Gobierno del Principado de Asturias
Consejería de Economía y Empleo

Unión Europea
Fondo Europeo
de Desarrollo Regional

Resultados Potenciación y Refuerzo

Línea especialización análisis inteligente de datos

En Gijón , a 31 de Diciembre de 2017

1. MEMORIA TÉCNICA

2016

Hito 1: Análisis

T1.1: Análisis de requisitos

En la primera tare de este hito se han analizado distintos requisitos necesarios que debe satisfacer la herramienta desarrollada en el proyecto, de modo que la valoración automática sea certera y útil.

En la siguiente tabla se muestran los requisitos del sistema, junto a su prioridad de aplicación (Alta/Obligatorio, Media/Recomendable, Baja/Opcional).

Adquisición e integración de datos		
RF-1	El sistema debe ser capaz de integrar datos cuyas especificaciones estén determinadas como estipula el Banco de España.	Alta
RF-2	El sistema debe ser capaz de geoposicionar el datos en estudio para la aplicación del método de comparación para la valoración.	Alta
RF-3	El sistema debe ser capaz de recopilar información de comparables.	Alta
RF-4	El sistema debe proporcionar a través de sistemas GIS comparables similares al datos en estudio de cara a la aplicación del método de comparación para la valoración.	Alta
RF-5	El sistema debe ser capaz de recopilar información de comparables a través de portales inmobiliarios en zonas donde ERP Gestión no pueda proporcionar un número suficientemente representativo.	Alta
RF-6	El sistema debe ser capaz de recoger información de redes sociales tales como *Twitter* para el posterior análisis de opinión de la zona.	Alta
Presentación y almacenamiento de la información		
RF-7	El sistema desarrollado debe ser capaz de presentar los resultados obtenidos a través de un informe interpretable, que utilice gráficos y tablas que permitan la comprensión del análisis por parte de cualquier usuario no experto en la materia.	Alta
RF-8	El sistema debe proporcionar al usuario una descripción detallada del estado demográfico de la zona a la que pertenece el dato.	Alta
RF-9	El sistema debe proporcionar al usuario un mapa geoposicionando al dato, así como a los comparables utilizados en caso de haberse utilizado el método de comparación.	Alta

RF-10	El sistema debe proporcionar al usuario información acerca de los comparables utilizados en caso de aplicar el método de comparación.	Alta

Extrapolabilidad		
RNF-1	El sistema será extrapolable a cualquier sector comercial para el que la estimación del valor de un producto dependa de forma directa de parámetros asociados a éste.	Alta
Interpretabilidad		
RNF-2	El sistema desarrollado debe permitir, en la medida de lo posible, que los usuarios finales interpreten la información recibida.	Alta
Rendimiento		
RNF-3	Los tiempos de ejecución de los algoritmos han de ser aceptables, proporcionando el valor de forma rápida.	Alta

Tabla 1: Requisitos que debe cumplir el sistema desarrollado

T1.2: Análisis de datos de entrada

ESTADO DEL ARTE

Información requerida

Si se pretende obtener una valoración, dichos datos han de estar correctamente recogidos y actualizados. Cada uno debe disponer de cierta información esencial. En este caso, estos atributos se dividen en dos grupos:

- **Datos geográficos:** son de gran importancia los mapas catastrales actualizados. Dicha información es muy importante en la aplicación de algunos métodos de valoración que se desarrollarán a continuación, tales como el método por comparación.
- **Datos sobre las características:** para la correcta valoración, toda la información característica de éste ha de estar disponible, tales como:
  - Superficie
  - Antigüedad
  - Calidad constructiva
  - Superficies secundarias (garajes, terrazas, etc.)
  - Características de la construcción (como el número de dormitorios y de baños)
  - Servicios disponibles
  - Instalaciones o equipamientos del sitio

  Dependiendo del modelo de valoración, serán necesarios datos adicionales que no serían indispensables mediante otros métodos.

La recogida de los datos ha de ser realizada por parte de los propios tasadores, o por personas formadas para tal fin.

Métodos de valoración convencionales

En este apartado se describirán distintos métodos existentes para la valoración: los métodos del coste, de comparación, de actualización de rentas y residual.

- **Método del coste**: este método es considerado uno de los más fiables para nuevas estructuras, produciendo estimaciones a partir del valor del terreno (obtenido mediante los métodos de comparación o residual), del coste de la edificación o de las obras de rehabilitación (obtenido mediante el coste de construcción por contrata) y de los gastos para el reemplazamiento (tales como impuestos para la obra o costes de licencias).
  La correcta aplicación de este método exige que los datos estén bien recogidos a la vez que actualizados. El mayor problema de dicho método reside en el cálculo de valores tales como el del terreno, al poder implicar la existencia de subjetividad.

- **Método de comparación**: dicho método estima el valor mediante el análisis de otros cuyas propiedades constructivas sean similares, así como sus características geográficas. Se trata de un método ampliamente utilizado, así como aquellas clases de construcciones suficientemente representadas.
  Este método tiene como principal característica tomar como comparables aquellos datos que satisfacen ciertas propiedades. Se seleccionan que estén en la misma zona, siempre y cuando haya suficiente representación. En caso contrario, se buscarán casos en zonas con unas características similares.
  Se seleccionarán aquellos con unas características parecidas, y en función del grado de semejanza de cada uno respecto al principal, se obtendrá el valor.

- **Método de actualización de rentas**: este método es uno de los más adecuados para propiedades que producen ingresos, siempre y cuando se dispongan de datos fiables respecto a dichos ingresos y gastos.

- **Método residual**: este método es utilizado principalmente en casos en los que el método de comparación no resulta factible para la estimación del valor del suelo, como puede ser el caso de terrenos destinados a posibles usos lucrativos.
  Dicho método basa la obtención del valor de la propiedad en los siguientes términos: valor de suelo, coste de la construcción, gastos y beneficio, siendo el valor de venta la suma de los valores previos.

FUENTES DE ORIGEN

Se han agrupado las distintas fuentes en cinco apartados, relacionadas cada una de ellas con una de los ámbitos del Proyecto.

Visor GIS

El Visor GIS proporcionará para cada una de las informaciones a analizar su zona homogénea de valor, a partir de la cual, también debe suministrar los comparables más similares al estudio. A través de dichos datos, se podrán desarrollar de forma óptima los modelos.

Twitter

Para el análisis de opiniones, desarrollado de forma más detallada al final de esta tarea, será necesaria la recogida de información de redes sociales. En este caso, la más informativa es la red social de *microblogging Twitter*.

Se procede a la recogida de los *tweets* (mensajes de máximo 140 caracteres) a través de la API de *Twitter* (https://dev.twitter.com/). Mediante el software R, se ha procedido a hacer uso de dicha API para poder obtener los distintos mensajes de interés para el análisis de sentimientos.

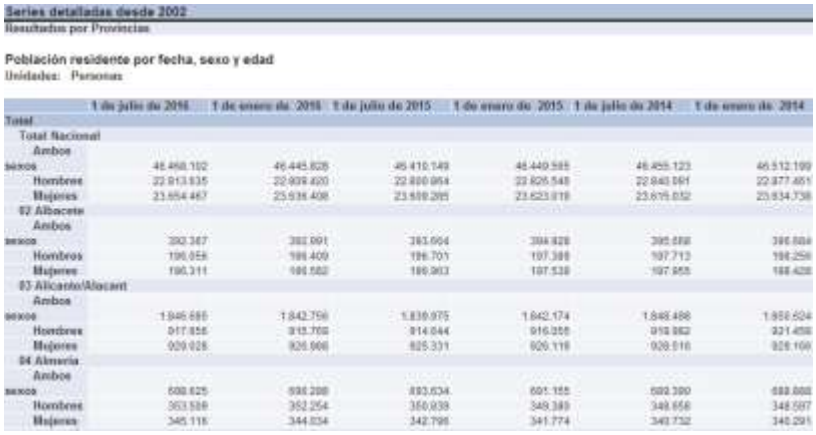
INE (Instituto Nacional de Estadística)

El Instituto Nacional de Estadística (INE) proporciona diversos datos a nivel nacional que son actualizados periódicamente, y que se requieren para la generación de diversas gráficas asociadas a la localización y el número de habitantes del dato estudiado.

Población residente por provincia, fecha, sexo y edad

Proporciona los datos de población residente de España, distinguiendo por:

- **Provincia**: 50 provincias, 2 ciudades autónomas, total nacional.
- **Fecha**: datos hasta 2002, actualizados semestralmente.
- **Sexo**: hombres, mujeres y ambos.
- **Edad**: desde los 0 años, hasta los 99, agrupando finalmente en 100 o más. Incluye el caso general.



Series detalladas desde 2002
Resultados por Provincias

Población residente por fecha, sexo y edad
Unidades: Personas

	1 de julio de 2016	1 de enero de 2016	1 de julio de 2015	1 de enero de 2015	1 de julio de 2014	1 de enero de 2014
Total						
Total Nacional						
Ambos sexos	46.468.102	46.445.828	46.410.149	46.449.565	46.455.123	46.512.199
Hombres	22.813.935	22.809.420	22.800.954	22.826.548	22.840.091	22.877.461
Mujeres	23.654.467	23.636.408	23.609.295	23.623.018	23.615.032	23.634.738
02 Albacete						
Ambos sexos	392.347	392.091	393.604	394.828	395.658	396.604
Hombres	196.956	196.409	196.701	197.388	197.713	198.259
Mujeres	195.311	195.682	196.903	197.538	197.955	198.428
03 Alicante/Alacant						
Ambos sexos	1.846.685	1.842.756	1.839.975	1.842.174	1.848.498	1.858.624
Hombres	917.856	915.708	914.644	916.055	919.882	921.459
Mujeres	928.828	926.998	925.331	926.119	928.616	928.166
04 Almería						
Ambos sexos	698.625	698.288	693.634	691.155	689.399	688.888
Hombres	353.539	352.254	350.939	349.389	348.656	348.597
Mujeres	345.116	344.034	342.795	341.774	340.732	340.291

Figura 4: Fragmento de la tabla de población residente por provincia, fecha, sexo y edad

Cifras de población por municipio

Proporciona los datos de población residente en cada municipio de España. Incluye los siguientes datos:

- Código provincial.
- Nombre de la provincia.
- Código del municipio.
- Nombre del municipio.
- Población total.
- Población de hombres.
- Población de mujeres.

Dichos datos son obtenidos a través de la revisión del padrón municipal anual.

Cifras de población resultantes de la Revisión del Padrón municipal a 1 de enero de 2016

CPRO	PROVINCIA	CMUN	NOMBRE	POBLACION	HOMBRES	MUJERES
02	Albacete	001	Abengibre	761	366	395
02	Albacete	002	Alatoz	582	325	257
02	Albacete	003	Albacete	172426	84366	88060
02	Albacete	004	Albatana	714	367	347
02	Albacete	005	Alborea	710	365	345
02	Albacete	006	Alcadozo	688	367	321
02	Albacete	007	Alcalá del Júcar	1217	653	564
02	Albacete	008	Alcaraz	1463	741	722
02	Albacete	009	Almansa	24800	12332	12468
02	Albacete	010	Alpera	2289	1177	1112
02	Albacete	011	Ayna	690	368	322
02	Albacete	012	Balazote	2346	1202	1144
02	Albacete	014	Ballestero, El	427	218	209
02	Albacete	013	Balsa de Ves	155	87	68
02	Albacete	015	Barrax	1903	971	932
02	Albacete	016	Bienservida	634	332	302
02	Albacete	017	Bogarra	920	486	435

Figura 5: Fragmento de la tabla de población por municipio y sexo

Precio medio del metro cuadrado de suelo urbano

Estos datos son actualizados trimestralmente, proporcionando la información desde el año 2004, para comunidades autónomas, provincias y nivel nacional. Dichos datos se agrupan en dos clases:

- Caso general. Proporciona el precio medio del metro cuadrado de suelo urbano para cada provincia, comunidad autónoma y el total nacional, además de la variación trimestral e interanual.

Tabla 4.
Precio medio del metro cuadrado de suelo urbano por comunidades autónomas y provincias

Figura 6: Fragmento de la tabla de precio medio del metro cuadrado de suelo urbano por comunidades autónomas y provincias

- Por franjas de habitantes. Se trata de 5 tablas en las cuales se tiene en cuenta el precio medio del metro cuadrado de suelo urbano en municipios de una determinada población a nivel de comunidad autónoma, provincia y nacional. Las franjas tenidas en cuenta en dichas tablas son:

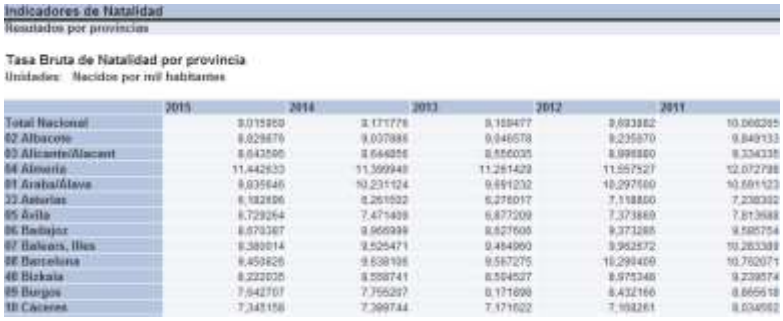
- Municipios de hasta 1000 habitantes.
- Municipios entre 1000 y 5000 habitantes.
- Municipios entre 5000 y 10000 habitantes.
- Municipios entre 10000 y 50000 habitantes.
- Municipios de más de 50000 habitantes.

Tabla 4.2.
Precio medio del metro cuadrado de suelo urbano en municipios de entre 1.000 y 5.000 habitantes

Figura 7: Fragmento de la tabla de precio medio del metro cuadrado de suelo urbano por comunidades autónomas y provincias en municipios entre 1000 y 5000 habitantes

Tasas de natalidad por provincia

Datos de tasa bruta de natalidad (nacidos por 1000 habitantes) por provincia y a nivel nacional con resultados anuales, y un histórico hasta 1975.



Indicadores de Natalidad
Resultados por provincias

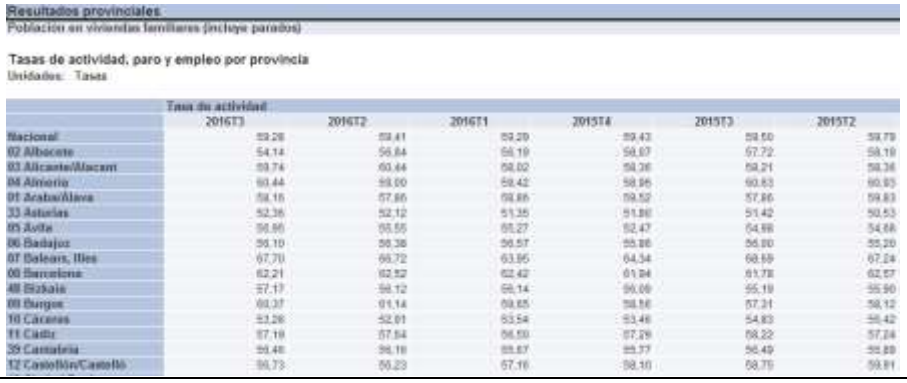
Tasa Bruta de Natalidad por provincia
Unidades: Nacidos por mil habitantes

	2015	2014	2013	2012	2011	
Total Nacional	9,015969	9,171778	9,169477	9,693882	10,066295	
02 Albacete	8,829876	9,007886	9,046578	9,235870	9,849133	
03 Alicante/Alacant	8,643596	8,644856	8,656035	8,999880	9,334335	
04 Almería	11,442633	11,399940	11,261429	11,557527	12,072798	
01 Araba/Álava	8,935646	10,231124	9,691202	10,297160	10,691123	
33 Asturias	6,182896	6,261032	6,276017	7,118800	7,238302	
05 Ávila	6,729264	7,471408	6,877209	7,373869	7,813588	
06 Badajoz	8,670367	8,965939	8,927606	9,373285	9,595754	
07 Balears, Illes	8,380014	9,525471	9,464960	9,962572	10,283389	
08 Barcelona	9,459826	9,638106	9,567275	10,299409	10,762071	
48 Bizkaia	8,222035	8,588741	8,594527	8,975348	9,239574	
09 Burgos	7,642707	7,755287	8,171699	8,432166	8,665518	
10 Cáceres	7,345158	7,399744	7,171622	7,958261	8,034982	

Figura 8: Fragmento de la tabla de tasas de natalidad por provincia

Tasas de actividad, paro y empleo por provincia

Datos de las tasas de actividad, paro y empleo por provincia y a nivel nacional con resultados trimestrales, y un histórico hasta 2002.



Resultados provinciales
Población en viviendas familiares (incluye parados)

Tasas de actividad, paro y empleo por provincia
Unidades: Tasas

	Tasa de actividad					
	2016T3	2016T2	2016T1	2015T4	2015T3	2015T2
Nacional	59,28	59,41	59,29	59,43	59,50	59,79
02 Albacete	54,14	56,84	56,19	56,87	57,72	58,19
03 Alicante/Alacant	59,74	60,44	58,02	59,36	59,21	58,36
04 Almería	60,44	59,00	59,42	58,96	60,63	60,85
01 Araba/Álava	58,16	57,86	58,86	59,52	57,86	59,83
33 Asturias	52,36	52,12	51,35	51,80	51,42	50,53
05 Ávila	56,86	55,55	55,27	52,47	54,68	54,68
06 Badajoz	56,19	56,38	56,57	55,86	56,00	55,20
07 Balears, Illes	67,70	66,72	63,95	64,34	66,69	67,24
08 Barcelona	62,21	62,52	62,42	61,94	61,78	62,57
48 Bizkaia	57,17	56,12	56,14	56,09	55,19	55,90
09 Burgos	56,37	61,14	59,65	58,56	57,31	58,12
10 Cáceres	53,26	52,61	53,54	53,46	54,83	55,42
11 Cádiz	57,19	57,54	56,59	57,29	58,22	57,24
39 Cantabria	56,46	56,19	55,67	55,77	56,49	55,89
12 Castellón/Castelló	56,73	56,23	57,16	58,10	58,75	59,91

Figura 9: Fragmento de la tabla de tasas de actividad, paro y empleo por provincia

Callejero del censo electoral

Callejero de toda España, incluyendo información de todas las vías y tramos de vía. Es actualizado semestralmente, pudiendo acceder a un fichero con las variaciones exclusivamente entre el semestre actual y el anterior. Dicho callejero está formado por cuatro archivos:

- Fichero de vías.
- Fichero de pseudovías.
- Fichero de tramos de vías.
- Fichero de unidades poblacionales.

FUENTES DE OPINIÓN

El análisis de opiniones y sentimientos en las redes sociales se utiliza cada vez más con el objetivo de "monitorizar" o conocer la opinión pública sobre ciertos temas (marcas, productos, sucesos, etc.). Se ha optado por el análisis de la red social *Twitter* por ser la red social que más volumen de opinión recoge, así como por su dinamismo y facilidad a la hora de recabar la información.

La propia aplicación de *Twitter* ofrece un apartado dedicado para desarrolladores ("*Twitter Developer Documentation*", www.dev.twitter.com/docs), donde se documenta la API que ofrecen, su alcance, sus limitaciones y política de cobros. La licencia gratuita limita el ratio de consultas a un límite de llamadas cada 15 minutos. Si se sobrepasa ese límite, la propia aplicación bloquea al usuario hasta que se finalicen los 15 minutos.

Software utilizado

Para realizar las tareas de consulta, descarga, registro y análisis de opiniones a partir de la red social *Twitter* se ha utilizado el lenguaje de programación R.

Este lenguaje o entorno de programación dispone de un paquete llamado "twitteR" que permite acceder a la API de *Twitter* con el objetivo de consultar, descargar y recopilar los *tweets* consultados. La principal ventaja de empleo de este lenguaje de programación es integrar este registro de *tweets* con el análisis de opiniones y sentimientos posterior.

Como principal limitación de la API de *Twitter* con licencia gratuita, mencionar que solamente se pueden consultar los *tweets* de los últimos 9 días en las búsquedas usuales, si bien permite descargar mensajes con mayor antigüedad si se centra en los mensajes publicados por un usuario concreto, conocidos comúnmente por *timeline*, cuya limitación en dicho caso es de un total de 3200 mensajes.

Consulta, descarga y registro de opiniones

Como se ha especificado en el apartado anterior, desde el propio entorno de R, y gracias al paquete "twitteR", consultamos y descargamos los *tweets* que nos interesan (aquellos que cumplen unas determinadas características, es decir, se han hecho en una fecha determinada dentro de los últimos 9 días, contienen alguna palabra clave o *hashtag*, etc.).

Para conectarse a la API de *Twitter* es necesario crear una nueva aplicación (apps.twitter.com). Para ello será necesario tener una cuenta de *Twitter*. Una vez creada la aplicación, *Twitter* le asignará una "key".

Estos y algunos otros datos disponibles desde el panel de administrador de la aplicación de *Twitter*, son los que se introducen en nuestro entorno de desarrollo de R, para poder conectarnos a la API y descargar la información deseada.

Análisis de opiniones y sentimientos

Una vez que se ha accedido a la API de *Twitter* y se han descargado los *tweets* deseados, se procede a su análisis, con el objetivo de extraer el contexto de aplicación, así como realizar un análisis de sentimientos. Para ello, será necesario hacer una fuerte labor de procesado que separe todos las palabras de un *tweet*, elimine todas aquellas sin contenido o que no aporten valor analítico, lo que se conoce como "palabras vacías" o en inglés *stopwords* (artículos, pronombres, preposiciones, etc.), signos de puntuación, eliminación de símbolos no alfanuméricos, etc.

Nube de palabras

Con el conjunto o bolsa de palabras resultante podemos hacer un análisis estadístico de la frecuencia de aparición de cada una de ellas, lo que indica de manera directa los temas más importantes o que generan más volumen de opinión en referencia al término o *hashtag* de búsqueda. Una manera muy popular y gráficamente muy visual de representar este estudio estadístico es mediante nubes de palabras o *wordclouds*. Una nube de palabras es una representación visual de las palabras que conforman un texto o conjunto de textos, donde el tamaño es mayor para las palabras que aparecen con mayor frecuencia y la ubicación nos indica la relación que tiene con las palabras contiguas. A modo de ejemplo podemos ver la nube de palabras resultante si rastreamos el *hashtag* #construccion:

Figura 10: Nube de palabras correspondiente al *hashtag* #construccion

A simple vista, y sin entrar a analizar manualmente cada uno de los n *tweets* descargados, sabemos que la opinión pública se centraba en los temas "arquitectura", "ingeniería", "grandes obras", "industria", "empresa", etc., así como entidades como "eurofinsa".

Análisis de sentimientos

A continuación, y a partir del conjunto de palabras resultante, se realiza el proceso de *stemming*, que consiste en extraer la raíz o *stem* de cada palabra, con el objetivo de que el género, el número o las conjugaciones verbales no afecten al análisis posterior.

Una vez realizada la tarea de *stemming*, se procede a comparar cada palabra con un corpus de expresiones catalogadas como positivas y con otro corpus de expresiones catalogadas como negativas. En función del número de coincidencias que presente cada *tweet* con palabras positivas o palabras negativas, se establece la polaridad del mensaje, pudiendo ser un *tweet* positivo, negativo o neutro (en caso de que ninguna polaridad destaque por encima de la otra).

De este modo podemos comprobar la reacción popular que provoca un determinado tema, así como analizar la tendencia de esa reacción con el paso de los días, semanas o meses. A modo de ejemplo, la siguiente imagen muestra la evolución durante nueve días de la polaridad de opinión pública asociada al *hashtag* #construccion. En el eje de abscisas se representa el porcentaje de *tweets* clasificados con cada polaridad, mientras que en el eje de ordenadas se representa la evolución temporal:

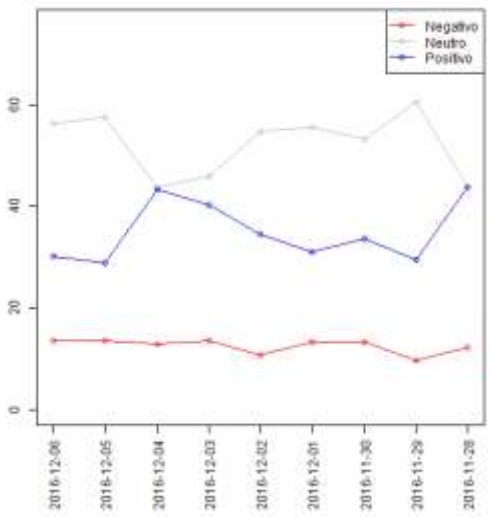


Figura 11: Evolución temporal de la polaridad de las opiniones relacionadas con el *hashtag* #construccion

Tal y como se puede comprobar, en este caso el término rastreado es muy genérico, y no crea especial controversia. De hecho, la mayor parte de las opiniones son catalogadas como neutrales o positivas, ya que no está referido a ningún escándalo, desastre o accidente. El análisis de sentimientos es especialmente interesante en aquellos casos en los que un tema es especialmente sensible a la opinión pública, ya sea de manera positiva o negativa, pudiendo cuantificar la dimensión.

Hito 2: Desarrollo

T2.1: Desarrollo del sistema maestro

Software utilizado

Para realizar las tareas de pre-procesado de los datos y modelado de los algoritmos se ha utilizado el lenguaje de programación R.

R es un lenguaje y entorno de programación estadístico basado en software libre, que ofrece una amplia variedad de técnicas estadísticas (modelado lineal y no lineal, test estadísticos, análisis de series temporales, clasificación, *clustering*, etc.) y técnicas de representación gráficas. Además, está basado en código abierto y permite su conexión con multitud de aplicaciones, tales como *Google Maps, Twitter*, etc.

R está disponible como software libre bajo los términos de la Licencia Pública General de GNU de la *Free Software Foundation* en forma de código fuente. Se compila y se ejecuta en una amplia variedad de plataformas *UNIX* y sistemas similares (incluyendo *FreeBSD* y *Linux*), *Windows* y *MacOS*.

Tal y como puede verse en la siguiente figura, R es la herramienta principal a la hora de realizar análisis inteligente de datos:

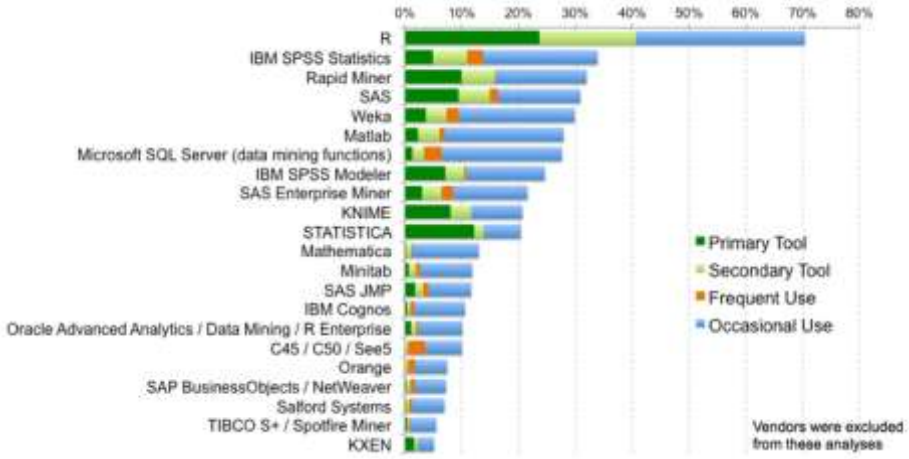


Figura 12: Tasa de uso de las diferentes herramientas de análisis

R es un conjunto integrado de servicios de software para la manipulación de datos, cálculo y representación gráfica que incluye:

- Un manejo eficaz de los datos y su almacenamiento.
- Un conjunto de operadores para cálculos con matrices.
- Una gran colección integrada de herramientas para el análisis de datos.
- Herramientas gráficas para el análisis y visualización de datos, ya sea en la pantalla o en copia impresa.
- Un lenguaje de programación simple y eficaz que incluye expresiones condicionales, bucles, funciones recursivas definidas por el usuario y las funciones de entrada y salida de datos.

R está diseñado como un lenguaje de programación, y permite a los usuarios añadir funcionalidad adicional mediante la definición de nuevas funciones. Además, R puede ser extendido fácilmente a través de paquetes. Hay más de 6000 paquetes disponibles a través de la web CRAN (*The Comprehensive R Archive Network*, un conjunto de servidores donde se almacena toda la documentación y código fuente

actualizado de R) que cubren una gama muy amplia de la estadística moderna. En particular, para este proyecto ha sido especialmente interesante, entre muchos otros, la utilización de los siguientes paquetes:

- **RODBC**: permite la conexión con la base de datos a través del puerto ODBC, que en este caso se ha construido en *Oracle*.
- **ggmap**: permite la conexión con la API de *Google Maps*, para geoposicionar y los comparables que no incorporen esa información.
- **plotGoogleMaps**: permite la elaboración de mapas interactivos a partir del visor de *Google*.
- **twitteR**: permite la conexión, consulta y descarga de mensajes publicados en la red social *Twitter*.

Por último, para las tareas computacionalmente intensivas, R puede ser vinculado con interfaces *Big Data*.

2017

Hito 2: Desarrollo

T2.1: Desarrollo del sistema maestro (continuación)

PRE-PROCESADO

Filtrado

La primera fase ha consistido en analizar cuidadosa y detalladamente los datos recibidos. A continuación, se presenta un listado de algunas de las operaciones de filtrado llevadas a cabo sobre el conjunto de datos de entrada, correspondiente a la categoría del dato, con el objetivo de preparar los datos para un posterior entrenamiento de los algoritmos:

- Eliminación del estudio de todos los elementos que no correspondan a viviendas (protegidas, primera residencia y segunda residencia).
- Eliminación de todas aquellas instancias que vienen etiquetadas como pisos (V94) pero corresponden a otros elementos constructivos, como por ejemplo terrazas.
- Eliminación de aquellas instancias cuyo número de baños y número de dormitorios es igual a 0.
- Reemplazo del valor “999” por el valor “0” en aquellos que presentan este dato en el atributo “planta”.
- Creación del atributo de entrada “antigüedad” en base al atributo “año de construcción”.
- Cálculo de la geoposición (latitud + longitud) de aquellos inmuebles que no tienen estos atributos informados o con un formato correcto.

Se ha trabajado con algunas aplicaciones, como la API de Google u Open Street Map, para extraer esta información a partir de la dirección completa (ciudad, calle, portal, etc.).

Selección de atributos

A parte de lo anterior, en esta etapa de pre-procesado se analiza el conjunto entero de atributos de entrada, analizando dependencias, correlaciones, grado de relevancia de cada uno, etc. Este proceso es conocido como "selección de atributos" y se realizó utilizando la herramienta "Weka", software desarrollado por la Universidad de Waikato.

A partir de este proceso de selección de atributos, se obtuvo que el listado de atributos a tener en cuenta en el proceso de entrenamiento de algoritmos es el siguiente:

- Tipo de vivienda
- Estado de construcción
- Antigüedad
- Superficie adoptada
- Número de baños
- Número de dormitorios
- Existencia o no de calefacción
- Existencia o no de aire acondicionado
- Calidad constructiva
- Estado de conservación
- Existencia o no de ascensor
- Número de planta
- Número de plazas de garaje
- Existencia o no de trastero
- Rehabilitación
- Precio medio unitario de los comparables seleccionados dentro de su zona homogénea de valor

T2.2: Desarrollo del modelo matemático

Con la información filtrada y procesada, entrenamos y validamos los siguientes tipos de algoritmos: regresión lineal, M5P, M5Rules, red neuronal, Random Forest y reglas difusas. Como se puede comprobar, se ha buscado entrenar una representación de los algoritmos más característicos empleados en Inteligencia Artificial, atendiendo a su interpretabilidad, precisión, etc. De este modo, algunos de los anteriores algoritmos corresponden a conjuntos de reglas, árboles de decisión, o "cajas negras". A continuación, se detalla brevemente cada uno de los algoritmos seleccionados.

Regresión lineal

La regresión lineal es una técnica estadística muy utilizada para estudiar la relación entre variables, pues se adapta a una amplia variedad de situaciones. Tanto en el caso de dos variables (regresión simple) como en el de más de dos variables (regresión multivariable), el análisis de regresión lineal puede utilizarse para explorar y cuantificar la relación entre una variable llamada dependiente y una o más variables llamadas independientes o predictoras, así como para desarrollar una ecuación lineal con fines predictivos.

En nuestro caso en particular, se trata de un problema multivariable, donde la ecuación de regresión no la define una recta en el plano, sino un hiperplano en un espacio multidimensional. El modelo de regresión

lineal multivariable es idéntico al modelo de regresión lineal simple, donde la única diferencia es la aparición de más variables explicativas.

- Modelo de regresión simple:

$$y = b_0 + b_1 \cdot x + u$$

- Modelo de regresión multivariable:

$$y = b_0 + b_1 \cdot x_1 + b_2 \cdot x_2 + b_3 \cdot x_3 + \ldots + b_k \cdot x_k + u$$

Árboles de regresión

Los árboles de regresión son modelos que predicen el valor de una variable de destino en función de diversas variables de entrada. Dada una base de datos se elaboran diagramas de construcciones lógicas, muy similares a los sistemas de predicción basados en reglas, que sirven para representar y categorizar una serie de condiciones que ocurren de forma sucesiva. Estos diagramas se utilizarán en la resolución de un problema en concreto. Un árbol de regresión tiene unas entradas, que pueden ser un objeto o una situación descrita por medio de un conjunto de atributos, y, a partir de estos datos, devuelve una respuesta, que en última instancia es la decisión tomada. Un árbol de decisión lleva a cabo un test a medida que se recorre desde la raíz hacia las hojas para alcanzar así la decisión.

El árbol de regresión suele contener nodos internos, nodos de probabilidad, nodos hojas y arcos. Un nodo interno contiene el test sobre algún valor de una de las propiedades. Un nodo de probabilidad indica que debe ocurrir un evento aleatorio de acuerdo a la naturaleza del problema. Este tipo de nodos es redondo, mientras los demás son cuadrados. Un nodo hoja representa el valor que devolverá el árbol de regresión y, finalmente las ramas brindan los posibles caminos que se tienen de acuerdo a la decisión tomada.

Los resultados de los árboles de regresión dependen de la profundidad de los mismos. Es necesario ajustar este parámetro para evitar la existencia de *overfitting.*

En este proyecto se han abordado tres aproximaciones a los árboles de regresión. Cada una de estas aproximaciones se describe a continuación:

- **M5P**

El M5P es una reconstrucción del algoritmo M5 de Quinlan para inducir árboles a partir de modelos de regresión. El M5P combina un árbol de regresión convencional con la posibilidad de tener funciones de regresión lineal en cada una de los nodos del árbol.

- **M5Rules**

Este algoritmo genera una lista de decisión para problemas de regresión utilizando el algoritmo *divide-and-conquer*. En cada iteración, se construye un árbol modelo utilizando el método M5 de Quinlan y transforma la mejor hoja en una regla.

- **Random Forest**

Es una combinación de árboles de regresión tal que cada árbol depende de los valores de un vector aleatorio probado independientemente y con la misma distribución para cada uno de estos, de modo que el resultado final se calcula a partir de una media ponderada del conjunto de árboles.

Los resultados de los modelos de *Random Forest* dependen del número de árboles a considerar, así como de la profundidad de los mismos. Es necesario ajustar estos parámetros para evitar la presencia de *overfitting.*

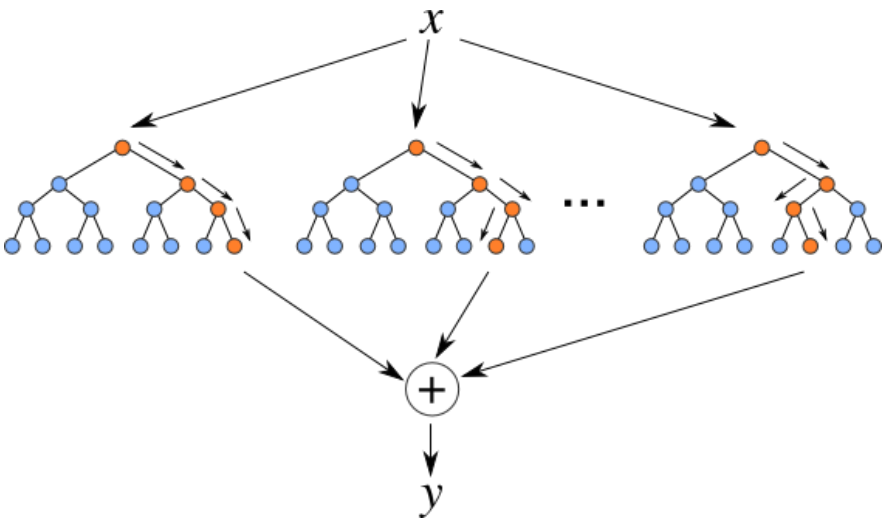


Figura 13: Representación gráfica de un Random Forest

Red neuronal

Las redes de neuronas artificiales son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales. Se trata de un sistema de interconexión de nodos que colaboran entre sí para producir un estímulo de salida. En inteligencia artificial es frecuente referirse a ellas como redes de neuronas o redes neuronales.

Una red neuronal se compone de una sucesión de capas formadas por distintas unidades llamadas neuronas. Cada neurona recibe una serie de valores de entrada a través de interconexiones con las neuronas de la capa precedente y emite una salida que servirá de entrada en capas sucesivas. En la Figura 14 se muestra el esquema general de una red neuronal con n neuronas de entrada y una única neurona de salida.

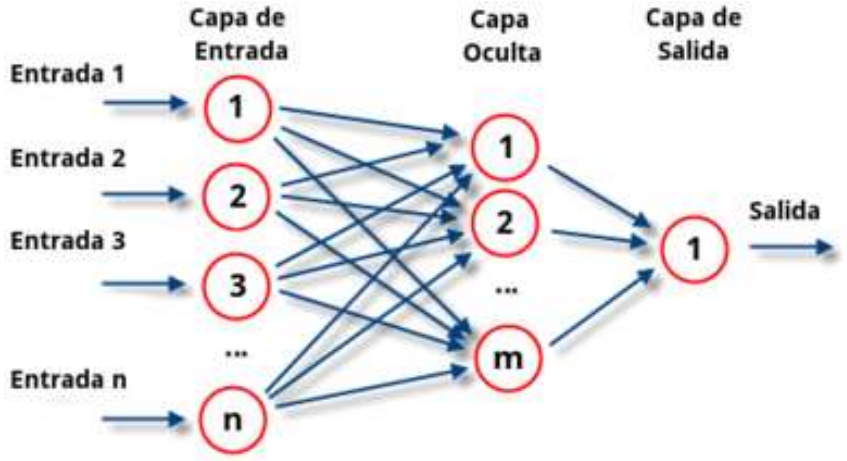


Figura 14: Esquema general de una red neuronal

Los valores de entrada a cada una de las neuronas sufrirán varias transformaciones antes de su posterior salida. Estas transformaciones vienen dadas por las siguientes funciones:

- Función de propagación o excitación: generalmente, consiste en el sumatorio de cada entrada por el peso de su interconexión. En el caso de que el peso sea positivo, se dirá que la conexión es excitatoria, mientras que, si es negativa, se dirá que la conexión es inhibitoria.
- Función de activación: es una función que modifica el valor generado por la función de propagación. Servirá para acotar los valores de salida de la neurona. Su definición dependerá de la interpretación que se quiera dar a la salida. Por ejemplo, algunas de las funciones de activación más comunes son:
  - Función sigmoide: sirve para obtener valores en el intervalo [0, 1]. Se puede interpretar como una probabilidad.
  - Función tangente hiperbólica: se utiliza para que los valores queden acotados en el intervalo [-1, 1].

Lógica difusa

La lógica difusa o *Fuzzy Logic* tiene como principal objetivo manejar la imprecisión y la variabilidad inherente a los datos, así como modelar aquellos fenómenos en los cuales obtener y plantear los modelos matemáticos que explican el comportamiento del mismo resulta una tarea difícil o imposible de realizar.

Por tanto, la lógica difusa puede ser empleada a la hora de modelar la estimación del precio de una vivienda mediante el uso de variables lingüísticas asociadas a las variables de entrada dentro de sistemas de reglas difusos, con el fin de describir la estimación mediante el uso del lenguaje natural. El empleo de variables lingüísticas y reglas difusas en lenguaje natural facilitan la interpretabilidad del modelo, haciendo posible entender las razones de ese valor estimado.

Máquinas de vectores soporte

Las máquinas de vectores de soporte (SVM, del inglés *Support Vector Machines*) son un conjunto de algoritmos de aprendizaje supervisado que se aplican en problemas de clasificación y regresión. La

aplicación más sencilla de una SVM consistiría en un problema de clasificación binaria, donde la SVM separaría las clases en dos espacios lo más amplios posibles mediante un hiperplano de separación definido como el vector entre los dos puntos, de las dos clases, más cercanos al que se llama vector soporte. Cuando las nuevas muestras se ponen en correspondencia con dicho modelo, en función de los espacios a los que pertenezcan, pueden ser clasificadas a una o la otra clase.

Más formalmente, una SVM construye un hiperplano o conjunto de hiperplanos en un espacio de dimensionalidad muy alta que puede ser utilizado en problemas de clasificación o regresión. Para ello, dado un conjunto de puntos en el que cada uno de ellos pertenece a una de dos posibles categorías, un algoritmo basado en SVM construye un modelo capaz de predecir si un punto nuevo (cuya categoría se desconoce) pertenece a una categoría o a la otra.

La SVM busca un hiperplano que separe de forma óptima a los puntos de una clase de la de otra, que eventualmente han podido ser previamente proyectados a un espacio de dimensionalidad superior. En ese concepto de separación óptima es donde reside la característica fundamental de las SVM: este tipo de algoritmos buscan el hiperplano que tenga la máxima distancia (margen) con los puntos que estén más cerca de él mismo. Por eso también a veces se les conoce a las SVM como clasificadores de margen máximo. De esta forma, los puntos del vector que son etiquetados con una categoría estarán a un lado del hiperplano y los casos que se encuentren en la otra categoría estarán al otro lado. La Figura 15 muestra un ejemplo de clasificación binaria donde los hiperplanos H_2 y H_3 son capaces de separar ambas clases mientras que el hiperplano H_1 no lo es. La principal diferencia entre los hiperplanos H_2 y H_3 es que el primero separa ambas clases con un margen muy pequeño, mientras que el segundo maximiza el margen entre las dos clases.

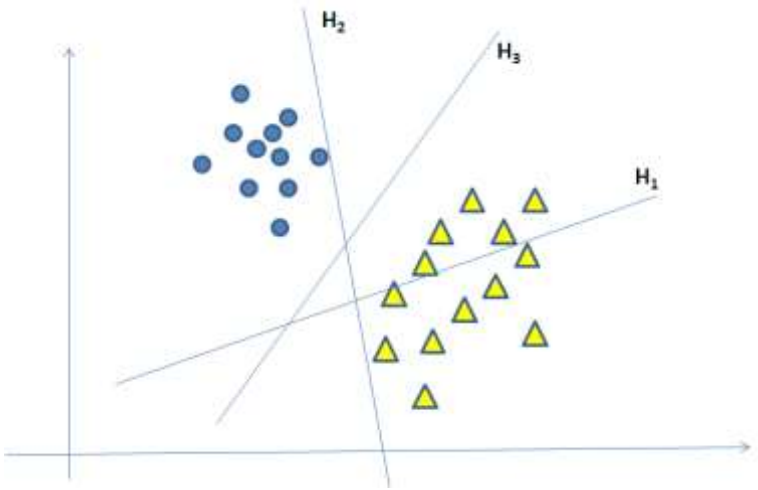


Figura 15: Ejemplo de clasificación binaria mediante las SVM

Aunque las SVM se utilizan frecuentemente para problemas de clasificación, existe una versión para regresión, cuya idea básica consiste en realizar un mapeo de los datos de entrenamiento $x \in X$, a un

espacio de mayor dimensión F a través de un mapeo no lineal ϕ: X → F, donde pueda realizarse una regresión lineal.

Hito 3: Validación

T3.1: Validación estadística de resultados

RESULTADOS OBTENIDOS

En este apartado se muestran los resultados obtenidos para cada uno de los modelos desarrollados correspondiente al Hito 2: "Desarrollo".

La Tabla 3 resume los errores de validación obtenidos para cada uno de los algoritmos, donde MAE (en inglés, *mean absolute error*) y MAPE (en inglés, *mean absolute porcentaje error*) representan el error medio absoluto y el error porcentual medio absoluto, respectivamente:

Tabla 3: Resultados de validación para todo el conjunto de datos disponible

Algoritmo	Correlación	MAE (€)	MAPE (%)
Regresión Lineal	0,9150	20.417	20,08
M5P	0,9815	9.672	8,49
M5Rules	0,9838	10.218	7,18
Red Neuronal	0,9794	10.354	8,51
Random Forest	0,9418	13.308	12,6

A continuación, en la Figura 15 se muestra gráficamente la distribución de errores de validación en base al número de ejemplos vistos en la fase de entrenamiento, con respecto al algoritmo M5P. Como se puede observar, los mayores errores se producen en aquellas zonas en las que prácticamente no hay ejemplos de entrenamiento (valores atípicos) y, por lo tanto, los modelos ajustan mal.

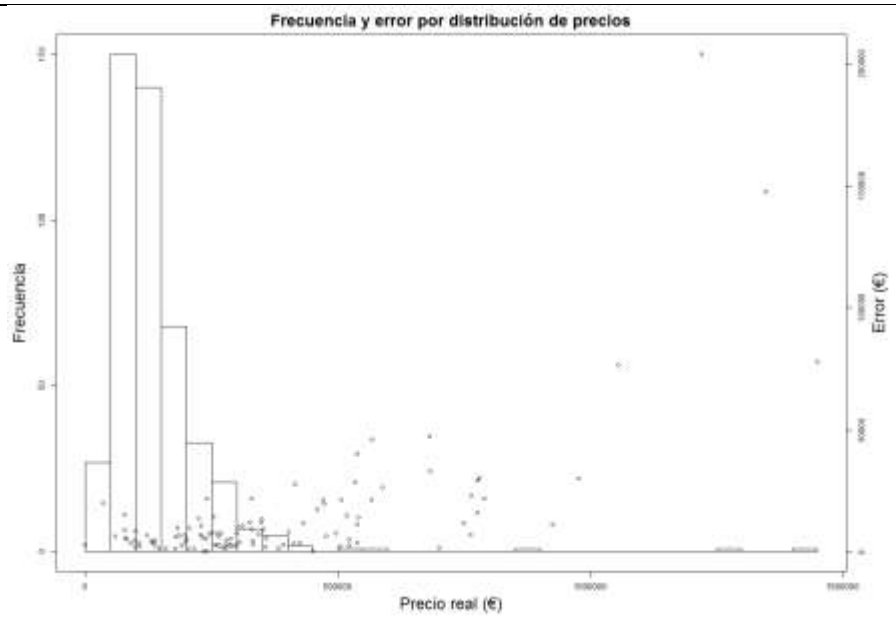


Figura 15: Distribución de errores cometidos en la fase de entrenamiento

A continuación, filtramos los datos disponibles con el objetivo de entrenar los algoritmos que mejores resultados han proporcionado en el estudio anterior. De este modo, nos quedamos con aquellos datos cuya frecuencia de aparición es "razonable", es decir, eliminamos los datos atípicos, de manera que haya muestras suficientes como para que el modelo pueda ajustarse correctamente cada situación. Los errores obtenidos en esta ocasión se muestran en Tabla 4.

Tabla 4: Resultados de validación para el conjunto de datos filtrado

Algoritmo	Correlación	MAE (€)	MAPE (%)
Regresión Lineal	0,9388	14.596	13,71
M5P	0,9875	7.471	6,35
M5Rules	0,9807	8.602	6,54
Red Neuronal	0,9829	9.373	7,66
RandomForest	0,9612	10.583	9,35

Tal y como se puede observar en la tabla anterior, los errores medios disminuyen considerablemente, obteniendo que los algoritmos que mejores resultados ofrecen son los correspondientes a los modelos M5P y M5Rules. En este caso se ha calculado el sesgo medio de todas las estimaciones individuales realizadas por los modelos, calculando la media de aquellas estimaciones realizadas a la baja, y la media de las estimaciones realizadas al alza.

Con el objetivo de optar por uno de estos dos algoritmos, se realiza un estudio que refleja la tendencia de error de cada uno de ellos, es decir, si son algoritmos que tienden a equivocarse al alza o a la baja. Por recomendación de Aesval, son preferibles los errores a la baja, es decir, es preferible que, en igualdad de

condiciones, el algoritmo escogido sea "prudente". Los resultados obtenidos son los mostrados en la Tabla 5.

Tabla 5: Error medio absoluto, al alza y a la baja de los modelos M5P y M5Rules

Algoritmo	MAE	ME+	ME-
M5P	7.417	7.340	7.626
M5Rules	8.602	8.791	8.370

Donde la columna MAE representa el error medio absoluto reflejado en la tabla anterior, la columna ME+ representa los errores medios cometidos al alza y la columna ME- representa los errores medios cometidos a la baja.

DISTRIBUCIONES DE VARIABLES Y ERRORES

Una vez seleccionado el modelo a utilizar, el árbol de regresión M5P, pasamos a analizar su comportamiento detalladamente a lo largo de este apartado.

La Tabla 6 muestra los percentiles correspondientes al error relativo, calculado como el cociente entre el error cometido en la estimación y el valor real de la misma, expresado en porcentaje.

Tabla 6: Percentil del error relativo para el modelo M5P

Algoritmo	Percentil error relativo (%)																			
	1	2	3	4	5	10	20	30	40	50	60	70	80	90	95	96	97	98	99	100
M5P	0	0	0	0	0	1	2	3	4	5	6	7	9	13	17	17	19	21	26	100

Tal y como se puede observar, un 80% de las muestras tienen un error relativo menor o igual al 9%. A partir de ahí los errores empiezan a subir de manera considerable, debido a la escasez de ejemplos disponibles en la fase de entrenamiento.

T3.2: Pruebas de concepto de otros casos de estudio

Este apartado presenta dos pruebas de concepto asociadas a dos casos de uso particulares. Por un lado, se ha estudiado la detección de fatiga en conductores mediante el análisis de patrones en imágenes, donde se han estudiado dos características: que el conductor tenga los ojos abiertos o cerrados, y que esté bostezando o no.

Por otro lado, se ha considerado el estudio de un caso de *pricing* dinámico. Se ha procedido mediante el uso de datos recogidos sobre costes de vuelos para una compañía aérea en particular, donde se proporciona tanto el precio base como el generado a partir de descuentos.

FATIGA EN CONDUCTORES

Este caso de estudio está centrado en el análisis de la fatiga en conductores a partir del análisis del reconocimiento de patrones en imágenes. Para ello, se han utilizado dos argumentos de entrada obtenidos mediante metodologías complementarias.

La primera de ellas permite detectar en una imagen de una cara, si dicha persona tiene los ojos abiertos o cerrados. Por otro lado, se comprobará la aparición de bostezos mediante un análisis de la imagen de la cara de dicha persona.

Detección de ojos cerrados

Este algoritmo trata de detectar si la persona tiene los ojos abiertos o cerrados en cortos espacios de tiempo de cara a generar un posible patrón que pueda predecir que el usuario empieza a sentirse cansado.

Para su desarrollo se ha hecho uso de las siguientes librerías:

- **OpenCV**[1] (versión 3.2.0): librería principal para el tratamiento/procesamiento de imágenes mediante el uso de técnicas de visión artificial.
- **Flandmark**[2]: librería que permite extraer de una región facial encontrada, 8 puntos característicos referentes a ojos, nariz y boca (Se utiliza en el algoritmo de detección de ojos abiertos o cerrados).
- **CLandmark**[3]: librería que permite extraer de una región facial encontrada, 68 puntos característicos referentes a cejas, ojos, nariz, boca y rebordes de la propia cara (Se utiliza en el algoritmo de detección de bostezos).
- **Boost**[4] (versión 1.60.0): librería para poder manejar directorios y archivos en disco.
- **LibSVM**[5] (versión 3.21): librería para crear clasificadores SVM mediante C++.

Como entorno de desarrollo se ha utilizado Visual Studio Community 2015. El algoritmo está implementado en Visual C++.

Para este caso, se ha provisto de 2 tipos de clasificaciones, aunque ambas siguen la misma estructura de uso. Por un lado, se propone detectar la cara de la persona para posteriormente normalizar la región de los ojos, la cual será la que se procesará para la construcción del modelo.

En cambio, se ha buscado una alternativa tratando de detectar cada ojo por separado gracias al clasificador Haarcascade proporcionado por la librería OpenCV. En este caso cada ojo tiene su propio clasificador, ya que se entrenan con las imágenes proporcionadas por la base de datos comentada anteriormente.

El algoritmo que se desarrollará dispone de 3 modos de funcionamiento, los cuales se detallan a continuacion de forma somera:

[1] http://opencv.org/
[2] http://cmp.felk.cvut.cz/~uricamic/flandmark/
[3] http://cmp.felk.cvut.cz/~uricamic/clandmark/index.php
[4] http://www.boost.org/
[5] https://www.csie.ntu.edu.tw/~cjlin/libsvm/

1. **Pre-procesado inicial**: en este modo se aplica de forma "pura" las diferentes técnicas usadas de visión por computador a través de la librería OpenCV. Se pide como parámetro de entrada "directorio de imágenes" a analizar, el algoritmo devuelve según la división (entrenamiento/prueba) realizada un fichero CSV donde se guardarán los datos procesados de las imágenes dadas.
2. **Creación de modelo**: una vez se tiene el fichero CSV, este se pide como parámetro de entrada en este modo. Se adapta dicho fichero al formato pedido por la librería para construir un fichero "**.**model" que será el clasificador entrenado contra el que harán pruebas siguientes.
3. **Test de modelo**: tras obtener el modelo, solo queda hacer pruebas para comprobar la precisión del algoritmo. El algoritmo analiza un directorio concreto con sus diferentes imágenes categorizadas previamente. Devuelve un resultado en caso positivo o negativo según las pruebas realizadas contra el modelo. Esto se aplica sobre el porcentaje de *testing* empleado cuando se hace la división inicial entre las imágenes destinadas al entrenamiento del clasificador y las restantes que se realizan como pruebas para saber si la precisión del clasificador entrenador es la correcta.

Pre-procesado inicial

Para la generación de un modelo, previamente se deben hacer una serie de transformaciones en la imagen. Para ello se siguen una serie de pasos que se detallan a continuación:

1. Se toma una imagen.
2. Se redimensiona la imagen.
3. Se convierte a escala de grises.
4. Se detectan los puntos de interés de la región facial (ojos, nariz y boca).
5. En base a los puntos anteriormente detectados, se alinea la imagen correctamente.
6. Se vuelven a detectar los puntos de interés de imagen alineada.
7. Se detecta y normaliza la región colindante de los ojos.
8. Se aplica el algoritmo "*Tan&Triggs*"[6], el cual se aplica para el reconocimiento facial en condiciones difíciles de iluminación mediante el uso de funciones de textura locales mejoradas.
9. Se divide la imagen en regiones (3 filas x 3 columnas).
10. Se aplica algoritmo "*LBP*"[7]**,** el cual permite clasificar la textura de cada una de las regiones en la que se divide la imagen. Para este caso particular se utiliza una versión de este algoritmo, denominada "uniforme", la cual disminuye el número de características extraídas mejorando el rendimiento posterior del clasificador.
11. Se normaliza si se pide el vector de la región.
12. Se concatena los vectores extraídos por cada región para formar el vector final de características de la imagen.

[6] TAN, Xiaoyang; TRIGGS, Bill. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE transactions on image processing*, 2010, vol. 19, no 6, p. 1635-1650.
[7] https://en.wikipedia.org/wiki/Local_binary_patterns

Una vez que se obtiene el vector de características final, se debe etiquetar dicho vector con el valor que atribuye que esa imagen pertenece a una determinada expresión facial:

- *Cara_ojos_abiertos (0).*
- *Cara_ojos_cerrados (1).*

Tras asignar dicho valor, el vector se graba dentro de un fichero CSV que se crea, formando una fila con tantas columnas como se haya asignado en el algoritmo LBP pedido con anterioridad. Al final de la última fila, se añade la etiqueta con la expresión facial asignada. Este proceso se repite tantas veces como imágenes tenga el directorio sobre el que se pide analizar como parámetro de entrada.

Creación del modelo de clasificación

Para crear un modelo, solo hay que indicar como parámetro de entrada el fichero CSV pre-procesado con los datos extraídos. En este proceso, se usa la librería *LibSVM* anteriormente mencionada. Esta librería permite entrenar un clasificador SVM (Maquinas de soporte vectorial). Antes de empezar a entrenar el modelo exige realizar una serie de pasos:

- Transformar el archivo CSV a otro fichero con un formato determinado como indica la librería.
- Escalar los datos: para un buen entrenamiento resulta fundamental un buen balanceo de los datos. Por ello se indica a la librería valores mínimo y máximo para escalar (0 y 1 como caso típico). Automáticamente crea otro archivo donde guardará el *dataset* escalado.
- A través de un fichero "*gridsearch*", el cual la propia librería (*LibSVM*) provee en su instalación, se encargará de calcular los mejores valores de entrenamiento (valor "*C*" y "*gamma*") para encontrar el modelo con la precisión más alta.
- Una vez que encuentra dichos valores, estos se aplican sobre la librería que ya está lista para entrenar el modelo y guardar el mismo en un fichero para después realizar pruebas contra el mismo.

Detección de bostezos

Este segundo algoritmo trata de detectar si la persona en la imagen está bostezando de forma continuada durante ciertos intervalos de tiempo. Así, se puede determinar si el operario está mostrando signos de fatiga que precisen un descanso en la tarea que esté realizando.

Para la detección de bostezos, se ha optado por una solución basada únicamente en el uso de técnicas de visión por computador. Por ello, se ha utilizado la librería "*CLandmark*" para poder extraer puntos faciales característicos de la cara.

Esta librería, tras detectar en primer lugar la región facial, detecta 68 puntos de la cara. El principal motivo del uso de esta librería es que la misma permite detectar puntos faciales referentes a los labios de la persona. Esto supone una ventaja en este caso particular, ya que para determinar si una persona está bostezando, podremos realizarlo sabiendo la distancia que hay entre los labios inferior y superior. A través de sus coordenadas, se puede predecir si la persona está bostezando o no.

Para este apartado, dado que el objetivo es saber si una persona está bostezando o no, se necesita encontrar los puntos relativos a los labios de la persona. Una vez se obtienen las coordenadas de las mismas, se busca la distancia existente entre ambas, comprobando que supera cierto umbral para saber si existe bostezo.

Clasificador de estado de fatiga

Una vez desarrolladas las dos herramientas previas, se procede a clasificar el estado de fatiga del conductor. El objetivo de los módulos de análisis de datos es doble:

- Se desea **extraer el patrón de comportamiento normal** para cada uno de los conductores a partir de los indicadores obtenidos de fatiga y estrés. Esto se hará a partir de un conjunto de datos previamente registrados por cada conductor estudiado durante un período de tiempo en el que se ha evaluado el estado del conductor como normal. Como resultado del proceso de análisis de datos, en este caso se obtendrá un modelo que permite definir lo que se considera normal.
- Durante la fase de operación o funcionamiento normal del sistema, se aplica el modelo de datos extraído para cada conductor, y se aplica a todos los nuevos datos generados durante un periodo de tiempo, con el objetivo de **detectar lo que se salga de lo normal**. El resultado de este proceso es la clasificación de los parámetros monitorizados en dos grupos diferentes: *normal* frente a *anomalía*.

En ambos casos, el comportamiento de una persona quedará asociado a los dos indicadores de fatiga utilizados por los módulos de visión artificial: **ojos abiertos/cerrados**, **bostezos detectados**.

Para el desarrollo de estos módulos, es necesario desarrollar un descriptor que determine el nivel de fatiga/estrés a partir de los dos parámetros monitorizados. Este descriptor servirá como fórmula para extraer los patrones normales, aplicándose a un conjunto de datos conocido, y detectar desviaciones sobre el comportamiento normal, dado un conjunto de datos nuevo generado durante el uso del vehículo.

El objetivo fundamental de los módulos planteados consiste en que, cada cierto intervalo de tiempo, se envíen los resultados generados por ambos algoritmos a un módulo de análisis de datos donde sean interpretados de manera que sea posible obtener, en un intervalo de tiempo, el porcentaje de veces que una persona en una imagen tiene los ojos abiertos o cerrados y el número de bostezos producidos en ese tiempo. Dados esos datos es, por tanto, posible inferir también si la persona se ha quedado dormida en algún momento. Este descriptor resulta un **indicador de nivel de fatiga** obtenido a partir de la combinación de los dos indicadores monitorizados.

Cada imagen relativa a una persona determinada viene caracterizada por un identificador numérico, esto es, cada persona presente en una imagen tiene una clave única, de manera que se pueda analizar el estado de fatiga para un individuo determinado en un intervalo de tiempo.

El sistema recibe un conjunto de imágenes con un identificador dado, guardándose el instante de tiempo al cual llegan las imágenes que pasan a ser clasificadas por el clasificador de ojos y por el clasificador de bostezos.

El clasificador de estado de fatiga está formado por dos módulos:

- **Pre-Procesador de datos**. Encargado, por un lado, de almacenar en una base de datos la información resultado de los clasificadores de ojos y de bostezos respectivamente. Por otro lado, de procesar esta información para, posteriormente, almacenarla en otro repositorio.
- **Clasificador de estado de fatiga**. Este módulo es el encargado de entrenar, utilizando algoritmos de aprendizaje automático supervisado, el conjunto de datos tratados por el módulo de pre-procesado, creando un clasificador o modelo que es capaz de predecir, ante un ejemplo de una imagen, si la persona representada en ella presenta fatiga o no.

A continuación, se detallan los dos subsistemas desarrollados.

Subsistema pre-procesador de datos

Se dispone de una base de datos donde se van almacenando los vectores que van obteniéndose de los clasificadores. El subsistema *pre-procesador de datos* se encarga de procesar estos vectores de manera que cuantifica, en un intervalo de tiempo fijado previamente por el usuario, el porcentaje de veces que una persona identificada por una imagen presenta ojos abiertos, el porcentaje de veces que una persona identificada por una imagen presenta ojos cerrados y el número de bostezos de una persona identificada por una imagen. Para realizar este cálculo, el subsistema ha de recorrer cada vector almacenado en la base de datos donde se guardan los resultados de los clasificadores.

Para todos los vectores con identificador de imagen igual y que estén dentro del intervalo de tiempo fijado, se calcula:

- El número de ceros obtenidos por el clasificador de ojos, esto es número de ojos cerrados.
- El número de unos obtenidos por el clasificador de ojos, esto es número de ojos abiertos.
- La suma del número de ojos cerrados y del número de ojos abiertos.
- El número de unos obtenidos por el clasificador de bostezos.

Una vez obtenidos los resultados, se almacenan en un vector con una estructura similar a la original, incluyendo dichos valores previamente descritos.

Subsistema clasificador de estado de fatiga

Este subsistema es el encargado de entrenar, mediante algoritmos de aprendizaje automático supervisado, el conjunto de datos procesados, etiquetados ya con el “estado de fatiga” y almacenados en el sistema, dando como resultado un clasificador a partir del cual se puede predecir el estado de fatiga

(cero si no presenta fatiga y uno en caso contrario) que sufre una persona, dado un conjunto de ejemplos de imágenes en un tiempo dado, cuyo estado de fatiga es desconocido y se desea inferir.

En este tipo de aprendizaje, los algoritmos requieren el suministro de un conjunto de ejemplos, cada uno de los cuales ha de estar descrito por un conjunto de atributos y una etiqueta asociada que describe alguna propiedad importante o decisión relativa al ejemplo etiquetado. La tarea del algoritmo de aprendizaje automático supervisado consiste en construir un modelo que genere predicciones de forma precisa ante futuros ejemplos.

Así, los sistemas de clasificación supervisados son aquellos que se usan para predecir una categoría, grupo o clase determinado. Tal como se ha explicado, a partir de un conjunto de ejemplos clasificados (conjunto de entrenamiento), se asigna una clasificación a otro conjunto de ejemplos dado. Cuando solamente hay dos opciones o clases a clasificar o predecir se denomina clasificación binaria.

Para crear el modelo clasificador, se han implementado tres algoritmos: el algoritmo de los K Vecinos, Árbol de Decisión y el algoritmo de Máquinas de Soporte Vectorial o *Support Vector Machine* (SVM).

Medidas de rendimiento de los algoritmos

Para medir el rendimiento de los algoritmos de clasificación implementados se guarda un conjunto de los ejemplos iniciales para ser utilizados posteriormente como validación del clasificador. Es decir, el conjunto de ejemplos (de los que se conoce el resultado que se debe obtener) *D*, se particiona en dos subconjuntos: *train* y *test*, de forma que al algoritmo de entrenamiento solo se le proporcionan los ejemplos del subconjunto *train*, y una vez realizado el entrenamiento completo, se mide cómo de buenos son los resultados sobre los datos de *test*, que el clasificador entrenado "no conoce". El error cometido es cuantificado teniendo en cuenta el resultado que devuelve el modelo sobre ellos y el dato conocido.

Así, se obtiene la denominada matriz de confusión, que indica para cada una de las posibles clases, cuántos ejemplos del subconjunto de *tes*t se clasifican en cada una de las posibles opciones: cada columna de la matriz representa el número de predicciones de cada clase, mientras que cada fila representa las instancias en la clase real.

Es importante mantener el equilibrio en el proceso de aprendizaje para que el modelo no aprenda tanto de los datos proporcionados como para distorsionar el posible patrón general que éstos representan.

Resultados de las pruebas con distintos algoritmos

Hay programas que ofrecen entre sus servicios el entrenamiento de clasificadores de forma rápida, con unos resultados preliminares que permitan obtener datos relativos a la precisión que puedan confirmar o avisar del trabajo en el pre-procesado de las imágenes, si ha sido adecuado o no antes de comenzar a entrenar el clasificador.

Uno de estos programas es Matlab, el cual dispone de una serie de aplicaciones internas entre las que destaca "*Classification Learner*", mediante la cual se puede elegir el clasificador que se considere y

empezar a entrenar el mismo. El único requisito es tener a disposición un banco de datos entrantes para poder crear el clasificador, justamente el mismo fichero CSV resultante de esta fase puede usarse para la clasificación.

Para las pruebas de los algoritmos se han utilizado imágenes obtenidas de la grabación de dos personas, en estado de no fatiga y fatiga, durante 2 minutos. Para cada una de las imágenes se ha generado manualmente el descriptor de fatiga, indicando si los ojos están abiertos o no en cada imagen, y si se bosteza o no en cada imagen.

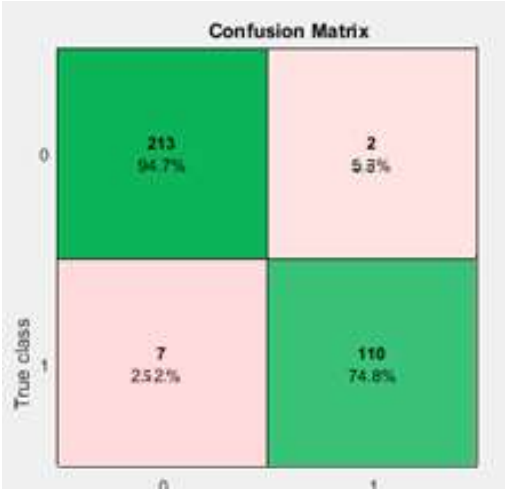


Figura 17. Ejemplo de uso "Classification Learner" en Matlab (matriz de confusión)

En la Figura 17 se puede observar el resultado de clasificar mediante el algoritmo de k-NN en dos grupos: fatiga (1) / no fatiga (0), mostrándose en la matriz por una parte los datos reales *(true class)* y por otro los resultados del clasificador *(predicted class)*.

Como puede verse, el mejor resultado se produce al clasificar una imagen como “no fatiga” cuando éste es el resultado correcto (94.7% de acierto). En cambio, este clasificador presenta más fallos cuando ha de reconocer un caso de fatiga (clase verdadera: fatiga, clase predicha: no fatiga en un 25.2%).

Tabla 7. Resumen resultados evaluación clasificadores

Clasificador	KNN		Árbol decisión		SVM	
True class/Predicted	**0**	**1**	**0**	**1**	**0**	**1**
0	94.7%	5.3%	93.6%	6.4%	95.1%	4.9%
1	25.2%	74.8%	18.4%	81.6%	16.3%	83.7%

En la Tabla 7 se proporcionan los resultados para los clasificadores. Se ha optado por el clasificador basado en árboles de decisión, ya que. ante unos resultados similares, es más rápido en cuanto a su aplicación, además de proporcionar una mayor interpretabilidad.

PRICING DINÁMICO

Este segundo caso de estudio tiene como temática el análisis del *pricing* dinámico. Esta estrategia tiene como base la variación de los precios de los productos en función de diversas características que varían con el tiempo. A modo de ejemplo, existen diferentes campos donde se aplica esta metodología, tales como en páginas de venta online tan conocidas como Amazon, así como precios de medios de transporte, como ocurre con diferentes compañías aéreas.

El objetivo será, a partir de datos reales de un caso de *pricing* dinámico particular, predecir qué valores de precio serán los finales a partir de los iniciales y otros parámetros de influencia. Este caso de uso se estructurará en tres apartados. El primero, describirá los datos utilizados para el ejemplo. El segundo, detallará cada uno de los modelos implementados para su tratamiento. Finalmente, se mostrarán los resultados obtenidos.

Datos utilizados

El caso de *pricing* dinámico utilizado en este proyecto ha sido el asociado a la variación de precios de vuelos. En concreto, se ha analizado la compañía Ryanair. Se han recogido datos en noviembre de 2017 de todos los vuelos disponibles en su página oficial. Dicha disponibilidad es de casi un año, obteniendo información de vuelos que no se realizarán hasta octubre de 2018.

Dicho proceso de recogida de datos se ha realizado mediante un proceso de *scraping* automático que ha permitido recoger dicha información paulatinamente. A través de dicho proceso, se ha recogido un total de 725775 vuelos.

El objetivo del problema base será **estimar el precio final del vuelo** a partir de las distintas variables disponibles, incluyendo el precio base de dicho trayecto. Se obviarán por razones lógicas las variables que informan de la existencia de descuentos como del porcentaje descontado.

Tras un análisis de la información proporcionada por cada una de las variables presentadas, se han seleccionado para el desarrollo del modelo las siguientes:

* **Date**. Proporciona la fecha de salida del vuelo. Es de importancia en la estimación del precio final con respecto a la importancia que tiene la antelación de la reserva del vuelo en cuanto a precios reducidos se refiere. Contando con que estos datos han sido recogidos a 5 de noviembre de 2017, se ha procedido a generar una nueva variable que determine los días de antelación con los que se haría dicha reserva, extrayendo los días entre la fecha de salida del vuelo y la de recogida
* **Segment Duration**. La duración del vuelo es de suma importancia ya que permite caracterizar de forma clara la tipología del trayecto. Dado que dicha información es proporcionada en formato HH:MM, ha sido preciso transformarla a minutos para su posterior análisis
* **Segment Distance (km)**. Del mismo modo que la duración del vuelo, caracteriza también la tipología del trayecto, pudiendo llegar a existir una alta correlación entre ambas variables.
* **Fare Published**. Precio base que proporciona la aerolínea, a partir del cual se obtendrá el precio final con los descuentos estimados a través del resto de parámetros en estudio.
* **Fare Amount**. Precio final a estimar. Será necesario disponer de dicha variable para ambas fases del análisis. En la fase de entrenamiento, el modelo ha de conocer valores reales de modo que

pueda aprender a estimarlo con casos no tratados. En la fase de test, será imprescindible para la evaluación de los resultados de cada uno de los casos analizados.

Además de las variables previamente listadas, se ha necesitado aplicar un atributo extra, siendo éste la moneda en la que se ha proporcionado el precio final. Debido a que existen varias, ha sido necesario realizar una conversión de éstas de modo que queden todos los casos representados por una misma moneda, en este caso, el Euro.

En la Tabla 8 se muestran las diferentes monedas identificadas para dicho parámetro, donde se muestran tanto la codificación proporcionada, su correspondencia real, y la tasa al cambio de dicha moneda al Euro.

Tabla 8. Tabla de monedas identificadas y tasa de conversión a Euro

Código moneda	Nombre moneda	Tasa de cambio (Euro)
CHF	Franco suizo	0.85134741
CZK	Corona checa	0.0393284828
DKK	Corona danesa	0.13427914
GBP	Libra esterlina británica	1.13770641
HUF	Florín húngaro	0.00321660644
MAD	Dirham de Marruecos	0.0883797054
NOK	Corona noruega	0.105038
PLN	Zloty polaco	0.236829739
SEK	Corona sueca	0.0992691741

Así, dichas conversiones han sido aplicadas a los parámetros *Fare Published* y *Fare Amount* con respecto a sus correspondientes monedas.

Además, como parte del pre-procesado, también se han eliminado aquellos casos donde no se disponga de valores para alguno de los parámetros en análisis. De este modo, los datos finales disponen de un total de 706459 vuelos.

Modelado de algoritmos

Una vez procesados los datos seleccionados, disponemos de cuatro variables predictoras (longitud en kilómetros del trayecto, antelación de la consulta en días, duración del trayecto en minutos, precio base sin descuentos), a partir de las cuales el objetivo será predecir el valor final del vuelo con los descuentos pertinentes que pueda llegar a aplicar la aerolínea.

Para ello, se procederá mediante el uso de algoritmos de *machine learning* de regresión, de modo que sea posible generar un valor de salida dentro de un dominio continuo. Dichos algoritmos serán regresión multivariable, diversos tipos de árboles de regresión (M5P, M5Rules, Random Forest) y redes neuronales.

Scripts

La generación del modelo M5Rules está hecha mediante la función "M5Rules", y a partir de dicho objeto, se proporcionan los resultados para los conjuntos de entrenamiento y test. De modo similar, se ha procedido con el resto de modelos.

Resultados

En la Tabla 9 se muestran todos los resultados obtenidos con respecto a cada uno de los algoritmos para los datos en estudio sobre *pricing* dinámico.

Tabla 9. Resultados de validación

Algoritmo	MAE (€)	MAPE (%)
Regresión Lineal	1.21	3.79
M5P	0.86	2.2
M5Rules	0.51	1.53
Red Neuronal	0.54	1.6
Random Forest*	1.56	4.65

* Nótese que, debido a la gran carga computacional que requiere el modelo *Random Forest*, no ha sido posible generarlo con el conjunto completo de datos de más de 700000 casos de estudio. Es por ello que los resultados de dicho algoritmo han sido generados con un subconjunto de estos de 100000 individuos.

Como se puede observar en los resultados previos, tanto el árbol de regresión M5Rules, como la red neuronal proporcionan resultados muy similares, ambos con un margen de error muy bajo, tanto en valores absolutos como porcentuales. La regresión no proporciona resultados tan ajustados al tratarse de un modelo con mayor complejidad y, por tanto, no dispone de suficiente potencia para acotarlo del mismo modo que el resto de modelos. En el caso del *Random Forest*, nos encontramos con problemas asociados a la falta de potencia computacional para el procesamiento de todos los datos, si bien sería un problema fácilmente solventable con una mejora en la infraestructura.

2. RESULTADOS CONSEGUIDOS

Durante el periodo de tiempo que cubre este informe se han conseguido todos los resultados planificados:

- Determinación de los requisitos mínimos que ha de cumplir la herramienta para la tasación automática de inmuebles.
- Estudio de los distintos métodos de valoración existentes.
- Selección de *Twitter* como fuente principal para el análisis de opinión de las zonas a las que pertenezcan los correspondientes inmuebles en estudio.
- Integración de las fuentes de información, pre-procesando los datos obtenidos mediante ERP Gestión.

- Integración con fuentes de información, pre-procesando los datos obtenidos mediante ERP Gestión.
- Selección de algoritmos con diferentes propiedades para la herramienta de valoración automática. Se han comparado de modo experimental para seleccionar el modelo más adecuado.
- Se han buscado casos de estudio similares en los que se pueda adaptar el modelo aquí desarrollado. (Pricing dinámico, fatiga en conductores)