H
	Generar conjuntos de entrenamiento y testeo	30H
FASE 5	Generación de la clasificación	25h
	Asignación de las nuevas clases	25H
FASE 6	Realizar pruebas	10H
	Recopilar documentación	30H
	Elaborar memoria	60H
	Elaborar presentación	30H

ILUSTRACIÓN 2: TABLA DE DEDICACIONES

1.2.3 EDT

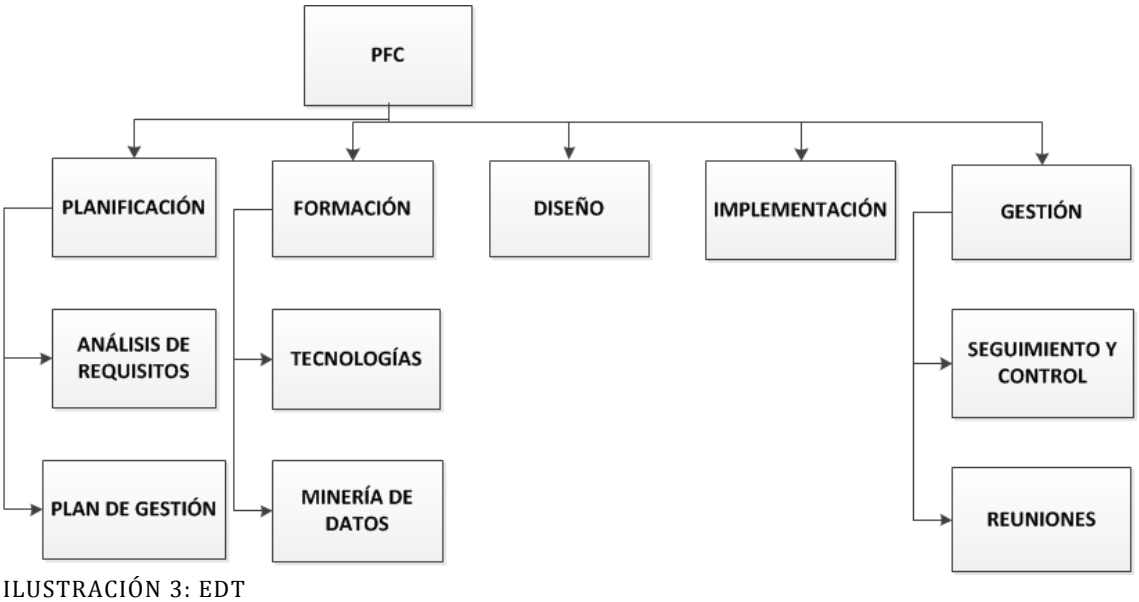

ILUSTRACIÓN 3: EDT

Análisis de Requisitos: se realiza una la recogida de los requisitos necesarios para la finalización correcta del proyecto.

Plan de Gestión: se elabora el plan para el desarrollo del proyecto, estimando todas las actividades y costes.

Tecnologías: formación en las tecnologías a utilizar durante el proyecto, Weka y Matlab.

Minería de Datos: conocimiento en el ámbito de la minería de datos.

Diseño: su función es cubrir las especificaciones que debe cumplir la interfaz, facilitando su manejo, para ello se genera un prototipo que será evaluado.

Implementación de la aplicación: actividades que se deben realizar para obtener el producto final con todas las funcionalidades.

Seguimiento y Control: se monitoriza el desarrollo a lo largo del proyecto y es comparado con la planificación realizada inicialmente. Con ello se busca detectarlas desviaciones y corregirlas.

Reuniones Gestión: reuniones que se realizan con el tutor para visualizar el desarrollo y llegar a acuerdos sobre el proyecto.

1.2.4 DIAGRAMA GANT

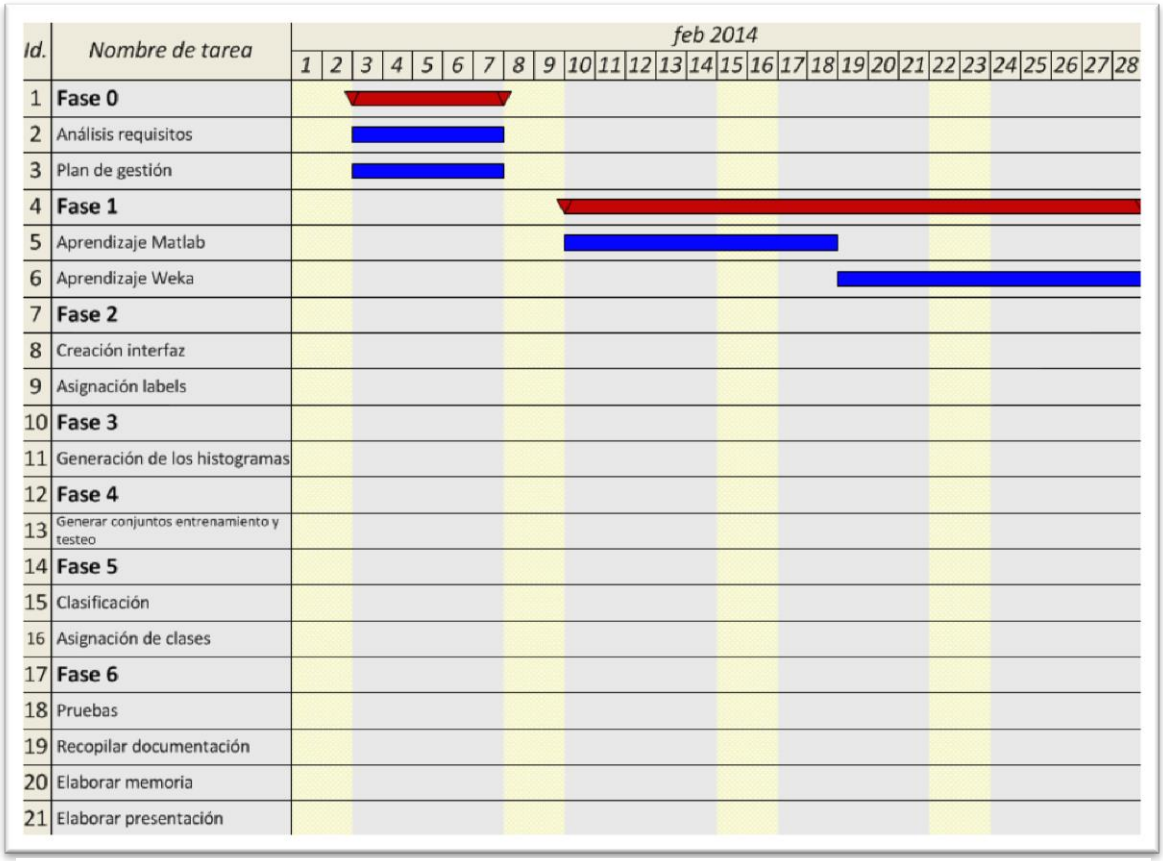

ILUSTRACIÓN 4: DIAGRAMA GANTT FEBRERO

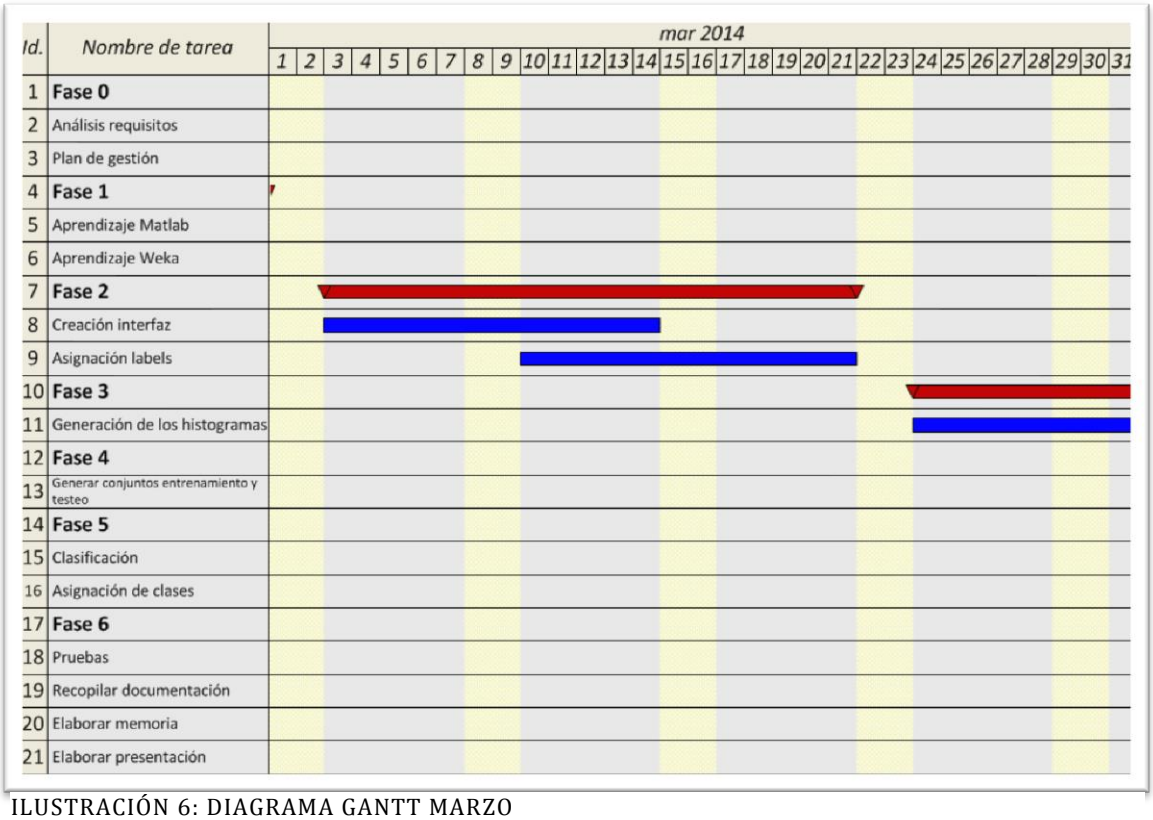

ILUSTRACIÓN 6: DIAGRAMA GANTT MARZO

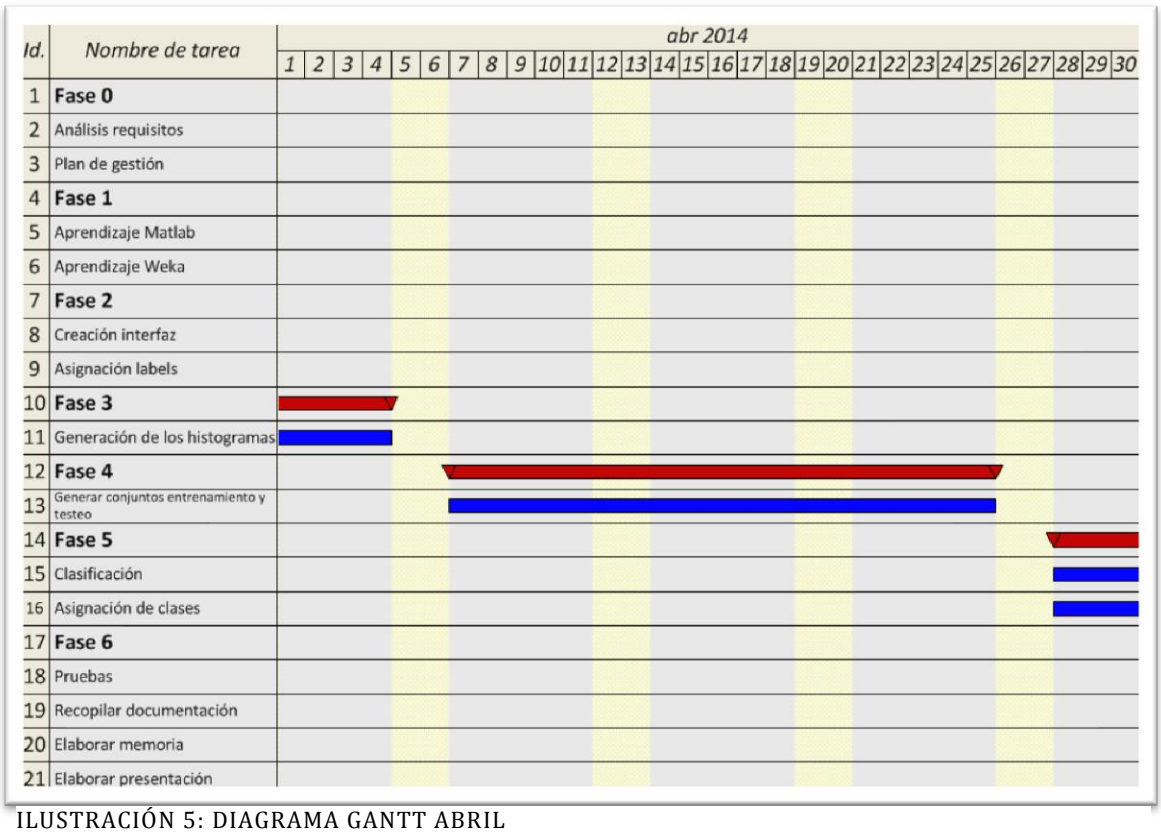

ILUSTRACIÓN 5: DIAGRAMA GANTT ABRIL

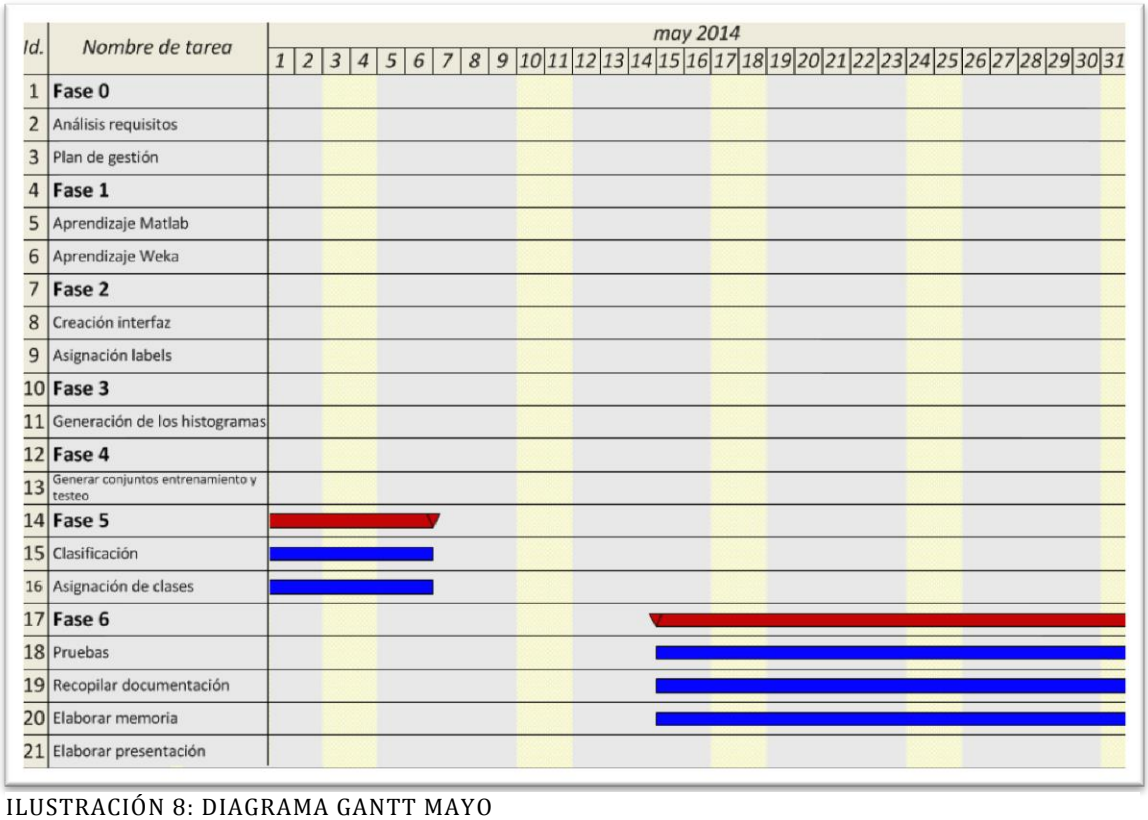

ILUSTRACIÓN 8: DIAGRAMA GANTT MAYO

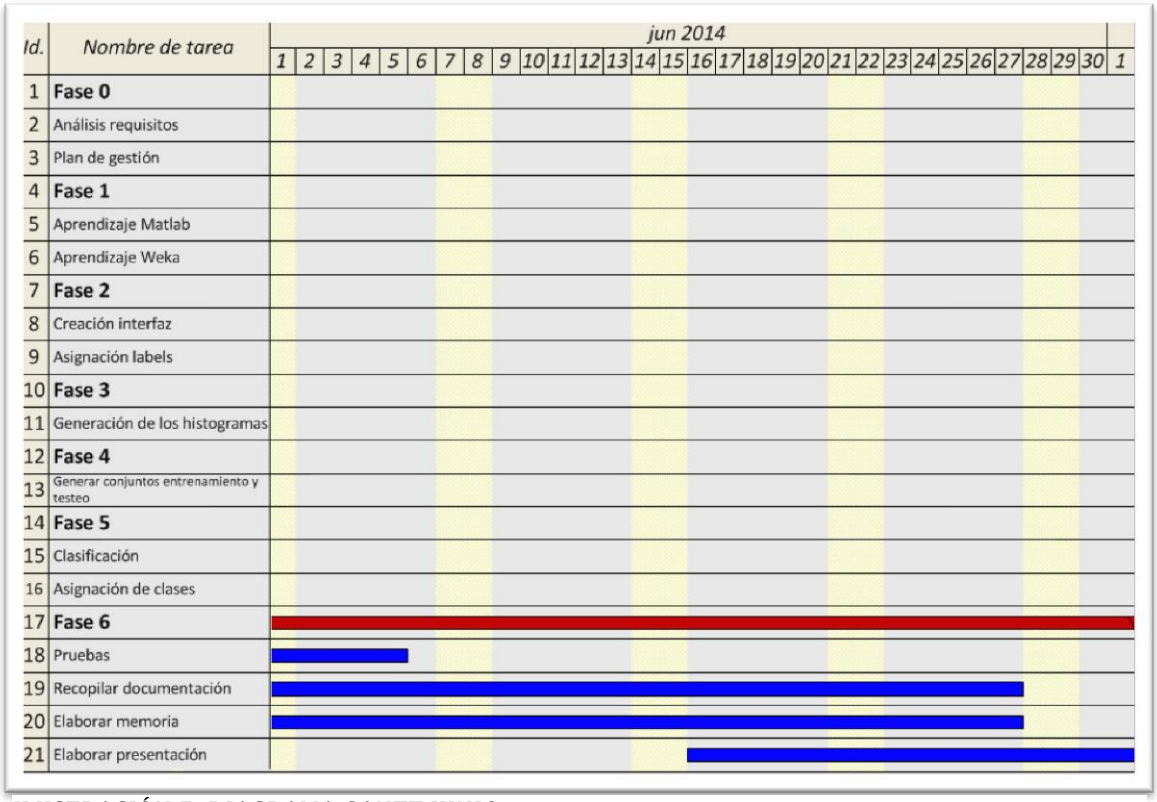

ILUSTRACIÓN 7: DIAGRAMA GANTT JUNIO

1.2.5 RIESGOS

Para minimizar los daños que pueden causar la aparición de riesgos, se ha creado un plan de contingencia en el cual se describen los posibles riesgos a los que enfrentarse y qué decisiones tomar en cada caso.

- *Descripción del riesgo:* costes imprevistos de adquisición de conocimiento.

  Frecuencia: media.

  Gravedad: media.

  Plan de respuesta: planificación exhaustiva al inicio del proyecto de los conocimientos a adquirir.

- *Descripción del riesgo:* retraso en los plazos establecidos.

  Frecuencia: alta.

  Gravedad: media.

  Plan de respuesta: se realizara una replanificación de los tiempos para llevar a cabo exitosamente el proyecto.

- *Descripción del riesgo:* cambios de orientación de especificaciones o elementos del alcance.

  Frecuencia: alta.

  Gravedad: alta.

  Plan de respuesta: planificará el desarrollo con un colchón suficiente para que se pueda asumir un 20% de ampliación o cambios en la especificación del alcance.

- *Descripción del riesgo:* pérdida de archivos del proyecto.

  Frecuencia: media.

  Gravedad: baja.

  Plan de respuesta: se tendrán copias de respaldo tanto en un dispositivo de almacenamiento externo como en un servicio en la nube. A diario se realizará una actualización para evitar la pérdida de información.

- *Descripción del riesgo:* cambios imprevistos en la interfaz de la aplicación.

  Frecuencia: baja.

  Gravedad: alta.

  Plan de respuesta: se hará un análisis para decidir si los cambios que afecten al diseño pueden llevarse a cabo de manera exitosa en los plazos establecidos.

CAPÍTULO 2: CONCEPTOS GENERALES DE MINERÍA DE DATOS

En este capítulo se introduce algunos conceptos de minería de datos, detallando el proceso de extracción de conocimiento, así como los clasificadores utilizados.

2.1MINERÍA DE DATOS

Las nuevas tecnologías han hecho que la recolección de datos se pueda realizar de una manera sencilla, y gracias a la creación de bases de datos estructuradas se ha conseguido poder almacenar los datos de una manera eficaz. Como bien se ha dicho, los datos pueden estar estructurados en bases de datos, pero también existen datos en forma de grafos (World Wide Web, estructuras moleculares, redes sociales,…) y en otro tipo bases de datos (espaciales, temporales y secuencias genéticas).

En la actualidad existe un gran abanico de aplicaciones para la minería de datos. A continuación se nombran algunos de los ámbitos en los que es más común su uso:

- *Financieras:* detección de uso fraudulento de tarjetas de crédito.
- *Comercio:* conocer patrones de compra del cliente para así diseñar mejores campañas de publicidad y aumentar ventas.
- *Seguros:* determinación de clientes potencialmente caros.
- *Educación:* detección de abandono o fracaso escolar.
- *Medicina:* detección de pacientes con riesgo a sufrir una patología concreta.

La minería de datos se basa en la aplicación de distintos algoritmos sobre un conjunto de datos, con el fin de obtener unos resultados acordes con el problema a resolver. Para ello no existe una técnica específica, sino que pueden ser utilizadas distintas aproximaciones, las cuales dependiendo del dominio mostrarán unos resultados

mejores o peores. De esta manera, se resuelven problemas analizando los datos, generando y evaluando distintos modelos de forma automática.

Las técnicas utilizadas en la minería de datos están basadas en la estadística y la inteligencia artificial. Por una parte, la estadística aporta una generalización de los resultados que se obtienen para así ofrecer una visión general del problema que se está resolviendo. Por otra, la inteligencia artificial es la disciplina encargada de dar soluciones algorítmicas con un coste computacional razonable.

Para entender mejor la minería de datos a continuación se definen dos conceptos fundamentales:

- *Data mining:* proceso de extraer conocimiento útil y comprensible previamente desconocido, desde grandes cantidades de datos almacenados en distintos formatos (Witten y Frank, 2000).

- *Knowledge discovery in databases*: descubrimiento de conocimiento en bases como proceso no trivial de identificar patrones válidos, novedosos, potencialmente útiles, y en última instancia, comprensibles a partir de los datos (Fayyad y col. 1996).

El proceso de extracción de datos KDD está compuesto por las siguientes etapas:

- *Selección de datos:* etapa en la que identifican aquellos datos que son relevantes para el futuro análisis. Esta etapa es importante ya que la calidad del conocimiento descubierto depende además de por el algoritmo de minería elegido por la calidad de los datos que son analizados.

- *Preprocesado de datos:* etapa en la que se filtran los datos, se debe diferenciar aquellos que son válidos y útiles de los que no se ajustan al comportamiento general de los datos, éstos últimos son eliminados

- *Transformación:* etapa en la que se realiza un tratamiento preliminar de los datos, se eligen las variables más representativas o variables independientes mediante un test de sensibilidad, algoritmos heurísticos o de distancia. Generando finalmente una estructura de datos apropiada.

- *Data Mining:* esta etapa consiste en construir modelos descriptivos o predictivos que representan patrones de comportamiento de los datos que constituyen el problema.

- *Interpretación y evaluación:* etapa en la que se identifican los modelos descriptivos o predictivos que mejor solucionen el problema, obteniendo con ellos unos resultados satisfactorios.

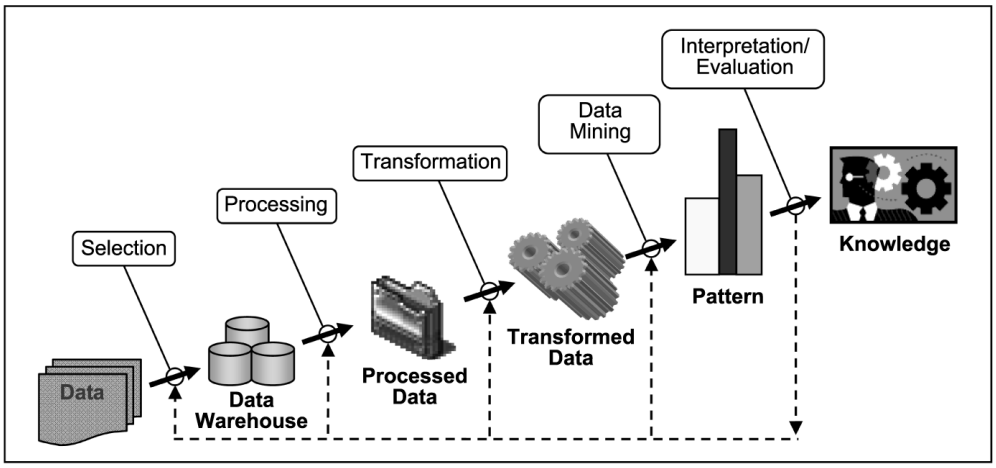

ILUSTRACIÓN 9: ETAPAS DEL PROCESO KDD

2.2 ALGORITMOS DE APRENDIZAJE

Los algoritmos de extracción de conocimiento utilizados en este proyecto se engloban en la clase de algoritmos supervisados. El aprendizaje supervisado tiene como objetivo la predicción del valor de una variable o atributo conocido como clase a partir de la información codificada en el conjunto de datos. Se basa en la búsqueda de relación entre los datos para así asignar un valor a los datos desconocidos.

Los algoritmos supervisados se componen de dos fases distintas:

- *Entrenamiento:* Construcción de un modelo usando un subconjunto de datos con etiquetas conocidas.

- *Test:* Prueba del modelo sobre el resto de los datos.

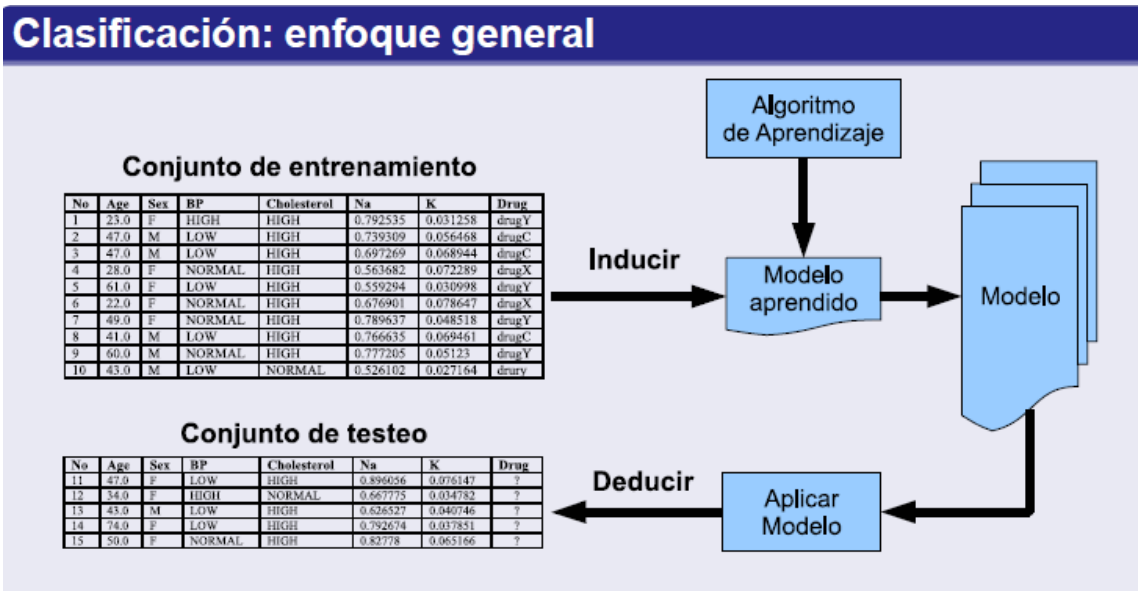

ILUSTRACIÓN 10: ESQUEMA GENERAL DEL APRENDIZAJE SUPERVISADO

En este proyecto, se han considerado dos paradigmas de clasificación supervisada que son la clasificación basada en vecindad (Knn) y el aprendizaje de Support Vector Machine (SVM).

SVM (Máquinas de vectores soporte)

Desarrollado por Cristianini y Shawe–Taylor. Se basa en la representación de las imágenes en un espacio vectorial, bajo la suposición de que las imágenes de cada tipo de clase son agrupadas en regiones separables del espacio de representación. [BUR98]El algoritmo busca un hiperplano que divida cada clase, habiendo una distancia máxima entre las imágenes y el hiperplano.

El hiperplano se define con la siguiente función: $f(x) = wx + b$

Suponiendo que el problema que se afronta está compuesto por datos separables, el primer paso es etiquetar los datos que pertenecen al conjunto de entrenamiento.

$$x_i;\ y_i;\ i = 1,\ldots,l;\ y_i \epsilon \{-1, 1\},\ x_i \epsilon R^d$$

Bajo la suposición de que existe un hiperplano que divida las muestras positivas de las negativas, se busca hallar el hiperplano con mayor margen existente.

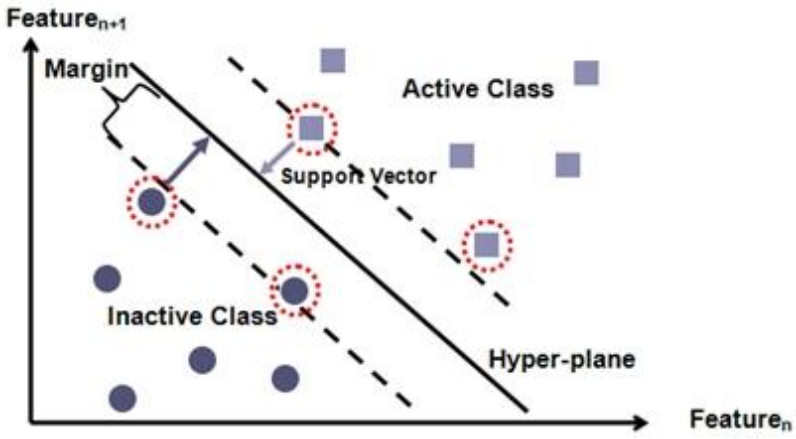

ILUSTRACIÓN 11: EJEMPLO ALGORITMO SVM

K-NN (K- NearestNeighbour)

Desarrollado por Covert, Harty Dasarathy. La idea básica sobre la que se fundamenta es que un nuevo caso se va a clasificar en la clase más frecuente a la que pertenecen sus K vecinos más cercanos [FIX51]. El paradigma se fundamenta por tanto en una idea muy simple e intuitiva, junto a su fácil implementación hace que sea un clasificador muy extendido.

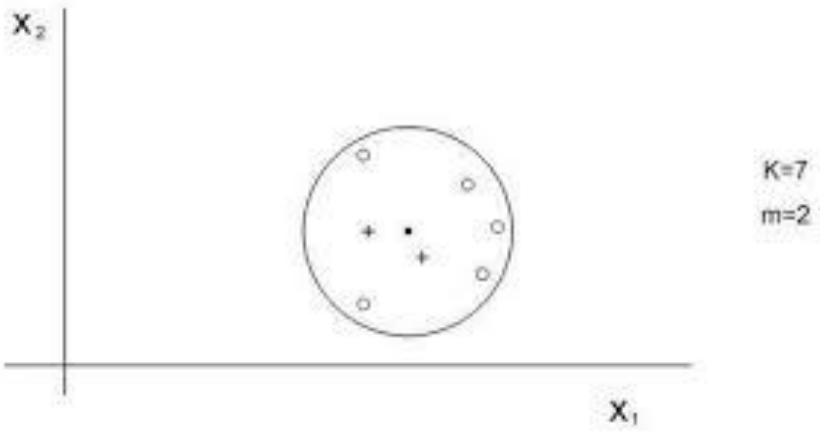

Figura 5: Ejemplo de ilustración del K-NN con distancia media

ILUSTRACIÓN 12: EJEMPLO ALGORITMO KNN

2.4 EVALUACIÓN EN LA CLASIFICACIÓN SUPERVISADA

Matriz de confusión:

En el aprendizaje supervisado, la mayoría de las métricas utilizadas para estimar la bondad de un clasificador pueden estimarse a través de la matriz de confusión. Permitiendo así, comprender la naturaleza y la distribución del error cometido Es un problema de dos clases, es decir, clase positiva y clase negativa, los elementos de esta matriz de confusión viene dados por:

- *TP:* número de instancias positivas que han sido clasificadas de manera correcta.
- *FP:* número instancias negativas que han sido clasificadas de manera incorrecta.
- *TN:* número de instancias negativas que han sido clasificadas de manera correcta.
- *FN:* número de instancias positivas que han sido clasificadas de manera incorrecta.

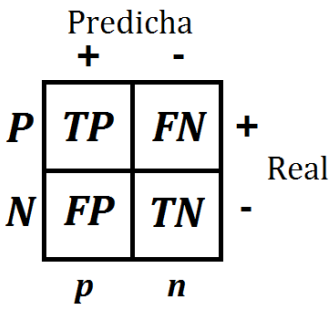

ILUSTRACIÓN 13: MATRIZ DE CONFUSIÓN

Error:

Es una representación del número de casos que han sido clasificados de una manera incorrecta.

$$\frac{FP\ +\ FN}{TP +\ FP +\ FN\ + TN}$$

Recall:

Porcentaje de elementos que han sido correctamente clasificados de la clase positiva en relación con el total de elementos de clase positiva.

$$\frac{TP}{P}$$

Precisión:

Porcentaje de elementos que han sido correctamente clasificados de la clase positiva en relación con el total de elementos predichos como supuestos positivos.

$$\frac{TP}{p}$$

Accuracy:

Se mide como el porcentaje de aciertos sobre el conjunto de testeo, estimando cómo se comportarán los nuevos datos.

$$\frac{TP + TN}{P + N}$$

2.5 TÉCNICAS DE VALIDACIÓN

El problema a afrontar consiste en validar un modelo en todas las circunstancias posibles. Al no tener a disposición tantos datos de entrenamiento que agrupen todas las circunstancias, se resuelve analizando si el comportamiento que se obtiene al generar un clasificador con un conjunto de datos y validarlo con otro conjunto distinto es correcto. Se busca que el modelo validado tenga una tasa de acierto elevada sin ser demasiado ajustado a los datos aprendidos para evitar problemas de overfiting y ganar en generalización.

A continuación se detalla la técnica de validación utilizada en este proyecto:

- *Validación cruzada (k-foldcross-validation):* garantiza que el conjunto de entrenamiento y test son independientes, para ello, el conjunto de datos se divide en *k* subconjuntos del mismo tamaño. Estos subconjuntos son mutuamente excluyentes. Para el entrenamiento se utilizan *k-1* subconjuntos los cuales sirven para inducir el modelo clasificatorio. El subconjunto restante será el test, el cual es utilizado para evaluar la predicción verdadera.

  El proceso de validación es repetido durante *k* iteraciones, con cada uno de los subconjuntos.A continuación, se realiza la media aritmética de los resultados obtenidos en cada iteración para obtener un único resultado.

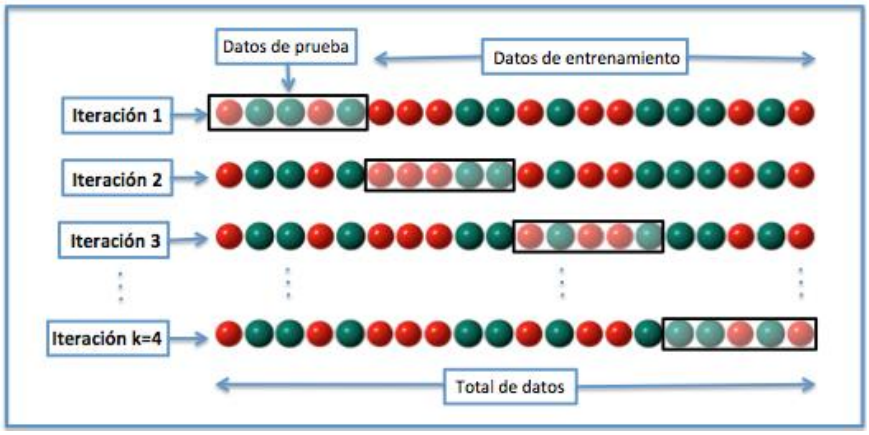

ILUSTRACIÓN 14: K-FOLD CROSS-VALIDATION

- *Validación cruzada dejando uno fuera (Leave one out):* el conjunto de datos se divide en *n* subconjuntos. Siendo *n* el número de muestras del conjunto de datos. Para el entrenamiento se utilizan *n-1* muestras de un subconjunto y la muestra restante para el test. El proceso de validación es repetido durante *n* iteraciones, con cada una de las muestras, por lo que a nivel computacional tiene un elevado coste.

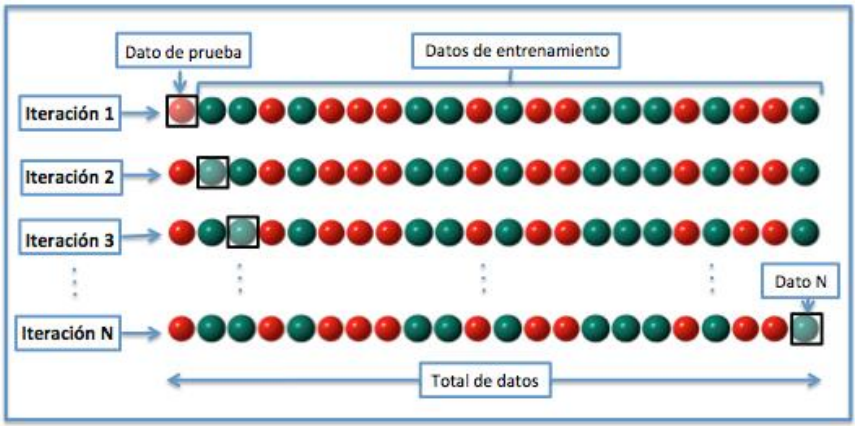

ILUSTRACIÓN 15: LEAVE ONE OUT

CAPÍTULO 3: DETECCIÓN DE REGIONES DE INTERÉS

En este capítulo se describen las imágenes, objetos de interés y el proceso de detección.

3.1 DESCRIPCIÓN DE IMÁGENES Y OBJETOS DE INTERÉS

La manera más intuitiva de detectar y extraer información de imágenes aéreas es mediante la interpretación visual, la cual se basa en la capacidad que las personas tienen para relacionar colores, tonos y patrones espaciales que aparecen en una imagen.

Las imágenes aéreas se componen por objetos de distintas formas y tamaños, algunos de ellos son identificados correctamente mientras que otros no. Los factores que influyen son la experiencia del modelo, ya que en el proceso de entrenamiento debe poseer un conocimiento amplio del estudio que se está realizando, y la calidad de la imagen.

En este caso, los objetos de interés que se buscan detectar de manera automática son los tejados.

ILUSTRACIÓN 16: IMAGEN AÉREA

Como se puede ver en la ilustración anterior la imagen se compone por diversos objetos:

- *Tejados*
- *Suelo*
- *Piscinas*
- *Coches*
- *Vegetación*

Toda imagen tiene unas características las cuales pueden ayudar en la identificación de los objetos de interés:

- *Tono:* brillo de un objeto.
- *Textura:* frecuencia con la que aparecen cambios de tono en la imagen.
- *Color:* elemento básico para la detección de objetos en imágenes.
- *Tamaño:* aunque los objetos de pertenezcan a una misma clase pueden tener distintos tamaños.
- *Forma:* característica que ayuda mucho a diferenciar los objetos, ya que hay formas que caracterizan a muchos de los objetos que componen las imágenes.
- *Patrón:* si la distribución de los objetos tiene una forma repetitiva puede ayudar a su identificación.

- *Sombras:* la oscuridad de un espacio de la imagen puede ocultar información importante, pero a su vez, ayuda a la percepción tridimensional de los objetos, lo cual puede ser de gran ayuda a la hora de distinguirlos.

3.2 DESCRIPCIÓN DEL PROCESO DE DETECCIÓN

El primer paso para la detección de las regiones de interés es asignar un valor a cada región existente. Para ello se trabaja con la imagen ya segmentada asociada a la original. A todos los píxeles que componen una misma región se les asigna un mismo valor. Este valor es incremental y su rango es de 1 hasta el número de regiones existentes.

Se generan los histogramas de la imagen según el descriptor seleccionado y a continuación se asigna la clase a la que pertenece cada región de la matriz. Será 0 en caso de considerarse background, 1 si es el objeto de interés y -1 en el caso de considerarlo de otro tipo. Para ello se compara con la imagen segmentada, si el número de píxeles con valor 255 es mayor que el número de píxeles con valor 0 con un porcentaje mínimo del 90% la región es clasificada como objeto de interés. Si el porcentaje de píxeles con valor 255 es menor al 10% se considera background, y si no se encuentra entre ninguno de los dos casos anteriormente descritos se considera de otro tipo.

Se generan los conjuntos de entrenamiento y testeo utilizando 6 cross validation. La imagen seleccionada es la que se utilizara para el testeo y el resto de imágenes serán el conjunto de entrenamiento.

A la hora de generar el conjunto de entrenamiento, se desechan aquellos histogramas que pertenecen a la clase -1, ya que para el aprendizaje no ofrecen

información útil. Además se eliminan aquellos atributos que están compuestos completamente por elementos de valor 0, ya que no aportan ningún tipo de información.

Los conjuntos anteriormente generados se utilizan para la clasificación con el clasificador escogido. Una vez generada la clasificación, para cada región de la imagen se asigna el valor de la clase predicha por el clasificador. Una vez obtenida la imagen resultante se superpone con la imagen original para mostrar si los resultados obtenidos son satisfactorios.

CAPÍTULO 4: DESCRIPTORES CARACTERÍSTICOS DE LA IMAGEN

En este capítulo se describe los descriptores basados en histogramas color y LBP utilizados para caracterizar las diferentes regiones de una imagen.

A la hora de identificar los objetos que componen una imagen, no solamente el color ofrece información, también se debe tener en cuenta la textura de ésta. La textura muestra si en las regiones que componen una imagen existe la repetición de uno o varios patrones. Pudiendo repetirse exactamente o teniendo pequeñas variaciones.

El operador LBP puede entenderse como un acercamiento unificador hacia los modelos estadísticos y los modelos estructurales del análisis de texturas.

4.1 PATRÓN LBP

Este patrón es un operador que asocia a una imagen una matriz de etiquetas de enteros que describen el aspecto de ésta a pequeña escala. Las regiones de etiquetas, o comúnmente histogramas, son las utilizadas para el análisis posterior.

Su uso se extiende tanto en imágenes monocromáticas como en imágenes color, ya que obtiene unos resultados notables en términos de precisión y a nivel computacional. A continuación detallan las diversas variaciones del operador LBP

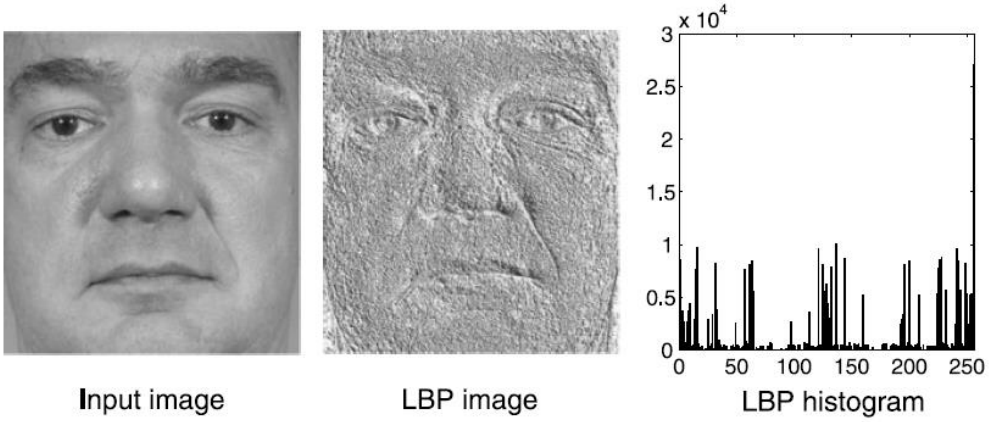

ILUSTRACIÓN 17: EJEMPLO LBP

LBP básico

Este operador fue introducido por Ojala[OPH96], basándose en la idea de que la textura de una imagen se compone por dos características complementarias, el umbral y el peso:

- *El umbral*: es el valor que se toma como referencia y es propio de cada vecindad, siendo en este caso el valor del píxel central.

- *El peso:* es el valor que tiene asignado cada píxel. La vecindad está compuesta por 8 píxeles, lo que da una rango de 2^8 = 256 valores distintos (0,…,255) que dependen del valor de gris del píxel central.

El mecanismo seguido por el patrón consta de una división en bloques de tamaño 3x3. Los píxeles de los bloques son etiquetados y se establece el umbral de la vecindad respecto al valor del píxel central.

Inicialmente se identifica el píxel central y su valor; los valores de los píxeles que componen la vecindad son comparados con el valor del píxel central teniendo en cuenta las siguientes reglas:

- Si $P_v < P_c$; $P_v = 0$ siendo, P_v el valor del píxel de la vecindad.
- Si $P_v \geq P_c$;$P_v = 1$ P_c el valor del píxel central.

Finalmente los nuevos valores de los píxeles obtenidos son multiplicados por su correspondiente peso. En la siguiente ilustración, se muestra un ejemplo en el cual el peso de cada píxel es 1.

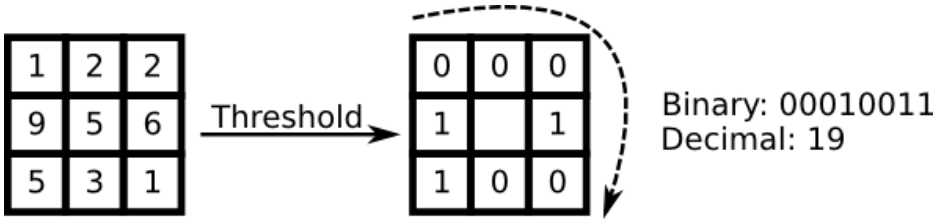

ILUSTRACIÓN 18: LBP BÁSICO

Derivaciones del LBP básico

Considerando que la textura de una imagen monocromática I(*x,y*) está definida por una distribución de niveles de grises de P + 1 píxeles, siendo P > 0. Se define la textura de la siguiente manera[OPM02]:

$$T = t(g_c, g_0, \ldots g_{p-1})$$

Se define g_c como el nivel de gris de un pixel aleatorio *(x,y)*, tal que $g_c=I(x,y)$.

Y g_p como el nivel de gris de P igualmente espaciado sobre un círculo de radio R siendo R>0.

A continuación se ilustra tres ejemplos de conjuntos circulares simétricos con diferentes valores de *P* y *R*. Los valores de los vecinos que no se encuentran exactamente en los pixeles que definen el círculo se estiman por interpolación lineal. El hecho de trabajar con entornos de tamaño variable no es un problema, cualquier radio es válido si se utilizan vecindades circulares y se interpolan los valores de los píxeles.

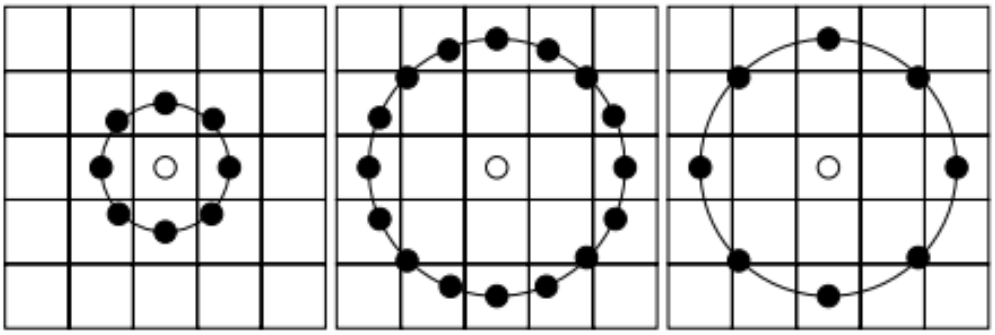

ILUSTRACIÓN 19: EJEMPLOS LBP CON DISTINTOS VALORES DE P Y R

UniformPatterns:

El patrón uniforme es otra versión del patrón básico, el cual implementa un descriptor sencillo e invariante frente a rotaciones con cierta solidez estadística. Se caracteriza por el reducido número de muestras necesarias para la estimación de la distribución, y su estabilidad, ya que son menos propensos al ruido.

La razón principal por la que son empleados con mayor frecuencia, es que una considerable parte de las imágenes siguen patrones binarios uniformes.

Los patrones LBP, se obtienen tomando muestras circulares alrededor del pixel central, la rotación de la imagen tiene las siguientes consecuencias:

- Cada vecindad local es rotada a otra situación de píxeles.
- Dentro de cada vecindad, los píxeles que rodean al píxel central son también rotados en una orientación diferente.

Para que un patrón binario se considere uniforme, éste debe contener un máximo de dos transiciones a nivel de bits cuando el patrón se recorre de manera circular. Por ejemplo, el patrón ‘11111111’ (0 transiciones), '00001111’ (1 transición) y 11001111 (2 transiciones) son uniformes, mientras que los patrones de 11010110 (6 transiciones) y 01010101 (8 transiciones) no lo son.

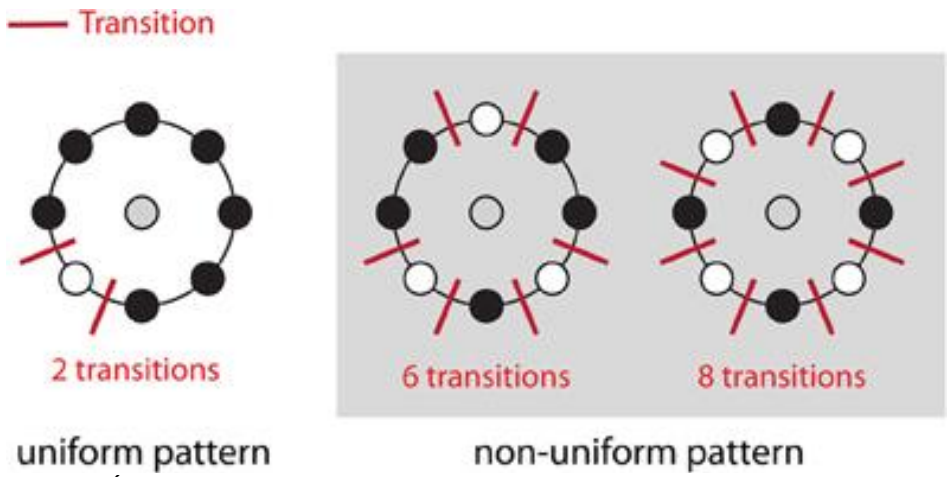

ILUSTRACIÓN 20: PATRONES UNIFORMES Y NO UNIFORMES

Cuando se utiliza el patrón uniforme, se asigna una etiqueta para cada uno de aquellos patrones que son uniformes, y para los patrones no uniformes se asigna una misma etiqueta.

RotationInvariant:

Este patrón cuantifica las ocurrencias de rotaciones invariables correspondientes a ciertas características de la imagen, pudiendo considerarse un método de detección de rasgos.

La rotación de una imagen da lugar a un $LBP_{P,R}$ diferente ya que los niveles de gris correspondientes se ubican en una posición distinta girando alrededor del píxel de origen. Para normalizar la traslación y rotación, se aplica un mapeo invariante de la rotación, el cual asigna un identificador único de valor mínimo para cada rotación:

$$LBPri\ P,R = \min\{ROR(LBP_{P,R,\,i})\ \{i = 0, 1, \cdots, P - 1\}$$

donde ROR(x, i) denota el desplazamiento circular hacia la derecha en secuencias de *P*bits e *i* etapas.

UniformRotationInvariant:

Este patrón se basa en la rotación invariante, la cual ha sido descrita anteriormente. Se caracteriza por no proporcionar una buena discriminación cuando la frecuencia de aparición de un mismo patrón varía mucho [MTZ00], ya que los histogramas utilizados son uniformes.

Considerando $h_I\ (U_P\ (n, r))$, siendo h_1 el valor del histograma y$U_P(n, r)$ el número de apariciones del patrón uniforme en la imagen *I*, si la imagen *I* es rotada $\alpha = a\ 360^{\circ}/P$, esta rotación provoca un desplazamiento cíclico en el histograma a lo largo de cada una de las filas:

$$hI\ \alpha\circ\ (U_P\ (n, r + a)) = h_I\ (U_P\ (n, r)).$$

4.2 PATRÓN RGB

Los modelos de color son ampliamente utilizados en bases de datos de gran tamaño. La razón de esto es que el color es un perfecto descriptor que facilita la identificación y extracción de objetos en una imagen. Por otra parte, las personas son capaces de percibir un amplio abanico de tonos e intensidades de color, en comparación con la cantidad reducida de tonos de grises existentes.

El patrón RGB se caracteriza por la distribución del color, está compuesto por un histograma por cada canal de color: rojo, verde y azul.

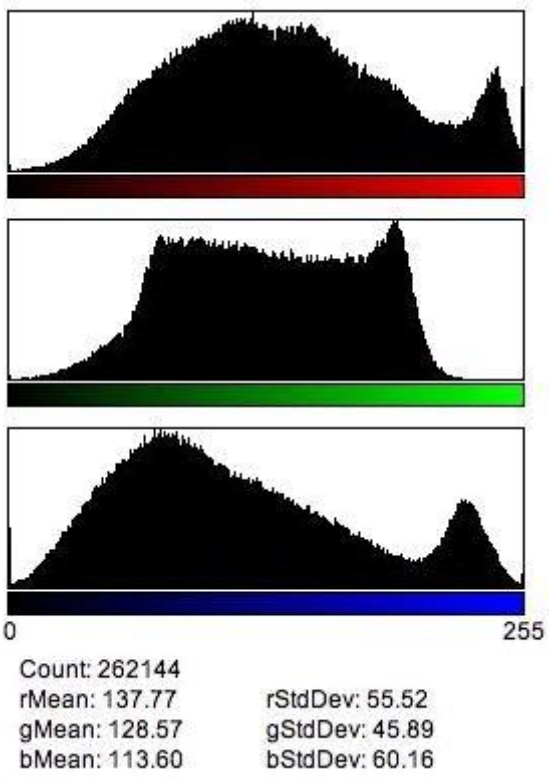

ILUSTRACIÓN 21: HISTOGRAMAS RGB

Este método se utiliza para la comparación del contenido de imágenes. Para ello, se calculan los histogramas de color asociados a cada imagen y se identifican las proporciones similares de cierto color. La idea básica es que las imágenes similares estarían compuestas por proporciones similares de los colores.

Esta idea tiene sus ventajas e inconvenientes, los cuales pasan a detallarse a continuación:

- *Ventajas:*
    - El color es un perfecto descriptor que facilita la identificación y extracción de objetos en una imagen. Siendo un método muy eficaz en la identificación de objetos cuando previamente se conoce su localización o apariencia.
    - La información de color es rápida a la hora realizar tratamientos.
- *Inconvenientes:*
    - La representación de histogramas es dependiente del color de los objetos, sin tener en cuenta su textura y forma. Esto puede generar problemas ya que dos imágenes totalmente distintas pueden tener histogramas idénticos.
    - Los histogramas de color describen una alta sensibilidad a la interferencia de ruido, como por ejemplo los cambios de intensidad de iluminación.

CAPÍTULO 5: RESULTADOS OBTENIDOS

Este capítulo comienza con la descripción de los resultados experimentales obtenidos.

A continuación, se muestran una comparativa entre resultados y objetivos, y una segunda comparativa con otras alternativas.

5.1 RESULTADOS

Inicialmente los descriptores utilizados han sido histograma color y LBP. Los resultados de este último patrón no eran los deseados, por lo que se buscó una alternativa que fue la combinación de LBP y RGB, obteniendo un descriptor híbrido.

A la hora de generar los conjuntos de entrenamiento y testeo se utiliza como conjunto de datos la concatenación de todas las combinaciones utilizando LBP sobre la misma imagen y los histogramas obtenidos con RGB. Para que no sea muy pesada la matriz generada, son eliminados aquellos atributos en los que todos sus componentes tienen valor 0, ya que no aportan ninguna información.

Las imágenes mostradas corresponden al patrón LBP con modo riu2 y radio 3. En los ejemplos que se muestran a continuación se utiliza el clasificador KNN con K=3

ILUSTRACIÓN 22: IMAGEN1 LBP

ILUSTRACIÓN 23: IMAGEN2 LBP

ILUSTRACIÓN 26: IMAGEN3 LBP

ILUSTRACIÓN 27: IMAGEN4 LBP

ILUSTRACIÓN 24: IMAGEN5 LBP

ILUSTRACIÓN 25: IMAGEN6 LBP

A continuación se muestran los mejores resultados obtenidos con LBP y SVM Como se puede comprobar la tasa de acierto es bastante baja.

LBP	RECALL	PRECISION	ACCURACY
IMAGEN 1	0.5906	0.2207	0.7415
IMAGEN 2	0.5541	0.7740	0.8373
IMAGEN 3	0.6100	0.7938	0.8721
IMAGEN 4	0.5089	0.6946	0.8776
IMAGEN 5	0.6413	0.6097	0.7949
IMAGEN 6	0.4673	0.5364	0.8485

ILUSTRACIÓN 28: RESULTADOS LBP

A continuación, se muestran los mejores resultados obtenidos con RGB. Como se puede comprobar la tasa de acierto es mayor que con LBP, pero aún no se acerca a los resultados esperados.

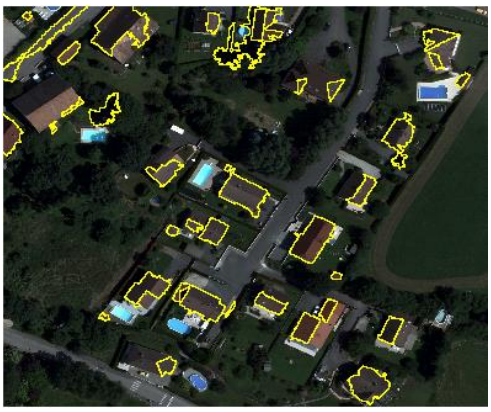

ILUSTRACIÓN 29: IMAGEN1 RGB

ILUSTRACIÓN 30: IMAGEN2 RGB

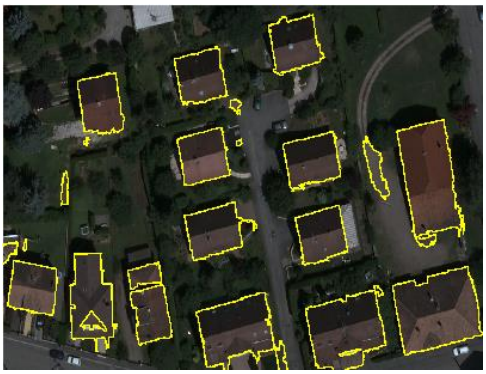

ILUSTRACIÓN 31: IMAGEN3 RGB

ILUSTRACIÓN 32: IMAGEN4 RGB

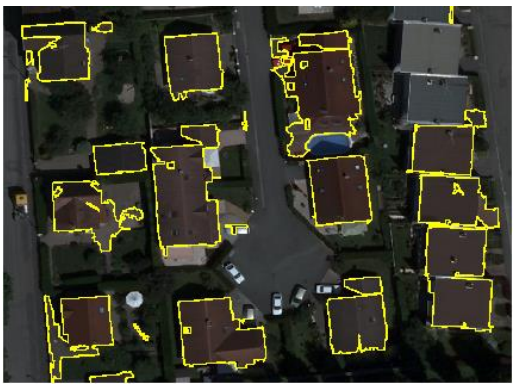

ILUSTRACIÓN 34: IMAGEN5 RGB

ILUSTRACIÓN 33: IMAGEN6 RGB

A continuación se muestran los datos obtenidos con el descriptor RGB y clasificador KNN, siendo K= 1, 3 y 7

RGB	K = 1			K = 3			K = 7		
	RECALL	PRECISION	ACCURACY	RECALL	PRECISION	ACCURACY	RECALL	PRECISION	ACCURACY
IMAGEN 1	0.5974	0.6725	0.9281	0.5555	0.7375	0.9334	0.5640	0.7258	0.9327
IMAGEN 2	0.7269	0.8032	0.8792	0.7010	0.8954	0.8980	0.7010	0.8954	0.8980
IMAGEN 3	0.9584	0.8679	0.9563	0.9783	0.9150	0.9737	0.9127	0.9032	0.9568
IMAGEN 4	0.8524	0.8295	0.9447	0.8738	0.9220	0.9657	0.8471	0.8885	0.9556
IMAGEN 5	0.8002	0.8298	0.9029	0.8632	0.8706	0.9293	0.8385	0.8773	0.9257
IMAGEN 6	0.7647	0.7329	0.9168	0.6957	0.8734	0.9344	0.6579	0.8994	0.9327

Datos obtenidos con el descriptor RGB y descriptor SVM:

RGB	RECALL	PRECISION	ACCURACY
IMAGEN 1	0.4854	0.8874	0.9403
IMAGEN 2	0.6033	0.9510	0.8854
IMAGEN 3	0.7463	0.9536	0.9324
IMAGEN 4	0.7029	0.8796	0.8951
IMAGEN 5	0.7029	0.8796	0.8951
IMAGEN 6	0.8716	0.8807	0.9601

A continuación, se muestran los mejores resultados obtenidos con el descriptor híbrido. Como se puede comprobar la tasa de acierto es mayor que con RGB.

ILUSTRACIÓN 35: IMAGEN1 LBP+RGB

ILUSTRACIÓN 36: IMAGEN2 LBP+RGB

ILUSTRACIÓN 37: IMAGEN3 LBP+RGB

ILUSTRACIÓN 38: IMAGEN4 LBP+RGB

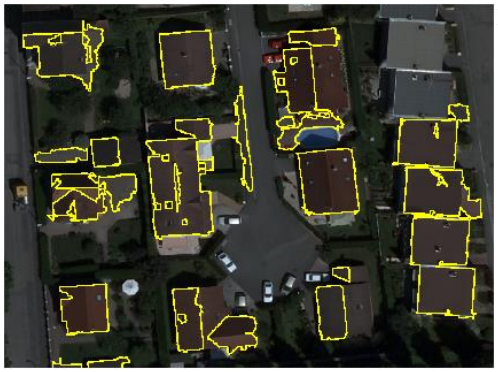

ILUSTRACIÓN 39: IMAGEN5 LBP+RGB

ILUSTRACIÓN 40: IMAGEN6 LBP+RGB

A continuación se muestran los datos obtenidos con el descriptor LBP+RGB y el clasificador KNN, siendo K= 1, 3 y 7

LBP + RGB	K = 1			K = 3			K = 7		
	RECALL	PRECISION	ACCURACY	RECALL	PRECISION	ACCURACY	RECALL	PRECISION	ACCURACY
IMAGEN 1	0.9380	0.8050	0.9700	0.9588	0.7974	0.9705	0.9547	0.5638	0.9188
IMAGEN 2	0.9093	0.9204	0.9547	0.9254	0.9360	0.9631	0.9258	0.9169	0.9577
IMAGEN 3	0.9645	0.9052	0.9682	0.9599	0.9045	0.9670	0.9593	0.9029	0.9664
IMAGEN 4	0.9480	0.9279	0.9785	0.9422	0.9237	0.9768	0.9405	0.9219	0.9762
IMAGEN 5	0.8477	0.8539	0.9207	0.8398	0.8711	0.9241	0.8973	0.8558	0.9323
IMAGEN 6	0.9057	0.8309	0.9549	0.9240	0.8396	0.9591	0.9541	0.8351	0.9583

A continuación se muestran los datos obtenidos con el descriptor LBP+RGB y clasificador SVM,:

LBP + RGB	RECALL	PRECISION	ACCURACY
IMAGEN 1	0.9380	0.8050	0.9700
IMAGEN 2	0.9264	0.9559	0.9688
IMAGEN 3	0.9722	0.9253	0.9752
IMAGEN 4	0.9515	0.9065	0.9749
IMAGEN 5	0.8304	0.8632	0.9197
IMAGEN 6	0.9455	0.8663	0.9676

CAPÍTULO 6: DESCRIPCIÓN DE LA INTERFAZ MATLAB

En este capítulo se proporciona una descripción detallada de la herramienta informática para la detección automática de los tejados.

6.1 DESCRIPCIÓN

Como se ha dicho anteriormente, esta interfaz persigue la detección automática de objetos de interés en imágenes aéreas, concretamente se trata de identificar los tejados de edificios en imágenes.

La base de datos con la que se trabaja está compuesta por las imágenes originales, las imágenes de verdad del terreno, e imágenes de segmentación ya generadas.

Una vez abierta la aplicación, el usuario tiene que realizar las siguientes acciones:

1. Seleccionar la imagen a clasificar.
2. Seleccionar y configurar el descriptor a utilizar
3. Seleccionar y configurar el clasificador a utilizar.

El proceso interno que sigue la aplicación se describe a continuación.

El primer paso es asignar un valor a cada región existente. Para ello se trabaja con la imagen ya segmentada asociada a la original. A todos los píxeles que componen una misma región se les asigna un mismo valor. Este valor es incremental y su rango es de 1 hasta el número de regiones existentes. Estos datos son guardados en una matriz llamada *labelMatxix*.

En segundo lugar se generan los histogramas de la imagen según el descriptor seleccionado por el usuario. Para que el usuario sepa en qué punto se encuentra del

proceso, una vez que han sido generados los datos, se van incluyendo en la interfaz de la aplicación.

El tercer paso consiste en asignar la clase a la que pertenece cada región de la matriz *labelMatrix*. Será 0 en caso de considerarse background, 1 si es el objeto de interés y -1 en el caso de considerarlo de otro tipo. Para ello se compara con la imagen segmentada, si el número de píxeles con valor 255 es mayor que el número de píxeles con valor 0 con un porcentaje mínimo del 90% la región es clasificada como objeto de interés. Si el porcentaje de píxeles con valor 255 es menor al 10% se considera background, y si no se encuentra entre ninguno de los dos casos anteriormente descritos se considera de otro tipo.

El resultado obtenido se superpone con la imagen original, mostrándose en rojo los objetos de interés y en azul los clasificados como otro tipo.

Los histogramas y la clasificación son generados de manera on-line, y los nuevos datos son guardados. Si los histogramas y la clasificación han sido generados anteriormente con esos mismos parámetros los datos guardados anteriormente son cargados para disminuir el tiempo de espera.

Además se genera una matriz llamada *buildingdata*, para manejar de manera más sencilla los datos que son necesarios para generar los conjuntos de entrenamiento y testeo. Cada columna corresponde a una imagen Para cada imagen esta matriz almacena la imagen segmentada, la imagen binaria, el vector de etiquetas y todos los datos obtenidos con los descriptores.

El cuarto paso es la generación de los conjuntos de entrenamiento y testeo. A la hora de generar el conjunto de entrenamiento, se desechan aquellos histogramas que pertenecen a la clase -1, ya que para el aprendizaje no ofrecen información útil.

Para trabajar con matrices menos pesadas, se eliminan aquellos atributos que están compuestos completamente por elementos de valor 0, ya que no aportan ningún tipo de información.

Estos conjuntos son los utilizados para generar la clasificación, pudiendo trabajar desde Weka o desde el propio Matlab.

- Si se trabaja desde Weka, los resultados obtenidos deben ser importados a Matlab, lo que supone que la generación de las clasificaciones no pueden ser realizada desde la interfaz de Matlab.

- Al trabajar desde Matlab, la ventaja es que se pueden generar las clasificaciones de manera automática pero ciertos clasificadores no funcionan de manera correcta desde Matlab. Esta fue la razón por la que no se pudo introducir el clasificador SVM desde la interfaz, pero está recogido en el Capítulo de los resultados obtenidos, ya que fue generado desde Weka. Para el clasificador SM tras un análisis de cuáles son los valores de los atributos que mejores resultados obtienen, se establece el valor de *gamma* = 0.05 y *cost*= 300. El resto de atributos toman el valor que traen por defecto.

A continuación se describe el funcionamiento con los dos casos anteriormente mencionados.

En el caso de trabajar desde Matlab, simplemente se llama al método de clasificación escogido, en este caso *knnclassify(Sample, Training, Group, k)*, donde *Sample* es el conjunto de testeo, *Training* el conjunto de entrenamiento, *Group* es la matriz de labels, y *k* es el número de vecinos colindantes utilizados en la clasificación. Una vez generada la clasificación, para cada región de la imagen se asigna el valor de la clase predicha por el clasificador. Una vez obtenida la imagen resultante se superpone con la imagen original para mostrar si los resultados obtenidos son satisfactorios.

En el caso de trabajar desde Weka, se escoge el clasificador que ha seleccionado el usuario. Se carga en el apartado *Preprocess*el conjunto de entrenamiento y en *Classify*, en la opción *supplied test set*, el conjunto de testeo. Las opciones de evaluación que han sido utilizadas se muestran en la siguiente ilustración.

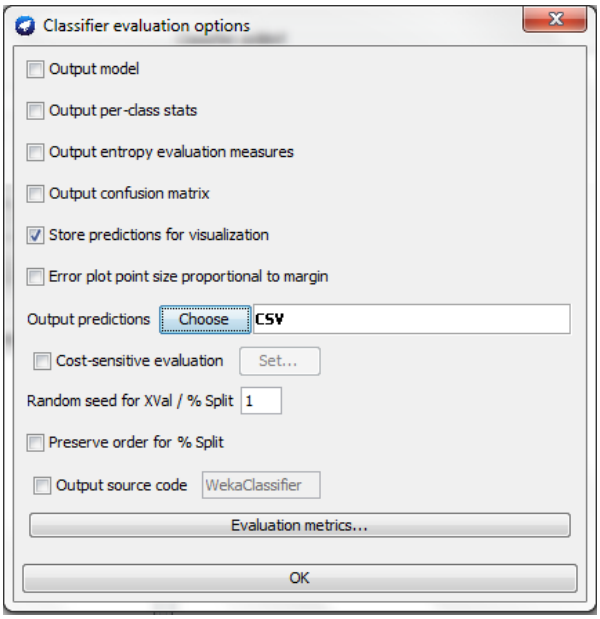

ILUSTRACIÓN 41: OPCIONES DE WEKA

Una vez guardado los datos obtenidos con Weka, éstos son cargados en Matlab. Para ello es necesario convertir los datos, especificando desde qué línea debe comenzar a leer los datos y dónde terminan. Para cada región de la imagen se asigna el valor de la clase predicha por el clasificador. Una vez obtenida la imagen resultante se superpone con la imagen original para mostrar si los resultados obtenidos son satisfactorios.

Archivos Weka

En el caso de trabajar con Weka, los archivos generados deben ser importados a Matlab para generar las imágenes resultantes de la clasificación. Estos archivos se componen por los siguientes atributos:

- *Instancia:* cada histograma de la imagen evaluada está representada por una instancia.
- *Actual:* Muestra la clase a la que pertenece ese histograma, siendo 0 background y 1 techo. Al generar una clasificación sobre un conjunto de testeo del cual se desconocen sus clases, se les asigna un símbolo de interrogación.
- *Predicted: muestra la clase a la cual ha sido asignada, 1:0 representa background y 2:1 techo.*
- *Error: en el caso de que un histograma no haya podido ser clasificado muestra un +.*
- *Prediction: informa de con que fiabilidad se ha realizado la asignación de la nueva clase Eeste atributo toma valores entre 0 y 1,*

	A **inst** NUMBER	B **actual** NUMBER	C **predicted** TEXT	D **error1** TEXT	E **prediction** NUMBER
1	=== Run in...				
2	Scheme: ...				
3	Relation: ...				
4	Instances: ...				
5	Attributes: ...				
6	[list of attri...				
7	Test mode:...				
8	=== Predic...				
9	inst#	actual	predicted	error	prediction
10	1	1:?	1:0		1
11	2	1:?	1:0		1
12	3	1:?	1:0		1
13	4	1:?	1:0		1
14	5	1:?	1:0		1
15	6	1:?	1:0		1
16	7	1:?	1:0		1
17	8	1:?	1:0		1
18	9	1:?	1:0		1
19	10	1:?	1:0		1
20	11	1:?	2:1		1
21	12	1:?	1:0		1
22	13	1:?	1:0		1
23	14	1:?	1:0		1
24	15	1:?	1:0		1

ILUSTRACIÓN 42: ARCHIVO WEKA

Requisito software:

Matlab permite generar ejecutables de las aplicaciones generadas, pero para poder ejecutarlos se necesita tener *Matlab* instalado o en su defecto *Matlab Compiler Runtime* (MCR).

MCR es un conjunto independiente de bibliotecas compartidas que permite la ejecución de aplicaciones o componentes de *Matlab* compilados en ordenadores que no lo tienen instalado. A continuación, se detalla como instalar MCR, el cual es completamente gratuito.

http://www.mathworks.es/products/compiler/mcr/

Se elige la versión y la plataforma correspondiente a la aplicación que se está utilizando, en este caso *R2013a*. A continuación simplemente hay que seguir las instrucciones que aparecen en el asistente de instalación.

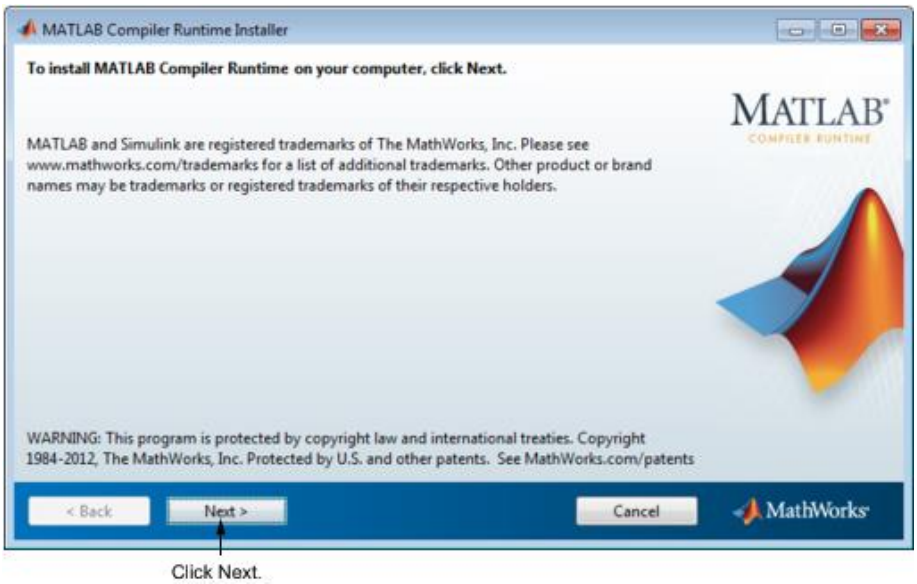

ILUSTRACIÓN 43: INSTALACIÓN MCR I

Se especifica la carpeta en la que se desea instalar el MCR en la *Selección de carpeta* del cuadro de diálogo.

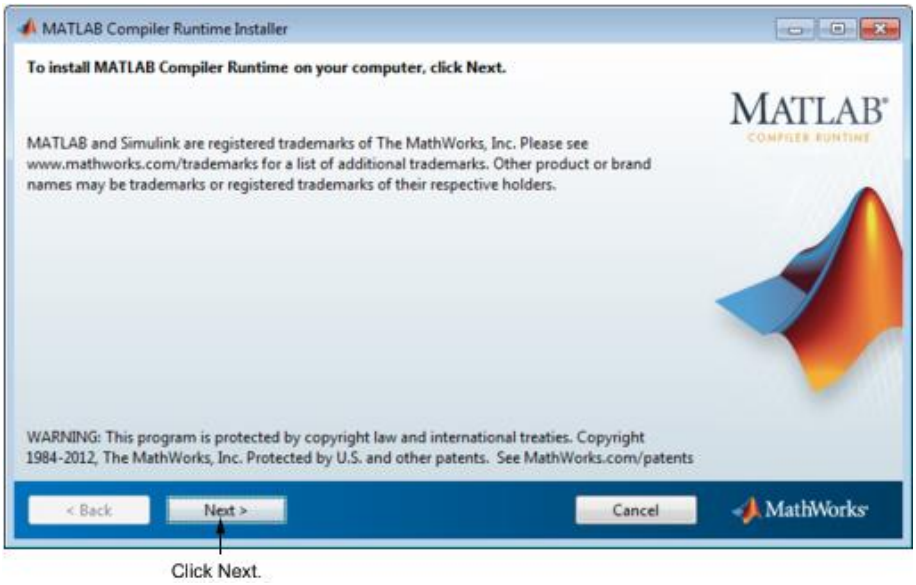

ILUSTRACIÓN 44: INSTALACIÓN MCR II

Requisito hardware:

Se ha probado el correcto funcionamiento con un equipo de 8GB de RAM, y un procesador Inter Core I7.

Aplicación

En este apartado describe la aplicación realizada, tanto los parámetros que pueden seleccionarse, los resultados que se obtienen y los mensajes emergentes. Una vez que se ejecuta aparece una interfaz como la que se muestra a continuación:

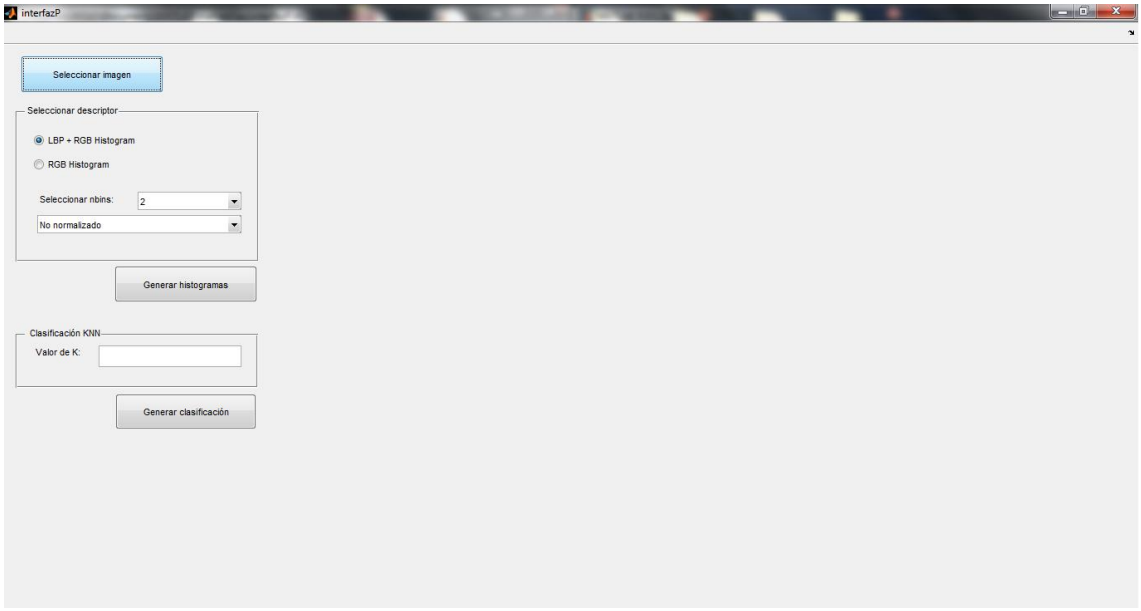

ILUSTRACIÓN 45: INTERFAZ DE LA APLICACIÓN I

Como se puede observar inicialmente la interfaz está compuesta por los siguientes elementos:

- *Botón para Seleccionar imagen:* se escoge una de las imágenes con extensión .tiff, Una vez seleccionada la imagen, se cargan en la interfaz la imagen seleccionada, y a su vez la imagen segmentada e imagen de verdad del terreno correspondientes.

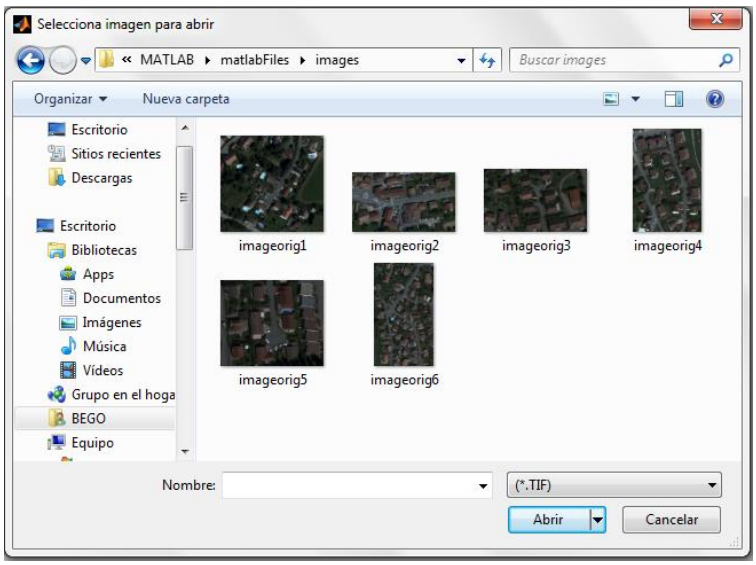

ILUSTRACIÓN 46: SELECCIONAR IMAGEN

Como se ha descrito anteriormente, una vez seleccionada la imagen, se cargan en la interfaz la imagen seleccionada, y a su vez la imagen segmentada e imagen de verdad del terreno correspondientes. La interfaz queda inicialmente de este modo.

- *Imagen segmentada:* la imagen original se divide en regiones, asignando un valor real a cada píxel que compone la región para su posterior clasificación.

- *Imagen verdad del terreno* muestra cuales son los techos que se encuentran en la imagen original. Los píxeles de esta imagen de intensidad de grises toma el valor 0 si es background y 255 si es objeto.

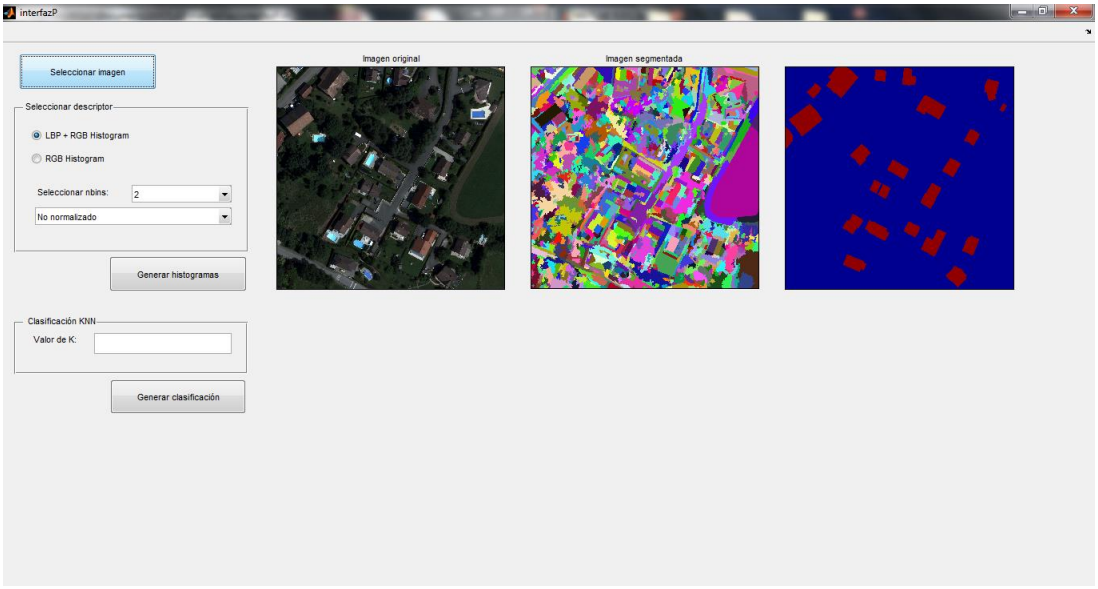

ILUSTRACIÓN 47: INTERFAZ DE LA APLICACIÓN III

- *Se puede escoger qué descriptor será el utilizado:*
    - *Color histogram RGB:* se selecciona el valor de Bins entre los valores 2, 4, 8 y 16 (Bins debe ser una potencia de 2). Además se escogerá si los datos deben ser normalizados de forma L1, L2, o no ser normalizados.

      *La normalización L1:*H=H./sum(H), se divide elemento a elemento el histograma H entre la suma de H.

      La normalización L2: se utiliza la función *normc*, *H=nomrc(H),* el cual normaliza las columnas del histograma *H* a una longitud de 1.

    - *Descriptor híbrido LBP + RGB:* se selecciona los bins y la normalización de RGB. Para el patrón LBP se utilizará todas las combinaciones posibles con radio = 1, 2, 3, y modo = uniform

patterns (u2), rotation invariant (ri) o uniform rotation invariant (riu2).

- *Botón para Generar histogramas:* una vez escogida la imagen, y seleccionado el descriptor se selecciona este botón. Se asignan las labels a cada región y se generan los histogramas. Los resultados comienzan a generarse, para saber en qué punto se encuentra del proceso, se muestran una serie de mensajes con dicha información. A su vez se cargan los nuevos datos en la interfaz. La iteración de los mensajes se detalla a continuación.

- *Se puede escoger qué valor tomará K del clasificador KNN:* se selecciona el valor 1, 3 o 7. Los atributos de este clasificador traen el valor por defecto, ya que no existe una mejora significativa variando sus valores.

- *Botón Generar clasificación*: una vez generado los histogramas y escogido el valor de K, se selecciona este botón. El cual genera los conjuntos de entrenamiento y testeo, realiza la clasificación y muestra en forma de imágenes los resultados de las clases predichas para cada región de la imagen.

ILUSTRACIÓN 48: INTERFAZ DE LA APLICACIÓN IV

Mensaje Generando Labels: se está asignando un valor a cada región de la imagen seleccionada.

Mensaje Generando los histogramas: se están generando los histogramas asociados a la imagen y al patrón seleccionado.

Mensaje Generando conjunto de entrenamiento: es el paso previo a la clasificación, en el cual se genera el conjunto de entrenamiento y testeo para trabajar con Weka. Se realiza un *6-cross validation*, en el cual 5 de las imágenes se utilizan para generar el conjunto de entrenamiento y la restante para el testeo.

Mensaje Generando clasificación: se utiliza el clasificador seleccionado con los conjuntos anteriormente generados.

El resto de datos que son cargados durante la ejecución de la aplicación son los siguientes:

- Se describe el tamaño de la *imagen original* y el número de regiones de la *imagen segmentada*.

- *Histogramas:* se muestran los histogramas obtenidos, pudiendo seleccionarse el histograma de una región concreta.

- *Imagen obtenida:* es la composición de dos imágenes superpuestas, la imagen original y la imagen generada en base a los resultados obtenidos con el patrón seleccionado. Para una mejor interpretación, como se puede observar en la leyenda los resultados correctos se muestran en color rojo y los errores en color azul.

- *Imagen clasificación*: es la composición de dos imágenes superpuestas, la imagen original y la imagen generada en base a los resultados obtenidos al realizar la clasificación con Weka.

- *Resultados del clasificador*: para comprobar la eficacia del clasificador utilizado se muestra por pantalla el *recall*, *precisión* y *accuracy* obtenidos.

- *Tiempo de ejecución:* se muestra el tiempo que ha tardado en obtener los resultados.

A continuación se muestra un ejemplo de cómo queda la interfaz una vez que han finalizado las ejecuciones internas.

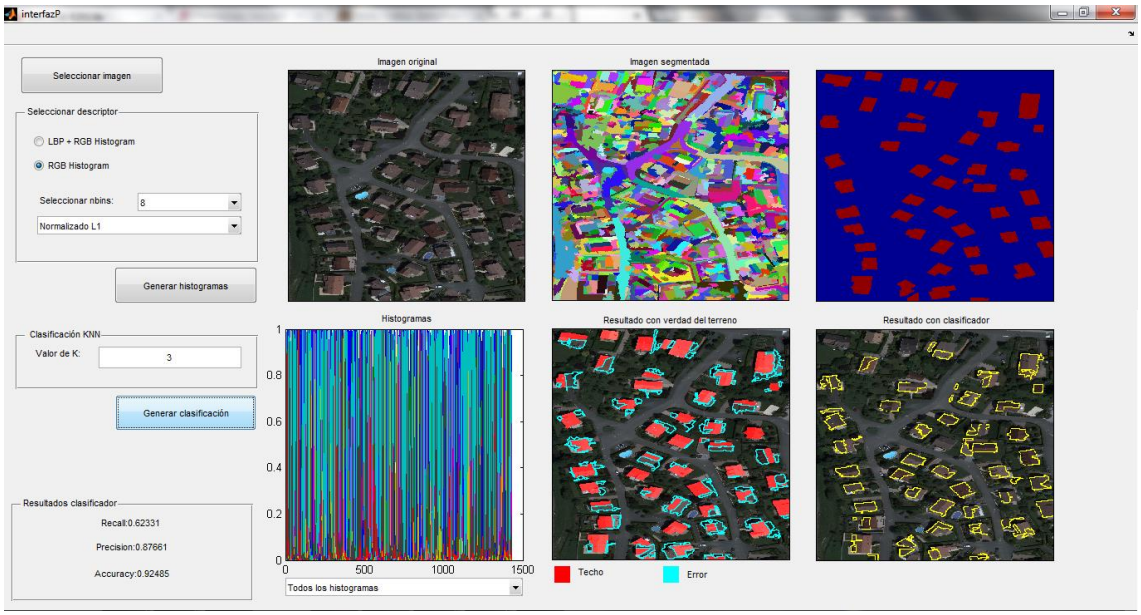

ILUSTRACIÓN 49: INTERFAZ DE LA APLICACIÓN V

CONCLUSIONES

En este proyecto se ha realizado la detección automática de objetos de interés en imágenes aéreas mediante el uso de descriptores característicos, siendo los tejados los objetos de interés. Los descriptores considerados para esta tarea han sido los histogramas color y descriptores LBP (Local Binary Pattern).

La extracción automática de objetos como edificios, rutas en imágenes aéreas es un campo que ha suscitado el interés de muchas investigaciones. Es bien conocido que la segmentación de edificios en imágenes aéreas es una tarea difícil. En el trabajo se proponen diferentes descriptores y clasificadores, en busca de aquel que ofrezca una tasa de acierto más alta.

La base de datos con la que se trabaja está compuesta por las imágenes originales, las imágenes de verdad del terreno, e imágenes de segmentación ya generadas.

Los resultados muestran que una combinación híbrida entre el descriptor LBP y color histogram ofrece los mejores resultados, existiendo en este caso una diferencia mínima entre el uso del clasificador KNN y SVM.

REFERENCIAS BIBLIOGRÁFICAS

[OPH96] T. Ojala, M. Pietik¨ainen, and D. Harwood. A comparativestudy of texturemeasureswithclassificationbasedonfeatureddistributions.PatternRecognition, 29:51–59, 1996.

[PIE2011] Local BinaryPatternsforStillImagesM. Pietikäinen et al., ComputerVisionUsing Local BinaryPatterns, ComputationalImaging and Vision 40, DOI 10.1007/978-0-85729-748-8_2.

[OPM02] T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution gray-scale and rotationinvarianttextureclassificationwith local binarypatterns. IEEE Trans. onPatternAnalysis and Machine Intelligence, 24(7):971–987, 2002.

[BUR98] Christopher J.C. Burges. A tutorial onsupport vector machines forpatternrecognition.

[FIX51] HodgesFix. Discriminatoryanalysis, nonparametricestimation: consistencyproperties. Technicalreport.

[EERMM2014]AnEficientAutomaticObjectRecognitionMethodBasedonRegion Similarity: Application to roofdetectionfromorthophotoplans.

[WEB 1] Web del departament of computerscience and engineering. www.cse.oulu.fi/mvg/downloads/lbpmatlab.

[WEB 2] Web de Weka. University of Waikatohttp://www.cs.waikato.ac.nz/ml/weka/

[WEB 3] Web de Matlab http://www.mathworks.es/

ANEXOS

ANEXO A: MATLAB

Matlab es un lenguaje de alto nivel que sirve para el cálculo numérico, la visualización y la programación. Permite analizar datos, desarrollar algoritmos y crear modelos. Además existen muchas toolboxes que sobre la base del núcleo de Matlab proporcionan nuevas funciones.

A continuación se dan ciertas pinceladas sobre el funcionamiento de Matlab que pueden ser de mucha utilidad:

- La ventana principal está compuesta por *Launch Pad, Command History y Command Window*. El símbolo >> que aparece en el *Command window* es el "prompt" de Matlab y significa que el programa está preparado para recibir órdenes.

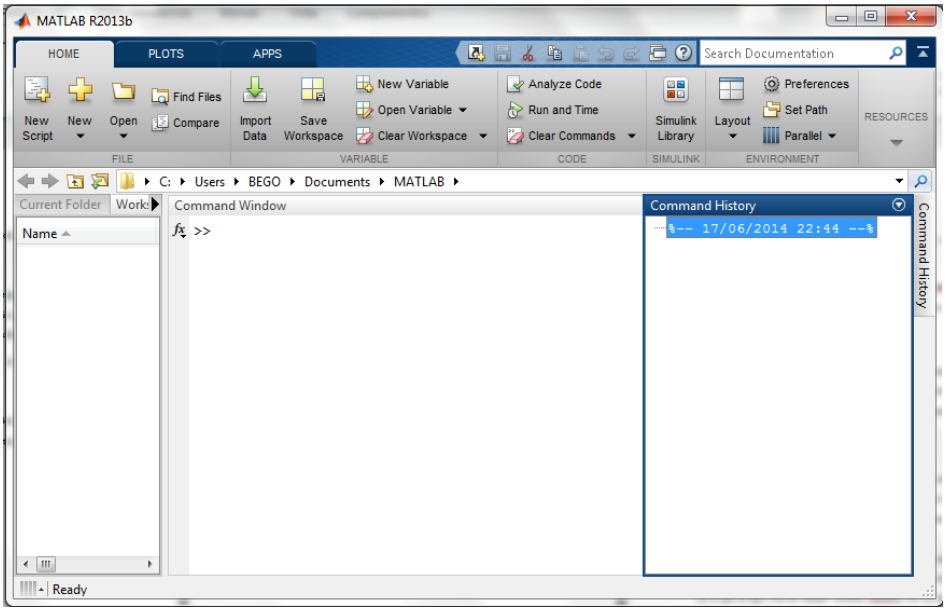

ILUSTRACIÓN 50: INTERFAZ MATLAB

- En el *workspace* se encuentra el área de memoria al cual se tiene acceso mediante los comandos *who* y *whos*.
- Al cerrar el programa todo el contenido de la memoria se borra automáticamente. Los comandos *save* y *load* permiten seguir trabajando con el mismo contenido.
- El carácter % indica el comienzo de un comentario.
- Mediante el *Path*se define la lista de directorios donde MATLAB debe buscar los ficheros de comandos y las funciones, tanto del sistema como de usuario.

Matrices:

La estructura de datos utilizada por Matlab es una matriz. Pueden ser introducidas de varias formas.

- Mediante una lista explícita de elementos.

  >> [16 3 2 13; 5 10 11 8; 9 6 7 12; 4 15 14 1]
  MATLAB responde:
  ans =
  16 3 2 13
  5 10 11 8
  9 6 7 12
  4 15 14 1

- Generando matrices con funciones predefinidas.
- Cargando matrices desde un fichero de datos externo.
- Creando matrices con funciones definidas por el usuario a través de un fichero.

Ficheros:

Los ficheros más comunes en Matlab son los *M-files*, que permiten guardar scripts y funciones. Una de sus características más importante es la posibilidad de referenciar otros ficheros del mismo tipo e incluso a sí mismos:

- *Sripts o ficheros de comandos*: no tienen argumentos de entrada ni de salida y operan con las variables del espacio de trabajo.

- *Funciones:* se diferencian de los scripts en que aceptan entradas y devuelven salidas. La estructura de una función sigue el siguiente esquema:

  function variables de salida= Nombre de la función(variables de entrada)
  %
  % Comentarios que definen completamente las variables que deben pasarse a la función
  % (variables de entrada) y el resultado que produce la función (variables de salida)
  %
  Cuerpo de la función: comandos de MATLAB

Imágenes

Se representa por una matriz, siendo cada elemento un píxel de la imagen. Matlab soporta distintos tipos de imágenes: binarias, RGB, indexadadas y de intensidad de gris:

- *Imágenes binarias:* es una imagen de intensidad que contiene los valores correspondientes al blanco y al negro. Para su representación los píxeles toman el valor 0 o 1.

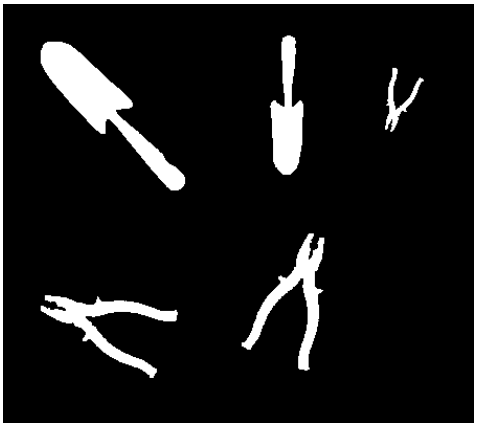

ILUSTRACIÓN 51: IMAGEN BINARIA

- *Imágenes RGB:* cada píxel de la imagen está representada por un conjunto de tres valores: intensidad de rojo, azul y verde, los cuales conforman el color. Por ello, son almacenadas en matrices de 3 dimensiones.

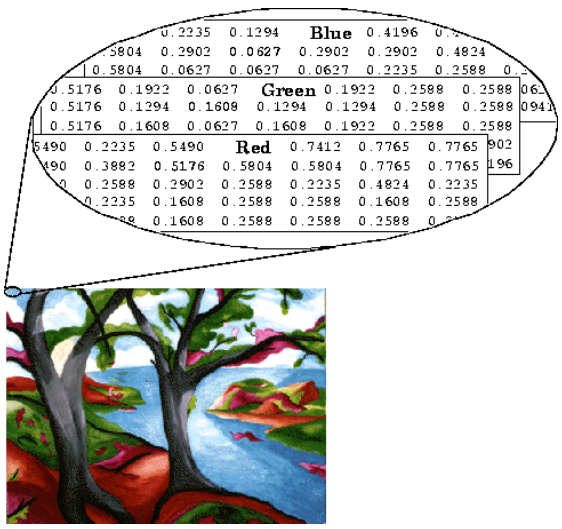

ILUSTRACIÓN 52: IMAGEN RGB

- *Imágenes indexadas:* está compuesta por dos matrices, la imagen y *colormap*. Siendo esta última un conjunto ordenado de los valores que representan los

colores de la imagen. Cada fila de la matriz *colormap* muestra los valores RGB de un determinado color.

- *Imágenes de intensidad de gris:* es una imagen de intensidad en la que cada píxel puede tomar un valor entre 0 y 255.

ILUSTRACIÓN 53: IMAGEN DE INTENSIDAD DE GRIS

Además cada imagen tiene asociado un histograma, es una representación de los distintos valores que puede tomar los píxeles de una imagen y cuántos píxeles se encuentran para ese valor en la imagen.

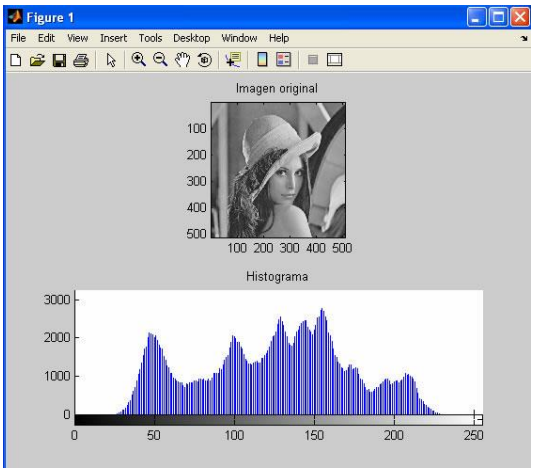

ILUSTRACIÓN 54: EJEMPLO DE HISTOGRAMA

Sistema de ayuda:

El sistema de ayuda que ofrece Matlab es muy completo, se pueden encontrar desde demos hasta ayuda de determinadas funciones o comandos. Para ello se debe escribir en *command window*, *help* seguido del nombre del comando o función.

Si no se sabe exactamente que se está buscando, el comando *helpwin* mostrara la estructura de carpetas en la que se organiza la ayuda y una descripción de cada una de ellas.

Interfaz gui:

Las GUI (interfaces gráficas de usuario) permiten la creación de interfaces de manera sencilla. Incluyen controles tales como menús, barras de herramientas, botones y controles deslizantes.

Una vez seleccionado la opción de GUI se presentan las siguientes opciones:

- *Blank GUI (Default):* interfaz gráfica de usuario en blanco.
- *GUI with Ui controls:* Esta opción presenta un ejemplo en el cual se han utilizado los *UI controls.*
- *GUI with Axes and Menu*: presenta un ejemplo en el cual se ha utilizado el menú File con las opciones Open, Print y Close.
- *Modal Question Dialog:* presenta un ejemplo de un cuadro de diálogo común, que se compone de una imagen, una etiqueta y dos botones.

Los ejemplos presentados anteriormente permiten ser ejecutados y modificados.

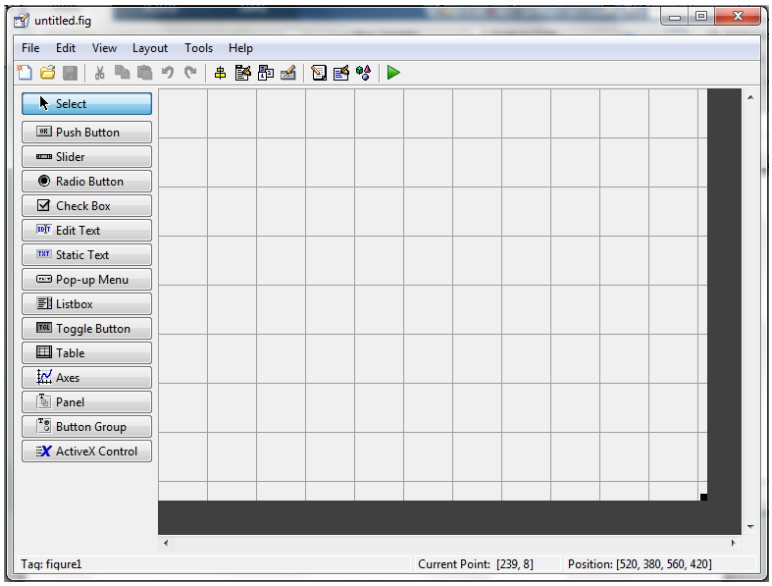

ILUSTRACIÓN 55: MATLAB GUI

Una aplicación se compone de dos archivos, un archivo con la parte gráfica (extensión .fig) el cual está asociado con un archivo ejecutable (extensión .m). Están unidas mediante las subrutinas *callback*. Para ejecutarlo basta con escribir el nombre del programa.

ANEXO B: WEKA

Weka (*Waikato Environment for Knowledge*) es una plataforma de software que se basa en el la minería de datos y el aprendizaje automático. Consta de un conjunto de herramientas que posibilita que la visualización y ejecución de algoritmos para el análisis de datos.

La gran ventaja es su licencia GNU (General Public Licence) pudiendo modificar el propio código e incluye una interfaz gráfica con las herramientas integradas. A continuación se describen las diferentes opciones que Weka ofrece.

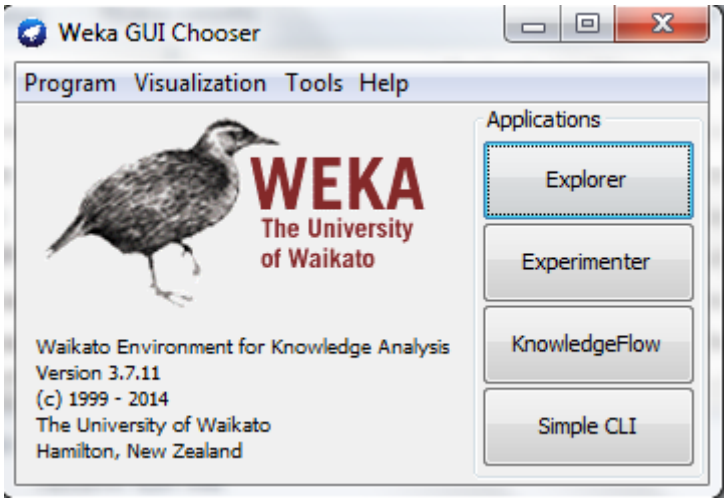

ILUSTRACIÓN 56: APLICACIÓN WEKA

Como puede observarse en la ilustración 3, Weka ofrece cuatro aplicaciones, Explorer, Experimenter, KnowledgeFlow y Simple CLI. A continuación se detalla las múltiples opciones que ofrece la aplicación Explorer que es la utilizada en este proyecto.

Explorer es una aplicación sencilla que permite manejar un único archivo de datos, ejecutando algoritmos de análisis ya implementados y realizando una ejecución independiente por cada prueba. Las tareas que permite realizar son Preprocess (aplicación de filtros), Classify (clasificación), Clustering, Associate (búsqueda de asociaciones), SelectAttributes (selección de atributos) y Visualize (visualización):

- *Preprocess:* en esta tarea se especifica el origen de los datos pudiendo escoger entre las siguientes opciones:
    - *Open file:* cargar un fichero de datos, el cual debe tener una extensión .arff,, .arff.gz, .names, .data, .csv, .libsvm, .dat, .bsi, .xrff o .xrff.gz.
    - *Open Url:* el fichero a cargar se encuentra en una dirección de Internet.
    - *Open BD:* se cargan los datos de una base de datos, definiendo su URL, usuario y contraseña.

Posteriormente si se selecciona *Choose*del apartaado *Filter*, se puede aplicar un filtro sobre los datos que sean seleccionados.

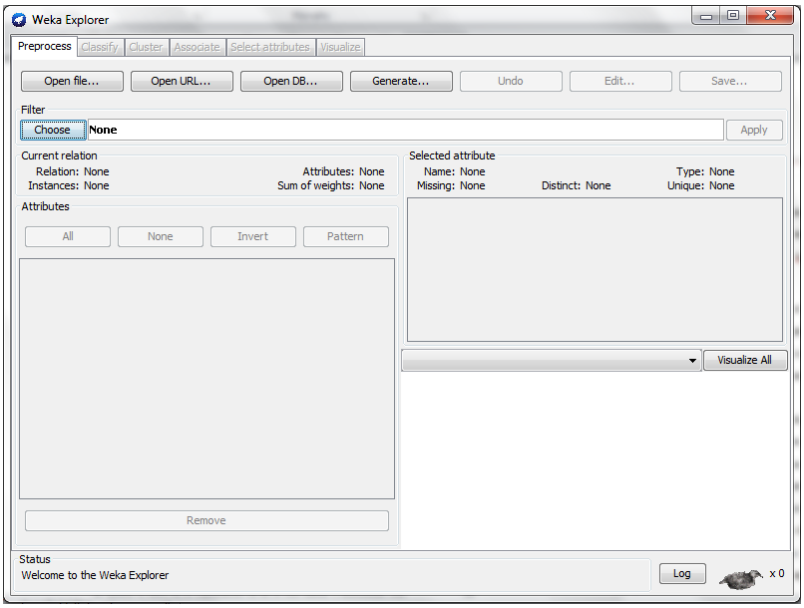

ILUSTRACIÓN 57: WEKA PREPROCESS

Una vez cargado los datos se puede acceder a la clasificación. Con la opción *Choose*se configuran los parámetros del clasificador seleccionado. Los modelos de evaluación que pueden ser utilizados son los siguientes:

- *Use training set*: el entrenamiento del algoritmo se realiza con todos los datos. Una vez generado el modelo, éste se evaluará con el mismo conjunto de datos.

- *Supplied test set*: la evaluación del modelo generado se realiza sobre un otro conjunto de datos distinto.

- *Cross-validation:* la evaluación del modelo generado se realiza con unavalidación cruzada.

- *Percentagesplit:* con este método se divide el conjunto de datos inicial en dos, el tamaño de los conjuntos está basado en el porcentaje seleccionado. El primer

conjunto será utilizado para la construcción del modelo, y el segundo conjunto para la evaluación del mismo.

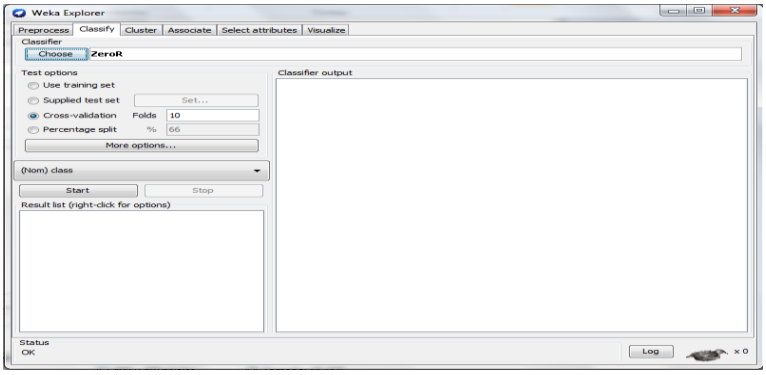

Ilustración 58: wekaclassify