Trabajo de Final de Máster

Máster Universitario de Ingeniería industrial

Ciencia de Datos aplicada al análisis de alojamientos turísticos en la ciudad de Barcelona

MEMORIA

Autor: Joel Medina Jiménez
Director: Luis José Talavera Méndez

Convocatoria: Septiembre 2022

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Resumen

El presente documento trata del análisis de los alojamientos turísticos en Barcelona utilizando ciencia de datos. Se predice la ocupación de los diferentes alojamientos que se encuentran en la página web *Airbnb* a partir de algoritmos de predicción.

Se utilizará la metodología CRISP-DM, está compuesta de diversas etapas las cuales se aplicaran en este trabajo. Siguiendo esta metodología se desea extraer la información más relevante de los datos, como patrones y tendencias.

Durante todo el proceso de elaboración del proyecto se programará en lenguaje Pyhton en *Jupyter*. Python es uno de los lenguajes más empleados para programar. Proporciona una serie de librerías, las cuales permiten tratar los datos y crear modelos.

Índice

1. Introducción

Estamos en una época donde las personas suelen viajar por todo el mundo, ya sea para descubrir lugares nuevos o por circunstancias de trabajo. Normalmente se aprovechan las vacaciones para volar a países con culturas diferentes al sitio de residencia y descubrir nuevas experiencias. Habitualmente se realiza acompañado de la familia o, en alguna ocasión, de forma solitaria. Cuando viajan, las personas buscan romper con la rutina de su día a día y conseguir desconectar.

Debido a la COVID, se impusieron restricciones donde se llegó al extremo de solo poder salir a la calle para comprar comida o trabajo. Cuando las restricciones fueron menos estrictas, compañías como *Vueling* empezaron a ofrecer ofertas de vuelos para aumentar la demanda y recuperar las pérdidas de meses anteriores. Negocios que vivían de la hostelería pudieron ir recobrando la normalidad a medida que fue pasando el tiempo.

A la hora de viajar, uno se puede hospedar en diversos sitios, entre ellos, en un hotel o en una residencia que una persona alquila durante un tiempo determinado. Al aumentar los precios tanto de la comida como del transporte, cada vez es más común que se comparen los lugares donde hospedarse. *Airbnb* es una compañía y plataforma digital que actúa como intermediaria entre particulares que quieren alquilar su vivienda con turistas que buscan un lugar donde estar durante unos días; a su vez, los propietarios de las residencias publicitan mediante fotos e información el lugar que ofrecen.

Hoy en día se dispone de muchos datos de viviendas donde se muestra sus características. Los datos aportan información valiosa y fácil de procesar de manera adecuada. La ciencia de datos es un campo con el que se puede extraer conocimiento y mejor entendimiento de todos estos datos. Este conocimiento se obtiene por medio de un análisis a partir de métodos o métricas, con las que se identifican patrones o parámetros en los datos. Del mismo modo que se puede usar para estos datos, se aplica en diversos sectores como, por ejemplo, el transporte, la medicina o el deporte. En el sector de la medicina se está aplicando para identificar de forma rápida posibles infecciones peligrosas en heridas: al realizar una foto se puede observar y concluir si es dañina para la salud.

La ciencia de datos suele ser una herramienta importante tanto para empresas privadas como para organizaciones orientadas a tomar decisiones respecto al futuro. Proporciona una visión más exacta de cómo se encuentra el sector, pudiendo predecir en muchos casos escenarios o peligros a tener en cuenta. En definitiva, se entiende como una continuación de la minería de datos, aprendizaje automático y la analítica predictiva.

1.1. Metodología CRISP-DM

Para realizar un análisis satisfactorio y cumplir con los objetivos, se debe realizar de forma metódica. El método más usado es el CRISP-DM (utilizado en este trabajo), que proporciona una descripción del ciclo de vida de un proyecto de análisis de datos. Consta de las siguientes seis fases:

Comprensión del negocio: esta fase es la base del proyecto, donde se definen los objetivos, se valora la situación actual y se obtiene el plan del proyecto, en definitiva, de qué forma se ejecutará.

Comprensión de los datos: proceso que se realiza establecidos los objetivos y el plan del proyecto. Consiste en obtener un buen conocimiento de los datos, tanto la estructura como la distribución, y conocer la calidad de los mismos. Se ejecuta el proceso de la captura de datos y se realiza la exploración.

Preparación de los datos: finalizada la exploración, se da paso a la preparación. El objetivo en esta etapa es procesar los datos para obtener los definitivos con los que se aplicarán los modelos de predicción. Se realiza una limpieza de los datos que se consideran irrelevantes y se transforman a la forma que se desea.

Modelaje: obtenidos los datos finales, se procede a construir modelos de predicción para lograr el objetivo del proyecto. Se seleccionan las técnicas de modelado más aptas para los datos que se disponen y se crean los modelos. Además, se establece una verificación para saber la calidad de los modelos. Después, se va ajustando el modelo valorando la fiabilidad que se proporcionan y el impacto que tienen en el objetivo que se ha fijado con anterioridad.

Evaluación: esta etapa se focaliza en valorar el grado de acercamiento de los diversos modelos con los objetivos. Se evalúan los modelos, se revisa todo el proceso realizado hasta el momento y se toman decisiones como volver a fases anteriores para hacer modificaciones

o continuar con la siguiente fase.

Despliegue: esta es la última parte del método, en la que se despliegan los resultados, se distribuyen a los usuarios interesados y, normalmente, se monitorizan. Es importante realizar un seguimiento y mantenimiento de esta fase por posibles cambios que pueda haber en los datos.

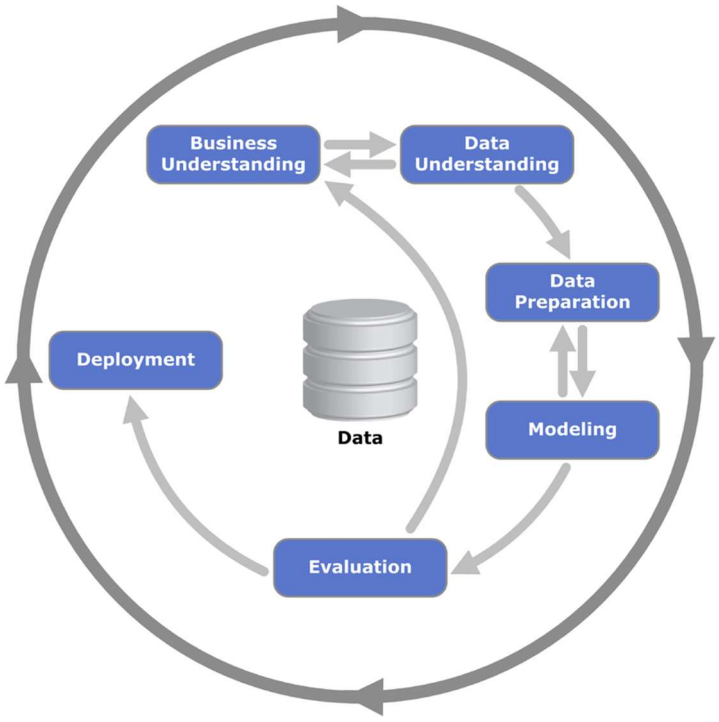


Figura 1.1.1. Representación de la metodología CRISP-DM

1.2. Exploratory Data Analysis

En la fase de comprensión de los datos se realiza el EDA, *Exploratory Data Analysis*, consiste en analizar el conjunto de datos para ver con claridad las características que tienen. Usa técnicas estadísticas, logra maximizar la comprensión y resume la información que contienen los datos. Consta de cinco etapas, expuestas a continuación. No hace falta que se realice en este orden.

1. Análisis descriptivo. en esta etapa se aplican funciones estadísticas descriptivas, mediante las cuales se visualiza la estructura de los datos. Se pueden observar los patrones de los datos y examinarlos, saber los tipos y las variables que presentan.

2. Ajuste de tipos de variables: se procede a codificar cada una de las variables del conjunto. Se comprueba que el tipo de variable sea el adecuado para el proyecto y, en caso de una mala codificación, puede repercutir negativamente en los resultados.

3. Detección y tratamiento de datos ausentes: es una etapa importante en el análisis exploratorio de datos, pues se buscan los valores ausentes que haya en el conjunto de datos y que puedan provocar una mala funcionalidad de los modelos. Se pueden sustituir estos datos por la media, la mediana, con el valor anterior o posterior de la columna, o bien eliminando la fila o columna.

4. Identificación de datos atípicos: es relevante identificar datos con valores muy diferentes a los otros. Consiste en disminuir la influencia que puedan tener estos valores en el objetivo del proyecto. En algunos casos se eliminan, ya que se consideran anormales y puede deberse a fallos en la adquisición de datos.

5. Análisis de correlación: se analiza la relación que pueda haber entre las variables y la relación entre las variables y el resultado. Se calcula el coeficiente de correlación y se observa la correlación si los valores son próximos a -1 o 1. En estos casos, si la relación es entre dos variables, se decide eliminar una de ellas.

1.3. Objetivo del proyecto

El principal objetivo de este trabajo es realizar un estudio mediante el uso de técnicas estadísticas y de minería de datos para analizar factores relevantes en los listados de alojamientos turísticos. Como objetivo secundario, se quiere predecir el nivel de ocupación en los alojamientos. Se desea utilizar dos tipos de modelos de predicción para predecir la ocupación: Regresión Logística y *Random Forest*. Al utilizar dos modelos, se comparan entre sí para poder observar la tasa de acierto de uno respecto al otro y sus características.

Los datos disponen de varias columnas, las cuales proporcionan diversos parámetros de los alojamientos como, por ejemplo, la zona donde está ubicado y el número de habitaciones o camas de las que se dispone. Las columnas son variables que los modelos utilizan. Cada fila de la base de datos se refiere a un alojamiento que ha publicado un propietario de la zona de Barcelona. Cabe destacar que todas las columnas no se pueden usar, ya sea porque la información no es adecuada o porque no son útiles para el modelo de predicción. Se pretende crear una nueva columna llamada *Ratio,* la columna a predecir. Esta columna estará compuesta por la categoría de Alta y la categoría de Baja. Indica si el alojamiento tendrá una ocupación Alta o Baja. Este trabajo enfatizará la categoría de Alto, pues siempre es beneficioso saber cuáles son los alojamientos que tendrán un nivel de ocupación alto al aportar mayor beneficio para el propietario; además, proporciona un valor añadido respecto a los demás competidores de la zona. No obstante, podría haber otros tipos de perspectivas respecto a la categoría en la que centrarse.

Para realizar el análisis se aplicará una metodología, mencionada anteriormente, la CRISP-DM y, además, la EDA. Es importante que el trabajo pueda evolucionar teniendo todos los factores a destacar en cuenta. El último objetivo es obtener un conocimiento más complejo en programación con lenguaje Python. Durante todo el proceso se manipularán los datos con la librería Panda y se utilizará la librería *Sklearn* para generar los modelos de predicción. La finalidad es familiarizarse con el entorno de trabajo *Jupyter*, así como con la metodología utilizada durante la realización del trabajo.

1.4. Abasto del proyecto

Como se ha expuesto, este proyecto se realizará mediante la metodología CRISP-DM. A continuación, se procede a mostrar las fases de la metodología adaptadas.

Comprensión del negocio. Se ha realizado un estudio de la situación del negocio para poder definir el objetivo y los problemas con las viviendas. Al estar trabajando en una empresa de proyectos, se entiende las necesidades y preocupaciones de los dueños. Por ello, es adecuado considerarlos con los conocimientos adquiridos para efectuar un proyecto de esta envergadura.

Comprensión de los datos. La estructura de los datos se explicará posteriormente. Las variables están bien definidas y ordenadas. Se aplicará EDA para profundizar más y así eliminar posibles fallos que puedan haber.

Preparación de los datos. En esta fase se manipularán los datos, creando nuevas columnas. Una de las nuevas columnas a generar será la ocupación que tendrán las viviendas durante el año. Se crearán dos categorías para predecir con los modelos. Todo se hará mediante lenguaje *Python*.

Modelaje y evaluación. Obtenidos los datos definitivos, se crearán dos tipos de modelos de predicción, los cuales se explicarán posteriormente. Se dividirá el conjunto de datos, una parte del conjunto se usará para entrenar a los modelos y la restante para validar si los modelos realizan predicciones de forma correcta. Los resultados obtenidos de cada modelo se utilizarán para compararlos y obtener conclusiones.

Respecto al despliegue, es una fase que no se hará en este proyecto. Esto se debe a que no se dispone de suficiente tiempo y sería necesario estar durante un periodo de un año observando cómo van modificando los datos.

1.5. Herramientas utilizadas

1.5.1. Python

Python es uno de los lenguajes de programación más usado en todo el mundo. Se utiliza para desarrollar una gran multitud de aplicaciones. Java y .Net son lenguajes que es necesario compilados, en cambio Python se puede ejecutar directamente cuando es llamado por el interpretador.

Python es un lenguaje perfectamente inteligible y fácil de escribir de forma rápida, posee una gran igualdad con los lenguajes humanos. Es un lenguaje gratuito que permite programar sin ninguna restricción. Su popularidad ha ido aumentando al poder usarse tanto para inteligencia artificial como para big data, además de proporcionar un aprendizaje automático y ciencia de datos.

Dispone de una gran variedad de librerías que facilitan la programación. Por otro lado, permite programar sin necesidad de muchas líneas de programación y tener documentos que ocupen grandes tamaños, lo que simplifica la programación. Proporciona a los usuarios una página donde se explican las librerías y su aplicación en el lenguaje mediante ejemplos.

Python es la herramienta que se enseña en el grado GETI, en las asignaturas *Fundamentos de informática* y *Informática*. En el máster MUEI, se aplica en asignaturas para obtener resultados y no realizar tantos cálculos numéricos.

1.5.2. Pandas

Pandas es un paquete disponible en Python y que se utilizará en este proyecto. Proporciona herramientas para poder manipular grandes bases de datos, y es la que se usará más en la parte de preparación de datos.

Pandas permite leer y escribir en diferentes formatos como Excel, bases SQL o CSV. Posibilita seleccionar de manera simple las tablas de datos, dependiendo de la posición o valor. Se pueden fusionar diversas tablas y unir datos. Facilita la creación de gráficas para poder observar la distribución de los datos y así llegar a mejores conclusiones.

1.5.3. Sklearn

Sklearn es una librería de Python y la más usada para ciencia de datos. Ofrece la posibilidad de crear modelos de aprendizaje automático. Incluye algoritmos de clasificación, regresión y análisis de grupos. Ayuda en el preprocesado de datos, la reducción de dimensionalidad, es decir, la selección de los parámetros y la selección de modelos.

Permite separar los datos de la siguiente manera: una parte para entrenar y otra para comprobar si los modelos realizarán las predicciones correctas. Para cada modelo proporciona una gran variedad de datos donde se muestra la precisión, la tasa de aciertos o los fallos que se han cometido.

1.5.4. Jupyter Notebook

Jupyter es un entorno informático interactivo que permite a los usuarios probar el código que han creado y poder compartirlo con los otros usuarios de la red. Las siglas de Jupyter provienen de Julia, Python y R, los primeros lenguajes con los que se empezó a programar en *Jupyter*, hoy en día se puede programar en muchos más.

Permite registrar código, ejecutarlo y poder observar los resultados. Es un entorno que es apropiado para limpieza de datos, modelado estadístico y construcción de modelos de Comprensión de los datos.

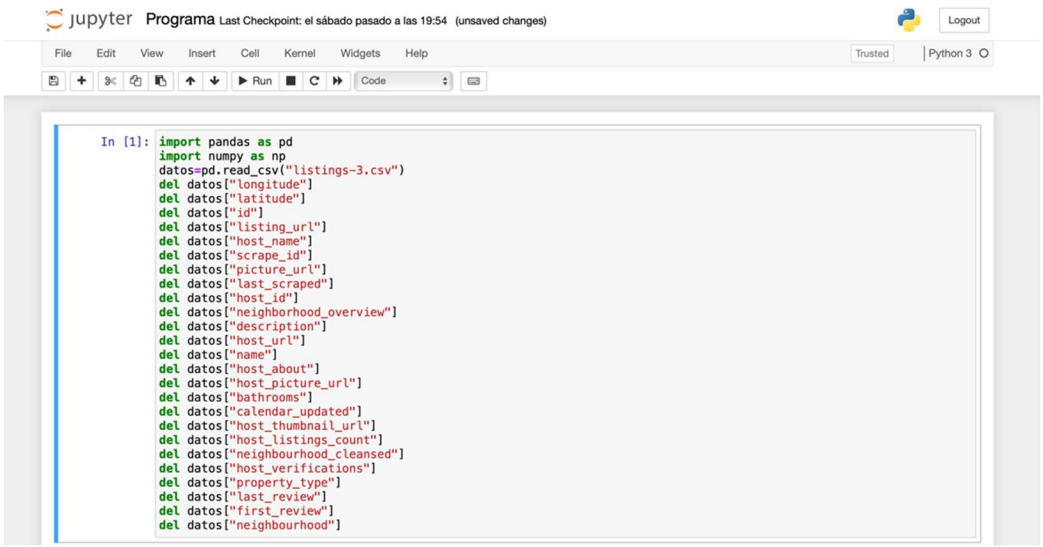


Figura 1.5.4.1. Notebook Jupyter

2. Comprensión de los datos

La comprensión de los datos es una etapa clave en el resultado del proyecto. Ayuda en la toma de decisiones en la selección, pues al transformar los datos en un formato correcto permite que se obtengan mejores resultados. En proyectos de esta magnitud suele ser una de las fases más largas, por lo que debe de dedicarse una gran cantidad de tiempo y entender todos los datos y variables.

Los datos fueron descargados de la página de *Inside Airbnb* . El objetivo de *Inside Airbnb* es mostrar los datos para así comprender, decidir y controlar el papel del alquiler de los alojamientos en viviendas residenciales que están destinadas para turistas. Los datos provienen de la información disponible de forma pública en *Airbnb*, se analizan y se agregan en *Inside Airbnb* para poder debatirlos. Los autores de esta página no tienen ninguna relación con *Airbnb*.

2.1. Análisis descriptivo

A continuación, se realizará una descripción de las columnas del documento “listings-3.csv”, documento en el que se basará el proyecto. Estos datos son de enero de 2021, dispone de 15705 filas y 74 columnas.

[1] ID: Indica el identificador del alojamiento.

[2] LISTING_URL: Enlace que dirige a la pagina para ver el anuncio

[3] SCRAPE_ID: Columna que posee el mismo valor para todas las filas

[4] LAST_SCRAPED: Indica el día que se añadió al listado.

[5] NAME: Nombre del anuncio del alojamiento.

[6] DESCRIPTION: Pequeña descripción del anuncio en *Airbnb*

[7] NEIGHBORHOOD_OVERVIEW: Descripción de la Zona del *Airbnb*

[8] PICTURE_URL: Enlace de las imágenes apartamento.

[9] HOST_ID**:** Identificador del dueño

[10] HOST_URL: Página del dueño

[11] HOST_NAME: Nombre del dueño/s

[12] HOST_SINCE: Cuando se unió a Airbnb el dueño

[13] HOST_LOCATION: Ubicación del dueño

[14] HOST_ABOUT: Información de dueño

[15] HOST_RESPONSE_TIME: Tiempo de respuesta de dueño

[16] HOST_RESPONSE_RATE: Ratio de tiempo de respuesta

[17] HOST_ACCEPTANCE_RATE: Ratio de aceptaciones que realiza el dueño a reservas

[18] HOST_IS_SUPERHOST: Indica si el dueños tiene buenas valoraciones y fiables { t/f}

[19] HOST_THUMBNAIL_URL: Enlace de foto del dueño tamaño pequeño

[20] HOST_PICTURE_URL: Enlace de foto del dueño tamaño mediano

[21] HOST_NEIGHBOURHOOD: Indica la zona del dueño

[22] HOST_LISTINGS-COUNT: Indica el número de alojamiento en *Airbnb* que tiene el dueño

[23] HOST_TOTAL_LISTING_COUNT: Indica el número de alojamiento en *Airbnb* que tiene el dueño

[24] HOST_VERIFICATIONS: Características verificadas del dueño, es una lista con parámetros.

[25] HOST_HAS_PROFILE_PIC: Indica si el dueño tiene foto de perfil { t/f}

[26] HOST_IDENTIFY_VERIFIED: Indica si el dueño está verificado { t/f}

[27] NEIGHBOURHOOD: Ciudad del alojamiento

[28] NEIGHBOURHOOD_CLEANSED**:** Zona exacta de la Ciudad del alojamiento

[29] NEIGHBOURHOOD_GROUP_CLEANSED**:** Zona conocida de la Ciudad del alojamiento, agrupa zonas de la ciudad.

[30] LATITUDE: Indica la latitud del alojamiento

[31] LONGITUDE**:** Indica la longitud del alojamiento

[32] PROPERTY_TYPE: Tipo de propiedad, la selecciona el dueño cuando crea el anuncio

[33] ROOM_TYPE: 4 tipos; [Entire home/apt|Private room|Shared room|Hotel]

[34] ACCOMMODATES**:** Capacidad máxima de personas en el alojamiento

[35] BATHROOMS**:** No hay valor

[36] BATHROOMS_TEXT: Número de baños y tipo de baños

[37] BEDROOMS: Cantidad de Habitaciones

[38] BEDS: Número de camas

[39] AMENITIES: Servicios que proporciona el alojamiento, es una lista de parámetros

[40] PRICE: Precio del alojamiento por día

[41] MINIMUN_NIGHTS: Número mínimo de noches

[42] MAXIMUM_NIGHTS: Número máximo de noches

[43] MINIMUN_MINIMUN_NIGHTS: El valor mínimo de noches que hay en el calendario 365

[44] MAXIMUM_MINIMUM_NIGHTS: El valor máximo mínimo de noches que hay en el calendario 365

[45] MINIMUM_MAXIMUM_NIGHTS: El valor mínimo máximo de noches que hay en el calendario 365 dias

[46] MAXIMUM_MAXIMUM_NIGHTS: El valor máximo de máximas de noches que hay en el calendario 365 dias

[47] MINIMUM_NIGHTS_AVG_NTM: media de mínimo de noches en el *Airbnb*

[48] MAXIMUM_NIGHTS_AVG_NTM: media de máximo de noches en el *Airbnb*

[49] CALENDAR_UPDATED: Columna sin valor

[50] HAS_AVAILABILITY: Indica si está disponible { t/f}

[51] AVAILABILITY_30: Indica la disponibilidad en 30 días

[52] AVAILABILITY_60: Indica la disponibilidad en 60 días

[53] AVAILABILITY_90: Indica la disponibilidad en 90 días

[54] AVAILABILITY_365: Indica la disponibilidad en 365 días

[55] CALENDAR_LAST_SCRAPED: Indica la fecha de captación de datos

[56] NUMBER_OF_REVIEWS**:** Indica el número de valoraciones

[57] NUMBER_OF_REVIEWS_LTM: Número de valoraciones en los últimos 12 meses

[58] NUMBER_OF_REVIEWS_L30D: Número de valoraciones en los últimos 30 días

[59] FIRST_REVIEW: Fecha la primera reseña

[60] LAST_REVIEW: Fecha de la última reseña

[61] REVIEW_SCORES_RATING: Puntuación promedio

[62] REVIEW_SCORES_ACCURACY: Puntuación en exactitud

[63] REVIEW_SCORES_CLENLINESS: Puntuación en limpieza

[64] REVIEW_SCORES_CHECKIN: Puntuación en el registro

[65] REVIEW_SCORES_COMMUNICATION**:** Puntuación en comunicación con dueño

[66] REVIEW_SCORES_LOCATION: Puntuación de la localización de alojamiento

[67] REVIEW_SCORES_VALUE: Puntuación en el valor del alojamiento

[68] LICENSE: Tipo de Licencia del alojamiento

[69] INSTANT_BOOKABLE: Indica si una persona puede reservar un alojamiento sin que el dueño lo verifique/acepte

[70] CALCULATED_HOST_LISTING_COUNT: Cantidad de alojamiento que tiene el dueño en la zona.

[71] CALCULATED_HOST_LISITINGS_COUNT_ENTIRE_HOMES: Número de casas y apartamentos que tiene el dueño.

[72] CALCULATED_HOST_LISITING_COUNT_PRIVATE_ROOMS: Número de salas privadas que tiene el dueño en la zona.

[73] CALCULATED_HOST_LISTING_COUNT_SHARED_ROOMS**:** Número de habitaciones compartidas que tiene el dueño en la zona.

[74] REVIEWS_PER_MONTH: Reseñas por Mes

2.1.1. Eliminación columnas

Descritas todas las columnas iniciales se procedió a eliminar las columnas que no aportaban información para la creación de los modelos predictivos. Se muestra un listado de las columnas y la razón por la que se ha decidido borrarlas:

- ID: Solo contiene un número que indica el alojamiento que es, lo puede causar confusión en el modelo.
- LISTING_URL: Solo contiene un enlace y no sirve como parámetro
- HOST_NAME: Proporciona solo un nombre y no proporciona información para predecir.
- SCRAPE_ID: Columna con el mismo valor para todos los alojamientos, este tipo de parámetros el modelo no los puede interpretar de forma correcta y toma poca relevancia para predecir.
- PICTURE_URL: Contiene la dirección de la imagen del alojamiento y no se puede utilizar para predecir.
- LAST_SCRAPED: Indica la fecha de la última vez que se realizó la captación de datos y no es información importante para el modelo.
- HOST_ID: La codificación del propietario no es un dato relevante que se pueda usar en el modelo.
- DESCRIPTION: Proporciona información en formato texto, se elimina porque requiere un tratamiento específico para usar esta variable y queda fuera el trabajo.
- HOST_URL: Solo contiene un enlace y no aporta información.
- HOST_VERIFICATIONS: Proporciona la misma información que HOST_IDENTITY_VERTIFIED
- PROPERTY_TYPE: Proporciona la misma información que ROOM_TYPE
- NEIGHBOURHOOD_CLEANSED: Proporciona la misma información que NEIGHBOURHOOD_GROUP_CLEANSED
- CALENDAR_UPDATED: No hay valores en esta columna, no aporta información.
- BATHROOMS: No hay valores en esta columna, no aporta información
- NEIGHBORHOOD_OVERVIEW: Proporciona información en formato texto, se elimina porque requiere un tratamiento específico para usar esta variable y queda fuera el trabajo.
- NAME: Indica el nombre del alojamiento, información no útil para el modelo
- HOST_THUMBNAIL_URL: Proporciona solo un enlace de una imagen en

pequeño.

- HOST_PICTURE_URL: No aporta información el enlace de la foto del propietario del alojamiento.
- HOST_ABOUT: Proporciona información en formato texto, se elimina porque requiere un tratamiento específico para usar esta variable y queda fuera el trabajo.
- HOST_LISTINGS_COUNT: Aporta la misma información que HOST_TOTAL_LISTINGS_COUNT
- NEIGHBOURHOOD: Proporciona la misma información que NEIGHBOURHOOD_GROUP_CLEANSED
- FIRST_REVIEW: Solo indica la fecha no aporta más información.
- LAST_REVIEW: Solo indica la fecha no aporta más información.

2.1.2. Análisis columnas

Se muestra un análisis de varias columnas de la base de datos para tener una mejor idea y comprensión de los datos antes de continuar con el trabajo. Se exponen los gráficos de las columnas seleccionadas con una descripción.

Room_type

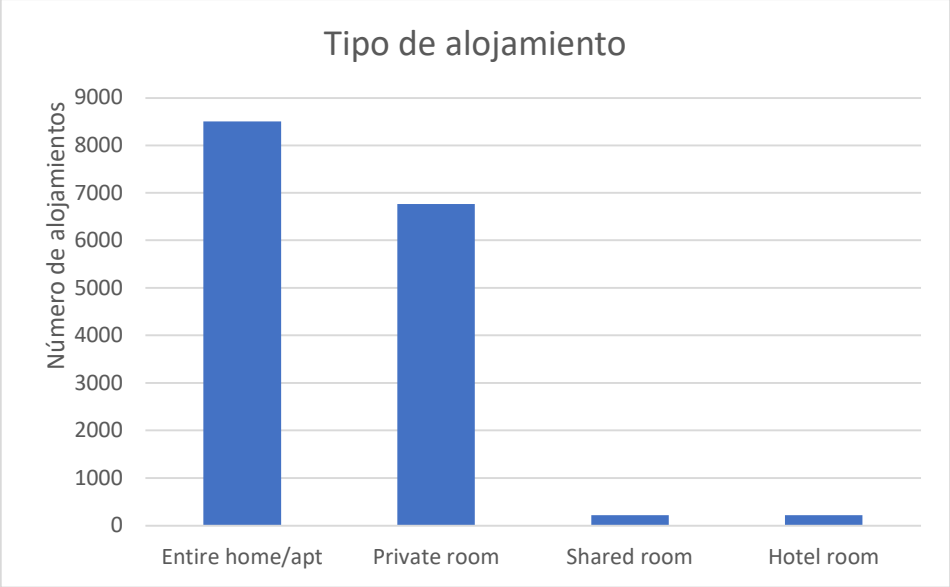


Gráfico 2.1.2.1. Se indica el número de valores en cada clasificación de la columna Room_type

Airbnb se centra en *Entire home/apt y Private room*. Hay una diferencia considerable respecto a *Shared room* y *Hotel room*. En el caso de la habitación de hotel, hoy en día existe una gran demanda, pero hay una gran cantidad de páginas especializadas en este tipo de alojamientos.

Host_is_superhost

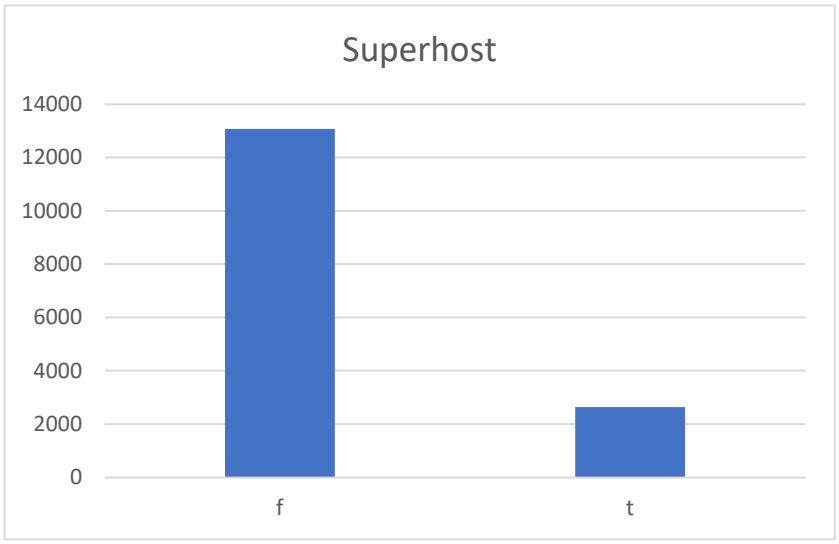
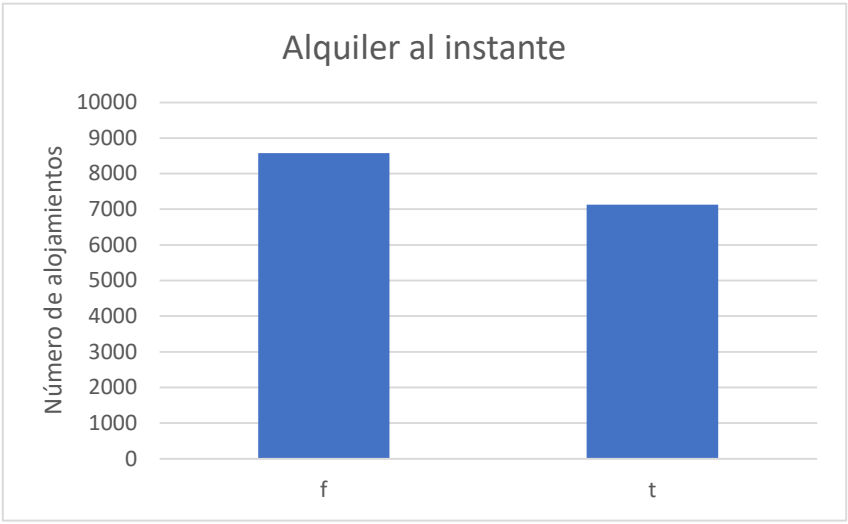


Gráfico 2.1.2.2. Representación del número de alojamientos de cada tipo en columna Host_is_superhost

La mayoría de propietarios no son considerados como propietarios a destacar respecto a los demás. En esta plataforma es muy difícil obtener este logro, los propietarios que son galardonados con este nombre, son personas que tienen las mejores valoraciones y con más experiencia en *Airbnb*.

Instant_bookable



Gráfica 2.1.2.3. Representación del número de alojamientos de columna Instant_bookable

Poder alquilar al instante es una ventaja respecto a otros alojamientos que no lo permiten, el cliente que desea alquilar puede concretar de forma más rápida donde se hospedará y reducir el tiempo de búsqueda. Si no se recibe una respuesta rápida las personas se decantan por otras opciones. *Airbnb* en Barcelona el número de alojamientos que permiten realizar la reserva de este modo está bastante igualado con los que no.

Host_has_profile_pic

Gráfico 2.1.2.4. Representación del número de alojamientos de cada tipo en columna Host_has_profile_pic

La cantidad de propietarios con foto de perfil es mucho mayor que el número de propietarios sin foto de perfil. Es un parámetro a enfatizar, para clientes que deseen alquilar el alojamiento les genera más seguridad el poder poner cara al propietario.

Minimum_nights

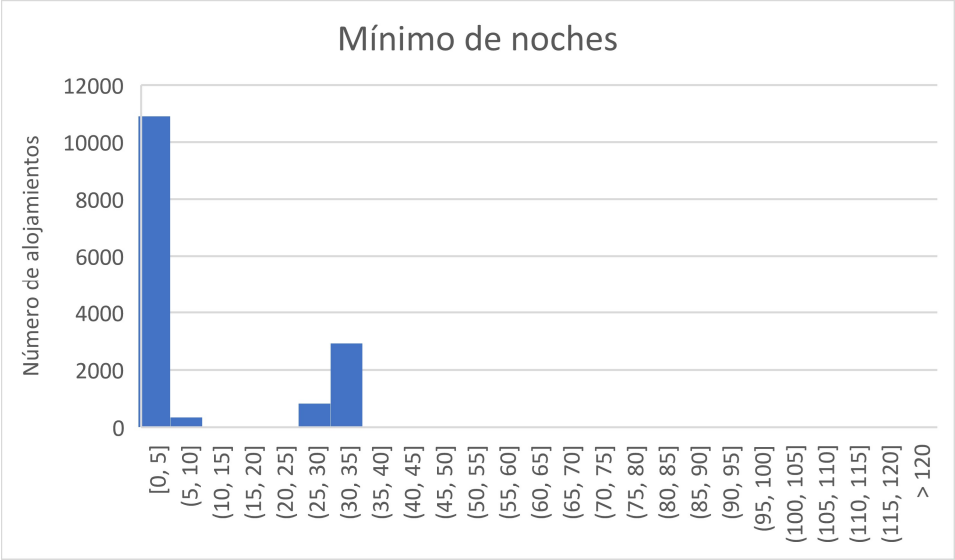


Gráfico 2.1.2.5. Histograma del número mínimo de noches en los alojamientos hasta un año

La mayoría de propietarios consideran no poner un número mínimo de noches para quedarse en el alojamiento. Esto aumenta la rotación de personas, pero puede aumentar el número de alquileres al no crear una limitación. Existen 183 alojamientos donde el numero mínimo supera los 120 días.

Maximum_nights

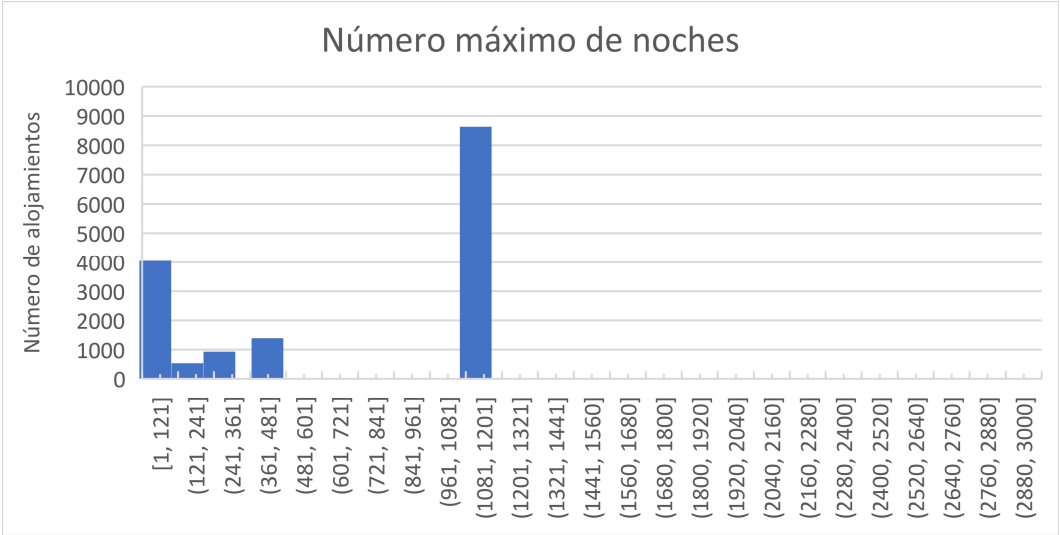


Gráfico 2.1.2.6. Histograma del número máximo de noches en los alojamientos

La gran mayoría de propietarios ponen el valor por defecto que proporciona *Airbnb* en su alojamiento, originando la barra más elevada. *Airbnb* es una página destinada para alquilar durante un periodo corto. Hay otras páginas destinadas para periodos largos de estancia.

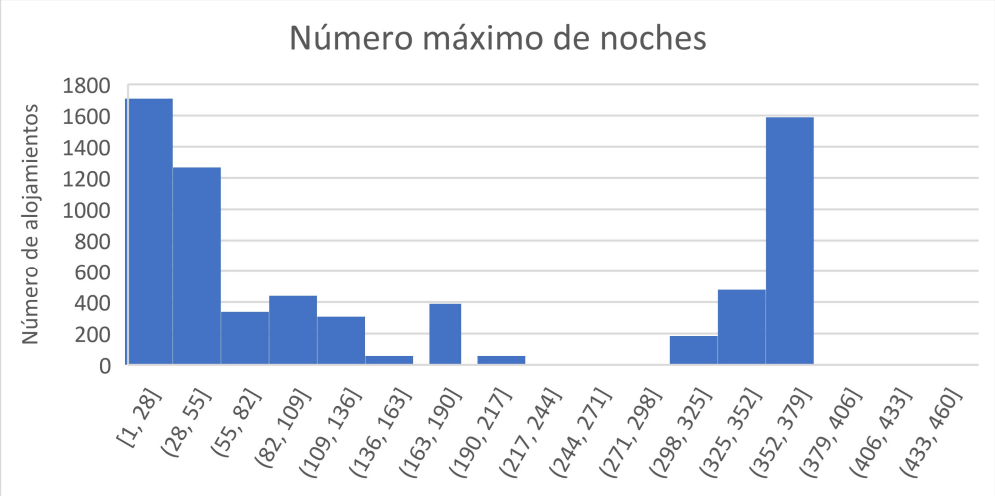


Gráfico 2.1.2.7. Histograma del número máximo de noches en los alojamientos, suprimiendo mayores de 400 días.

Los alojamientos que no se quiere superar el mes y el año de alquiler están muy igualados. Entre el mes y el año existe bastante variabilidad de propietarios.

Number_of_reviews

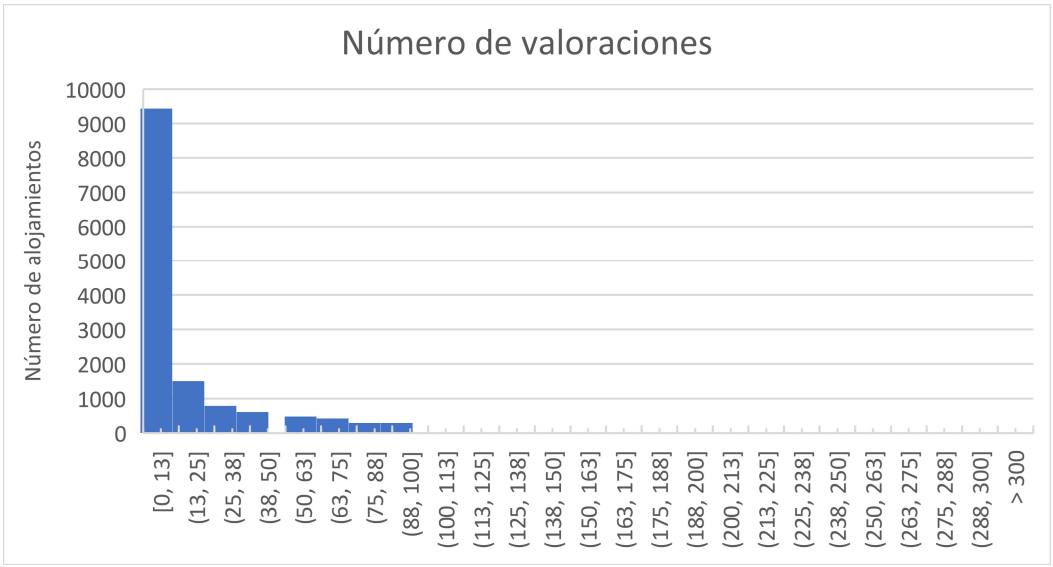


Gráfico 2.1.2.8. Histograma del número de valoraciones en alojamientos

Hay casos particulares donde el valor de las reseñas superan el valor de 300. Los alojamientos que suelen tener un gran numero de valoraciones es porque están bastante

solicitados. A medida que aumenta el número de valoraciones, el número de alojamientos disminuye, hay pocos alojamientos con muchas valoraciones.

Review_scores_cleanliness

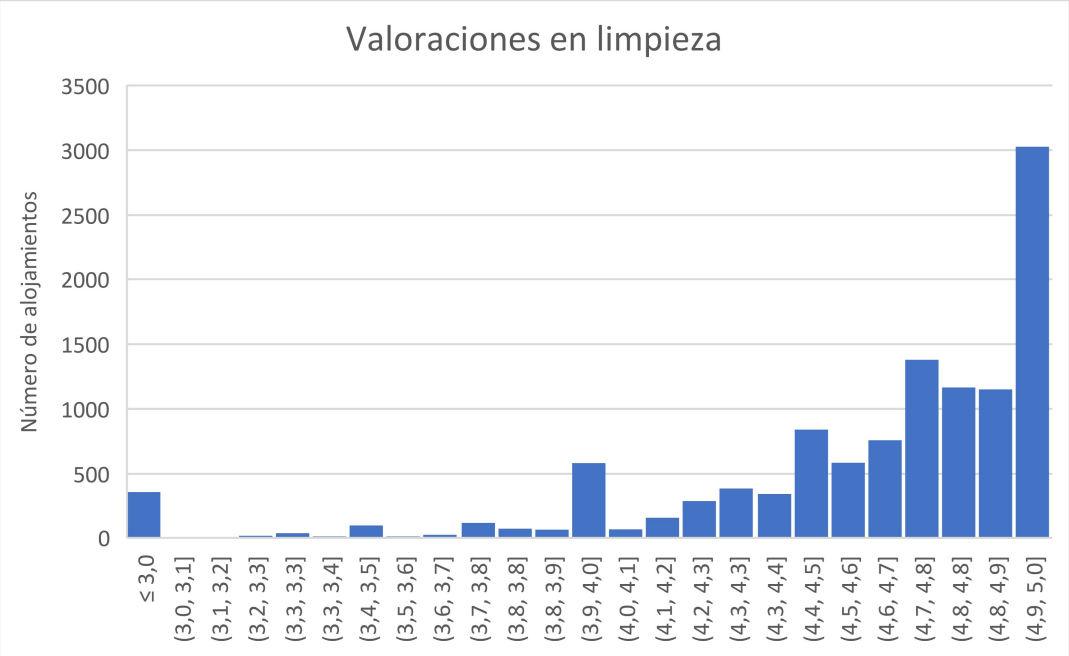


Gráfico 2.1.2.9. Histograma de la puntuación de las valoraciones en limpieza de alojamientos

Las valoraciones son muy positivas, destacan el número de alojamientos con valoraciones cercanas al 5, la puntuación máxima. Hay un pequeño número de alojamientos donde obtienen una mala valoración. La diferencia de la cantidad de valores es muy notable en las franjas con buenas valoraciones.

Review_scores_location

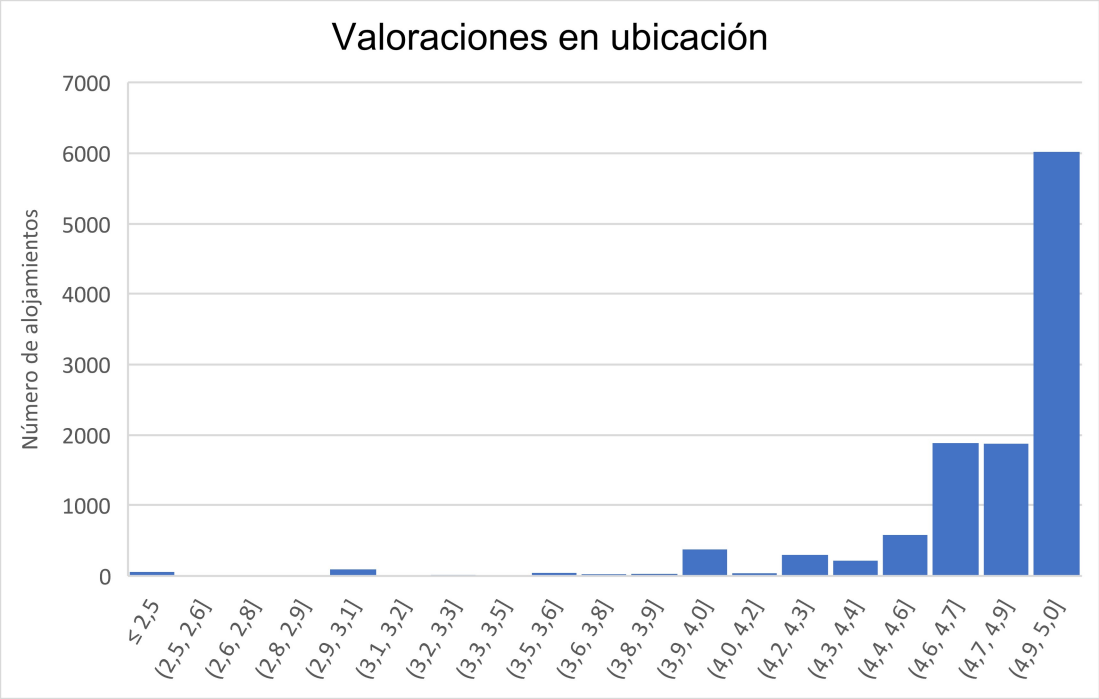


Gráfico 2.1.2.10. Histograma de la puntuación de las valoraciones en ubicación de alojamientos

En la mayoría de los alojamientos, los clientes piensan que están bien situados, indicando valoración 5. Hay pocos alojamientos con valoración inferior a 2,5. El número de alojamientos con buenas puntuaciones comienza a crecer a partir de 4,5. Hay una gran diferencia en el número de alojamientos con puntuación 5 respecto a las demás puntuaciones.

Bedrooms

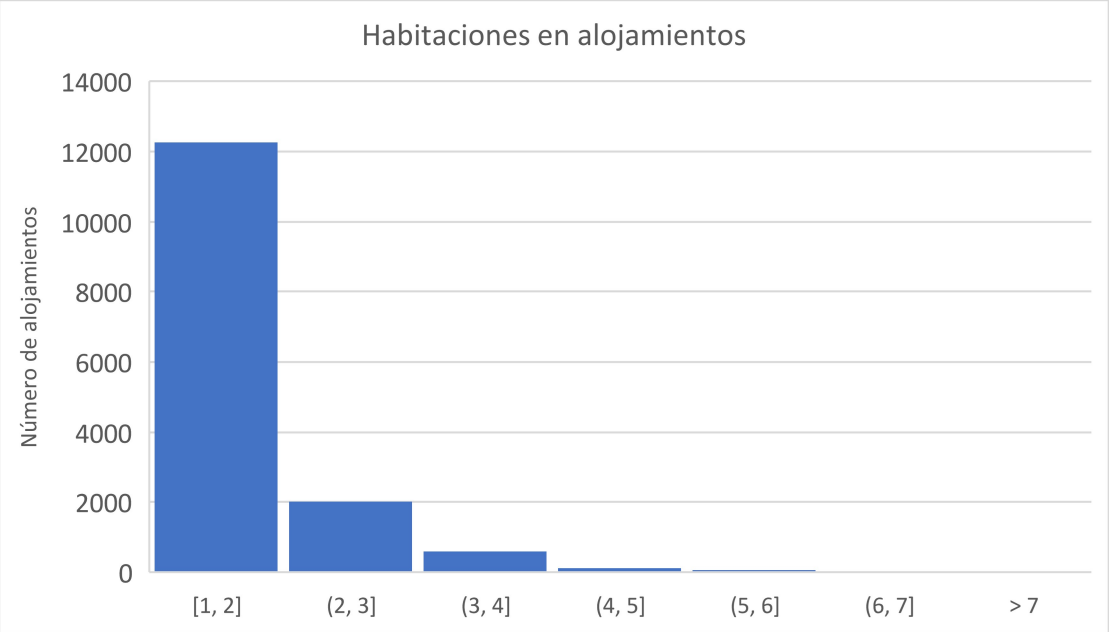


Gráfico 2.1.2.11. Histograma del número de habitaciones para dormir en alojamientos

Airbnb en Barcelona destacan los alojamientos con 1 o 2 habitaciones. Se dan pocos casos donde más de 7 habitaciones, son alojamientos bastante grandes. El número de alojamientos con 3 o 4 habitaciones es elevado pero no tanto como los de 1 o 2 habitaciones.

Neighbourhood_group_cleansed

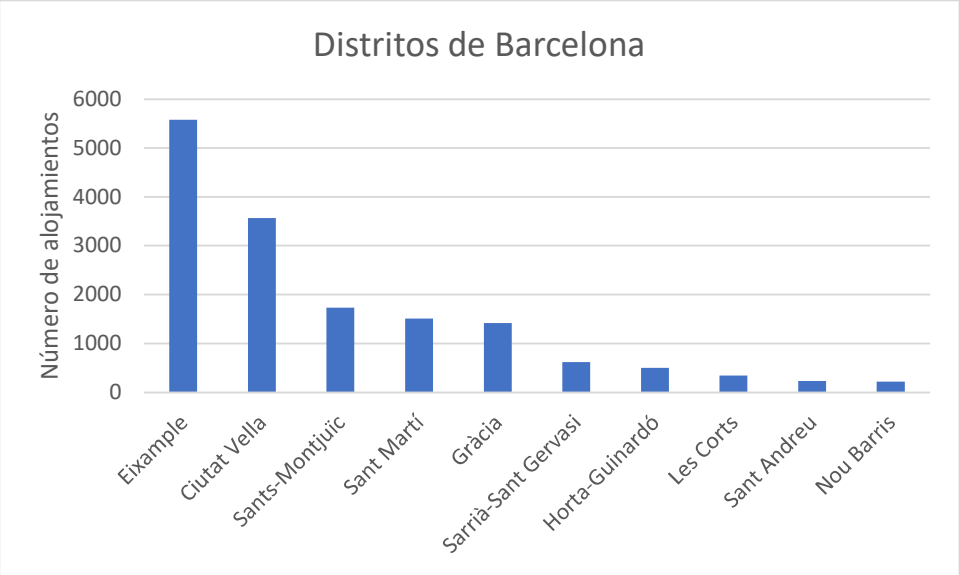


Gráfico 2.1.2.12. Representación del número de alojamientos en cada distrito de Barcelona

La zona de Barcelona donde hay más alojamientos es Eixample. Les corts, Sant Andreu y Nou barris son distritos donde hay pocos alojamientos. Eixample es una zona muy concurrida, donde hay muchos turistas y tiendas.

Price

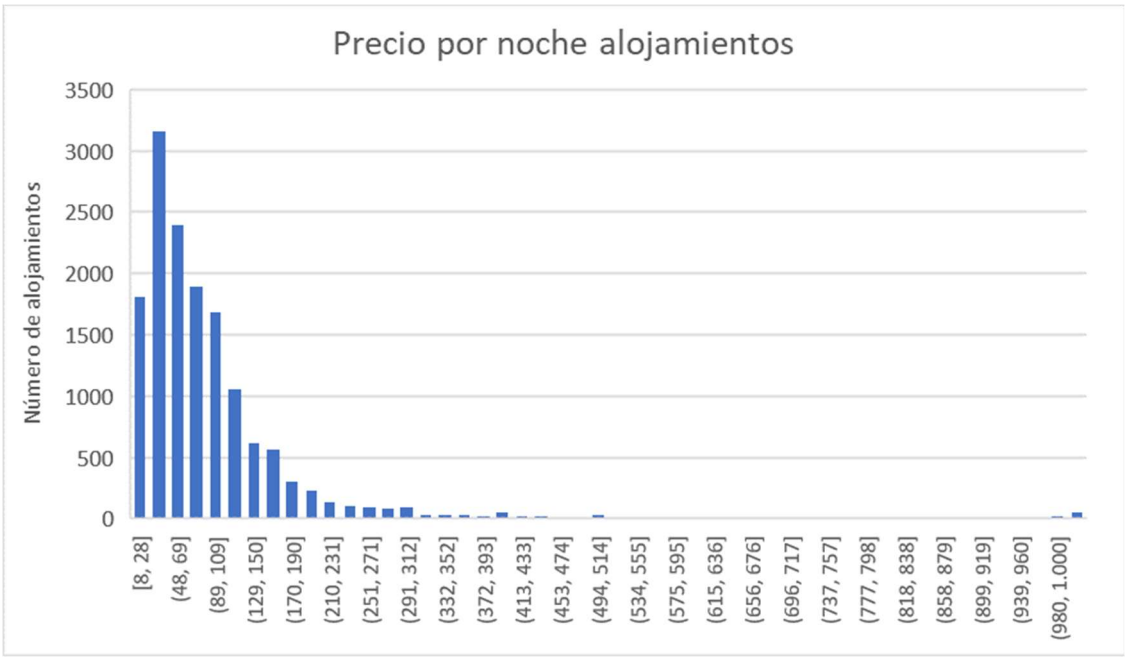


Gráfico 2.1.2.13. Histograma representando del precio noche de los alojamientos de Barcelona

Se hallan 54 alojamientos que superar los 1000€ la noche, son casos particulares y son para alojamientos de alta gama. Predomina los precios en la franja de 29 a 48 euros la noche. A medida que aumenta el precio, el número de alojamientos disminuye.

3. PREPARACIÓN DE LOS DATOS

3.1. Detección y tratamiento de datos ausentes

Los datos ausentes o *missing values* son posiciones donde debería estar un valor, pero no se especificó información. Los valores se nombran de esta forma porque no existen o no están registrados cuando se ha realizado la recopilación de datos. Detectarlos es una tarea a destacar en el proceso, ya que *sklearn* no permite generar modelos a partir de datos con valores ausentes.

Al estar introducidos los datos en un *Dataframe* para poder usar Pandas, se utilizó una función que indica la cantidad de valores ausentes que hay en cada columna. En el caso de que haya un valor de este tipo, toma un valor de *NaN*.

A continuación, se muestra en el gráfico 3.1.1 los valores de las columna que superan el número de 5 *missing values* .

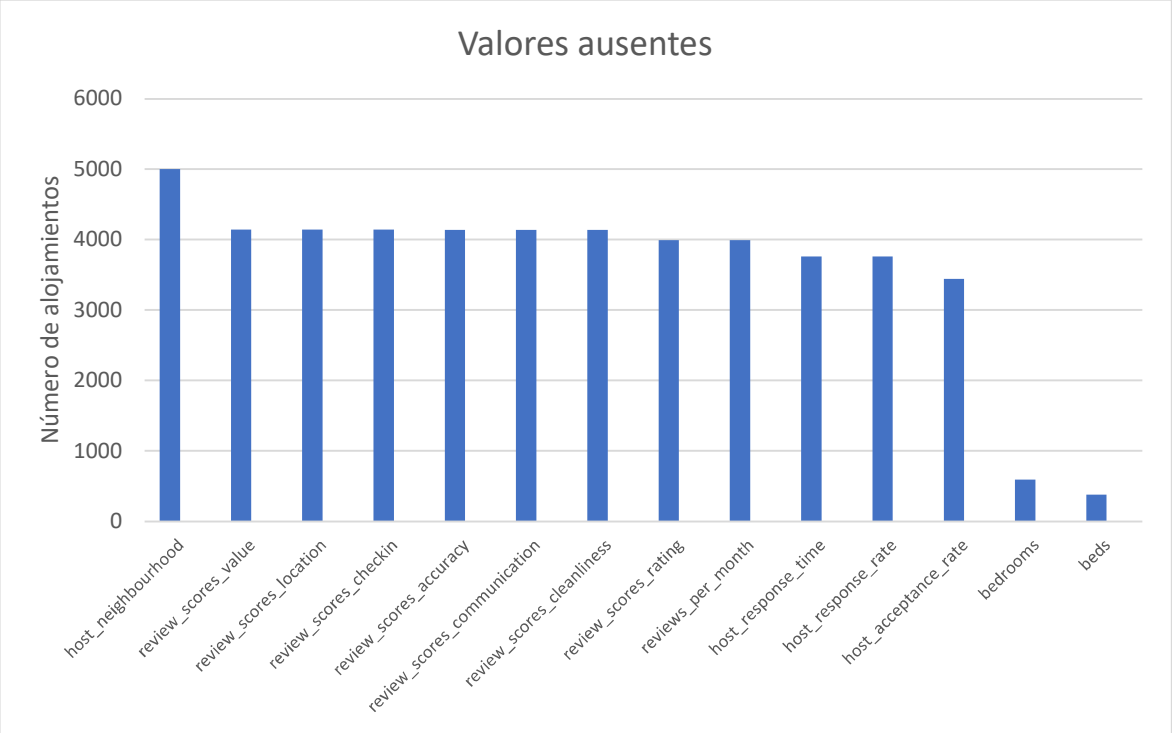


Gráfico 3.1.1. Histograma representando número de valores ausentes de cada columna

Hay varias columnas que poseen un gran número de valores ausentes. De las columnas *Review_scores_rating,Review_scores_accuracy,Review_scores_cleanliness, Review_scores_checkin,Review_scores_communication,Review_scores_location, Review_scores_value y Reviews_per_month* se decidió asignarles un valor 0 . Este valor 0 es un valor real, ya que si las filas no poseen valor es porque nadie les ha realizado una reseña. Un 35% de las filas totales de alojamientos poseen valores *NaN*.

Respecto a las columnas *Host_neighbourhood, Host_response_time, Host_response_rate, Host_acceptance_rate* se decidió eliminarlas ya que sus valores no se podían simular pudiendo provocar incongruencias en los modelos de predicción.

Respecto a las demás columnas, al presentar un valor muy pequeño de valores ausentes fueron eliminadas las filas que los contenían. Se asegura que no se crean datos que no son ciertos y pueden provocar malas predicciones o casos atípicos.

3.2. Ajuste de tipos de variables.

En esta fase del proyecto se procedió a codificar de forma correcta las columnas del archivo de datos que se dispone.

En primer lugar, se modificaron las columnas que estaban codificadas con true/false como columnas binarias. En el caso de True se pone un 1 y en False un 0, es necesario porque los modelos de predicción necesitan variables numéricas.

Este proceso se realizó en las columnas *Host_is_superhost, Host_has_profile_pic, Host_identity_verified, Has_availability y Instant_bookable*.

En segundo lugar, se procedió a modificar la columna *License*. A partir de esta columna se crearon 3 columnas nuevas, *Excepción, Sin licencia y Con licencia*. Las filas que se denomina con Licencia eran las que tenían HUTB y un código de números. Por otro lado, las que se categoriza como excepción eran las que estaban como Exempt. Las otras filas han sido clasificadas como Sin licencia. Se indica 1 si la fila pertenece a la columna o 0 si no, como mínimo, cada fila tiene una columna a la que pertenece.

De la misma forma que con la columna *License*, a partir de la columna *Host_location* se crearon 2 columnas, *Local y Extranjero*. Las dos columnas estaban compuestas por 1 y 0, se denomina Local cualquier propietario que sea Español, por lo contrario, se denomina extranjero.

Sobre la base de la columna *Neighbourhood_group_cleansed* se crearon 10 columnas nuevas. Las columnas son *Ciudad vella, Eixample, gracia, Horta-guinardó, Les corts, Nou barris, Sant andreu, Sant martí, Sants-montjuic y Sarrià-santgervasi*. Estas columnas igual que las anteriores son binarias, se indica 1 si el alojamiento pertenece a esa zona o 0 en caso contrario.

Se formaron 4 columnas, *Apartamento entero, Habitacion de hotel, Habitacion privada y Habitacion compartida*, derivadas de la columna *Room_type*. Las 4 columnas son binarias y indican el tipo de alojamiento que se desea alquilar.

Del mismo modo que las anteriores columnas descritas, partiendo de la columna *Bathrooms_text*, se formaron 3 columnas. La primera columna indica el número de baños que dispone el alojamiento. La segunda es una columna binaria e indica si el baño es compartido. La última columna, igual que la segunda, es binaria e indica si el baño es privado. Varias filas no poseen un valor entero respecto al baño, eso es porque el baño no posee ducha.

La columna *Amenities* para cada alojamiento proporciona una serie de servicios, se decidió crear columnas de los servicios más destacables. Es un parámetro importante debido a que muchas personas toman la decisión de alquilar o no según las características de este. En este caso se descompuso cada fila para poder identificar las características de forma correcta y crear columnas. Solo se generaron las columnas que la suma total de alojamientos que tenían ese servicio fuera inferior al 10% ya que se consideraban destacables respecto a los otros.

Generadas las columnas de los diferentes servicios, se decidió crear nuevas columnas derivadas de agrupaciones de los servicios. Analizando el listado de datos, se encontró que en muchos casos los propietarios de los alojamientos mencionaban muchas marcas. Se consideró que este podía ser un parámetro a considerar relevante porque una persona se puede ver atraída si se muestra la marca del objeto/servicio. Por lo tanto, se tomó la decisión de crear 4 columnas nuevas, Jabón cuerpo, Champú, Refrigerador y Acondicionador. Estas 4 columnas indicaban 1 si en el alojamiento se indicaba que había alguno de los objetos con su marca. Después se analizó que el número total de alojamientos que tenían valor 1 en estas

columnas no superara el 10% del total. Una vez se generaron estas 4 columnas, las derivadas se eliminaron para evitar correlaciones no deseadas.

A continuación, se crearon 3 columnas nuevas. Una fue apodada como *Video Consola* ya que se distinguió que varios alojamientos indicaba que había un tipo de video consola. Se determinó crear esta columna indicando 1 si había video consola en el alojamiento. En la otra columna que se creó, se indica 1 si el alojamiento disponía de televisión con alguna plataforma de reproducción de series o películas, como Amazon o Netflix. Se observó como en varios alojamientos lo ponía como servicio. Esta columna fue nombrada como *Televisón con reproductor.* Por último, se construyo una columna apodada como *Wifi describiendo velocidad.* Se generaron muchas columnas donde solo se indicaba la velocidad a la que iba el router del alojamiento, por lo que se determinó formar una columna que unificara todas, 1 se refería que en el alojamiento se señala la velocidad. De igual forma que la anteriores una vez creadas la columnas se eliminaron las columnas con las que se generaron las nuevas. Se estudiaron las columnas para saber si el número de alojamientos en cada una no superara el 10% del total. Se decidió eliminar la columna Wifi describiendo velocidad debido a que superaba este valor.

Por otro lado, se generó una columna nueva nombrada *Espacio de trabajo.* Analizando las columna se identifico que se formaban varias con el servicio Espacio de trabajo. Se decidió unificarlas y comprobar igual que las anteriores que el numero máximo de alojamientos que poseían este servicio no superara el 10% del total de alojamientos del listado. Además de eliminar las columnas que indicaban Espacio de trabajo, de la misma forma que las descritas antes.

Con estas acciones se redujo la cantidad de columnas creadas a raíz de la columna *Amenities*.

Adicionalmente, se generó una columna del número total de características que posee cada alojamiento, con el nombre *Total de características*.

Por añadir, se cambió el formato de la columna *Price*, se transformó de *string a float* y se renombró a *Precio*. Era necesario eliminar el signo $ del comienzo de cada fila para que el modelo pueda tomar la variable para realizar la predicción.

Se creó una columna que indica la Antigüedad del Propietario en la plataforma. Esta duración se obtiene de la resta entre la columna *Calendar_last_scraped y Host_since*, el día que se realizó la captación de datos y el día que el propietario se registró en *Airbnb*. Se ha llamado a la columna *Antigüedad del propietario*. Muchas personas quieren saber si el propietario lleva tiempo en la plataforma y ya está acostumbrado a la dinámica, genera más tranquilidad. Posteriormente, se eliminaron las columnas *Calendar_last_scraped y Host_since* debido a que poseen una correlación con la columna nueva y no es eficiente para los modelos.

3.2.1. Columna predecir

Esta columna se generó a partir de la sugerencia que crea uno de los autores en la página *Inside Airbnb.*

La columna a predecir, nombrada como *Ratio*, se construye a partir de una fórmula. Se multiplican las reseña por mes por 2 porque mediante un estudio se considera que el 50% de las personas que alquilan el alojamiento son las que realizan una reseña. Seguidamente, se multiplica ese valor por el número mínimo de noches que se debe de quedar una persona en el alojamiento y por 12, puesto que el valor de las reseñas es por mes y se quiere de forma anual. Por último, se divide el valor por 365, ya que se desea el ratio de forma diaria.

$$Ratio = \frac{2 * REVIEWS_{PER_{MONTH}} * MINIMUM_{NIGHTS} * 12}{365}$$

Ecuación 3.2.1.1. Proceso para obtener columna a predecir

Una vez creada la columna se analiza mediante un histograma la cantidad de valores en cada caso. Cabe destacar que los valores que se obtuvieron mayores a 1, se modificaron y se convirtieron en 1 al ser un ratio.

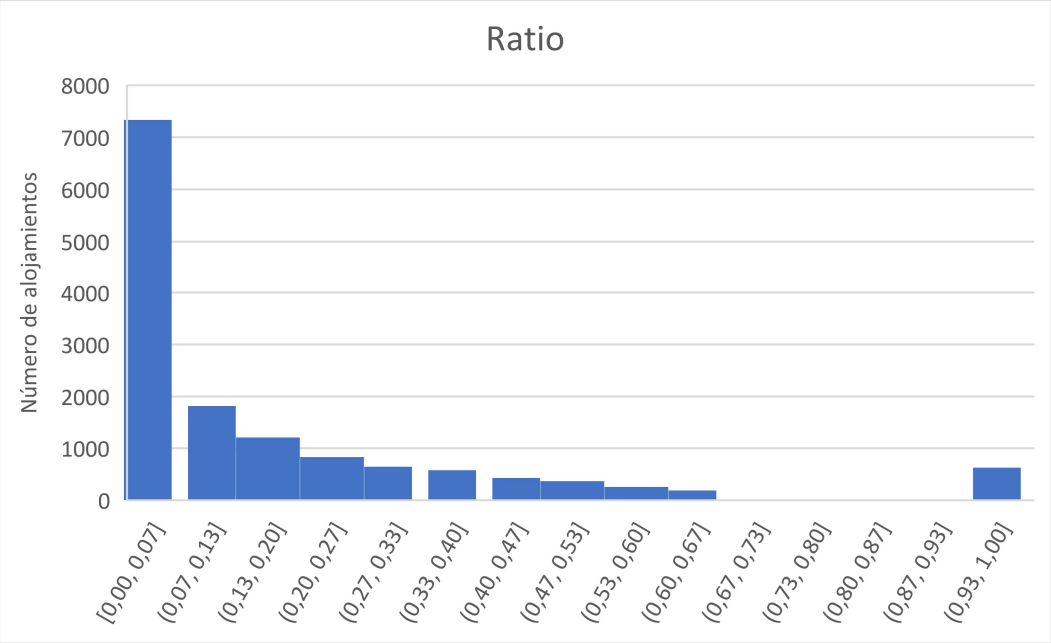


Gráfico 3.2.1.1. Histograma representando variable Ratio

En este caso hay una gran cantidad de valores cercanos al 0 o que su valor es exactamente 0. Cuando se realizó la división en categorías de la columna a ratio alto y ratio bajo, Se obtuvo una buena división a partir de 0.17. Ratio mayores o iguales a 0.17 se consideraron como ratios altas y inferior a 0.17 se consideraron como ratios bajas.

3.3. Análisis de correlación

Esta es la última fase de Preparación de datos, el objetivo es saber la correlación que existe entre todas las columnas. Es algo a destacar y hacer énfasis, ya que dos columnas con una correlación muy alta provoca que el modelo tenga muchos fallos.

Correlación es una medida estadística que indica cuanto están relacionadas linealmente dos variables, es decir, varían de forma alineada a un valor constante. El valor de la correlación está entre -1 y 1, siendo -1 y 1 los dos valores que indican mayor correlación.

En este trabajo cuando encuentra una correlación superior o igual a 0,98 se decide eliminar una de las dos columnas.

Analizando todos los datos se decidió quitar del listado un número total de 59 columnas que obtenían un valor de 1 o -1 con otra columna de correlación.

Cabe destacar que se eliminaron dos columnas formadas con anterioridad, *Local y Baño sin compartir*. Obtenían una correlación muy alta con las otras columnas generadas, *Extranjero y Baño compartido.* Se tomó la decisión de eliminarlas para no causar confusiones al programa y desajustar la tasa de acierto. El objetivo es disminuir la correlación global de la base de datos haciendo énfasis en los casos más extremos.

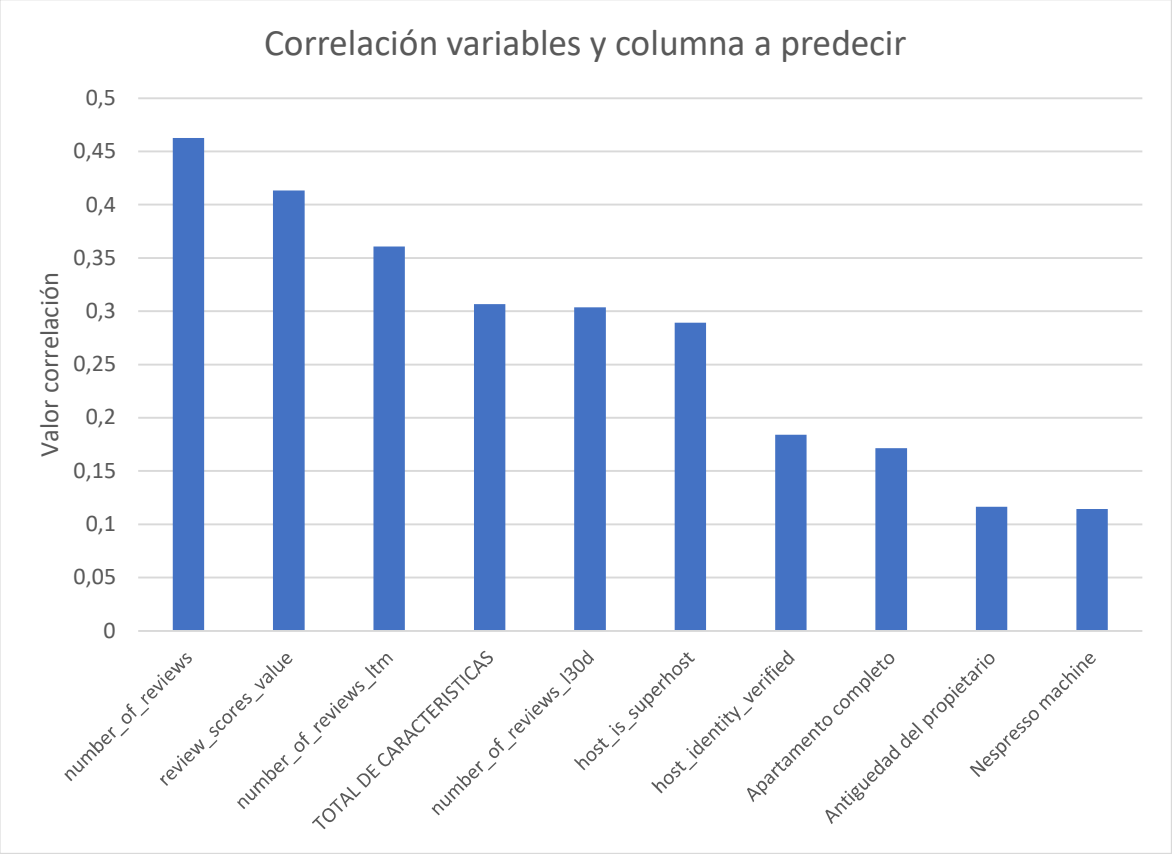


Gráfico 3.3.1. Representación correlación entre variables y columna a predecir

Las variables mostradas son las que mayor valor obtuvieron. En tal caso, se decidió eliminar las columnas con un valor superior a 0,35 porque se consideró que era una relación muy alta.

Al llegar en este punto del trabajo, la base de datos estaba compuesta por 14756 alojamientos distribuidos por toda Barcelona y 382 variables de cada uno de ellos.

4. Modelaje

Finalizadas las fases de Preparación y Compresión de datos se dio paso al Modelaje. Esta es la fase donde se generan los modelos de predicción. La finalidad es obtener un conocimiento mejorado de los modelos de predicción, saber como trabajan y los diversos tipos que hay.

4.1. Modelos seleccionados

Un modelo predictivo es utilizado para poder pronosticar resultados futuros a partir del análisis de patrones y relaciones entre las diferentes variables. Analiza los datos históricos y los actuales para predecir los comportamientos y las tendencias que hay en los datos.

Existen dos tipos de modelos de predicción, modelos de clasificación y modelos de regresión. Los modelos de clasificación predicen si una fila pertenece a una clase u otra. A diferencia de los modelos de regresión, los cuales predicen un valor numérico en concreto. Son empleados en muchas ocasiones para prever el beneficio que puede obtener una empresa con un producto o de un determinado cliente. Ayuda a estimar costes y saber si es viable continuar con el producto o cliente.

A continuación, se exponen los modelos seleccionados en este trabajo con la finalidad de conocer sus correspondientes comportamientos y saber diferenciarlos. En el caso de *Random Forest*, se aclaran los parámetros a modificar para detectar si origina una mejora en la tasa de acierto. Los modelos que se explicarán seguidamente son modelos supervisados, en el momento que se realiza el entrenamiento de los modelos se incluye la columna de *Ratio* que es la solución deseada.

4.1.1. Regresión Logística

La Regresión Logística es un modelo supervisado que se utiliza con el fin de predecir el resultado de una variable categórica, dicho de otro modo, para clasificar. La variable a predecir está limitada a un número de categorías, esta puede ser una variable binaria o continua. Este tipo de modelo permite informar sobre cuáles son las columnas más relevantes para obtener la predicción. Se aplica en múltiples sectores, a modo de ejemplo, en predecir qué tipo de sistema utiliza un usuario en su ordenador y en detectar el tipo de método de pago cuando un cliente realiza un gasto.

El modelo desempeña dos pasos importantes, en primer lugar realiza una relación lineal entre las columnas y la variable a predecir. Se suman todas las variables, excepto la que se desea predecir, de la base de datos, cada una multiplicada por un coeficiente que asigna el modelo. Seguidamente, en el segundo paso se le aplica la función logística al resultado obtenido del primer paso.

Se muestra la expresión de la relación lineal comentada en la ecuación 4.1.1.1.

$$f(x) = \beta_0 x_0 + \beta_1 x_1 + \cdots + \beta_n x_n \qquad \text{para i=0,1…n}$$

Ecuación 4.1.1.1. Polinomio de regresión logística

ß: indica el coeficiente que se le aplica al parámetro i

x: indica el valor del parámetro i

La función logística es la ecuación 4.1.1.2. que se muestra a continuación.

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

Ecuación 4.1.1.2. Función logística

El resultado de la función se interpreta como una probabilidad, se obtiene un valor entre 0 y 1, ambos incluidos. Un ejemplo aplicado al trabajo donde se quiere predecir dos categorías, los valores obtenidos menores de 0.5 correspondían al grupo Bajo y valores superiores a 0.5 correspondían al grupo Alto. Esto es debido a la curva de esta función representada en el gráfico 4.1.1.1.

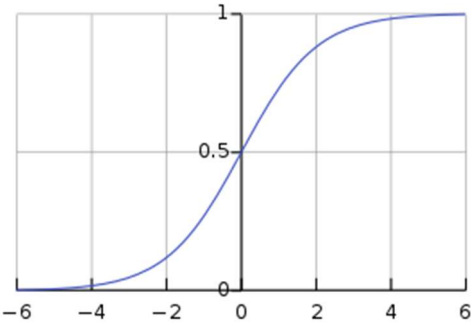


Gráfico 4.1.1.1. Gráfico de limites regresión logística

Se programará en lenguaje Python utilizando la librería *Sklearn*, *sklearn.linear_model.LogisticRegression()*. En este modelo se fijará un parámetro *random_state,* se requiere para controlar la creación de los números aleatorios usados en el modelo.

4.1.2. Árbol de decisión

Se explica el árbol de decisión para entender mejor el segundo modelo utilizado, Random Forest, en el trabajo.

El modelo árbol de decisión se interpreta como un diagrama de flujo. La estructura tipo diagrama de flujo permite tomar todas las decisiones al modelo predictor. Imita o simula el pensamiento a nivel humano, son fáciles de entender. Las variables a predecir pueden ser, igual que en la Regresión Logística, continuas o binarias.

Parte de un único nodo y seguidamente se divide en posibles resultados a la prueba realizada al parámetro elegido por el árbol. A raíz de cada resultado se generan nodos que se dividen en otros. La raíz sirve para unir nodos e indica la condición que debe de cumplir el parámetro.

El árbol está compuesto por diversos tipos de nodos, se muestran a continuación:

- **Nodo raíz**, es el nodo donde se inicia el árbol.
- **Nodo intermedio**, es el nodo que se encuentra entre el nodo inicial y nodo terminal, a partir de este nodo puede aparecer un nodo intermedio.
- **Nodo terminal**, es el último nodo e indica la decisión tomada por el árbol de decisión.

Respecto a la información que proporciona cada nodo se muestra a continuación:

- **Entropía:** proporciona un valor entre 0 y 1. Si se indica un valor cercano al 1, se refiere que los datos están correctamente distribuidos, es decir, existe la misma cantidad de cada clasificación de la prueba realizada. En cambio, si hay un valor cercano al 0, se refiere que los datos que hay solo pueden pertenecer a una clasificación.
- **Condición:** Indica la condición que se le realiza al parámetro seleccionado.
- **Muestras:** indica la cantidad de datos cumplen para llegar a ese nodo.
- **Valores:** indica la cantidad de datos que van a cada raíz del total de datos que llegan a ese nodo.
- **Clase:** esta información solo está en los nodos terminales, indica la decisión que se realiza final.

En la generación del árbol se forman diversas condiciones para las variables y se calcula la entropía de cada una. La condición que obtenga un valor de entropía 1 será la seleccionada como nodo raíz. Los siguientes nodos son los que tienen una entropía inferior. El nodo terminal es el nodo que tiene un valor cercano al 0, es cuando se realiza la clasificación.

Este tipo de modelos de la forma en la que se genera origina que sea un modelo inestable. Como punto de partida el modelo selecciona un parámetro de los que hay en la base de datos con la que se forma una condición, considera que es la mejor condición. Puede que esa condición no acabe siendo la más eficiente y correcta. Si se cambia los datos de entrenamiento o el parámetro de inicio, puede dar lugar a nuevas condiciones modificando por completo el árbol.

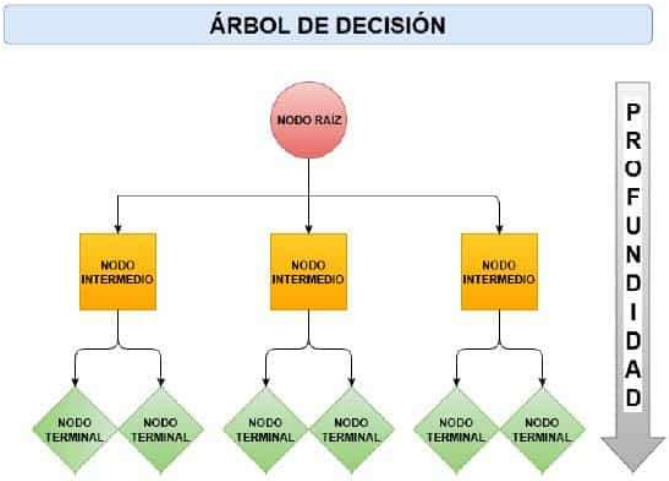


Figura 4.1.2.1. Representación forma árbol de decisión

4.1.3. Random Forest

Random Forest es un modelo que combina múltiples árboles de decisión con el objetivo de obtener una mejor predicción que con un solo árbol de decisión. Cada árbol de decisión ve una porción distinta de los datos, ninguno ve todos los datos de entrenamiento.

Se crean subconjuntos de datos a partir de los datos originales de forma completamente aleatoria, es decir, se utilizan un número limitado de filas y columnas de datos para cada árbol. Como es de esperar, hay datos que están en múltiples conjuntos de datos, no solo se utiliza para crear un subconjunto. La finalidad de utilizar los datos de esta forma es que cada árbol pueda entrenar y adaptarse a los datos con distintas muestras para realizar una misma predicción. Se logran árboles de decisión con diferentes perspectivas para predecir.

Generados todos los árboles de decisión, se juntan los resultados obtenidos de cada árbol. De esta forma, al unificarlos, los errores que hayan podido tener algunos árboles se compensan y se obtiene un modelo más fiable.

A la hora de combinar los árboles se utiliza una técnica llamada *Soft-voting*. Consiste en dar más importancia a las predicciones que realizan los árboles de decisión con mejores métricas de evaluación.

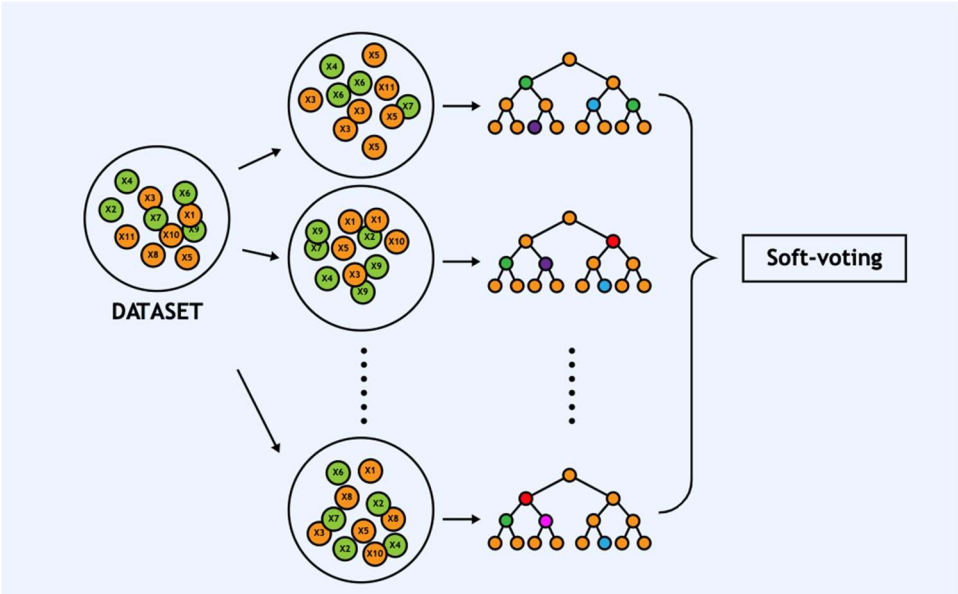


Figura 4.1.3.1. Representación forma Random Forest

Se programará en lenguaje Python utilizando la librería *Sklearn*, *sklearn.linear_model. RandomForestClassifier()*. En este modelo igual que con el modelo de Regresión Logística se fijará el parámetro *random_state.*

Se decidió modificar tres parámetros para la generación de los múltiples modelos de Random Forest, a continuación se explican:

- ***Max_depth***: indica la profundidad del árbol. Por defecto, los arboles se desarrollan tanto como sea posible.
- ***N_estimators***: indica la cantidad de árboles que se crean al generar el modelo. Un número elevado es correcto ya que se realiza una votación de más cantidades y se aumenta la eficacia. Por otro lado también se aumenta el tiempo de generación de los modelos. El valor por defecto es de 100 árboles.
- ***Max_features:*** indica la cantidad máxima de variables para cada árbol. Esto provoca que los arboles se entrenen con muestras aleatorias diferentes. El valor por defecto que *sklearn* asigna es la raíz de total de columnas que dispone el conjunto de datos que se introduce en el modelo.

Se modifican dos parámetro para el modelo *Random Forest* y uno para los árboles de decisión interiores.

5. Validación

La fase de validación es una de las más importantes en todo el proceso realizado. Permite saber si los modelos predicen de forma correcta, es decir, se valora la fiabilidad de los diferentes modelos. Además, se pueden realizar las comparaciones oportunas, ya que se prevé el funcionamiento de los modelos a partir de datos que no se han utilizado para formarlos.

Se debe de observar si los modelos se han ajustado de forma excesiva a los datos de entrenamiento, aquí es donde cabe destacar que es necesario tener en cuenta el concepto *Overfitting*.

5.1. Overfitting

El *overfitting* es un concepto que ocurre en muchos casos cuando se están generando modelos de predicción. Sucede cuando un modelo considera como válidos solo los datos que se han usado para entrenar el modelo. No reconoce ningún otro dato que sea diferente a la tendencia que tiene la base de datos con la que se ha entrenado. No generaliza para los nuevos datos que deba predecir, provocando unos malos resultados y dejando de usar el modelo.

Hay que tener en cuenta que un modelo se sobre ajusta cuando se entrena con una gran cantidad de datos, donde hay muchos detalles. En los árboles de decisión es un fenómeno bastante común, se desarrolla el modelo hasta una profundidad del árbol muy elevada, dando lugar a un sobreajuste muy alto. Por esto, es necesario que se limite este parámetro del modelo.

Para observar si un modelo presenta *overfitting* o sobreajuste, los datos se dividen en dos grupos. Un conjunto de datos va destinado al entrenamiento y otro para el test final. Generado el modelo se comprueba con el conjunto de datos de test, se analizan las métricas de evaluación. Si se identifica que hay sobreajuste se debe de modificar la estructura de los datos.

5.2. Holdout

En este apartado del trabajo se procede a explicar el método de validación Holdout. Este método consiste en dividir la base de datos en dos conjuntos: conjunto de datos de entrenamiento y conjunto de datos de test.

Esta división se realiza de forma aleatoria, pero hay un parámetro llamado *Stratify* el cual permite tener un factor importante en consideración. Stratify es un parámetro que permite una vez indicada una columna, realizar la división de forma que la proporción de categorías en un conjunto y otra sea la misma.

El conjunto de datos de entrenamiento es el que se utiliza cuando se genera el modelo de predicción. A raíz de estos datos, el modelo genera los patrones y se modela. Es un paso que se debe de realizar cada vez que se forme un algoritmo. Entrenado con este conjunto ya está disponible para realizar predicciones con otros datos. La calidad del modelo será proporcional a la calidad de los datos con los que realice el entrenamiento.

El conjunto de datos de test/prueba son los que se comprueba la fiabilidad del modelo. Se observa como el modelo creado funciona y si realiza predicciones correctas. Estos datos solo se utilizan para esta parte, es decir, el modelo no ha visto hasta creados estos datos. El conjunto de datos de test debe de tener un volumen suficiente como para poder crear resultados estadísticos significativos. Los datos son desconocidos para el modelo, en caso contrario se obtienen resultados que no son reales porque el modelo ya sabría la respuesta correcta a predecir.

La función en Python que permite realizar esta separación de datos es *train_test_split.*

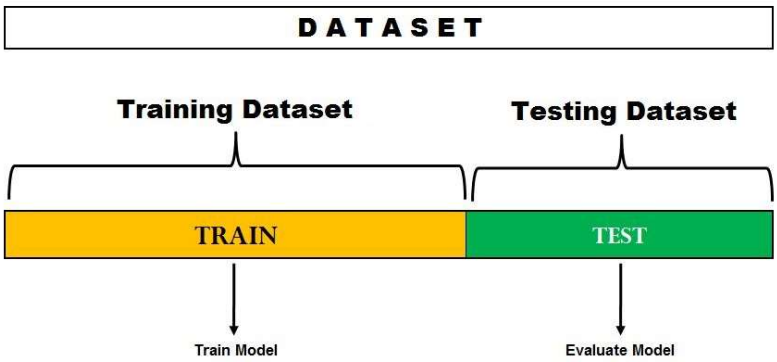


Figura 5.2.1. Representación validación Hold-out

5.3. Validación cruzada

La validación cruzada es una técnica que sirve para comprobar el rendimiento de los modelos de predicción. Garantiza que los resultados obtenidos en las métricas sean independientes entre los datos de entrenamiento y los datos de test. Es una técnica beneficiosa ya que permite saber si el modelo se está ejecutando de forma correcta. Este tipo de validación es más fiable que Holdout, pero es más lenta desde el punto de vista computacional.

Se pueden utilizar los dos tipos de validación a la vez, inicialmente se emplea Holdout para crear los dos subconjuntos y posteriormente se aplica la validación cruzada a la parte de entrenamiento.

Acto seguido se explican varios tipos de validación cruzada.

5.3.1. Validación cruzada de K iteraciones

La validación cruzada de K iteraciones consiste en dividir los datos en K conjuntos. Una vez realizada la división se procede a seleccionar uno de los conjuntos como datos de test y los otros son los datos con los que se entrena el modelo. Este proceso se lleva a cabo hasta k veces, en otros términos, todos los conjuntos de datos pasan a ser una vez datos de test. Finalizadas todas las iteraciones, se efectúa la media aritmética de los resultados de cada iteración para lograr un único resultado.

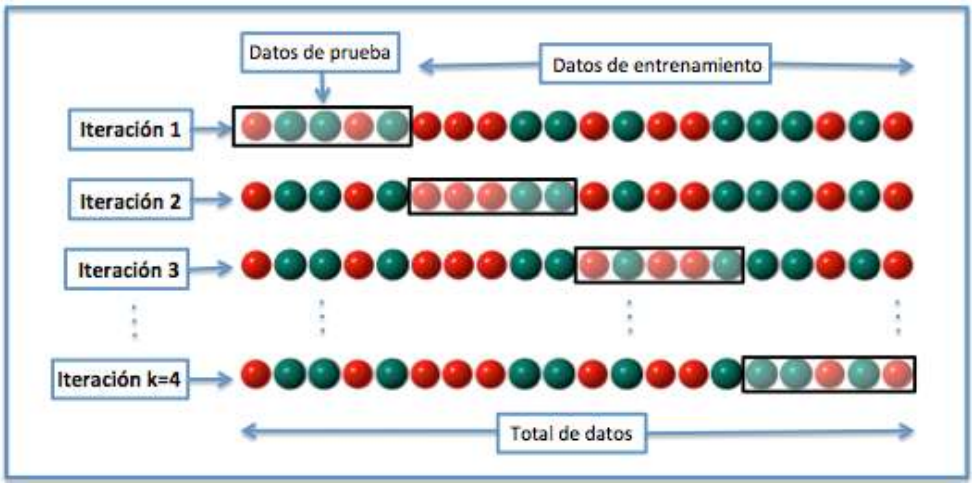


Figura 5.3.1.1. Representación validación cruzada de K iteraciones

5.3.2. Validación cruzada aleatoria

La Validación cruzada aleatoria es un tipo de validación donde se separa de forma aleatoria el conjunto de datos, del mismo modo que se muestra a continuación.

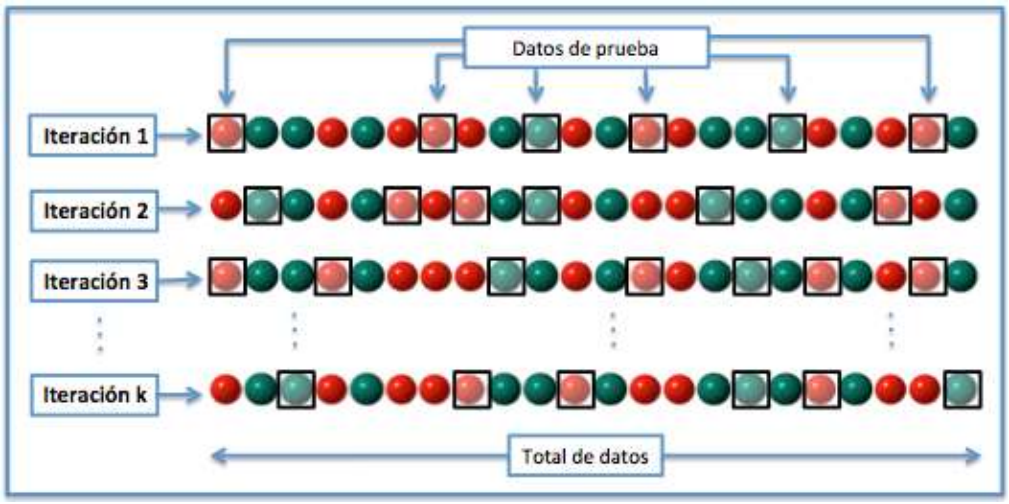


Figura 5.3.2.1. Representación validación cruzada aleatoria

Esta separación se hace el número de veces que se desee, finalizadas todas igual que en el anterior, se realiza la media aritmética de los valores obtenidos en las diferentes iteraciones. La ventaja de este tipo respecto a k-iteraciones es que la división no depende del número de iteraciones. En cambio, hay datos que quedan sin evaluar y otros que se evalúan un gran número de veces.

5.3.3. Validación cruzada dejando uno fuera [LOOCV]

La validación cruzada dejando uno fuera es un tipo de validación que divide los datos de tal manera que selecciona solo una muestra de datos para el entrenamiento del modelo y el resto para realizar el test. Cada iteración selecciona una muestra diferente. Efectuadas todas las iteraciones se procede igual que los anteriores, se calcula la media aritmética de los valores obtenidos dando lugar a un único resultado.

Esta es la validación de las descritas que mayor coste computacional, se debe de realizar un gran número de iteraciones.

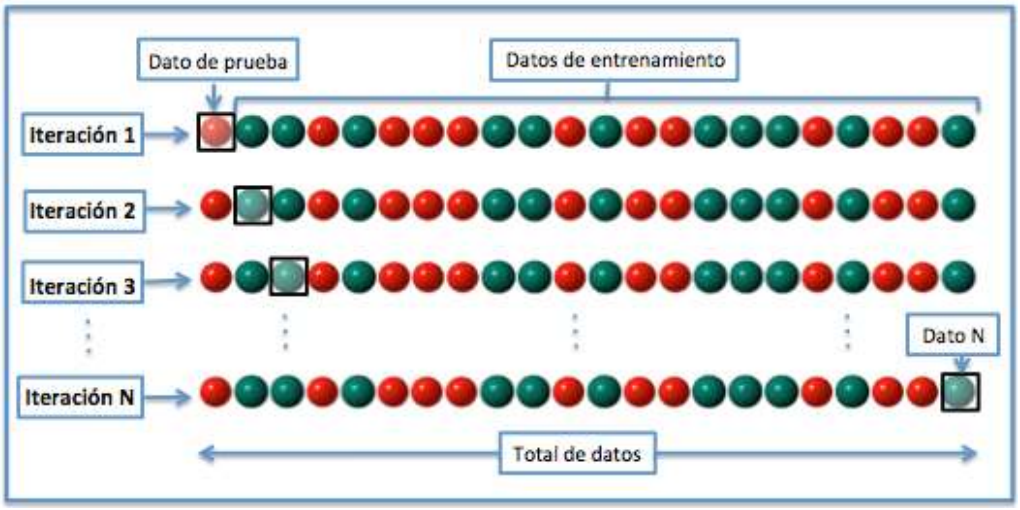


Figura 5.3.3.1. Representación validación cruzada dejando uno fuera

5.3.4. Validación dejando P fuera

La validación dejando P consiste en seleccionar un número de P muestras como datos de test. En este caso, si se selecciona un número P muy alto, será necesario un número no muy alto de iteraciones. En cambio, si el número P es un valor pequeño, será necesario un coste computacional mayor debido a realizar más iteraciones.

Una vez finalizadas las iteraciones, se procede a cuantificar la media aritmética de cada métrica de evaluación del mismo modo que en los casos anteriormente descritos.

5.3.5. Stratified K-Fold

Este tipo de validación cruzada es una variante del tipo K-iteraciones. La diferencia es que en este tipo de validación cuando se realiza la separación de datos en subconjuntos se tiene en cuenta mantener equilibradas las clases. El proceso es igual que en K-iteraciones pero teniendo en cuanta esta característica de los datos. Se suele utilizar en casos donde las clases de la variable a predecir no estén bien distribuidas.

5.4. Métricas de evaluación

Las métricas de evaluación sirven para valorar el rendimiento de un modelo de aprendizaje automático. A continuación se detallan las métricas que se han utilizado en este trabajo para poder observar la tasa de acierto de los modelos creados y obtener un mejor conocimiento de cada una. Como se comenta anteriormente, se desea predecir correctamente los alojamientos categorizados como ocupación Alta, de tal modo se centra en lograr un valor alto de *F1-score.*

5.4.1. Matriz de confusión

La matriz de confusión es la representación matricial de los resultados obtenidos de las predicciones realizadas del conjunto de datos de test del modelo. La matriz tiene las mismas columnas que filas y siempre coincide con el número de categorías a predecir.

Las columnas indican resultado obtenido en la predicción realizada. En cambio, las filas indican el valor que en realidad debe de obtener la predicción realizada. Por lo tanto, es importante que los valores que haya fuera de la diagonal sean pequeños.

La matriz de confusión en este trabajo es la siguiente:

AC	BE
AE	BC

Tabla 5.4.1.1. Matriz de confusión aplicada a los datos

AC: Número de muestras categorizada como altas y en realidad pertenecen a la categoría de altas

AE: Número de muestras categorizada como altas y en realidad pertenecen a la categoría de bajas

BC: Número de muestras categorizada como bajas y en realidad pertenecen a la categoría de bajas

BE: Número de muestras categorizadas como bajas y en realidad pertenecen a la categoría de altas

5.4.2. Accuracy

Accuracy indica el porcentaje total de elementos clasificados correctamente del modelo. Consiste en dividir los aciertos de cada categoría entre el total de muestras que hay en el conjunto de datos de test. Se muestra a continuación de la fórmula para mejor entendimiento. Es una de las métricas más usadas. No se puede observar exactamente cuál es el valor más alto, puede que el modelo realice buenas predicciones solo para una categoría.

$$Accuracy = \frac{AC + BC}{AC + AE + BC + BE}$$

Ecuación 5.4.2.1. Fórmula para obtener valor Accuracy

5.4.3. Precision

Precision indica en porcentaje cuanto preciso o exacto es el modelo de muestras clasificadas de forma correcta. Consiste en dividir el número total de aciertos en una categoría entre los aciertos en esa categoría más la cantidad de muestras que se debería de haber predicho en esa categoría, pero se ha fallado. A continuación se muestra la ecuación basada en la matriz de confusión descrita anteriormente:

$$Precision = \frac{AC}{AC + BE}$$

Ecuación 5.4.3.1. Fórmula para obtener valor Precision

5.4.4. Recall

Recall o sensibilidad indica el número de elementos identificados correctamente de una categoría. Consiste en dividir el número total de aciertos en una categoría entre el total de muestras que hay en el conjunto de datos de test. Se muestra la ecuación para obtener el valor.

$$Recall = \frac{AC}{AC + AE}$$

Ecuación 5.4.4.1. Fórmula para obtener valor Recall

5.4.5. F1-score

En la métrica F1-score utiliza los valores *Recall* y *Precisión* para realizar la media harmónica de estos dos. Consiste en dividir la multiplicación de *Recall* y *Precisión* entre la suma de los dos valores. Posteriormente, se multiplica por 2. Esta métrica al influirle dos métricas distintas, si se obtiene un valor alto es necesario obtener un valor alto de *Recall* y *Precision*, por esta razón ha sido la métrica en la que centrarse en los modelos construidos. Muestra el balance entre las dos métricas.

De igual forma que las anteriores métricas de muestra la ecuación para obtener el valor.

$$F1 - Score = 2 * \frac{Recall * Precision}{Recall + Precision}$$

Ecuación 5.4.5.1. Fórmula para obtener valor F1-score

6. Análisis de los resultados

6.1. Resumen experimentos realizados

Presentada las herramientas y modelos se procede a explicar los experimentos realizados.

En todos los experimentos se ha realizado el método de validación Hold-out. Se descartó la validación cruzada debido a que era necesario una gran cantidad de tiempo que no se disponía. Se separaron los datos, un 65% para entrenamiento de los modelos y un 35% para test.

El primer experimento que se realizó fue crear un modelo de Regresión logística a partir de los datos tratados anteriormente. Este experimento era el experimento base, se obtuvo un primer valor de *F1-score* para predecir la categoría de ratios altos de los alojamientos. No se modificó ningún parámetro del algoritmo, estaban todos con el valor por defecto de *sklearn*.

Seguidamente, se realizó el segundo experimento generando un modelo *Random Forest*. Igual que en el anterior, se mantuvieron los parámetros del modelo por defecto. Llegados a este punto, se compararon los dos modelos. Uno es un modelo lineal y otro no lineal.

A continuación, se procedió a realizar varios experimentos modificando los parámetros *Random Forest.* El objetivo era observar como cambiaba la tasa de acierto del modelo una vez variado ciertos parámetros.

Primero se crearon modelos de *Random Forest* modificando el parámetro de los árboles de decisión interiores. Se seleccionaron los parámetros que mejores puntuaciones de *F1-score* lograban y se crearon nuevos modelos modificando los parámetros elegidos de *Random Forest*.

Finalmente, de forma inversa, se crearon modelos de *Random Forest* modificando los parámetros descritos anteriormente de *Random Forest*. Igual que en el otro experimento, se seleccionaron los parámetros con mejor valor de *F1-score* y se generaron nuevos modelos cambiando el parámetro de los árboles de decisión interiores.

De cada experimento se comenta los resultados obtenidos.

6.2. Primer experimento

El modelo construido todos sus parámetros están por defecto. Se marcó este modelo como resultado base, el resultado que se utiliza para comparar y mejorar con *Random Forest*.

Construido el modelo de *Regresión Logística* los valores de las métricas son las siguientes:

	precision	recall	f1-score
Alto	0.59	0.35	0.44
Bajo	0.73	0.88	0.80
accuracy			0.71
macro avg	0.66	0.62	0.62
weighted avg	0.69	0.71	0.68

Figura 6.2.1. Métricas de evaluación para datos con Regresión Logística

Regresión logística es un modelo lineal, no se adapta correctamente a los datos al obtener métricas bastante bajas.

Tiene un valor *recall* de solo un 35%. Este valor significa que se han categorizado una gran cantidad de alojamientos como altos que en realidad eran bajos. Logra una tasa de acierto muy baja. Por otro lado, el valor *precision* logra un 59%, superior que *recall.* El modelo tiene bastantes fallos prediciendo alojamientos categorizados como bajos que en realidad son altos. Al ser el valor de *recall* en categorías altas inferior a 50%, penaliza mucho al valor de *f1-score*.

6.3. Segundo experimento

Igual que en el experimento anterior el modelo *Random Forest* generado es con los parámetros dejados por defecto.

	precision	recall	f1-score
Alto	0.78	0.69	0.73
Bajo	0.85	0.90	0.88
accuracy			0.83
macro avg	0.82	0.80	0.80
weighted avg	0.83	0.83	0.83

Figura 6.3.1. Métricas de evaluación para datos con Random Forest

Con el modelo de *Random Forest* se logró tasa de acierto mucho más elevadas. Es un modelo no lineal, en este caso se ajustó mucho mejor que Regresión logística.

El valor de *F1-score* es un 73%, dado por un 78% en precisión y un 69% de *recall*. Se obtiene un 34% más en *recall* y un 19% en *precision* de mejora en *Random Forest* respecto al modelo anterior. Aunque no se logra obtener un valor del 100% en *f1-score*, el modelo consigue buenos resultados.

Los valores exactos de las métricas son los siguientes:

- F1-score = 0,730
- Recall = 0,686
- Precision = 0,780

Randon Forest forma una recta con cada condición que impone a una variable de los datos, puede crear una gran cantidad de rectas. En cambio, *Regresión Logística* solamente a partir de los datos genera una recta.

A continuación, se indican las variables más importantes a partir de la función *feature_importances_* de *sklearn*.

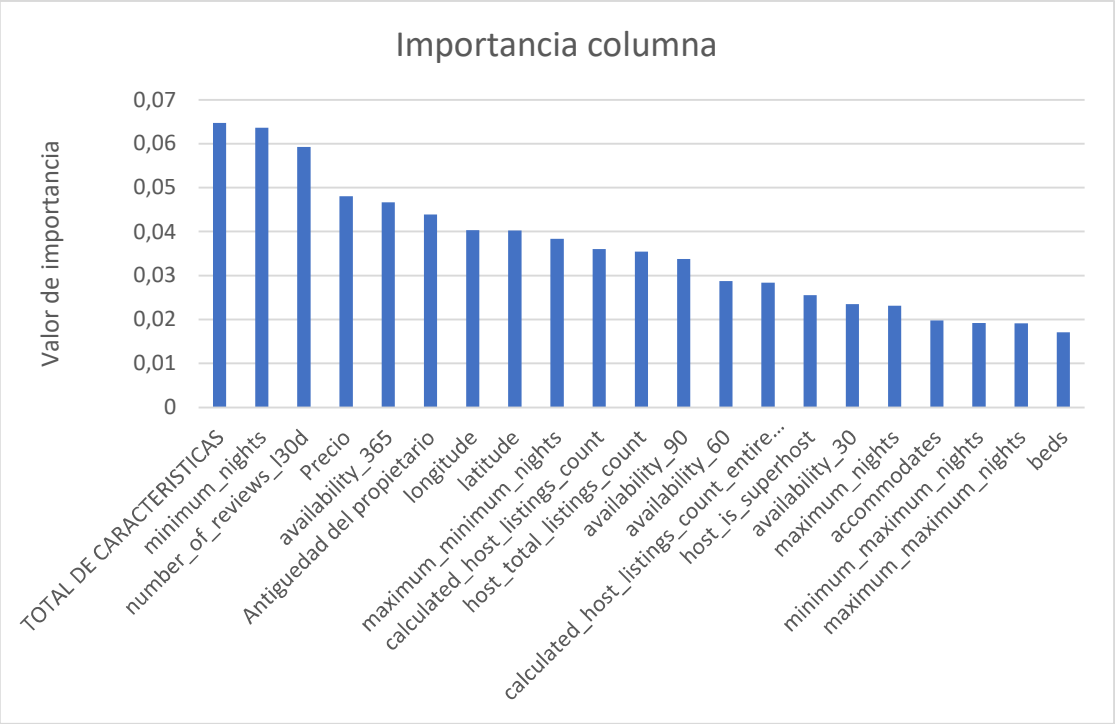


Gráfico 6.3.1. Gráfico donde se representa la importancia de variables para Random Forest

No hay ninguna columna que destaque mucho respecto a las demás. El modelo le asigna una importancia alrededor del 0,06 a *number_of_reviews_l30d, Total de características y Minimum_nights*, son las que mayor puntuación obtienen. La diferencia respecto a las siguientes columnas con mayor valor es de 0,01. La importancia está bastante equilibrada entre las columnas que mayor valor poseen.

6.4. Tercer experimento

6.4.1. Primera Fase: Parámetro Árbol de decisión

En esta parte del trabajo, se realizó un análisis de los parámetros mostrados del modelo *Random Forest*. Primero se modificó el parámetro de los árboles de decisión interiores de *Random Forest*, dejando los otros por defecto. Después, se seleccionó las 3 combinaciones que mejor valor de *F1-score* lograron.

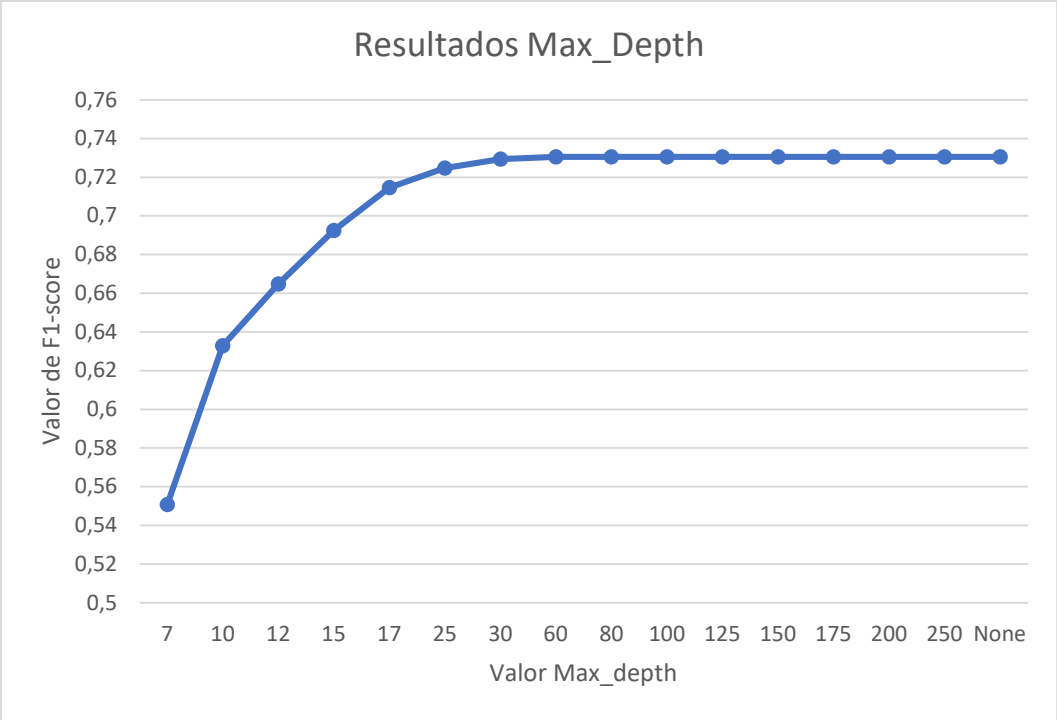


Gráfico 6.4.1.1. Gráfico donde se representa los valores de F1-score modificando Max_depth

En el eje Y se indica el valor de *F1-score* que se obtiene realizando la predicción del modelo *Random Forest* para los alojamientos de categoría alta. En el eje X, se representa los valores de *Max_depth* con los que se han creado los modelos.

El valor de *F1-score* aumenta a medida que se aumenta la profundidad del árbol. Llegado a una profundidad de 60 condiciones, el parámetro es estable. En este caso es mejor utilizar el primer parámetro para poder disminuir el tiempo de procesado. Se obtiene el mismo resultado que el obtenido en el experimento anterior, con el valor que mejor *F1-score* se logra.

Las mejores predicciones se obtuvieron con los siguientes valores:

- *Max_depth=60 → F1_score = 0,730 → Recall = 0,686 → Precision = 0,780*
- *Max_depth=30 → F1_score = 0,729 → Recall = 0,679 → Precision = 0,786*
- *Max_depth=25 → F1_score = 0,724 → Recall = 0,671 → Precision =0,786*

6.4.2. Segunda Fase: Parámetros Random Forest

De cada valor elegido, se realizó un estudio de la combinación de valores de los parámetros de *Random Forest*. Analizando los resultados de cada combinación y se escogieron los 5 que mejores resultados obtenían en *F1-score*.
Se representan en cada caso los valores de *Max_features* y *N_estimators* más importantes.

Gráfico 6.4.2.1. Gráfico donde se representa los valores de F1-score modificando parámetros Random Forest con Max_depth=60

En el eje Y se indica los valores que ha obtenido los diferentes modelos de *Random Forest* para predecir alojamientos como categoría alto. En el eje X se representa los valores de *N_estimators* utilizados para generar los modelos. Por otro lado, cada recta indica el valor de *Max_features* usados para construir los modelos.

Se logran los mejores resultados con valores de *Max_Features* altos, cuantas más variables se incluyan en los árboles mejores predicciones se hacen. Respecto a *N_estimators*, aumentando el número de árboles, los modelos logran obtener un mejor valor de F1-score. No obstante, llega un punto donde los modelos llegan a un valor máximo y no lo superan igualmente que se utilice un número superior de árboles. Cuando se construyen modelos con 25 árboles de decisión se produce una disminución del valor de *F1-score*. En cambio, con 20 árboles se produce un pico en el valor *f1-score* en todos los modelos.

Los mejores resultados se consiguieron con las siguientes combinaciones:

- *Max_features=115 – N_estimators = 150→ F1-score= 0.749 → Recall = 0,725 → Precision = 0,775*
- *Max_features=125 – N_estimators = 100→ F1-score= 0.747→ Recall = 0,719 → Precision = 0,777*
- *Max_features=115 – N_estimators = 175→ F1-score= 0.746 → Recall = 0,719 → Precision = 0,776*
- *Max_features=80 – N_estimators = 50→ F1-score= 0.746 → Recall = 0,721 → Precision = 0,774*
- *Max_features=80 – N_estimators = 75→ F1-score= 0.744→ Recall = 0,710 → Precision = 0,781*

La métrica que más varia con las combinaciones es *Recall*. Los modelos tienen fallos al predecir alojamientos como bajos que en realidad son altos.

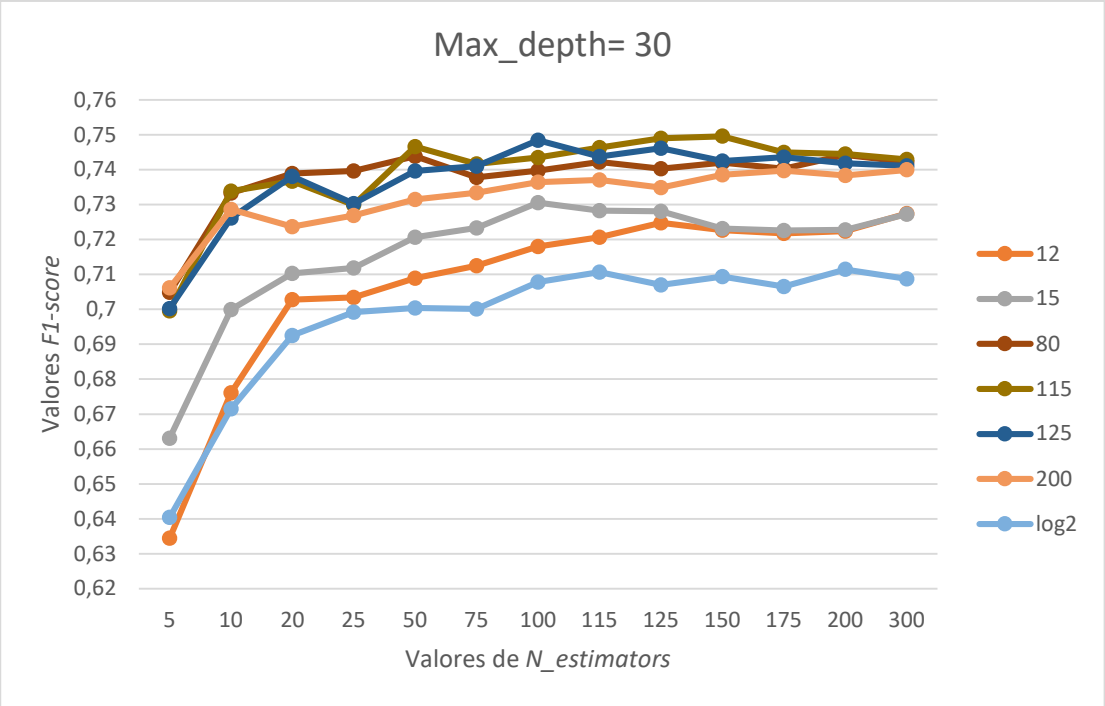


Gráfico 6.4.2.2. Gráfico donde se representa los valores de F1-score modificando parámetros Random Forest con Max_depth = 30

Los modelos con valor 30 en *Max_depth* tienen un comportamiento muy parecido a los anteriores. Aumenta el valor de *F1-score* a medida que aumenta tanto el parámetro *Max_features* como *N_estimators*. En este caso se obtienen dos valores de *F1-score* máximos muy parecidos. Se muestran menos rectas de *Max_features* debido a que en bastantes casos los valores son muy iguales y muchas quedan por debajo.

Los modelos con las mejores valores fueron a partir de las combinaciones siguientes:

- *Max_features=115 – N_estimators = 150→ F1-score= 0.749 → Recall =0,723 → Precision =0,777*
- *Max_features=115 – N_estimators = 125→ F1-score= 0.748 → Recall =0,723 → Precision =0,776*
- *Max_features=125 – N_estimators = 100→ F1-score= 0.748 → Recall =0,716 → Precision =0,779*
- *Max_features=115 – N_estimators = 115→ F1-score= 0.746→ Recall =0,717 → Precision =0,777*
- *Max_features=115 – N_estimators = 50→ F1-score= 0.744 → Recall = 0,726→ Precision =0,767*

Las combinaciones que logran mejor puntuación de *F1-score* son las que utilizan siempre 115 variables, exceptuando una que usa 125 variables.

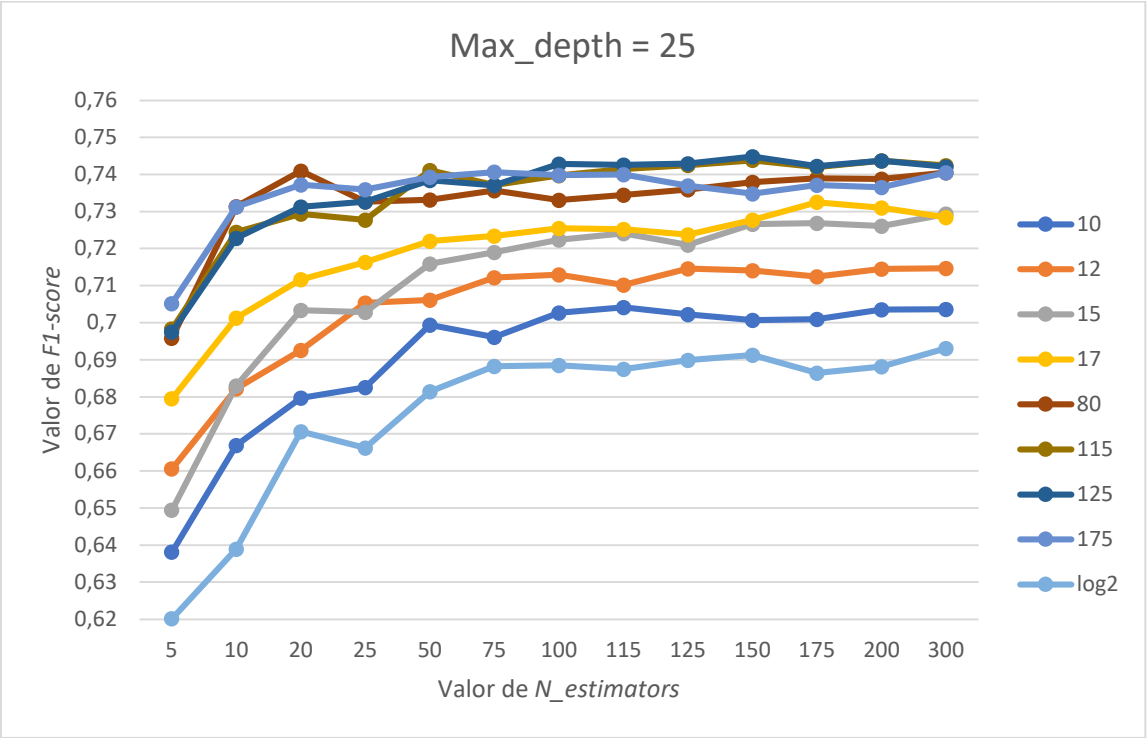


Gráfico 6.4.2.3. Gráfico donde se representa los valores de F1-score modificando parámetros Random Forest con Max_depth = 25

En este caso, existe una mayor diferencia entre las rectas que logran los valores inferiores de *F1-score*. Cuanto menor son las variables usadas con una profundidad pequeña de los árboles, se obtienen valores más pequeños de *F1-score*. No obstante, continua con la misma tendencia que los anteriores pruebas realizadas. Seguidamente, se exponen las combinaciones con los mejores valores de *F1-score*:

- *Max_features= 125 – N_estimators = 150→ F1-score= 0.744 → Recall =0,716 → Precision =0,775*
- *Max_features= 125 – N_estimators = 200→ F1-score= 0.743 → Recall =0,716 → Precision =0,772*

- *Max_features= 150 – N_estimators = 300 → F1-score= 0.743 → Recall =0,711 → Precision =0,778*
- *Max_features= 125 – N_estimators = 125 → F1-score= 0.742 → Recall =0,711 → Precision =0,776*
- *Max_features= 125 – N_estimators = 100 → F1-score= 0.742 → Recall =0,712 → Precision =0,775*

Usando el valor de 25 en *Max_depth*, se utilizan siempre un número elevado de árboles para lograr buenos valores de *F1-score*.

6.5. Cuarto experimento

Se planteó la idea de mejorar el modelo de *Random Forest* buscando los valores de *F1-score* como el experimento anterior, pero de forma inversa. En las gráficas que se exponen no se muestran todas las rectas de *Max_features*.

6.5.1. Primera Fase: Parámetros Random Forest

En esta fase se buscaron los mejores valores para los parámetros de *Random Forest*, los que creaban un valor de *F1-score* más alto. Se dejaron en todo momento los otros parámetros por defecto. Una vez construidos los modelos se seleccionaron las 5 mejores combinaciones.

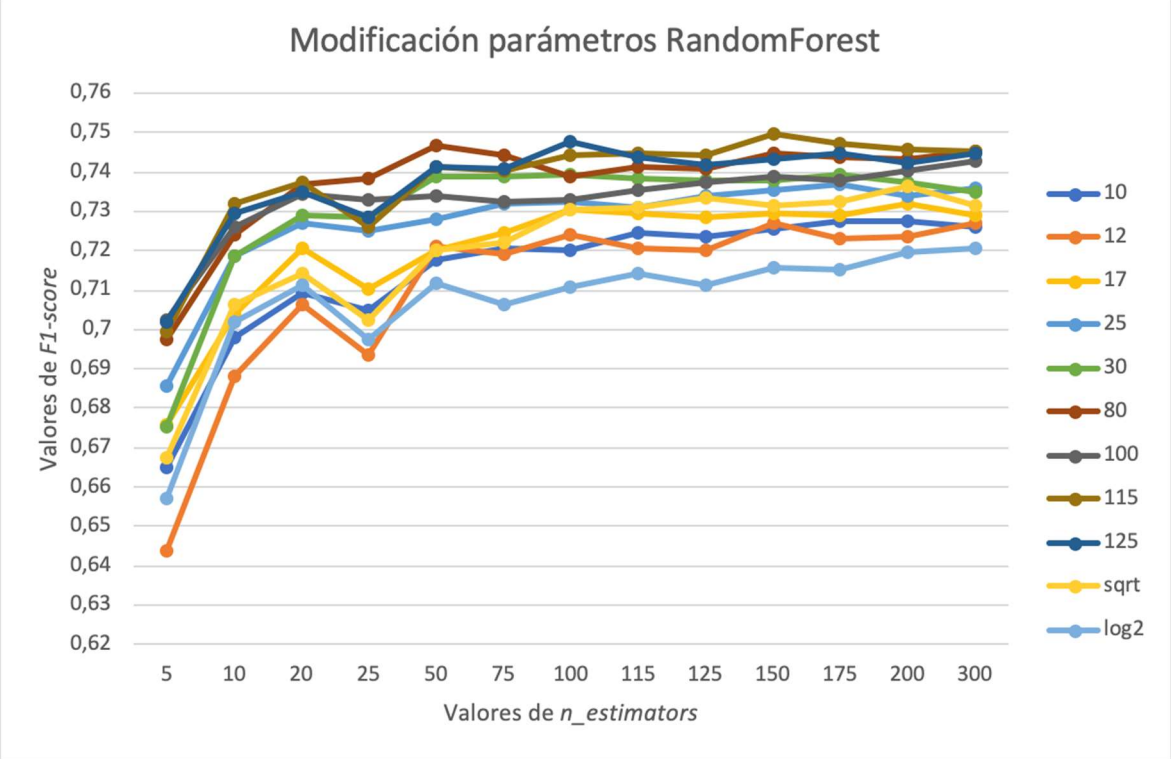


Gráfico 6.5.1.1. Gráfico donde se representa los valores de F1-score modificando parámetros Random Forest max_features y n_estimators

Con un número pequeño de árboles de decisión, el modelo realiza varias estimaciones erróneas. El valor de *f1-score* oscila entre el 0,64 y 0,7, para estos casos. A medida que el *n_estimators* aumenta, el modelo progresa y logra valores de f1-score mejores. Igual que antes, se produce una disminución de *f1-score* cuando se usan 25 árboles de decisión. Por otro lado, con 12 y 10 columnas a usar en los árboles, se obtienen resultados muy similares en los dos casos.

A continuación se muestran las combinaciones con mejor tasa de acierto:

- *Max_features= 115 – N_estimators = 150→ F1-score= 0.749 → Recall =0,716 → Precision =0,775*
- *Max_features= 125 – N_estimators = 100→ F1-score= 0.747 → Recall =0,716 → Precision =0,775*
- *Max_features= 115 – N_estimators = 175→ F1-score= 0.746 → Recall =0,716 → Precision =0,775*
- *Max_features= 50 – N_estimators = 80→ F1-score= 0.746 → Recall =0,716 →*

Precision =0,775

- *Max_features= 115 – N_estimators = 200 → F1-score= 0.745 → Recall =0,716 → Precision =0,775*

6.5.2. Segunda Fase: Parámetro Árbol de decisión

Obtenidas las mejores combinaciones se procedió a realizar el análisis con los diferentes valores del parámetro del Árbol de decisión.

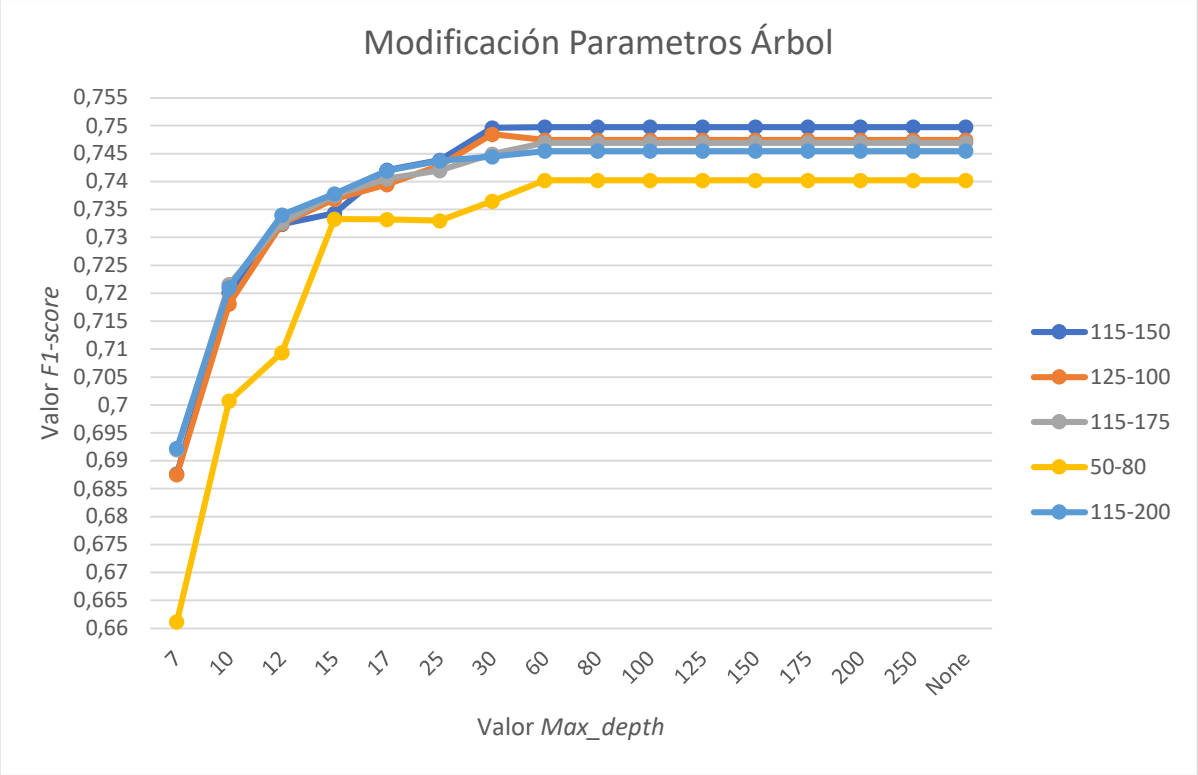


Gráfico 6.5.2.1. Gráfico donde se representa los valores de F1-score modificando el parámetro de los árboles de decisión max_depth

En el eje Y representa los distintos valores de *F1-score* para cada modelo de *Random Forest* creado. En el X se indica los diferentes valores del parámetro *Max_depth* que se utilizan para generar los modelos. Cada recta es la combinación seleccionada de los resultados logrados de la primera fase de este experimento. El primero número que está en la leyenda es el valor de *Max_features* y el segundo el de *N_estimators*.

La recta con valores bajos de *Max_features y N_estimators* es con las que se obtiene un *f1-score* inferior. En todos los casos, cuando se llega a 60 de *Max_depth*, *F1-score* se mantiene constante sin variar. Se produce en cada recta una pequeña disminución cuando *Max_depth* es 25. Por otro lado, en las rectas 4 rectas que lograran valores muy parecidos, se produce un aumento cuando se utiliza una profundidad de 30 en los árboles de decisión.

Las combinaciones con mejor f1-score en cada caso son los siguientes:

- *Max_features = 115 - N_estimators = 150 - Max_depth = 60 → F1-score = 0,749 → Recall = 0,725 → Precision = 0,775*
- *Max_features = 125 - N_estimators = 100 - Max_depth = 30 → F1-score = 0,748 → Recall = 0,716 → Precision = 0,783*
- *Max_features = 115 - N_estimators = 175 - Max_depth = 60 → F1-score = 0,745 → Recall = 0,719 → Precision = 0,776*
- *Max_features = 115 - N_estimators = 200 - Max_depth = 60 → F1-score = 0,745 → Recall = 0,722 → Precision = 0,770*
- *Max_features = 50 - N_estimators = 80 - Max_depth = 60 → F1-score = 0,740 → Recall = 0,707 → Precision = 0,775*

Exceptuando un combinación, en todos los demás se logran los mayores valores de F1-score con la profundidad de 60 en los árboles de decisión.

6.6. Comparativa de Resultados

Se realiza en este apartado una comparación de los cuatro experimentos realizados anteriormente.

A medida que se ha realizado los experimentos se ha podido observar como era el grado de influencia de los parámetros en los modelos y los dos tipos de modelos que se han usado. En ocasiones modificar los parámetros provocaba una mejora, pero en otras pasaba lo contrario, el modelo empeoraba sus predicciones.

Respecto a los dos primeros experimentos, si se hace énfasis en los resultados, el modelo *Random Forest* dejando sus parámetros por defecto logra un mayor porcentaje de aciertos. *Random Forest* es un modelo no lineal, puede hacer un gran número de particiones en los datos debido a las condiciones que genera. En cambio, *Regresión Logística* es un modelo mucho más limitado, solo realiza una división en los datos y a partir de esa divisoria ya predice la ocupación del alojamiento. Con una base de datos de las características que se emplea en este trabajo, *Regresión Logística* no proporciona el rendimiento necesario.

Random Forest es un modelo que construye una cantidad elevada de árboles para predecir. Posteriormente, hace una búsqueda interna de cuál es la mejor decisión una vez todos los árboles ha dado su categoría para el alojamiento. Por lo contrario, Regresión Logística no hace ninguna búsqueda, el resultado que logra es el que presenta como predicción. Es un modelo más sencillo, por eso se utilizaron sus métricas como resultado base a superar.

Referente a la primera fase del tercer experimento, el mejor valor de *f1-score* que se logra es el mismo que se consigue en el segundo experimento, además de mismos valores de *recall* y *precision*. Cabe destacar que el valor se logra con una profundidad inferior en los árboles de decisión, en el segundo experimento no se limitaba el máximo de profundidad y los árboles podían desarrollarse tanto como se pudiera. Se consigue el mismo resultado, pero reduciendo el tiempo de computación, es la combinación adecuada cuando no se dispone de mucha potencia computacional.

En la segunda fase del tercer experimento, en los resultados obtenidos de las combinaciones elegidas, fijado el valor del parámetro *Max_depth*, se superó el valor de *f1-score* del primer modelo de *Random Forest*. De igual manera, el valor de *Recall* también fue superior al anterior, se aumentó la fiabilidad de los modelos al categorizar un alojamiento como alta que

en verdad es baja. No obstante, se disminuyó el valor de *Precision.* A partir de esta fase es cuando se aumentó la tasa de acierto, por lo tanto, los parámetros de *Random Forest* tienen una mayor influencia en el modelo que el de los árboles de decisión con estos datos.

A continuación se realiza una comparación de los resultados en esta fase, haciendo énfasis en las combinaciones con mejor *f1-score.* Las combinaciones seleccionadas fijando el valor *Max_depth* a 60 y 30 proporcionaban el mismo valor de *f1-score*. Usando un valor de *Max_features* 115, *N_estimators* 150 y *Max_depth* 60 el modelo generaba un valor de *recall* superior que utilizando un valor *Max_features* 115 , *N_estimators* 150 y Max_depth 30. En cambio, se reducía *Precision*. Se puede utilizar uno u otro, dependiendo si se desea predecir de mejor forma alojamientos categorizados como altos y que su ocupación es baja o predecir alojamientos como bajos que en realidad son altos. La principal diferencia es la profundidad de los árboles si se quiere que sean más complejos o no. Por otro lado, asignando un valor de *Max_features* 125, *N_estimators* 150 y *Max_depth* 25 no se proporciona el mismo valor como en las otras combinaciones. *Recall* es el valor que más se reduce con este modelo.

Como conclusión, los modelos que mejor *f1-score* consiguen son los que como mínimo los árboles poseen una profundidad de 30. Además, si se generan 150 árboles, un número donde no se crea un sobre ajuste del modelo y puede generalizar de forma óptima, se obtiene una pequeña mejora en los resultados.

La diferencia de valores no era muy significativa, se diferenciaban por solo 0.001 décimas. No obstante, se buscó siempre la mejor situación.

Respecto a la primera fase del cuarto experimento, modificando los parámetros de *Random Forest*, a diferencia de la primera fase del tercer experimento, los modelos escogidos proporcionaban mejor valor de *f1-score*. Se mejora el valor *Recall,* pero disminuye un poco *Precision*.

Los modelos generalizan de forma correcta si no se genera un número excesivo de árboles y se supera el valor por defecto que está asignado. Asimismo, es necesario que los árboles de decisión se entrenen con un valor cercano a 125 columnas para que puedan predecir de forma correcta, un número muy pequeño no realiza buenas predicciones.

En la segunda fase del cuatro experimento, solo en una de las combinaciones escogidas los valores de *Recall y Precision* eran superiores a los que se mostraba en el segundo experimento.

En los dos experimentos se alcanza la misma combinación de valores con los que se obtiene el mejor *f1-score*. Durante los experimentos se han creado modelos con puntuaciones elevadas en las métricas de evaluación. La elección de cuál es el mejor modelo depende del criterio que se desee seguir, disminuir predecir un alojamiento como ocupación baja que en realidad es alta o minimizar fallos en predecir un alojamiento como ocupación alta que en realidad es baja.

7. Impacto ambiental

El presente trabajo tiene un mínimo impacto medioambiental, todo se realiza de forma digital.

El ordenador es la herramienta que ha hecho servir durante todo el proceso, el impacto que puede tener el ordenador es la electricidad que consume para que funcione y la cantidad de material que se ha utilizado para producirlo. Respecto a su consumo, la tecnología avanza y cada vez los ordenadores son más potentes y consumen menos energía.

Se puede considerar el consumo de fuentes de energía del entorno como la luz y el consumo del router que proporciona conexión a la red, pero es muy pequeño.

No se ha generado ningún gasto de papel porque en todo momento se ha usado la aplicación *notas* para apuntar todo lo necesario y relevante.

8. Planificación

En este apartado se muestra la planificación a partir de un diagrama de Gantt que se ha seguido para realizar el proyecto. En este Diagrama se muestran las etapas desde el inicio hasta el final del proyecto para así mostrar claramente y de forma precisa cuando se ha llevado acabo cada parte.

Fases	Actividad	Febrero			Marzo				Abril				Mayo				Junio				Julio				Agosto				Septiembre			
Inicio del trabajo	Planteamiento del proyecto																															
	Fijar los objetivos y familarizarse con metodología CRISP-DM																															
	Aumento de programación librería Pandas y sklearn																															
Manipulación de los datos	Comprensión de las variables																															
	Preparación de los datos																															
Modelaje y resultados	Estudiar los modelos de predicción																															
	Estudio tipos de validación																															
	Aplicación de algoritmos																															
	Analisis de resultados																															
Memoria	Redacción de memoria																															
	Presupuesto																															
	Conclusiones																															

Tabla 8.1. Tabla planificación del trabajo

9. Presupuesto

El coste del trabajo se debe de descomponer en coste de la mano de obra necesaria para realizar todo el trabajo y los costes de herramientas que se han utilizado.
Seguidamente, se exponen los costes para un mayor entendimiento.

9.1. Coste de mano de obra

Se muestra una tabla donde se desglosa cada acción realizada en el proyecto. Para cada acción se indica el tiempo de dedicación, el precio por hora de cada una y el coste total de cada una.
Los precios hora de cada acción varía ya que la destreza y el esfuerzo realizado es diferente para cada acción. No es lo mismo realizar la parte de redactar la memoria, describiendo todos los procesos, en comparación con hacer toda la parte de programación.

Acción	*Dedicación*	*Precio por hora*	*Coste total*
Investigación	110h	10€	1100€
Compresión	60h	20€	1200€
Preparación	160h	40€	6400€
Modelaje	100h	40€	4000€
Análisis	70h	30€	2100€
Redacción	80h	20€	1600€
TOTAL	**580h**		**16400€**

Tabla 9.1.1. Tabla costes mano de obra del trabajo

La cantidad de tiempo que se ha dedicado es diferente debido a que había acciones donde era necesario estar mucho más tiempo. Cuando se realizó la parte de preparación, se tuvo que estar un periodo largo programando y además observando si los resultados eran los deseados. Era primordial esperar a que el programa mostrara los datos y realizar una

verificación exhaustiva.

La redacción era importante una dedicación considerable, ya que es preciso explicar con detalle las acciones llevadas a cabo y justificarlas. Si no se realiza de esta forma, puede dar lugar a confusión y no entender el trabajo que se ha ejecutado.

9.2. Coste de herramientas

En este apartado se especifica las herramientas que se han utilizado durante la ejecución del proyecto.

Se ha usado un ordenador valorado en 1300€, según la agencia tributaria un ordenador se amortiza un 25% de forma anual para un uso de 1200 horas al año. Para este caso se ha usado para un total de 580 horas, en proporción es un 12,08%. Por lo tanto, el importe a sumar al presupuesto anterior respecto a la utilización del ordenador es de 157,04€.

Además del ordenador se han empleado el programa Word para la redacción de la memoria y un consumo de internet y energía eléctrica.

El programa Word cuesta 61€ la subscripción anual, de este modo como se ha usado durante 2 meses aproximadamente, conlleva a un coste de 10,16€.

Hoy en día, el precio de la energía está muy elevado, debido a que ha habido un aumento de la demanda, se ha encarecido el gas y se ha aumentado el precio de las emisiones de CO2. Pagando una media de 40€ mensuales de uso de energía, el coste para este proyecto por un total de 580 horas es de 32,2€.

Respecto al uso del router, se paga mensualmente 15€ por la fibra óptica, conlleva un cargo de 12,08€.

El coste total de las herramientas suma un total de 211,48 € y el coste total unidos los dos costes es de 16.611,48€.

10. Conclusiones

En el presente trabajo se ha conseguido el principal objetivo, realizar un análisis de los alojamientos turísticos en Barcelona de la página *Airbnb* proporcionados por *Inside Airbnb*. Además, se ha podido predecir la ocupación de los diferentes alojamientos en lenguaje Python y aplicar de forma adecuada la metodología CRISP-DM. No obstante, por falta de tiempo, no se ha incluido la fase de tratamiento de valores atípicos de la metodología EDA.

Mediante la utilización de la metodología CRISP-DM, se ha logrado estudiar las diferentes columnas de los datos, identificar los parámetros ausentes de la base de datos, tanto crear como eliminar columnas y generar diversos modelos con los que realizar predicciones. Al haber un número elevado de columnas en los datos era primordial saber qué significaba cada una de ellas y si se podía utilizar para realizar predicciones. En varias columnas, se tomó la decisión de no tratarlas, ya que era necesario de un tratamiento específico y no se disponía de todo el tiempo deseado para realizado de forma eficiente y correcta.

La elección de dos algoritmos de predicción diferentes ha sido beneficioso para el trabajo porque se ha observado diversas formas con las que realizar predicciones. En *Random Forest* se ha estudiado identificar, analizando los resultados obtenidos, como actuaban los parámetros y la influencia que tenían en el modelo. Los parámetros generalmente mejoraban las predicciones, pero había casos en los que el modelo no tenía suficiente información para predecir la ocupación y se disminuya la tasa de acierto. Cabe destacar que se podría haber utilizado una cuantía superior de valores para probar en los parámetros y construir nuevos algoritmos. Por otro lado, Regresión Logística se ha entendido su funcionamiento y las limitaciones que posee. Además, saber interpretar las métricas de evaluación y la importancia que conlleva poder utilizarlas para observar como actúan los modelos.

Como reflexión personal, haber realizado un trabajo de ciencia de datos beneficia mi futuro laboral. Una de mis metas es ser Project Manager, dirigir todo tipo de proyectos es una de las principales motivaciones por la que decidí realizar el máster y grado en ingeniería. La ciencia de datos es una rama muy demandada, las empresas están buscando a ingenieros que sepan analizar datos. Mi perfil se puede ajustar a las vacantes donde se pida saber ciencia de datos y aplicarlo, por lo tanto, aumentar las posibilidades de encontrar una buena posición laboral.

11. Trabajos Futuros

Dada la complejidad de estos datos se pueden estudiar desde diferentes ángulos. A continuación, se expondrán una serie de acciones con las que se puede crear un nuevo trabajo o ampliar el trabajo que se ha realizado.

Durante el proceso de elaboración, no se pudo hacer énfasis en los datos atípicos que mostraban las diferentes variables una vez tratadas, debido a la limitación de tiempo. Se puede utilizar este trabajo como referencia y eliminar o transformar filas o columnas con valores atípicos, con el objetivo de ver si se mejoran los modelos. En el modelo de regresión logística, uno de los modelos con peor tasa de acierto, realizar esta acción y observar si las métricas de evaluación aumentan o disminuyen.

Por otro lado, *Inside Airbnb* proporciona una base de datos donde varias variables están compuestas por una gran cantidad de texto. El texto lo escribe el propietario del alojamiento cuando está formando el anuncio que quiere publicar en *Airbnb*. Un trabajo futuro es ejecutar un tratamiento específico a estas columnas para extraer información de cada alojamiento. Dando lugar a nuevas columnas que pueden utilizar los modelos para realizar predicciones.

En este trabajo se destacó realizar buenas predicciones para los alojamientos con un ratio categorizado como alto utilizando dos algoritmos de predicción. Otro trabajo es utilizar los mismos datos pero usando dos modelos diferentes para hacer predicciones y modificar sus parámetros. Modelos como SVM o CNN, pueden ser útiles para aplicar con estos datos. Obtener diferentes métricas de evaluación de varios modelos sirve para saber cuál de ellos identifica de mejor forma los patrones. Además, permite que los propietarios sepan de una forma más fiable el ratio de ocupación de su alojamiento.

Asimismo, *Inside Airbnb* capta datos de varias ciudades, no solo de Barcelona. Se puede formar un trabajo analizando y prediciendo la ocupación enfocándose en otra ciudad y, posteriormente, comparar los resultados con los obtenidos en este trabajo. En las otras bases de datos puede que haya variables diferentes de las cuales se puede extraer más información.

12. Agradecimientos

Quiero agradecer todo el trabajo realizado ayudándome con el trabajo a mi tutor Luis Talavera y a toda mi familia y pareja por apoyarme en todo momento.

Bibliografía

[1] DAVID L. OLSON, DURSUN DELEN. Advanced Data Mining Techniques, 2008, 169 p

ISBN: 978-3-540-76916-3

[2] *Agencia Tributaria*. (s.f.). Obtenido de https://www.agenciatributaria.es/

[3] *Airbnb*. (2008). Obtenido de https://www.airbnb.cat/

[4] *Aprende Machine learning*. (s.f.). Obtenido de https://www.aprendemachinelearning.com/

[5] *Ciencia de datos*. (s.f.). Obtenido de https://www.cienciadedatos.net/

[6] *DataScientest*. (s.f.). Obtenido de https://datascientest.com/

[7] *IArtificial*. (s.f.). Obtenido de https://www.iartificial.net/

[8] IBM. (s.f.). Obtenido de https://www.ibm.com/

[9] *Inside Airbnb*. (2015). Obtenido de http://insideairbnb.com/

[10] *Jupyter*. (s.f.). Obtenido de https://jupyter.org/install

[11] *Machine learning* . (s.f.). Obtenido de https://machinelearningmastery.com/

[12] *Medium*. (s.f.). Obtenido de https://medium.com

[13] *Pandas*. (s.f.). Obtenido de https://pandas.pydata.org/docs/user_guide/index.html

[14] science, T. d. (s.f.). Obtenido de https://towardsdatascience.com/

[15] *Sklearn*. (s.f.). Obtenido de https://scikit-learn.org/stable/