

Memoria Proyecto Grupo Globo-Data

Ponle precio a tu propiedad en Airbnb

Keepcoding - Glovo Women in Tech

Sep-Feb 2023

Maria Aguilà, Alba Abaurrea, Maria Gimenez, Judit Gómez y Paula Gual

Índice

1. Introducción
2. Arquitectura y validación de los datos
3. Análisis Exploratorio
4. Visualización de las métricas
5. Pre-procesamiento y Modelado
6. Conclusiones

1. Introducción

Este estudio tiene como finalidad principal realizar un análisis detallado de los alojamientos en Airbnb en la ubicación de Madrid. Se busca comprender de manera más profunda si existe alguna relación entre el precio de los alojamientos y diversos factores como, por ejemplo, la ubicación en un vecindario determinado, las características de los alojamientos, entre otras variables. Además, se pretende identificar patrones y tendencias en el comportamiento de los precios, para poder comprender mejor los aspectos que los influyen.

Este dataset contiene información sobre los alojamientos de Airbnb en diversas ciudades del mundo. Hay un total de 89 columnas que describen diversas características de los alojamientos, los anfitriones y las reseñas. De todas ellas, filtraremos los datos únicamente a los de las propiedades que se encuentran en Madrid y seleccionaremos aquellos campos que sean de mayor utilidad e importancia para nuestro estudio de mercado.

En base al data set inicial, nuestro objetivo principal era resolver la pregunta de aquellos propietarios madrileños que quieren publicar sus inmuebles o estancias en Airbnb:

¿Qué precio debo ponerle a mi publicación?

Para ello, realizaremos una exploración inicial de los datos, seguida de la creación e implementación del modelo entidad-relación para elaborar el Data Warehouse. De esta manera, posteriormente, procederemos a analizar los datos y cómo se relacionan entre ellos para conocer los factores más influyentes en el precio y poder realizar un asesoramiento lo más exhaustivo posible a los propietarios que quieran realizar publicaciones en el futuro.

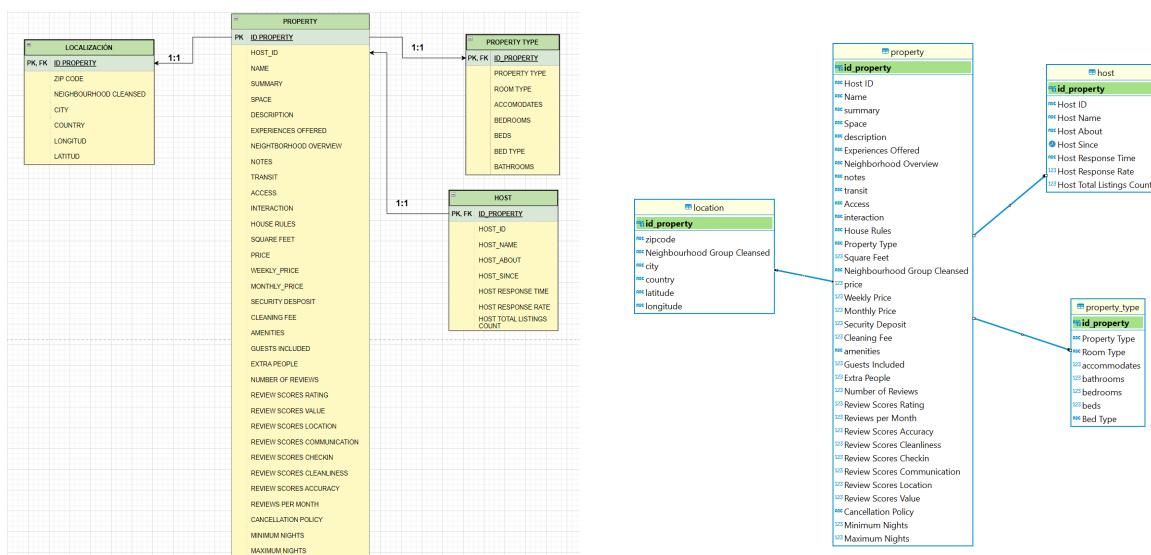
Finalmente, crearemos un modelo de predicción que permitirá fácilmente conocer el precio recomendado. Siempre teniendo en cuenta los factores estrechamente relacionados con el valor de mercado del inmueble, más allá de los precios en la ciudad.

2. Arquitectura y validación de los datos

Tras un primer contacto con los datos, detectamos dos inconvenientes a la hora de trabajarlos. El primero fue que, a causa del uso del emoji 😊 en algunas reviews, no era posible importar la información directamente desde el archivo original. Y, el segundo fue que los datos de origen no estaban sesgados y mostraban propiedades que se encuentran fuera de la ciudad de Madrid. Para resolverlos, hemos empezado por eliminar los emojis de las reviews, ya que no aportan valor al dataset, y utilizando pandas hemos filtrado los datos para dejar los de Madrid.

Durante la exploración inicial de los datos recabamos la información necesaria para proceder al siguiente paso y empezar a perfilar lo que será el Data Warehouse. Para comenzar, creamos el modelo entidad-relación que presenta como nexo de unión entre todas las tablas el dato único de cada propiedad, el ID de la propiedad. En un principio creímos que se podrían relacionar las tablas utilizando otros datos, como el Host ID, pero este no es único ya que algunos propietarios contaban con varias propiedades. Al no ser una relación 1:1 y repetirse se nos complicaba marcar otras claves primarias.

En la definición de atributos, decidimos desglosar la información en cuatro grupos relevantes para nuestro análisis: la **propiedad**, englobando toda la información directamente relacionada con ésta; el **host**, para poder analizar lo relativo a los propietarios y sus características; el **tipo de propiedad** y sus características; y la **localización**.



Esquema en Draw.io

Esquema en SQL

Una vez obtenido el esquema inicial nos pusimos manos a la obra para plasmarlo en la base de datos. Primero, creando las tablas que formarían el Data Warehouse tal y como estaba diseñado en el modelo. Y, posteriormente, resolviendo el escrollo de la importación de datos a cada una de las tablas desde el dataset original utilizando un código de inserción y seleccionando la información correspondiente de la tabla original, dando como resultado las siguientes agrupaciones de datos:

The screenshot shows four tables from a database interface:

- property_type**: Shows 17 rows of property types with columns: id_property, Room Type, accommodates, and bath.
- location**: Shows 17 rows of locations with columns: id_property, zipcode, Neighbourhood Group Cleansed, city, and cc.
- property**: Shows 18 rows of properties with columns: id_property, Host ID, Name, and summary.
- host**: Shows 18 rows of hosts with columns: Host ID, id_property, Host Name, and Host About.

3. Análisis Exploratorio

Este estudio tiene como finalidad principal realizar un análisis detallado de los alojamientos en Airbnb en la ubicación de Madrid. Se busca comprender de manera más profunda si existe alguna relación entre el precio de los alojamientos y diversos factores como, por ejemplo, la ubicación en un vecindario determinado, las características de los alojamientos, entre otros. Además, se pretende identificar patrones y tendencias en el comportamiento de los precios, para poder comprender mejor los aspectos que los influyen.

Este dataset contiene información sobre los alojamientos de Airbnb en diversas ciudades del mundo. Hay un total de 89 columnas que describen diversas características de los alojamientos, los anfitriones y las reseñas.

Algunas de las columnas más importantes incluyen:

- ID: identificador único de cada alojamiento
- Name: nombre del alojamiento
- Neighborhood: barrio en el que se encuentra el alojamiento
- Description: descripción detallada del alojamiento
- Host ID: identificador único del anfitrión
- Host Name: nombre del anfitrión
- Host Location: ubicación del anfitrión
- Property Type: tipo de propiedad (apartamento, casa, etc.)
- Room Type: tipo de habitación (privada, compartida, etc.)
- Accommodates: número de personas que puede alojar el alojamiento
- Price: precio por noche del alojamiento
- Latitude: coordenada latitud del alojamiento
- Longitude: coordenada longitud del alojamiento
- Review Scores Rating: puntaje promedio de las reseñas del alojamiento
- Review Scores Accuracy: puntaje promedio de la precisión de las reseñas
- Review Scores Cleanliness: puntaje promedio de la limpieza de las reseñas

Este es solo un ejemplo de algunas de las columnas importantes que se encuentran en el dataset, pero hay muchas más que podrían ser útiles para diferentes análisis.

La Metodología seguida para llevar a cabo este estudio, se siguieron los siguientes pasos:

1. Limpieza de datos:

La primera etapa de nuestro análisis exploratorio consistió en revisar la calidad de los datos y prepararlos para el análisis.

En primer lugar, se carga el conjunto de datos en un dataframe de Pandas:

```
import pandas as pd

# Cargar los datos en un dataframe de Pandas

df = pd.read_csv("airbnb.csv")
```

Luego, revisamos la calidad de los datos y detectamos los valores nulos o inconsistentes en las distintas columnas:

```
# Revisar la calidad de los datos

print(df.info())

# Detectar los valores nulos o inconsistentes en el precio y
en la ubicación

print(df.isnull().sum())
```

Identificamos los valores nulos o inconsistentes y decidimos sustituir las filas con valores nulos con la media o valores mínimos . Para los valores faltantes Con estos pasos, logramos preparar los datos para continuar con el análisis.

Además de limpiar los datos, es común que se necesite filtrar la información para enfocarse en una subsección específica del conjunto de datos. En este caso, se filtró la ciudad y el país para trabajar únicamente con los datos de los alojamientos Airbnb ubicados en Madrid, España.

Para llevar a cabo este proceso de filtrado, se utilizó la función de selección de filas en Pandas, que permite seleccionar solo las filas que cumplen con ciertas condiciones. En particular, se creó un nuevo dataframe llamado "df_madrid" a partir del dataframe original "df" utilizando la función de selección de filas. Se especificaron las condiciones de selección, que incluyen que el código de país sea "ES" (código de dos letras para España) y que la ciudad sea "Madrid".

Una vez que se obtuvo el subconjunto de datos de Madrid, se utilizó el método "reset_index()" para reiniciar los índices de las filas en el nuevo dataframe y, finalmente, se imprimió el resultado con la variable "df_madrid". Este proceso de filtrado permitió reducir el tamaño del conjunto de datos y enfocarse en los alojamientos Airbnb ubicados únicamente en Madrid, lo que

puede ser útil para el análisis exploratorio y otras tareas de modelado de datos.

Después de haber filtrado el conjunto de datos para trabajar únicamente con los alojamientos ubicados en Madrid, procedimos a realizar una revisión detallada de las distintas columnas que componen el dataset. En este sentido, identificamos aquellas que no proporcionaban información relevante para el análisis que deseábamos llevar a cabo.

Por lo tanto, se tomó la decisión de eliminar ciertas columnas que no aportan datos significativos para nuestro estudio. Estas columnas incluyeron el Scrape ID, el índice, la URL del listado y la imagen del anfitrión, entre otras. En total, se eliminaron 16 columnas del dataset.

Una vez eliminadas estas columnas, se obtuvo un nuevo dataframe denominado "df_madrid_clean" que contenía exclusivamente las columnas consideradas pertinentes para nuestro análisis. A continuación, se realizó una revisión adicional para detectar valores nulos en el dataset y garantizar que los datos estuvieran limpios y listos para el análisis.

En nuestro proceso de limpieza y procesamiento de datos, se tomó la decisión de abordar los valores nulos de manera adecuada para mejorar la calidad del conjunto de datos. Con el fin de lograr esto, se optó por reemplazar los valores nulos con espacios en blanco en el caso de las columnas de texto, y con ceros en el caso de las columnas numéricas correspondientes.

Es importante destacar que el uso de esta estrategia fue determinado tras un análisis detallado de los datos y una evaluación exhaustiva de las distintas opciones posibles. Se consideró que esta solución resultaba la más efectiva y apropiada para garantizar que la información que se proporcionará a los usuarios fuera lo más precisa y completa posible.

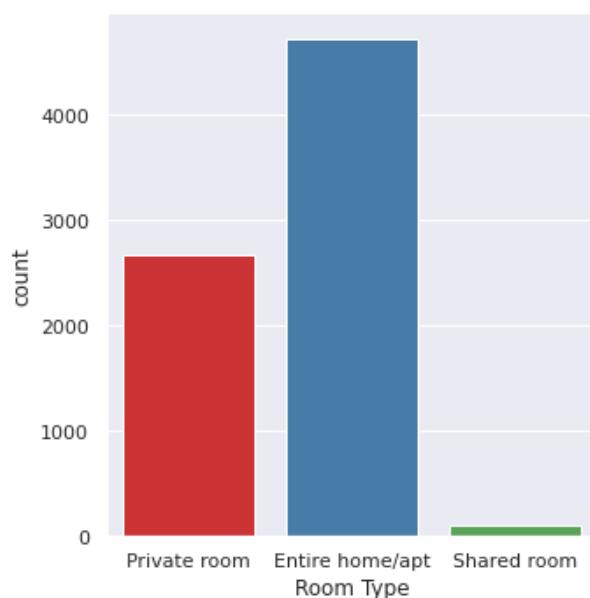
En resumen, nuestro equipo tomó la decisión de implementar un enfoque de reemplazo de valores nulos con espacios en blanco o ceros, según corresponda, en el proceso de limpieza y procesamiento de los datos, con el fin de mejorar la calidad del conjunto de datos y asegurar que los usuarios tengan acceso a la información más precisa y completa disponible.

En particular, en nuestro análisis, hemos llevado a cabo una importante limpieza de datos para garantizar la calidad de nuestra muestra. Hemos eliminado todas las entradas en las que se encontraron valores nulos o negativos en variables clave, incluyendo el número de habitaciones, baños, capacidad de alojamiento, precio y camas. De esta forma, hemos asegurado

que nuestra muestra esté compuesta únicamente por registros completos y fiables, lo que nos permitirá obtener resultados precisos y significativos.

2. Analysis Type Rooms:

Con el objetivo de obtener una comprensión más detallada de la composición de nuestro conjunto de datos, hemos realizado una visualización de los distintos tipos de Airbnb disponibles. Utilizando la función de conteo de valores de la biblioteca Pandas, hemos obtenido el número total de cada tipo de habitación presente en nuestros datos. Posteriormente, hemos creado un gráfico de barras utilizando la biblioteca Seaborn, para representar de manera visual esta información.



En particular, hemos encontrado que nuestro conjunto de datos incluye un total de `{len(df_madrid_clean["Room Type"].unique())}` tipos de habitaciones distintas. En orden de frecuencia, estas habitaciones son: `{df_madrid_clean["Room Type"].value_counts()}`. Este análisis preliminar nos proporciona información útil para el posterior análisis de nuestro conjunto de datos y nos permite tener una idea clara de la distribución de habitaciones disponibles en Airbnb en Madrid.

Para analizar los diferentes tipos de alojamiento en el conjunto de datos, contamos con 3 categorías distintas de alojamiento. En particular, se trata de alojamientos de viviendas completas, habitaciones privadas y habitaciones compartidas. Después de examinar los datos, encontramos que la mayoría de los alojamientos (aproximadamente el 64%) eran de tipo "Entire home/apt", seguidos de habitaciones privadas (alrededor del 36%) y habitaciones compartidas (menos del 1%).

3. Análisis Property Type

En este análisis, hemos identificado que el conjunto de datos de Airbnb Madrid incluye 18 tipos de propiedades diferentes. El tipo de propiedad más común en este conjunto de datos es el apartamento, con un total de 6218 entradas, seguido de la casa con 514 entradas. Hay un número significativamente menor de otros tipos de propiedades, incluyendo Bed & Breakfast, condominios, lofts, y otros. Estas observaciones se pueden

visualizar fácilmente en un diagrama de barras utilizando la biblioteca Seaborn.



4. Análisis de precios

En el conjunto de datos limpio de Airbnb Madrid, hemos calculado los precios medios para diferentes combinaciones de tipos de habitación y propiedad. Los 5 precios promedio más altos se observan para:

Tipo de propiedad: Hostal, Tipo de habitación: Habitación privada,
Precio promedio: 239,5

Tipo de propiedad: Hostal, Tipo de habitación: Habitación compartida,
Precio promedio: 115,0

Tipo de propiedad: Casa adosada, Tipo de habitación: Habitación privada, Precio promedio: 105,0

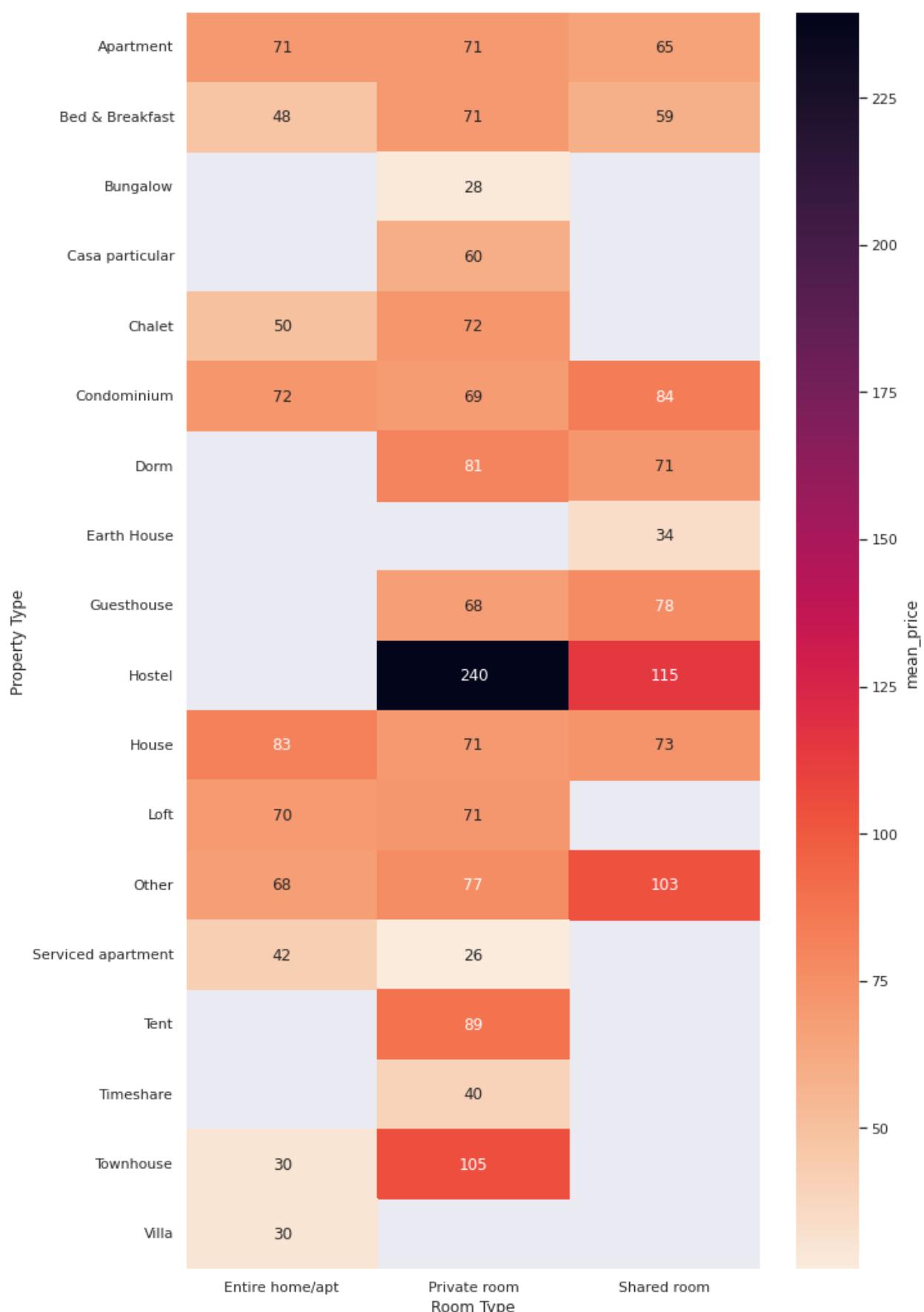
Tipo de propiedad: Otros, Tipo de habitación: Habitación compartida, Precio promedio: 103,0

Tipo de propiedad: Tienda de campaña, Tipo de habitación: Habitación privada, Precio promedio: 89,0

Podemos observar que el precio promedio más alto es para habitaciones privadas en un hostal, con un precio promedio de 239,5, seguido de habitaciones compartidas en un hostal con un precio promedio de 115. El más bajo de los 5 primeros es habitaciones privadas en una tienda de campaña con un precio promedio de 89,0.

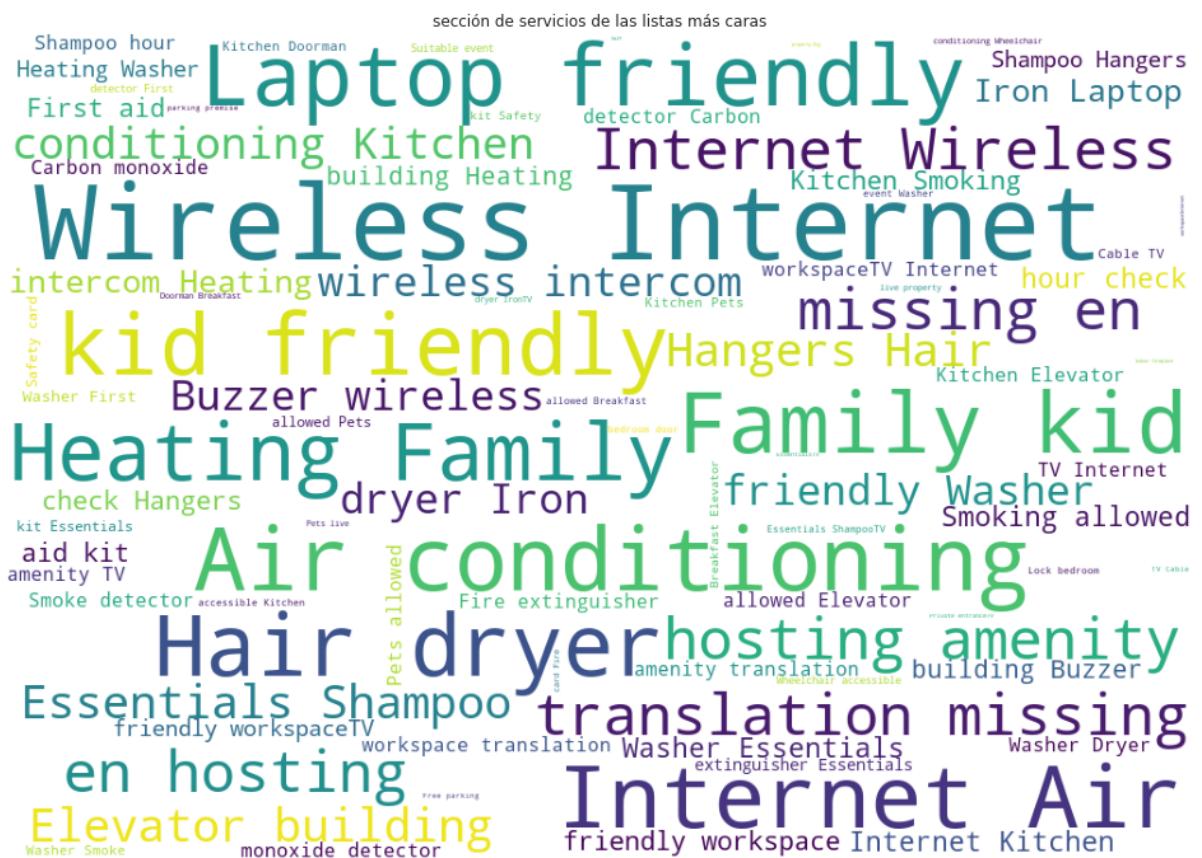
En el siguiente gráfico representa un mapa de calor (heatmap) que muestra la media de precios para diferentes combinaciones de tipos de habitación y propiedades.

La información se organiza en una tabla, donde las filas representan los tipos de propiedades y las columnas representan los tipos de habitación. Los colores en el mapa de calor indican la media de precios correspondiente a cada combinación. La leyenda de la barra de colores indica la media de precios y la intensidad del color corresponde a valores más altos o más bajos de precio.



En la siguiente visualización podemos ver un wordcloud o nube de palabras a partir de las palabras que aparecen en la sección de servicios de las 100 listas de alojamiento más caras del conjunto de datos de Airbnb Madrid.

Para hacer esto, el código lee los datos, filtra y limpia los datos para eliminar las palabras comunes que no son importantes, luego lematiza las palabras, lo que significa que se convierten a su forma base para que las palabras que se refieren a lo mismo se traten como iguales. Finalmente, todas las palabras se unen en una cadena y se utilizan para generar un wordcloud utilizando la biblioteca de WordCloud de Python. El resultado es una imagen que muestra las palabras más comunes utilizadas en la sección de servicios de las 100 listas de alojamiento más caras.

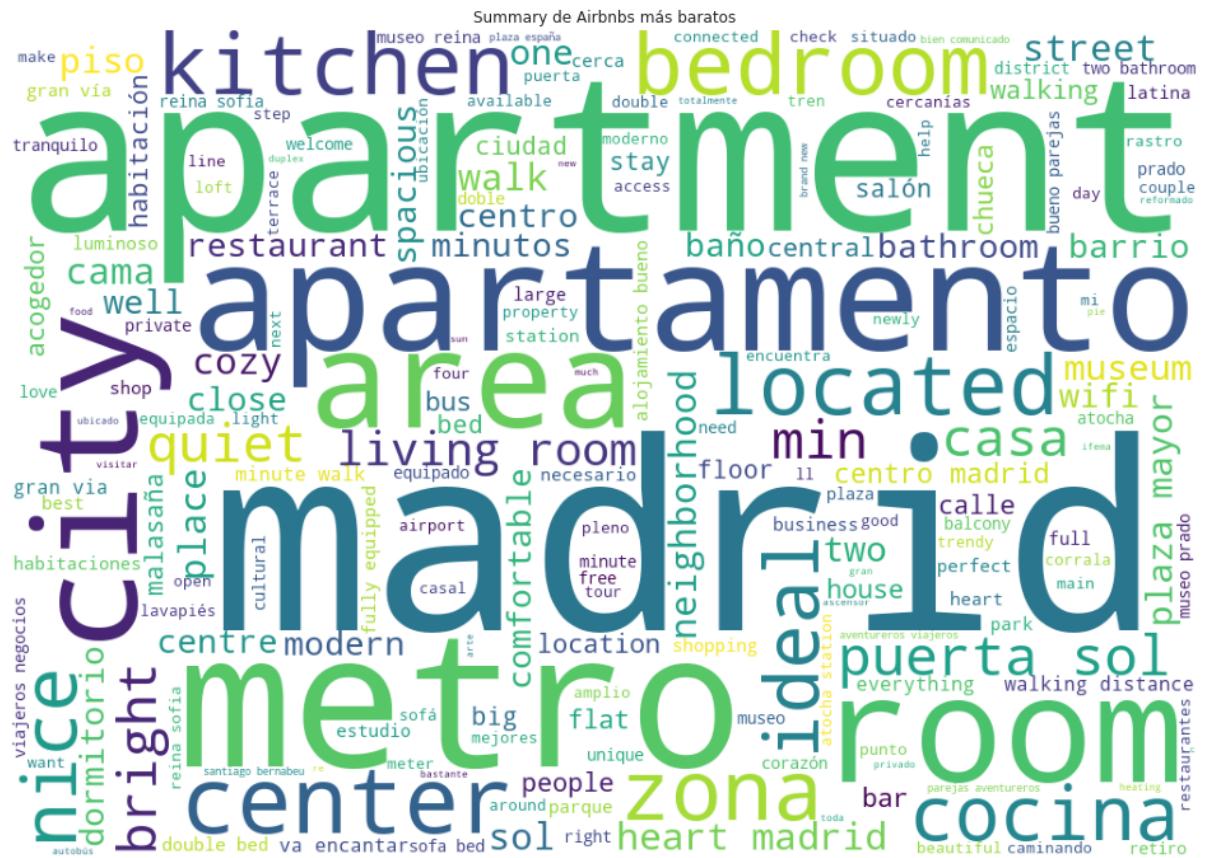


Para obtener las siguientes visualizaciones, el código tiene como objetivo limpiar el texto de la columna "Summary" del dataframe "df_madrid_clean", y generar una nube de palabras (wordcloud) de los resúmenes de los 100 listados más caros y más baratos en Madrid. Primero, se crea un nuevo dataframe "summaryDF" con los resúmenes y los precios correspondientes. Luego, se eliminan las filas que no tienen resúmenes y se ordena el dataframe por precio descendente. El nuevo dataframe "top100DF" contiene los 100 listados más caros y otro con los más baratos.

La función "clean_text" se encarga de limpiar el texto en la columna especificada del dataframe proporcionado. Elimina las palabras de parada (stopwords) en varios idiomas, signos de puntuación, caracteres innecesarios y lematiza las palabras. La función devuelve una cadena de texto limpia.

Finalmente, se llama a la función "wordcloud_generator" para generar una nube de palabras del texto limpio de los resúmenes de los 100 listados más caros y otro con los más baratos.

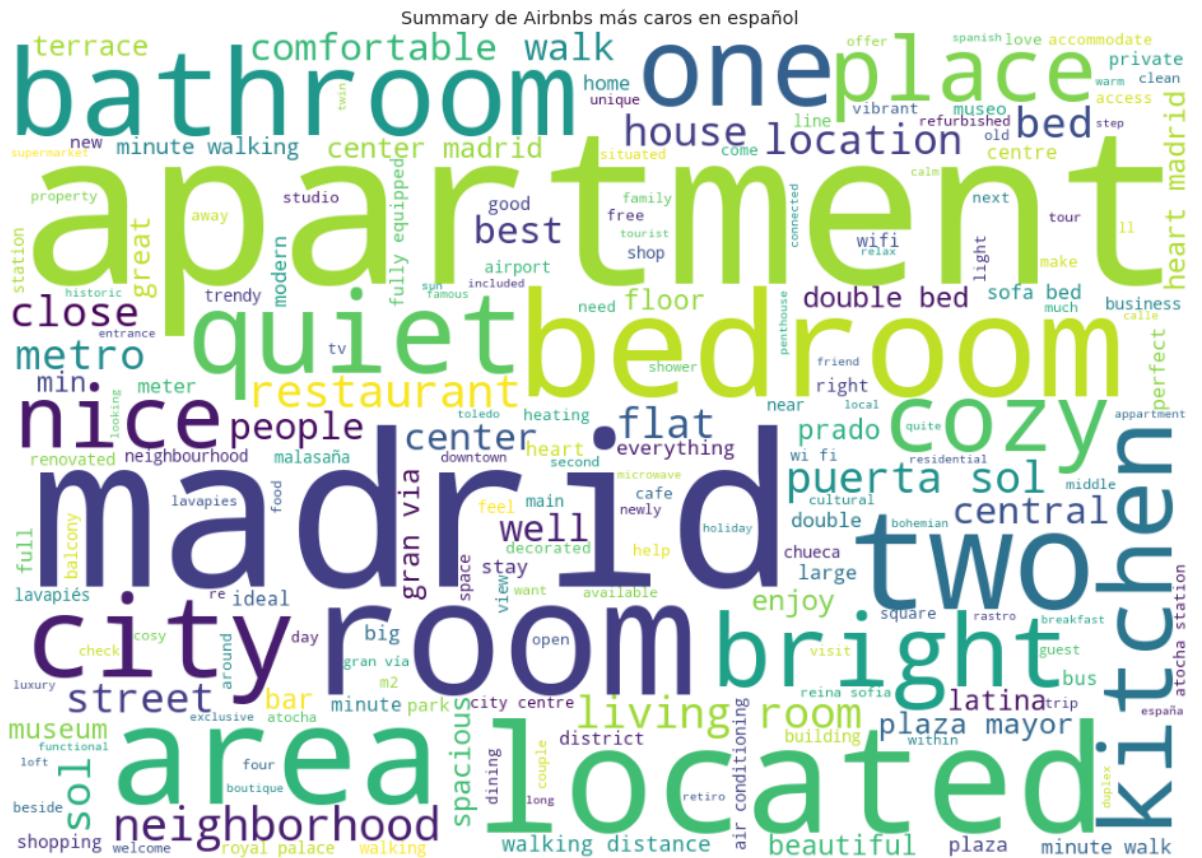
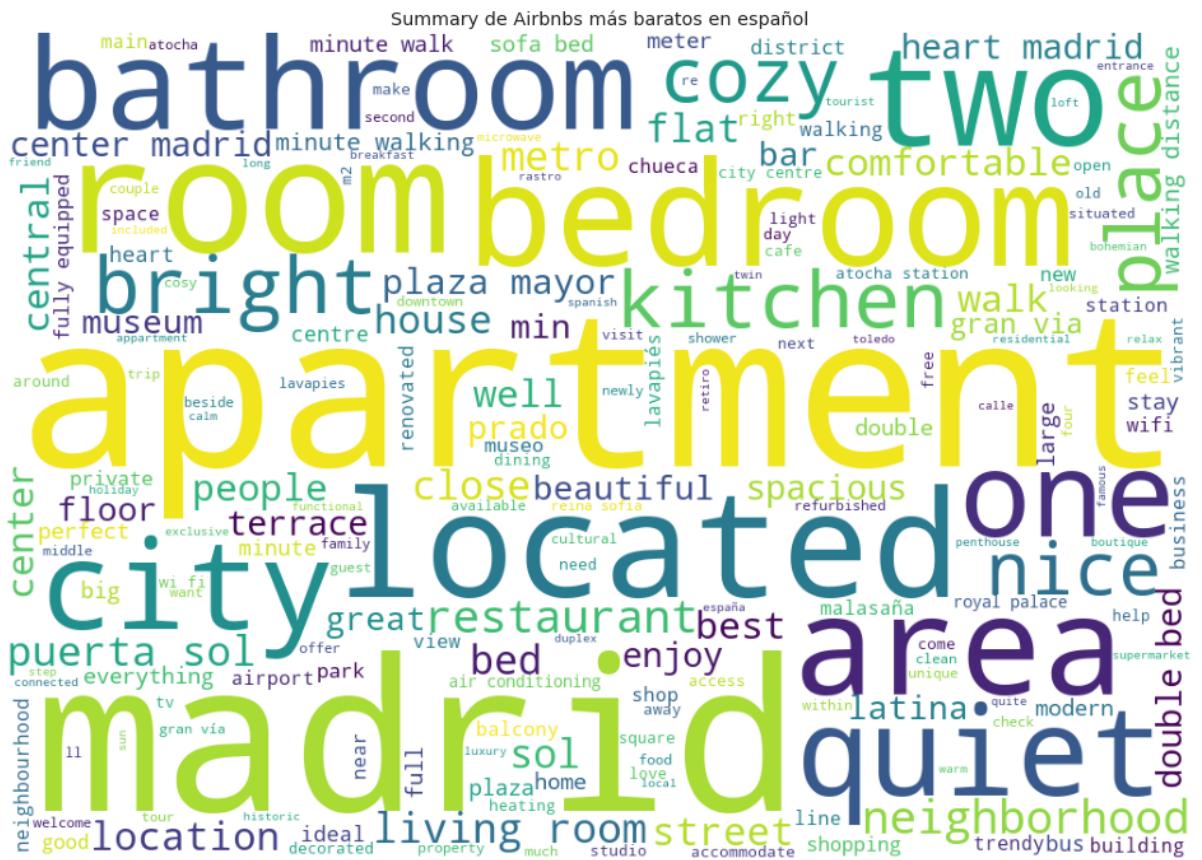




Este código crea dos nuevos data frames, uno para alojamientos cuyo resumen está en inglés y otro para aquellos en otros idiomas. Luego, filtra el data frame en inglés y el de otros idiomas para obtener los 100 alojamientos más baratos y los 100 alojamientos más caros según el precio. A partir de estos data frames, crea dos nuevos data frames que contienen solo el resumen y el precio de los alojamientos más caros y más baratos.

Luego, para cada uno de estos data frames, el código utiliza la función `clean_text()` previamente definida para limpiar los resúmenes y crear una lista de palabras. Finalmente, para cada uno de los dos data frames, genera un `wordcloud` que muestra las palabras más comunes en los resúmenes de los alojamientos más caros y más baratos, respectivamente.





5. Análisis vecindario y tipos de propiedad según el precio medio

Para analizar los siguientes datos hemos optado por generar un mapa de calor (heatmap) utilizando la librería Seaborn en Python. El mapa de calor muestra la mediana del precio de los alojamientos en Madrid agrupados por vecindario y tipo de propiedad.

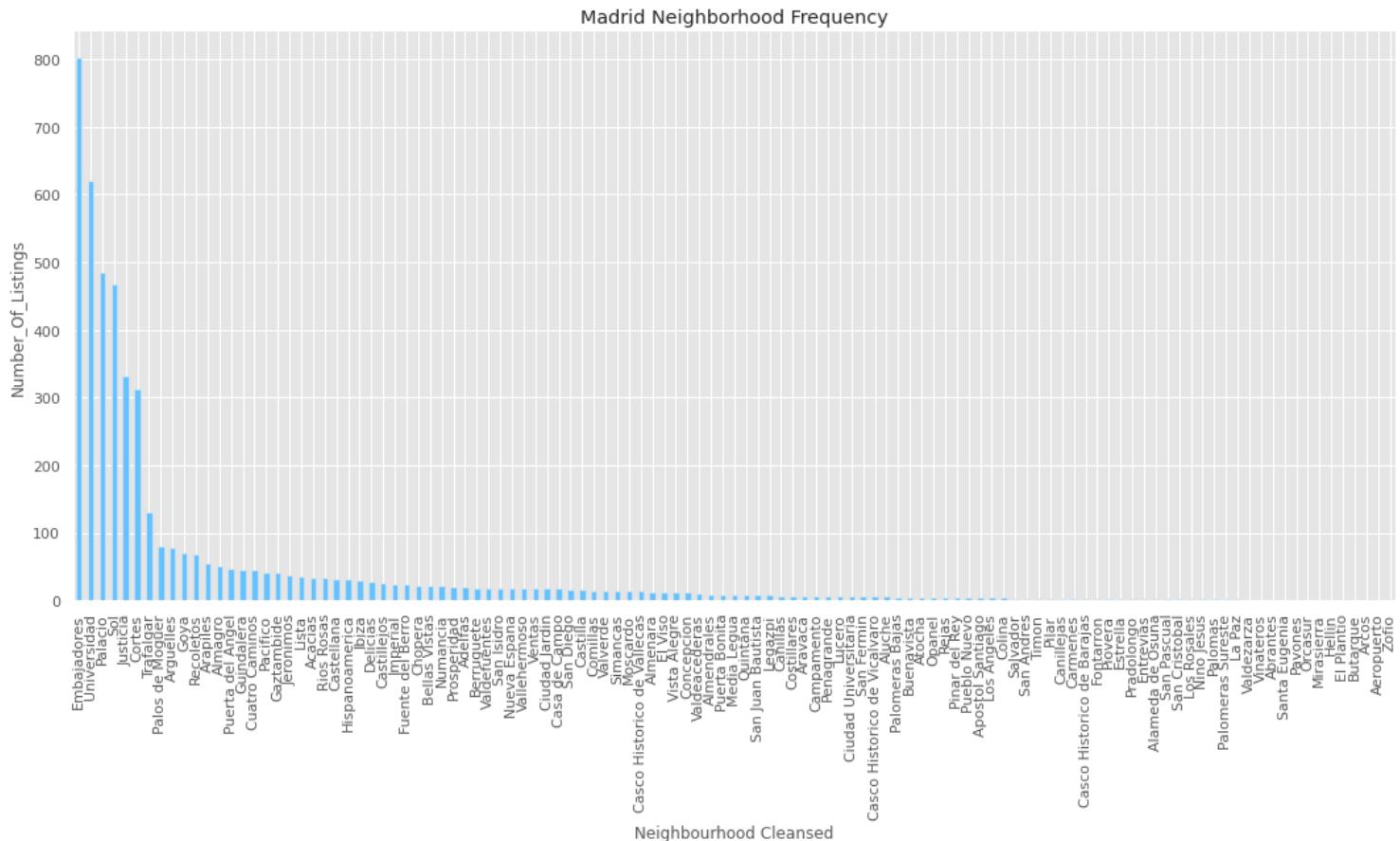
En el eje x del mapa de calor se encuentra el tipo de propiedad (por ejemplo, apartamento, casa, habitación compartida, etc.) y en el eje y se encuentra el vecindario (por ejemplo, Centro, Malasaña, Salamanca, etc.). Los cuadros de la matriz de la tabla de cada combinación vecindario-tipo de propiedad están coloreados según la mediana del precio de los alojamientos. La escala de colores está indicada en la barra de color que se encuentra en la parte inferior del mapa de calor.

La visualización obtenida permite comparar visualmente las medianas de precios de diferentes tipos de propiedades y vecindarios, permitiendo identificar patrones y tendencias en los precios.



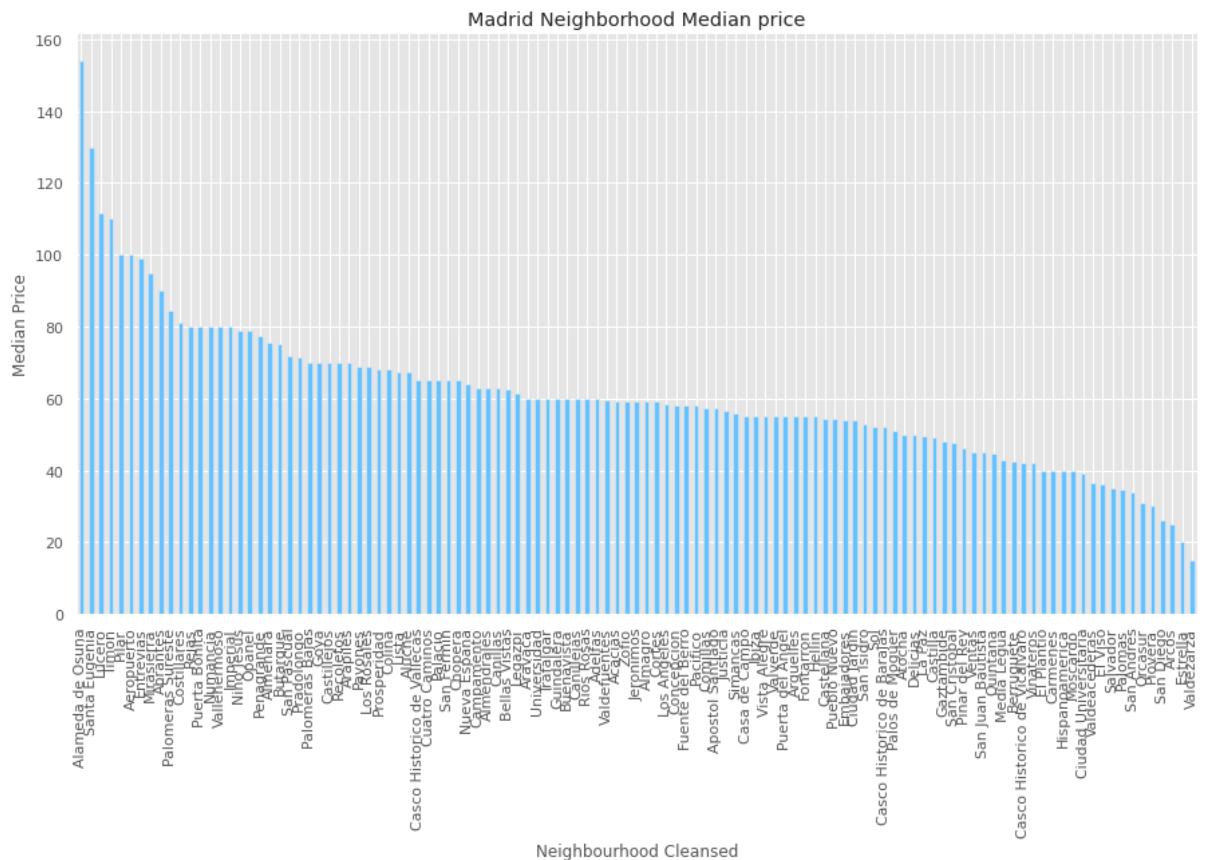
Para seguir analizando tomamos el dataframe limpio de las listas de Airbnb en Madrid y agrupamos las listas por tipo de habitación: casa/apartamento completo, habitación privada y habitación compartida. Luego, crea un nuevo dataframe 'neighbourhood_DF' que agrupa las listas de casa/apartamento completo por vecindario y cuenta el número de listas en cada vecindario, también el precio medio de los listados en cada vecindario. El dataframe resultante se ordena en orden descendente por el número de listas en cada vecindario, mostrando los 5 principales vecindarios con el mayor número de listas de casa/apartamento completo. La salida muestra el nombre de los 5 principales vecindarios junto con el número de listas en cada uno.

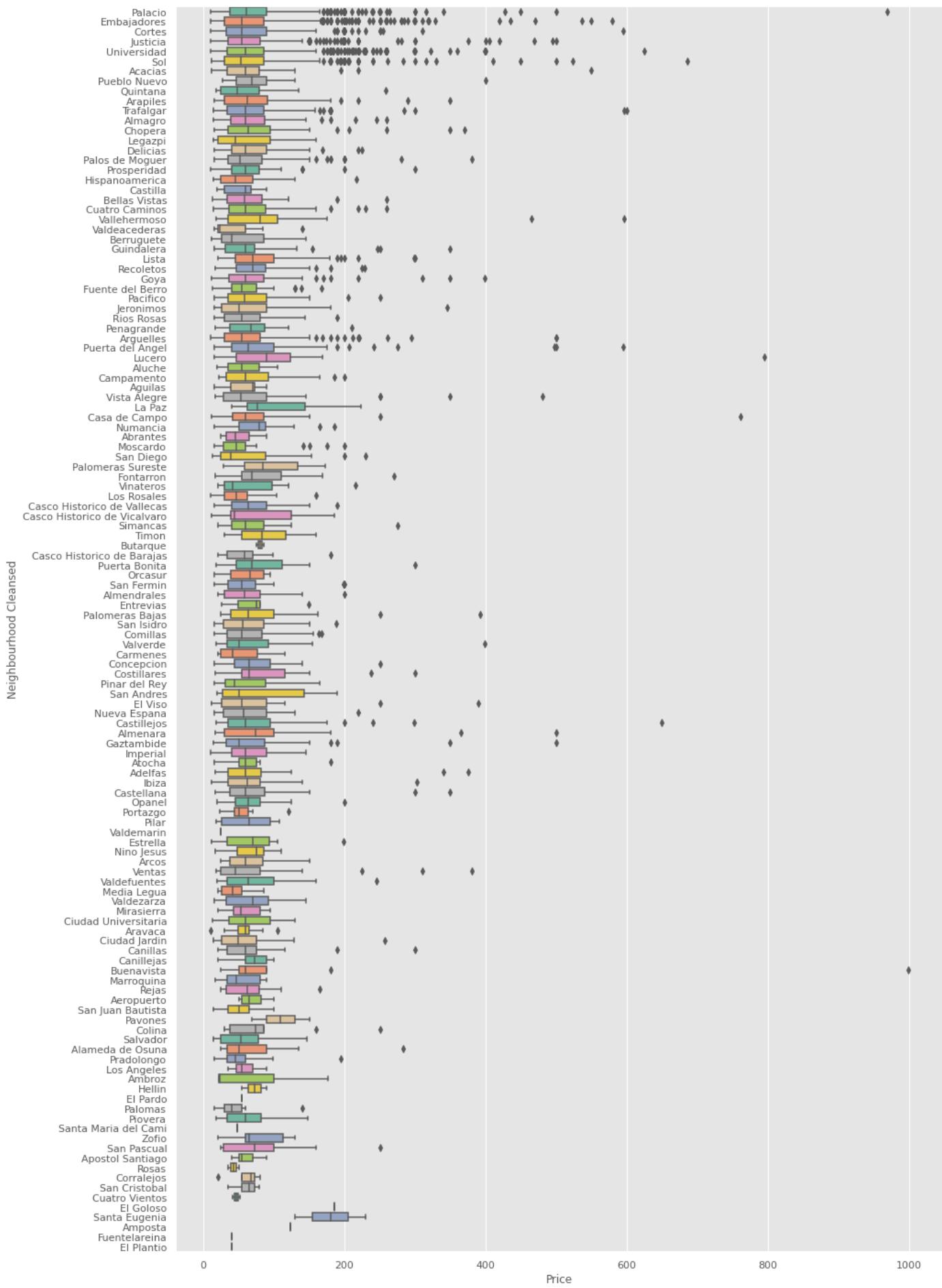
En la siguiente visualización se muestra el número de listados para cada vecindario de Madrid utilizando un gráfico de barras. Primero, se utiliza el DataFrame "neighbourhood_DF" que se creó previamente para obtener las etiquetas de los objetos (neighbourhoods) y las posiciones y alturas de las barras (y_pos). Luego se utiliza el método "plot" para crear el gráfico de barras y se especifican los detalles de la visualización, como el tipo de gráfico, los colores, el tamaño y el título. Finalmente, se agrega una etiqueta al eje y. El resultado es un gráfico de barras que muestra el número de listados para cada vecindario de Madrid.



También hemos analizado a través de un gráfico de caja (boxplot) usando la librería Seaborn de Python. El eje y muestra los nombres de los barrios de Madrid, mientras que el eje x muestra los precios de los alojamientos. Cada caja representa los precios de los alojamientos en un barrio específico, donde la línea central de la caja representa la mediana, la parte inferior de la caja representa el primer cuartil, la parte superior de la caja representa el tercer cuartil, y los puntos que están fuera de la caja representan los valores atípicos.

Para finalizar realizamos una visualización que muestra la mediana del precio de las viviendas en cada barrio, y genera un gráfico de barras que muestra los nombres de los barrios en el eje X y la mediana de precios en el eje Y. El gráfico está ordenado por la mediana de precios en orden descendente, por lo que muestra los barrios más caros en la parte superior. El resultado es una visualización que muestra la mediana del precio de la vivienda para cada barrio en Madrid.





4. Visualización de las métricas

El objetivo de este apartado es realizar un dashboard que permita al anfitrión una visualización sencilla de cómo está el mercado para poder posicionarse en él, tanto en precio como en tipo de habitación que desea ofrecer al inquilino.

Para ello, hemos realizado un análisis previo de los datos que ofrece el dataset de Airbnb, cuáles resultan interesantes y cuales afectan a los parámetros deseados. Además, también nos hemos apoyado en el análisis de cómo afectan los diferentes parámetros al precio, realizado en el siguiente apartado.

Ya que el análisis deseado es para Madrid, hemos decidido tener en cuenta los datos que estaban en la bajo la columna 'city' que indicaban que estaban ubicados en esta localidad. Por lo que hemos eliminado los datos situados en otras localidades y no hemos tenido en cuenta los valores vacíos, ya que estos implicaban un porcentaje menor al 0.04% del total. Tampoco hemos considerado los datos introducidos en otros idiomas como el chino para no introducir posibles errores, teniendo en cuenta que estos no llegan al 0.4% del total.

Con este análisis hemos procedido a realizar un primer borrador de las vistas que se desean en el dashboard determinando que como adecuadas tres vistas: una tabla con datos de los apartamentos, un gráfico de densidad que muestra el porcentaje de 'Room type' respecto el total y un mapa para ubicar cada una de las propiedades que incluye más detalles sobre las mismas.

VISTA 1: PROPERTIES.

En la primera vista, a la que hemos denominado 'Properties', hemos introducido una tabla con las propiedades por nombre, acompañadas de el precio total que incluye el precio por noche y la tasa de limpieza y por el depósito a ingresar. En este caso hemos decidido hacerlo así ya que el depósito es reembolsable y el precio final no.

Tanto en la columna del depósito como en la de la tasa de limpieza había celdas sin valor que hemos decidido convertir a cero, ya que hemos considerado que el valor de los mismos es cero y completar con otros valores como la mediana o la media podía inducir errores en los cálculos.

Además, en esta vista hemos añadido un desplegable con valores como número de baños o número de habitaciones que se han considerado que influyen a la hora de asignarle un precio a la propiedad.

A continuación, se muestra una imagen de esta vista en la que se muestra el desplegable que aparece al pasar el cursor por una de las propiedades.

PROPERTIES

Name	Total pric..	Security..	
1	133	200	
1 bedroom apartment in La Latina	140	300	
1 bedroom Apartment in the city center of ..	80	500	
1 bedroom apartment near the bull ring	39	100	
1 bedroom flat in the heart of Madrid	50	0	
1 Bedroom flat, sunny and very quiet in La..	86	180	
1 bedroom in Calle Zurbano with garage a..	139	400	
1 cozy bedroom with a nice view	37	0	
1 Dormitorio	30	0	
1 Double room	37	0	
1 MIN AL METRO - 4 dormitorios	80	0	
1 min de la estación de metro Argüelles	30	0	
1 or 2 rooms + breakfast in the city center.	40	0	
1 or 2 rooms with private bathroom in city..	35	0	
1 or 2 Rooms-Sharedflat. WI-FI	38	250	
1 or 2 rooms+breakfast in city center loft	40	0	
1 room Center of Malasana	20	0	
1 room in bright flat in malasaña	37	0	
1 room with bathroom	45	0	
1 Stop from Atocha	20	0	
1-4.Madrid Center.Las Cortes.Sol.50m2.Br..	125	150	
1-6Persons+MadridCentre+WIFI+Cozy+Ap..	50	0	
1-7.Las Cortes.Sol.Madrid Center. 70m2.B..	169	150	
1-8.Las Cortes.Sol.Madrid Center. 70m2.B..	175	150	
-----	---	---	

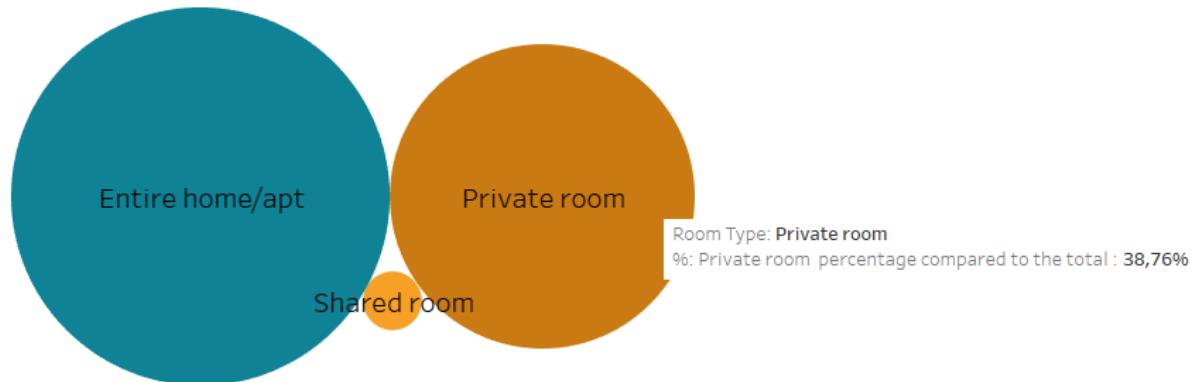
Total price (non-refundable): 86 €
 Cleaning Fee :30 €
 Price:56 €
 Accommodates: 3
 Bedrooms: 1
 Bathrooms: 1,00

VISTA 2: PERCENTAGE OF ROOM TYPE

En la segunda vista hemos considerado que un usuario no acostumbrado a tratar con datos puede manejarse mejor si estos se le proporcionan de manera más visual. Es por ello por lo que hemos añadido un gráfico de densidad para el porcentaje de tipo de habitación que hay respecto al total de los alojamientos. De esta manera el propietario puede visualizar que hacen los otros propietarios y decidir si le interesa más poner en alquiler la propiedad al completo, por habitaciones e incluso por camas.

A continuación se muestra una imagen de la vista.

Percentage of room type

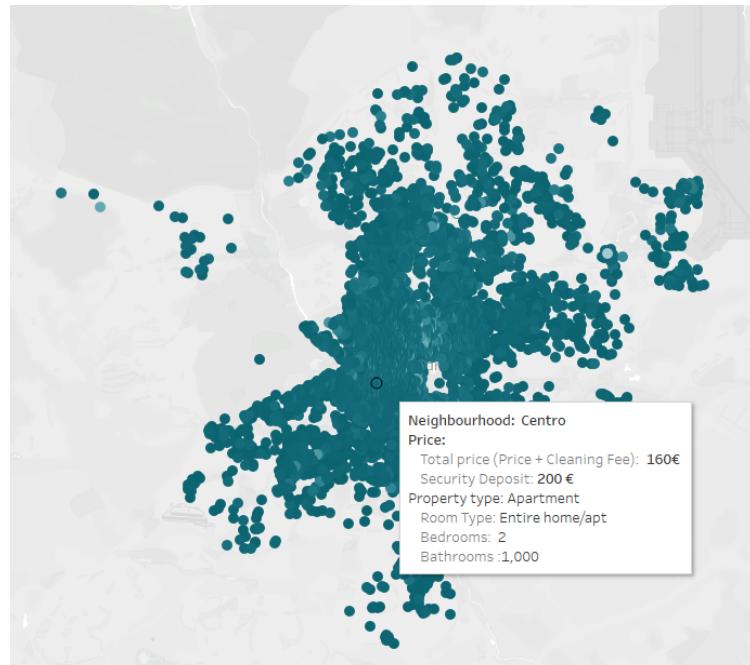


VISTA 3: PROPERTY DETAILS

En la tercera vista hemos decidido añadir un mapa interactivo con las distintas propiedades en función de la longitud y de la latitud. Para ello, hemos dado a los datos la función geográfica longitud y la función geográfica latitud respectivamente.

Además, hemos añadido datos como el precio, el tipo de propiedad o la cantidad de baños de la que dispone el alojamiento para que el usuario pueda visualizar de una manera sencilla las características de los mismos.

A continuación se muestra una imagen de la vista.



Finalmente hemos considerado necesario añadir filtros comunes a todas las vistas de manera que cuando el usuario quiera saber detalles o acotar los datos, todas las vistas muestren los resultados para esos datos seleccionados.

Por ello se han añadido tres filtros: por neighbourhood, por precio y por tipo de habitación.

Para neighbourhood hemos considerado la pestaña neighbourhood group cleansed ya que la cantidad de barrios es más reducida y de cara al usuario hemos considerado que es mejor a disponer de más datos en el caso que quiera filtrar por barrio. Las otras pestañas que incluían barrios añadían una cantidad más grande de barrios pero en muchos de los casos con datos insuficientes para poder comparar.

Visualmente hemos considerado más adecuado que el filtro sea una lista desplegable, en el que el usuario pueda seleccionar tantos barrios como considere necesarios. Además hemos considerado que era adecuado dejar los datos nulos por si el propietario quiere visualizarlos.

El segundo filtro que hemos añadido es un intervalo de valores pro precio, para que el anfitrión pueda seleccionar el rango de precios que considere y de esta manera poder ver los datos de las diferentes vistas para este rango de precios. Así, puede evitar datos con precios muy elevados o muy bajos, en función de lo que considere oportuno.

El tercer filtro es una lista de valores múltiples para el tipo de habitación, propiedad completa, habitación privada o habitación compartida. Todos ellos se pueden seleccionar de manera individual o conjuntamente.

A continuación se muestra una imagen de los diferentes filtros. De izquierda a derecha y de arriba a abajo, la lista desplegable, el intervalo de valores y la lista de valores múltiples.

FILTERS	
Neighbourhood	(Todo)
Total Price	0 1.565
Room Type	<input checked="" type="checkbox"/> (Todo) <input checked="" type="checkbox"/> Entire home/apt <input checked="" type="checkbox"/> Private room <input checked="" type="checkbox"/> Shared room

En la siguiente imagen se muestran dos ejemplos, uno del dashboard completo (sin filtros) y otro del dashboard con filtros aplicados.

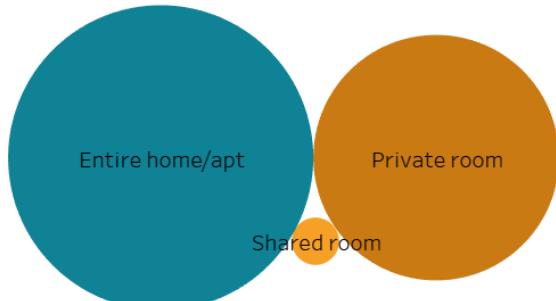
FILTERS

Neighbourhood <input type="text" value="Todo"/>	Room Type <input checked="" type="checkbox"/> (Todo) <input checked="" type="checkbox"/> Entire home/apt <input checked="" type="checkbox"/> Private room <input checked="" type="checkbox"/> Shared room
Total Price 0 <input type="range"/> 1.565	

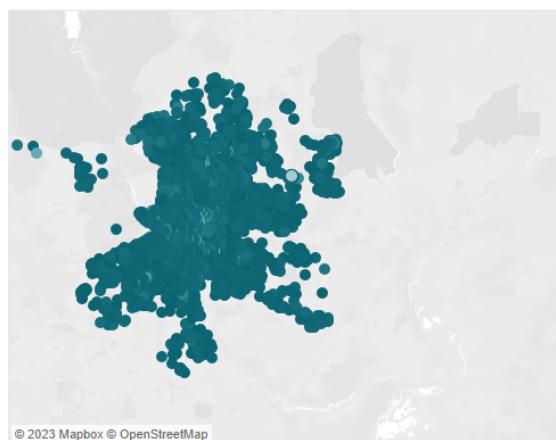
PROPERTIES

Name	Total pric..	Security..
1	133	200
1 bedroom apartment in La Latina	140	300
1 bedroom Apartment in the city center of...	80	500
1 bedroom apartment near the bull ring	39	100
1 bedroom flat in the heart of Madrid	50	0
1 Bedroom flat, sunny and very quiet in La...	86	180
1 bedroom in Calle Zurbano with garage a...	139	400
1 cozy bedroom with a nice view	37	0
1 Dormitorio	30	0
1 Double room	37	0
1 MIN AL METRO - 4 dormitorios	80	0
1 min de la estación de metro Argüelles	30	0
1 or 2 rooms + breakfast in the city center.	40	0
1 or 2 rooms with private bathroom in city..	35	0
1 or 2 Rooms-Sharedflat. WI-FI	38	250
1 or 2 rooms+breakfast in city center loft	40	0
1 room Center of Malasana	20	0
1 room in bright flat in malasaña	37	0
1 room with bathroom	45	0
1 Stop from Atocha	20	0
1-4.Madrid Center.Las Cortes.Sol.50m2.Br..	125	150
1-6Persons+MadridCentre+WiFi+Cozy+Ap..	50	0
1-7.Las Cortes.Sol.Madrid Center. 70m2.B..	169	150
1-8.Las Cortes.Sol.Madrid Center. 70m2.B..	175	150
1-12.Las Cortes.Sol.Madrid Center.120m2..	276	200

Percentage of room type



Property details



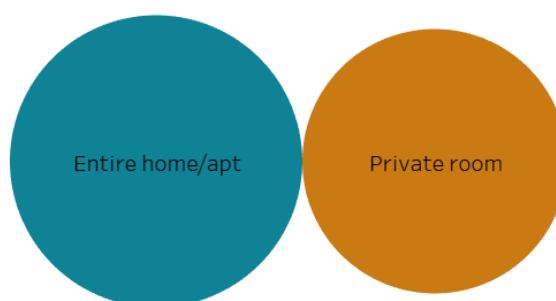
FILTERS

Neighbourhood <input type="text" value="Retiro"/>	Room Type <input type="checkbox"/> (Todo) <input checked="" type="checkbox"/> Entire home/apt <input checked="" type="checkbox"/> Private room <input type="checkbox"/> Shared room
Total Price 12,0 <input type="range"/> 706,0	

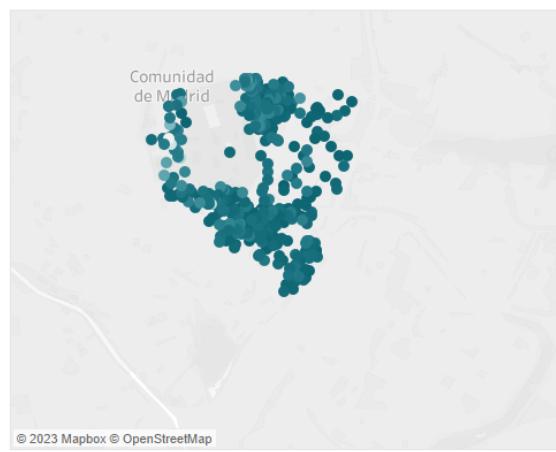
PROPERTIES

Name	Total pric..	Security..
1 or 2 Rooms-Sharedflat. WI-FI	38	250
2 bedroom apartment in central Madrid	165	500
2 habitaciones Mar y Tierra	70	0
2 persons private room	30	0
4/2 Pax Buhardilla en ATOCHA con Chimen..	81	99
140 m2 de lujo Retiro Park	120	150
(YA ALQUILADA 1p) Habitación zona Retir..	50	100
(YA ALQUILADA) Habitación(1p) zona Paci..	40	100
A charming corner at the Retiro Park	99	0
A cosy room nearby El Retiro	28	0
A dream in Madrid, surrounded by museu..	190	90
Acogedor apartamento parque Retiro	52	300
Acogedora habitación al lado Retiro	50	0
Acogedora habitación en zona Retiro	22	0
alojamiento Centrico y Low Cost, para 6 pe..	80	150
Alojamiento económico Retiro-Centro	110	300
alojamiento en centro de madrid	18	0
Alojate en mi casa	20	0
Alquiler de Habitacion	30	400
Alquiler habitacion individual	60	0
Amazing luxury penthouse in the centre of..	535	0
Amplia habitación con baño privado	31	0
Amplia habitación junto al Retiro	31	0
Amplia habitación privada & wifi	29	0
AMPLIO Y SOLEADO - RETIRO Y ATOCHA	91	100

Percentage of room type



Property details



5. Pre-procesamiento y Modelado

Durante el proceso de Pre-procesamiento y modelado, hemos realizado un análisis de las variables que vamos a seleccionar y de las variables con las que podemos generar otras variables.

5.1. Análisis de Columnas para Feature Selection, Feature Engineering y Modelado

En el dataset disponemos de 89 columnas, algunas de las cuales podremos utilizar para el modelo, otras a su vez nos servirán para crear otras columnas mediante el feature engineering que sean también útiles para el modelado. Finalmente, otras columnas no nos servirán y por tanto se podrán eliminar.

Dividiremos las columnas desde el punto de vista de su posible utilidad en el modelo de predicción, entre:

- **DROP FEATURE:** Columnas que no vamos a tener en cuenta.
- **FEATURE ENGINEERING:** Columna que no utilizaremos, pero podemos transformar en una feature que si utilizaremos.
- **FEATURE CATEGORICAL:** Columna categórica que podemos usar en el modelo.
- **FEATURE NUMERICAL:** Columna numérica que podemos usar en el modelo.
- **NEW FEATURE:** Columnas que no vamos a tener en cuenta, pero que nos dan ideas de una nueva feature con datos externos.
- **TARGET: Variable objetivo**

'Price'(target): NaN:9. La variable objetivo es el precio por día del alojamiento. La distribución está escorada a la izquierda ('positive skewness')y tiene bastante pico ('positive kurtosis'). Con el boxplot también vemos que tiene bastantes outliers. Todo ello puede dificultar la predicción de la variable con una Regresión Lineal.

También estudiamos la posible transformación de la variable Target a su logaritmo: $\log('Price')$, el resultado es una distribución más normalizada que probablemente nos sea más "sencilla" de predecir.

'ID' : NaN:0 - integer - Es el ID del alojamiento. no aporta información sobre el alojamiento. **DROP**

'Listing Url': NaN:0 - object - Es la url del alojamiento en Airbnb. Se podría usar para volver a scrapear el alojamiento y recabar más información. **DROP**

'Scrape ID': NaN:0 - object - Es el ID del escape, no aporta información sobre el alojamiento. **DROP**

'Last Scrapped': NaN:0 - object- La fecha de escraperio de los datos. En todos los casos es la misma fecha 08-04-17. No aporta información sobre el alojamiento. En caso de utilizarla, habría que transformarla a un formato datetime. **DROP**

'Name': NaN:0 - object - Título del Alojamiento - En sí mismo no nos da información para el modelado. Pero se pueden extraer diferentes nuevas features: longitud del título, idioma del título, palabras vinculadas al precio, tokenizacion NLP. **FEATURE ENGINEERING**

'Summary': NaN: 488 - object - Resumen del Alojamiento - En sí mismo no nos da información para el modelado. Pero se pueden extraer diferentes nuevas features: longitud del resumen, idioma del resumen, palabras vinculadas al precio, tokenizacion NLP. **FEATURE ENGINEERING**

'Space': NaN: 488 - object - Descripción del espacio del Alojamiento - En sí mismo no nos da información para el modelado. Pero se pueden extraer diferentes nuevas features: longitud de la descripción, idioma de la descripción, palabras vinculadas al precio, tokenizacion NLP. **FEATURE ENGINEERING**

'Description': NaN: 6 - object - En sí mismo no nos da información para el modelado. Pero se pueden extraer diferentes nuevas features: longitud de la descripción, idioma de la descripción, palabras vinculadas al precio, tokenizacion NLP. **FEATURE ENGINEERING**

'Experiences Offered': NaN:0 - object - Experiencias ofrecidas - 100% none - **DROP**

'Neighborhood Overview'; NaN:4938 - object - Descripción del barrio en el que está el alojamiento. No aporta por sí misma y por la gran cantidad de nulos tampoco podemos hacer mucho FE. **DROP**

'Notes': NaN:8201 - object - Notas de interés del Host. No aporta por sí misma y por la gran cantidad de nulos tampoco podemos hacer mucho FE. **DROP**

'Transit': NaN:4987 - object - Descripción de las opciones de transporte, no tabuladas. Se podría intentar averiguar qué línea de metro aparece en el texto, pero seguramente sería mejor hacerlo directamente a través de la API de la EMT Madrid (<https://apidocs.emtmadrid.es/>). **DROP. NEW FEATURE.**

'Access': NaN:5705 - object - Descripción de las zonas del alojamiento a las que tiene acceso el huésped. Hay muchos NaN y no parece dar mucha información adicional. **DROP**

'Interaction': NaN:5722 - object - Disponibilidad de interacción del host. Tiene muchos NaN y no aporta información adicional. **DROP**

'House Rules': NaN:4648- object - Normas de la casa, sin normalizar. Tiene muchos NaN y no aporta información adicional. **DROP**

'Thumbnail Url': NaN:2478 - object - Url de la miniatura de la foto del alojamiento. Podemos hacer una feature que nos diga si el alojamiento tiene o no Thumbnail. Mismos datos que en Medium Url y XL Picture Url. **FEATURE ENGINEERING**

'Medium Url': NaN: 2478 - object - Url de la foto mediana de la foto del alojamiento. Ya hemos usado Thumbnail. **DROP**

'Picture Url': NaN: 29 - object - Url de la foto mediana de la foto del alojamiento. Ya hemos usado Thumbnail. Podríamos analizar el contenido de la imagen con redes neuronales pre-entrenadas, por ejemplo para ver si sale una piscina o no en la foto, etc **DROP FEATURE. NEW FEATURE.**

'XL Picture Url': NaN: 2478 - object - Url de la foto XL de la foto del alojamiento. Ya hemos usado Thumbnail. **DROP**

'Host ID': NaN:0 - integer - Es el ID del host. no aporta información sobre el alojamiento. **DROP**

'Host URL': NaN:0 - object - Es la URL del host. no aporta información sobre el alojamiento. **DROP**

'Host Name': NaN:3 - object - Es el nombre del host. No aporta información sobre el alojamiento. **DROP**

'Host Since': NaN:3 - object - Fecha desde la que el host se dió de alta. Por sí misma, no nos aporta información. Pero nos puede aportar la antigüedad del host en días hasta la fecha de escape (2017-04-08) **FEATURE ENGINEERING**

'Host Location': NaN:42 - object - Localidad del host. No aporta información sobre el alojamiento. No es consistente, algunos pone país otros ciudad. Más del 90% son de Madrid/España . **DROP**

'Host About': NaN:49722 - object - información más o menos detallada del Host. Podemos crear una nueva Feature, tiene / no tiene presentación (35% No - 65% si) **FEATURE ENGINEERING**

'Host Response Time': NaN: 1655 - object - información sobre el tiempo de respuesta del Host. 4 opciones y NaN, se puede categorizar en 4 categorías + Unknown, **FEATURE CATEGORICAL** y se puede hacer feature engineering creando la columna is_ResponseInHours. **FEATURE ENGINEERING**

'Host Response Rate': NaN:1655. float. Tasa de respuesta del 0 al 100 del Host. Tiene muy poca variabilidad desde Q1 son todo 100. **DROP**

'Host Acceptance Rate': NaN:13207. Todo nulos. **DROP**

'Host Thumbnail Url': NaN: 3 - object - Url de la foto de la miniatura de la foto del host. No nos da información del Alojamiento. **DROP**

'Host Picture Url': NaN:3-object Url de la foto del Host. No nos da información del Alojamiento. Se podría usar la foto para una nueva feature, por ejemplo aparece una cara en la foto o no aparece. **DROP . NEW FEATURE.**

'Host Neighbourhood': NaN:3249. Barrio del Host. No nos da información del alojamiento. **DROP**

'Host Listings Count': NaN:3. float. Número de Alojamientos del host. se podría transformar a integer. **FEATURE NUMERICAL**

'Host Total Listings Count': NaN:3. float. Número de Alojamientos del host. se podría transformar a integer. Misma que la anterior. **DROP**

'Host Verifications': NaN: 6-object. Lista de elementos de verificación del Host. Posible para feature engineering. Se puede contar cuantas verificaciones tienen y dejarlo como columna. También se pueden separar los tipos de verificación y hacer una columna para cada tipo de verificación. **FEATURE ENGINEERING**

'Street': NaN:0. object.Campo compuesto del barrio, la ciudad, el código postal. Tenemos la misma información en otras columnas. **DROP**

'Neighbourhood': NaN: 4454.object. Nombre del barrio del alojamiento. 65 valores únicos. Utilizaremos 'Neighbourhood Group Cleansed' que es más reducido y no tiene NaN. **DROP**

'Neighbourhood Cleansed': NaN: 0.object. Nombre del barrio del alojamiento.125 valores únicos. Utilizaremos 'Neighbourhood Group Cleansed' que es más reducido. **DROP**

'Neighbourhood Group Cleansed': NaN:0 - object - Barrio del Alojamiento agrupado. 21 valores únicos. **FEATURE CATEGORICAL**

'City': NaN:0. object. Ciudad = Madrid. No hay variabilidad. **DROP**

'State': NaN:42. object. State = Comunidad de Madrid. No hay variabilidad. **DROP**

'Zipcode': NaN:0. object. Código postal. Categórica con 76 valores distintos. Usaremos Neighborhood Group Cleansed. **DROP**

'Market': NaN:62. Mercado. No hay mucha variabilidad. **DROP**

'Smart Location': NaN: 0. No hay mucha variabilidad. **DROP**

'Country Code': NaN:0. ES. No hay variabilidad. **DROP**

'Country': NaN:0. Spain. No hay variabilidad. **DROP**

'Latitude'. NaN:0. Latitud del alojamiento. **FEATURE NUMERICAL**

'Longitude': NaN:0. Longitud del alojamiento. **FEATURE NUMERICAL**

'Property Type': NaN:0. 22 tipos de propiedad. Lo podemos reducir a las más numerosas. **FEATURE CATEGORICAL**

'Room Type': NaN:0-object- Tipo de alojamiento. 3 variaciones. Se puede categorizar. **FEATURE CATEGORICAL** Además se puede crear una variable is_EntireHome. **FEATURE ENGINEERING**

'Accommodates': NaN:0. float. Personas que caben en el alojamiento. **FEATURE NUMERICAL**

'Bathrooms': NaN:49. float. Número de baños. **FEATURE NUMERICAL**

'Bedrooms': NaN:23. float. Número de habitaciones. **FEATURE NUMERICAL**

'Beds': NaN:49. float. Número de camas. **FEATURE NUMERICAL**

'Bed Type': NaN:0.object. Tipo de cama. Poca variabilidad. Se puede crear una variable is_Bed. **FEATURE ENGINEERING**

'Amenities': NaN: 107. Listado de Amenities del Alojamiento. Se pueden crear nuevas features como si tiene internet, lavadora, secadora, etc... **DROP FEATURE. NEW FEATURE.**

'Square Feet': NaN:12688. float. Metros cuadrados del alojamiento. 96% NaNs. **DROP**

'Weekly Price': Precio Semanal. DATA LEAK. **DROP**

'Monthly Price': Precio Mensual. DATA LEAK. **DROP**

'Security Deposit': NaN: 7572. Depósito de Seguridad. Diría que es 0 si es NaN, es decir, no hay depósito de seguridad. **FEATURE NUMERICAL**

'Cleaning Fee': NaN: 5387. Depósito de Limpieza. Diría que es 0 si es NaN, es decir, no hay depósito de limpieza. **FEATURE NUMERICAL**

'Guests Included': NaN:0 - integer - Huéspedes incluidos en el precio. **FEATURE NUMERICAL**

'Extra People': NaN:0. integer. Personas adicionales. Tiene bastantes outliers. **FEATURE NUMERICAL**

'Minimum Nights': NaN:0. integer. Mínimo de noches. **FEATURE NUMERICAL**

'Maximum Nights': NaN:0. integer. Máximo de noches. Distribución muy extraña.
DROP

'Calendar Updated': NaN:0. object. Tiempo desde la actualización del calendario. Cambiar a días desde los que se actualizó el calendario. También es una feature de `Is_UpdatedToday` **FEATURE ENGINEERING**

'Has Availability': NaN:13207. Todo NaNs. **DROP**

'Availability 30': NaN:0. Disponibilidad 30. **FEATURE NUMERICAL**

'Availability 60': NaN:0. Disponibilidad 60. **FEATURE NUMERICAL**

'Availability 90': NaN:0. Disponibilidad 90. **FEATURE NUMERICAL**

'Availability 365': NaN:0. Disponibilidad 365. **FEATURE NUMERICAL**

'Calendar last Scraped': NaN:0. object. Último escrapeo del calendario. Sin Variabilidad. **DROP**

'Number of Reviews': NaN:0. float. Número de Reviews. **FEATURE NUMERICAL**

'First Review': NaN:2713. object. Fecha de la primera review. Se puede crear una feature de días desde la first review. **FEATURE ENGINEERING**

'Last Review': NaN:2714. object. Fecha desde la última review. Se puede crear una feature de días desde la last review. **FEATURE ENGINEERING**

'Review Scores Rating': NaN:2838. Rating de los scores de los reviews. **FEATURE NUMERICAL**

'Review Scores Accuracy': NaN:2856. Rating de los scores de los reviews. **FEATURE NUMERICAL**

'Review Scores Cleanliness': NaN:2850 Rating de los scores de los reviews. **FEATURE NUMERICAL**

'Review Scores Checkin': NaN:2866 . Rating de los scores de los reviews. **FEATURE NUMERICAL**

'Review Scores Communication': NaN:2850 . Rating de los scores de los reviews. **FEATURE NUMERICAL**

'Review Scores Location', NaN:2858 . Rating de los scores de los reviews. **FEATURE NUMERICAL**

'Review Scores Value' NaN:2868 . Rating de los scores de los reviews. **FEATURE NUMERICAL**

'License'. NaN:12959. Número de licencia. 98% NaN . **DROP**

'Jurisdiction Names': NaN:13207. Todo NaNs. **DROP**

'Cancellation Policy': NaN:0. object. Política de Cancelación. Convertir en 3 categorías. **FEATURE CATEGORICAL**

'Calculated host listings count': NaN: 0. float. Alojamientos por host calculados. Muy similar a Host Listing count. **FEATURE NUMERICAL**

'Reviews per Month'. NaN: 0. float. Media de reviews. **FEATURE NUMERICAL**

'Geolocation': NaN:0. object. Tupla latitud, longitud. ya tenemos los valores en el df. **DROP**

'Features': NaN:0. object. Listado de detalles sobre el alojamiento. Se pueden crear nuevas features de cada ítem de la lista en una columna. **DROP FEATURE. NEW FEATURE.**

5.2. Feature Engineering

Según el análisis que hemos realizado, hemos podido crear las siguientes variables con

Name_Len: FLOAT - longitud del título del alojamiento

Hemos calculado la longitud del string del título del alojamiento, puede que una determinada longitud del mismo se relacione con más o menos precio.

Name_Lang: BOOL - es inglés del título del alojamiento

Hemos verificado si el idioma del string del título del alojamiento es inglés (True) o no (False) puede que una determinada longitud del mismo se relacione con más o menos precio.

Summary_Len: FLOAT - longitud del resumen del alojamiento

Hemos calculado la longitud del string del resumen del alojamiento, puede que una determinada longitud del mismo se relacione con más o menos precio.

Summary_Lang: BOOL - es inglés del resumen del alojamiento

Hemos verificado si el idioma del string del resumen del alojamiento es inglés (True) o no (False) puede que una determinada longitud del mismo se relacione con más o menos precio.

Space_Len: FLOAT - longitud del resumen espacio del alojamiento

Hemos calculado la longitud del string del título del alojamiento, puede que una determinada longitud del mismo se relacione con más o menos precio.

Space_Lang: BOOL - es inglés del resumen espacio del alojamiento

Hemos verificado si el idioma del string de la descripción del espacio del alojamiento es inglés (True) o no (False) puede que una determinada longitud del mismo se relacione con más o menos precio.

Description_Len: FLOAT - longitud del resumen espacio del alojamiento

Hemos calculado la longitud del string de la descripción espacio del alojamiento, puede que una determinada longitud del mismo se relacione con más o menos precio.

Description_Lang: BOOL - es inglés del resumen espacio del alojamiento

Hemos verificado si el idioma del string de la descripción del alojamiento es inglés (True) o no (False) puede que una determinada longitud del mismo se relacione con más o menos precio.

is_thumbnail: BOOL - tiene el alojamiento miniatura

Verificamos si el alojamiento tiene miniatura, son los mismos valores que definen si el anuncio tiene Foto XL. Podría ser que los alojamientos con foto de calidad, tuvieran más precio.

is_HostAbout: BOOL - tiene el alojamiento descripción de Host

Verifica si el host tiene rellenado el campo de descripción. Puede que un host con un campo de descripción relleno sea más confiable.

is_ResponseInHours: BOOL - contesta el host en horas o menos

Verifica si el host contesta en unas horas o menos, puede que si el host contesta más rápido, el precio sea más elevado.

num_Host_verifications: - FLOAT - número de verificaciones que tiene el host

Cuenta el número de verificaciones distintas que tiene el host. Por ejemplo, Número de identificación, Email, Teléfono, etc...

is_EntireHome: - BOOL - El alojamiento es la casa entera.

Verifica si el alojamiento es de una casa entera, Casa Entera (True), Habitación en alojamiento o habitación compartida(False)

is_Bed: - BOOL - son las camas camas (no sofa, couch, etc)

Verifica si el alojamiento dispone de cama o de otro tipo de sitio para dormir(sofá, futón, etc). Cama es True y los demás (sofá, futón, etc) False

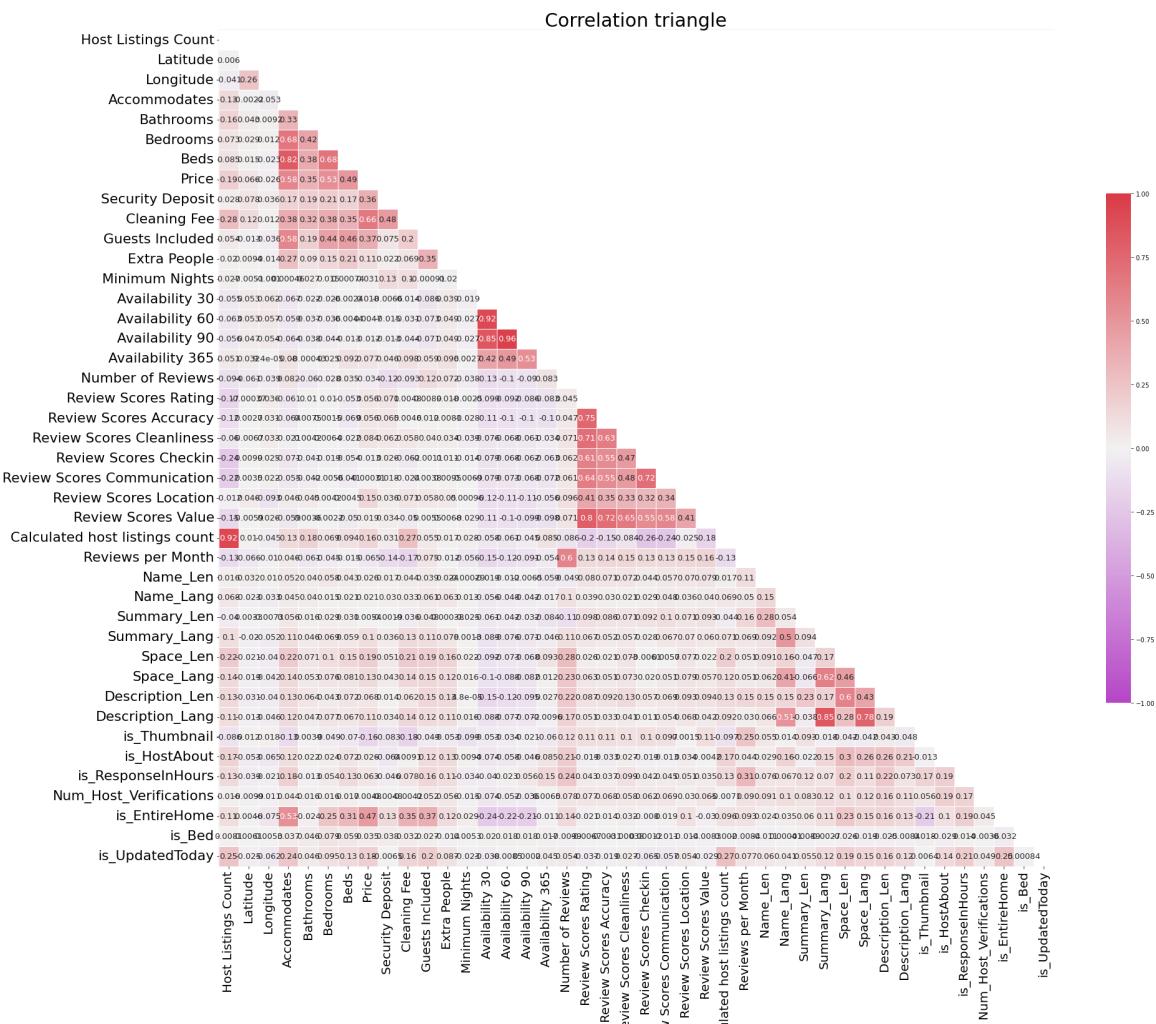
is_UpdatedToday: - BOOL - se ha actualizado el calendario hoy

Se ha actualizado el alojamiento hoy.

5.3. Relación entre columnas

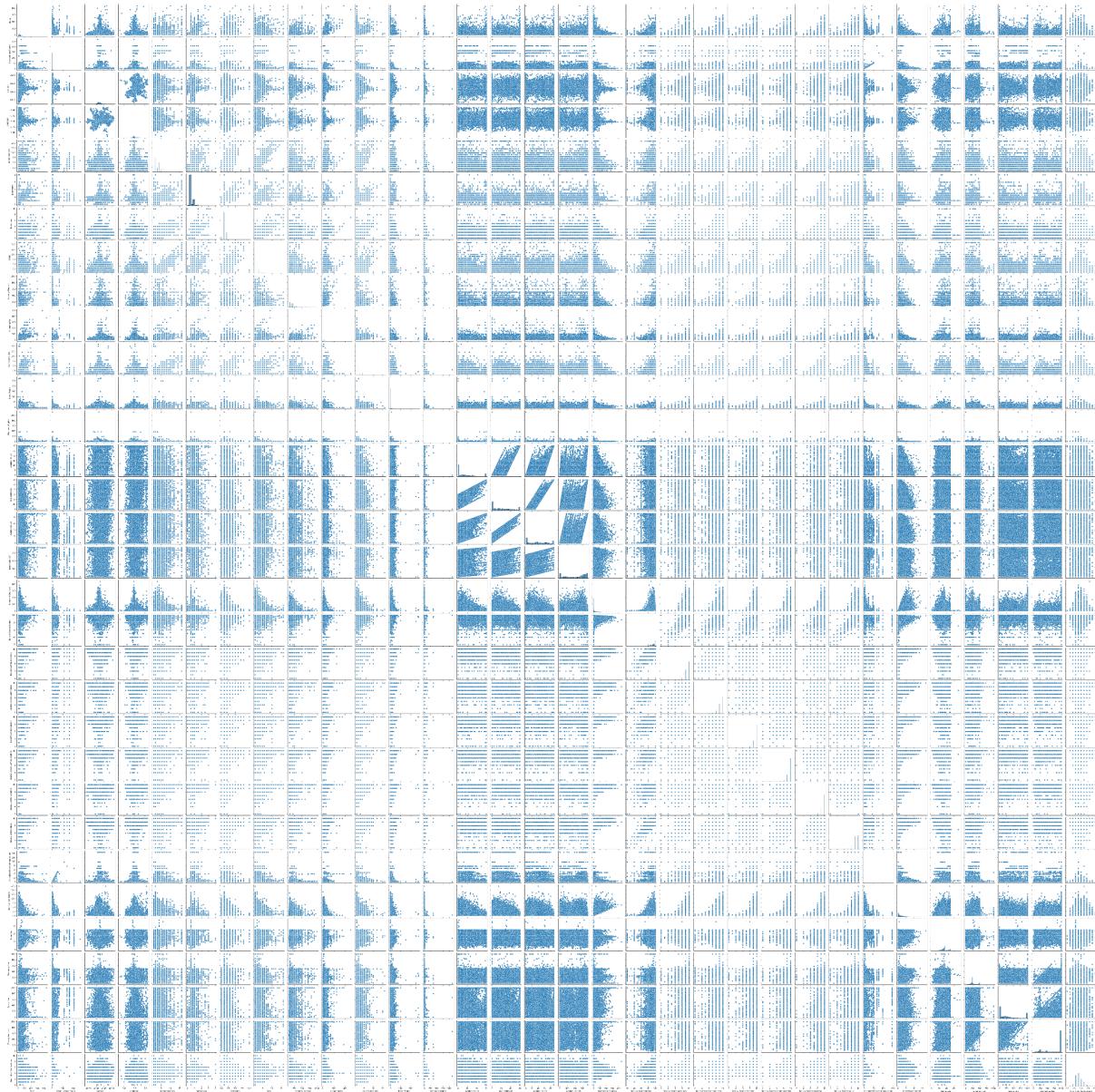
Hemos estudiado la relación entre columnas, tanto para ver cuales se correlacionan linealmente en mayor medida con la variable objetivo: el precio, como para ver si hay columnas muy correlacionadas linealmente entre sí, para quitar alguna columna y evitar el riesgo de colinealidad.

Correlaciones Numéricas



En el triángulo de correlación podemos ver algunos grupos de variables altamente correlacionados linealmente

También hemos realizado un Pairplot



A simple vista, también podemos ver que algunas variables están altamente correlacionadas linealmente.

Concretamente las variables, o grupos de variables más correlacionados linealmente son:

- Calculated host listings count vs Host Listings Count: 0.92 - riesgo de colinealidad
- Beds vs Accommodates: 0.82 - riesgo de colinealidad
- Review Scores Rating, Review Scores Accuracy, Review Scores Cleanliness, Review Scores Checkin, Review Scores Communication, Review Scores Location, Review Scores Value - riesgo de colinealidad

- Name_Lag, Summary_Lang, Space_Lang, Description_Lang - riesgo de colinealidad
- Availability 30, Availability 60, Availability 90, Availability 365 - riesgo de colinealidad

Después de analizar los scatterplots y distribuciones de las variables, decidimos borrar las siguientes:

- Host Listings Count
- Review Scores Accuracy
- Review Scores Value
- Name Lang
- Space Lang
- Description Lang
- Availability 90
- Availability 60

5.4. Feature Selection

En resumen, después del análisis, de la primera selección y del feature engineering, tenemos las siguientes variables:

TARGET:

- Price

FEATURE NUMERIC:

1. Latitude
2. Longitude
3. Accommodates
4. Bathrooms
5. Bedrooms
6. Beds
7. Security Deposit
8. Cleaning Fee
9. Guests Included
10. Extra People
11. Minimum Nights

12. Availability 30
13. Availability 365
14. Number of Reviews
15. Review Scores Cleanliness
16. Review Scores Checkin
17. Review Scores Communication
18. Review Scores Location
19. Review Scores Value
20. Calculated host listings count
21. Reviews per Month
22. Name_Len
23. Summary_Len
24. Spacee_Len
25. Description_Len
26. num_Host_verifications

BOOL:

1. Summary_Lang: BOOL - es inglés del resumen del alojamiento
2. is_thumbnail: BOOL - tiene el alojamiento miniatura
3. is_HostAbout: BOOL - tiene el alojamiento descripción de Host
4. is_ResponseInHours - BOOL - contesta el host en horas o menos
5. is_EntireHome - BOOL - es el alojamiento una casa entera.
6. is_Bed - BOOL - son las camas camas (no sofa, couch, etc)
7. is_UpdatedToday - BOOL - se ha actualizado el calendario hoy

FEATURE CATEGORICAL:

1. Host Response Time
2. Neighbourhood Group Cleansed
3. Property Type

4. Room Type
5. Cancellation Policy

Son muchas variables, así que las iremos dividiendo en grupos para poder realizar diferentes modelos desde el modelo base al modelo definitivo.

5.5. Imputación de Nulos

Para la imputación de nulos seguiremos diferentes estrategias en función de la columna ya que en algunos casos podremos imputarlos al dataset completo, y en otros casos deberemos hacerlo después del split entre train y test para evitar el data leak.

Target: Price

En la variable objetivo, el Precio, tenemos 9 nulos, hemos decidido borrarlos, ya que es la variable objetivo y son pocos.

Security Deposit y Cleaning Fee

En ambos casos, el NaN significa que no hay depósito de seguridad o cleaning fee, por tanto entendemos que hay que asignar un 0 a los NaNs.

Bathrooms, Bedrooms, Beds

Entre las 3 columnas, tenemos 83 filas (de 13198), al no ser muchas filas, decidimos borrarlas.

Reviews

Las columnas de scoring de reviews, creemos que hay que imputarlas con alguna estrategia que obtengamos de los valores de la propia columna o de las demás columnas.

Las columnas tiene aproximadamente 4000 nulos cada una y son las siguientes;

Review Scores Rating

Review Scores Cleanliness

Review Scores Checkin

Review Scores Communication

Review Scores Location

Reviews per Month

Las dos opciones que mejor nos han funcionado son la mediana de la columna y el KNN Imputer. Ya que los resultados de ambas estrategias son muy similares, hemos optado por la vía más sencilla: la mediana.

Para evitar el data leak, hemos aplicado la mediana del set de train de la columna, tanto al X_train como al X_test.

5.6. Creación de subdatasets

Como hemos comentado para poder diferentes modelos del más sencillo al más complejo, creamos diferentes datasets que incluyan diferentes columnas.

Numeric No Reviews

En este subdataset sepáramos solo las features numéricas y booleanas, sin incluir el grupo de features de review scores. Con este dataset crearemos nuestro modelo base, con el que comparar los demás modelos.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 13115 entries, 0 to 13206
Data columns (total 27 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Latitude         13115 non-null   float64 
 1   Longitude        13115 non-null   float64 
 2   Accommodates     13115 non-null   int64  
 3   Bathrooms        13115 non-null   float64 
 4   Bedrooms         13115 non-null   float64 
 5   Beds              13115 non-null   float64 
 6   Guests Included  13115 non-null   int64  
 7   Extra People     13115 non-null   int64  
 8   Minimum Nights   13115 non-null   int64  
 9   Availability 30  13115 non-null   int64  
 10  Availability 365 13115 non-null   int64  
 11  Security Deposit 13115 non-null   float64 
 12  Cleaning Fee    13115 non-null   float64 
 13  Calculated host listings count 13115 non-null   float64 
 14  Num_Host_Verifications 13115 non-null   int64  
 15  Name_Len          13115 non-null   int64  
 16  Summary_Len       13115 non-null   int64  
 17  Space_Len          13115 non-null   int64  
 18  Description_Len   13115 non-null   int64  
 19  Summary_Lang      13115 non-null   bool   
 20  is_Thumbnail      13115 non-null   bool   
 21  is_HostAbout     13115 non-null   bool   
 22  is_ResponseInHours 13115 non-null   bool   
 23  is_EntireHome    13115 non-null   bool   
 24  is_Bed            13115 non-null   bool   
 25  is_UpdatedToday   13115 non-null   bool   
 26  Price             13115 non-null   float64
```

Numeric con Reviews

En este subdataset separamos solo las features numéricas y booleanas, incluyendo el grupo de features de review scores. Con este dataset crearemos nuestro modelo numérico con más features, y con las imputaciones de NaNs.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 13115 entries, 0 to 13206
Data columns (total 34 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Latitude          13115 non-null   float64
 1   Longitude         13115 non-null   float64
 2   Accommodates      13115 non-null   int64  
 3   Bathrooms         13115 non-null   float64
 4   Bedrooms          13115 non-null   float64
 5   Beds               13115 non-null   float64
 6   Guests Included   13115 non-null   int64  
 7   Extra People      13115 non-null   int64  
 8   Minimum Nights    13115 non-null   int64  
 9   Availability 30   13115 non-null   int64  
 10  Availability 365  13115 non-null   int64  
 11  Number of Reviews 13115 non-null   int64  
 12  Review Scores Rating 10315 non-null   float64
 13  Review Scores Cleanliness 10303 non-null   float64
 14  Review Scores Checkin   10287 non-null   float64
 15  Review Scores Communication 10303 non-null   float64
 16  Review Scores Location   10285 non-null   float64
 17  Reviews per Month    10439 non-null   float64
 18  Security Deposit     13115 non-null   float64
 19  Cleaning Fee         13115 non-null   float64
 20  Calculated host listings count 13115 non-null   float64
 21  Num_Host_Verifications 13115 non-null   int64  
 22  Name_Len             13115 non-null   int64  
 23  Summary_Len          13115 non-null   int64  
 24  Space_Len            13115 non-null   int64  
 25  Description_Len      13115 non-null   int64  
 26  Summary_Lang          13115 non-null   bool   
 27  is_Thumbnail          13115 non-null   bool   
 28  is_HostAbout          13115 non-null   bool   
 29  is_ResponseInHours    13115 non-null   bool   
 30  is_EntireHome         13115 non-null   bool   
 31  is_Bed                13115 non-null   bool   
 32  is_UpdatedToday        13115 non-null   bool   
 33  Price                 13115 non-null   float64
dtypes: bool(7), float64(15), int64(12)
memory usage: 2.9 MB
```

All Features Categorical Encoded

Por último, tenemos un subset con todas las columnas numéricas, además de las columnas categóricas codificadas con el OneHotEncoder, a pesar de que pudiera quedar un dataframe un poco sparse. No hemos utilizado el LabelEncoder ya que con una Regresión Lineal puede crear incongruencias.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 13115 entries, 0 to 13206
Data columns (total 63 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Latitude         13115 non-null   float64 
 1   Longitude        13115 non-null   float64 
 2   Accommodates     13115 non-null   int64  
 3   Bathrooms        13115 non-null   float64 
 4   Bedrooms         13115 non-null   float64 
 5   Beds              13115 non-null   float64 
 6   Guests Included  13115 non-null   int64  
 7   Extra People     13115 non-null   int64  
 8   Minimum Nights   13115 non-null   int64  
 9   Availability 30  13115 non-null   int64  
 10  Availability 365 13115 non-null   int64  
 11  Number of Reviews 13115 non-null   int64  
 12  Review Scores Rating 10315 non-null   float64 
 13  Review Scores Cleanliness 10303 non-null   float64 
 14  Review Scores Checkin    10287 non-null   float64 
 15  Review Scores Communication 10303 non-null   float64 
 16  Review Scores Location   10285 non-null   float64 
 17  Reviews per Month      10439 non-null   float64 
 18  Security Deposit      13115 non-null   float64 
 19  Cleaning Fee          13115 non-null   float64 
 20  Calculated host listings count 13115 non-null   float64 
 21  Num_Host_Verifications 13115 non-null   int64  
 22  Name_Len              13115 non-null   int64  
 23  Summary_Len           13115 non-null   int64  
 24  Space_Len              13115 non-null   int64  
 25  Description_Len       13115 non-null   int64  
 26  Summary_Lang           13115 non-null   bool   
 27  is_Thumbnail           13115 non-null   bool   
 28  is_HostAbout           13115 non-null   bool   
 29  is_ResponseInHours    13115 non-null   bool   
 30  is_EntireHome          13115 non-null   bool   
 31  is_Bed                 13115 non-null   bool 
```

```

32  is_UpdatedToday           13115 non-null  bool
33  Price                     13115 non-null  float64
34  Neighbourhood_Barajas    13115 non-null  float64
35  Neighbourhood_Carabanchel 13115 non-null  float64
36  Neighbourhood_Centro     13115 non-null  float64
37  Neighbourhood_Chamartin   13115 non-null  float64
38  Neighbourhood_Chamberi    13115 non-null  float64
39  Neighbourhood_CiudadLineal 13115 non-null  float64
40  Neighbourhood_Fuencarral   13115 non-null  float64
41  Neighbourhood_Hortaleza    13115 non-null  float64
42  Neighbourhood_Latina      13115 non-null  float64
43  Neighbourhood_Moncloa     13115 non-null  float64
44  Neighbourhood_Moratalaz    13115 non-null  float64
45  Neighbourhood_PuenteVallecas 13115 non-null  float64
46  Neighbourhood_Retiro      13115 non-null  float64
47  Neighbourhood_Salamanca    13115 non-null  float64
48  Neighbourhood_SanBlas      13115 non-null  float64
49  Neighbourhood_Tetuán       13115 non-null  float64
50  Neighbourhood_Usera        13115 non-null  float64
51  Neighbourhood_Vicalvaro    13115 non-null  float64
52  Neighbourhood_VillaVallecas 13115 non-null  float64
53  Neighbourhood_Villaverde    13115 non-null  float64
54  PropertyType_BedAndBreakfast 13115 non-null  float64
55  PropertyType_Condominium    13115 non-null  float64
56  PropertyType_House         13115 non-null  float64
57  PropertyType_Loft           13115 non-null  float64
58  PropertyType_Other          13115 non-null  float64
59  BedType_PrivateRoom        13115 non-null  float64
60  BedType_SharedRoom         13115 non-null  float64
61  Cancellation_moderate     13115 non-null  float64
62  Cancellation_strict       13115 non-null  float64
dtypes: bool(7), float64(44), int64(12)
memory usage: 6.3 MB

```

5.7. Modelado

MODELO BASE

El modelo base es el que incluye las features numéricas (sin contar con las Reviews en las que tenemos que hacer imputación de los NaN) y sin las Features que hemos creado.

Dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13115 entries, 0 to 13114
```

```

Data columns (total 15 columns):
 #   Column            Non-Null Count Dtype  
--- 
 0   Latitude          13115 non-null  float64 
 1   Longitude         13115 non-null  float64 
 2   Accommodates      13115 non-null  int64   
 3   Bathrooms         13115 non-null  float64 
 4   Bedrooms          13115 non-null  float64 
 5   Beds               13115 non-null  float64 
 6   Guests Included   13115 non-null  int64   
 7   Extra People      13115 non-null  int64   
 8   Minimum Nights    13115 non-null  int64   
 9   Availability 30   13115 non-null  int64   
 10  Availability 365  13115 non-null  int64   
 11  Security Deposit 13115 non-null  float64 
 12  Cleaning Fee      13115 non-null  float64 
 13  Calculated host listings count 13115 non-null  float64 
 14  Price              13115 non-null  float64 

dtypes: float64(9), int64(6)
memory usage: 1.5 MB

```

Train Test Split:

Separamos el dataset entre Train 80% y Test 20%

Algoritmo:

Utilizamos el algoritmo LinearRegression de los linear models de sklearn

Métricas:

Como métricas de error utilizamos el MAE, el RMSE y el R2

Cross Validation:

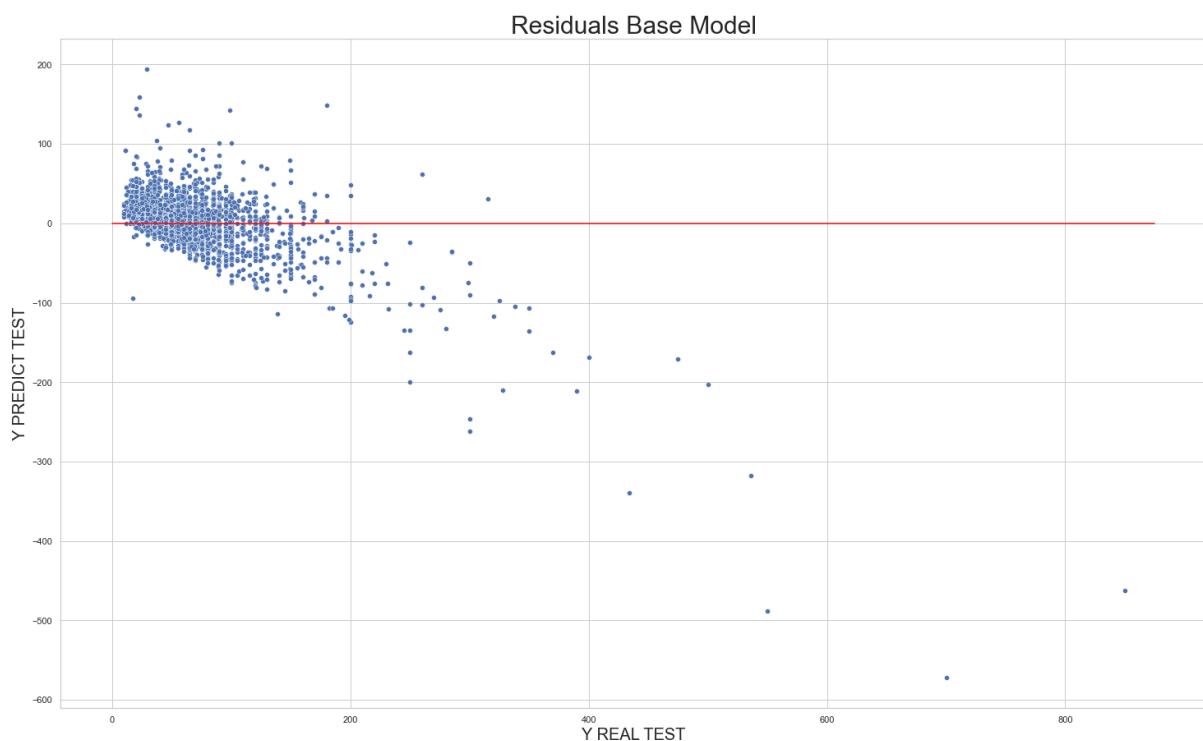
También utilizaremos la cross validation, para que el resultado no dependa del split.

Resultados:

Model	MAE	RMSE	R ²
Train Modelo Base	23,46	41,06	0,475
Test Modelo Base	21,88	36,99	0,517
Cross Validation Base Model	23,57	41,25	0,470

Residuales:

Vemos que los residuales aumentan a medida que aumenta el precio a predecir.



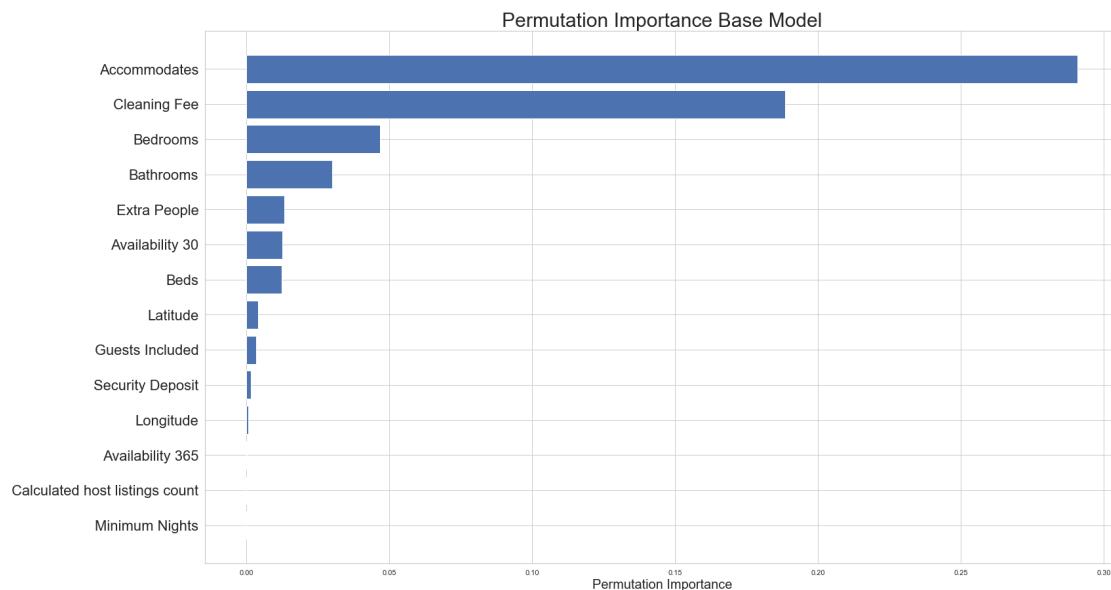
P Values:

Con los p values (que hemos calculado porque sklearn no te los da por defecto) vemos que algunas variables podrían obviarse ya que no aportan demasiado al modelo.

	Feature	P Values
0	Latitude	2.92232e-11
1	Longitude	3.45512e-11
2	Accommodates	0.0329724
3	Bathrooms	5.10427e-191
4	Bedrooms	2.65127e-59
5	Beds	3.92353e-64
6	Guests Included	2.68988e-12
7	Extra People	7.932e-05
8	Minimum Nights	3.70172e-31
9	Availability 30	0.831822
10	Availability 365	1.89629e-28
11	Security Deposit	0.524235
12	Cleaning Fee	2.86756e-06
13	Calculated host listings count	0

Permutation Importance:

En LinearRegression de sklearn no tenemos las feature importance como en otros modelos, pero si disponemos de la permutation importance, que nos dice cuánto del modelo depende de dicha feature.



Accommodates, Cleaning Fee, Bedrooms, Bathrooms son las features que más influyen en el modelo base.

MODELO NUMÉRICO

El modelo numérico incluye todas las features numéricas incluidas las de las Reviews y las Features que hemos creado.

Dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13115 entries, 0 to 13114
Data columns (total 34 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Latitude         13115 non-null   float64 
 1   Longitude        13115 non-null   float64 
 2   Accommodates     13115 non-null   int64  
 3   Bathrooms        13115 non-null   float64 
 4   Bedrooms         13115 non-null   float64 
 5   Beds              13115 non-null   float64 
 6   Guests Included  13115 non-null   int64  
 7   Extra People     13115 non-null   int64  
 8   Minimum Nights   13115 non-null   int64  
 9   Availability 30  13115 non-null   int64  
 10  Availability 365 13115 non-null   int64  
 11  Number of Reviews 13115 non-null   int64  
 12  Review Scores Rating 10315 non-null   float64 
 13  Review Scores Cleanliness 10303 non-null   float64 
 14  Review Scores Checkin 10287 non-null   float64 
 15  Review Scores Communication 10303 non-null   float64
```

```

16 Review Scores Location      10285 non-null float64
17 Reviews per Month          10439 non-null float64
18 Security Deposit           13115 non-null float64
19 Cleaning Fee                13115 non-null float64
20 Calculated host listings count 13115 non-null float64
21 Num_Host_Verifications     13115 non-null int64
22 Name_Len                     13115 non-null int64
23 Summary_Len                  13115 non-null int64
24 Space_Len                    13115 non-null int64
25 Description_Len              13115 non-null int64
26 Summary_Lang                 13115 non-null bool
27 is_Thumbnail                 13115 non-null bool
28 is_HostAbout                 13115 non-null bool
29 is_ResponseInHours           13115 non-null bool
30 is_EntireHome                 13115 non-null bool
31 is_Bed                        13115 non-null bool
32 is_UpdatedToday               13115 non-null bool
33 Price                         13115 non-null float64
dtypes: bool(7), float64(15), int64(12)
memory usage: 2.8 MB

```

Train Test Split:

Separamos el dataset entre Train 80% y Test 20%

Imputar los NaN:

En este dataset tenemos NaNs en las columnas de las Reviews, ya que al querer imputar con la mediana o con el KNN Imputer, queremos evitar el data leak y por tanto lo aplicaremos una vez realizado el split. En concreto, después de probar ambas opciones el que da mejor resultado es aplicar la mediana del Train de cada columna en los NaN de esa columna tanto en Train como en Test.

Algoritmo:

Utilizamos el algoritmo LinearRegression de los linear models de sklearn

Métricas:

Como métricas de error utilizamos el MAE, el RMSE y el R2

Cross Validation:

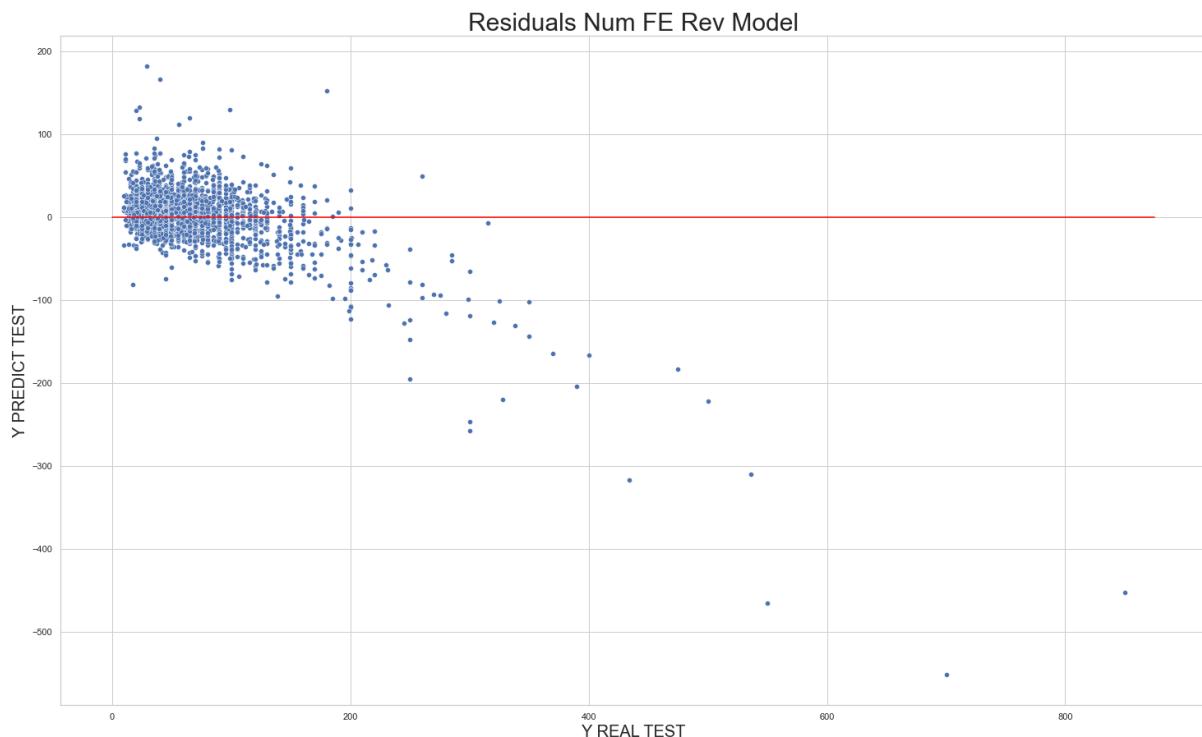
También utilizaremos la cross validation, para que el resultado no dependa del split.

Resultados

Model	MAE	RMSE	R ²
Train Modelo Numérico	21,48	38,74	0,533
Test Modelo Numérico	20,67	35,25	0,562
Cross Validation Numérico	21,64	38,98	0,520
Cross Validation Base Model	23,57	41,25	0,470

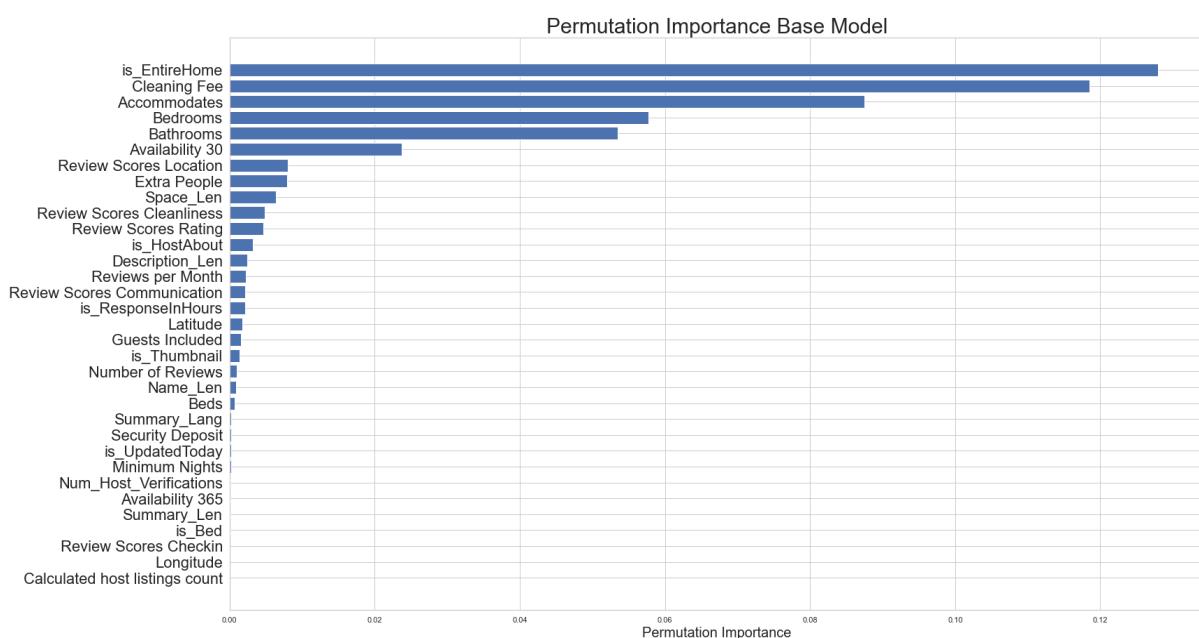
Residuales:

Vemos que los residuales aumentan a medida que aumenta el precio a predecir.



Permutation Importance:

En LinearRegression de sklearn no tenemos las feature importance como en otros modelos, pero si disponemos de la permutation importance, que nos dice cuánto del modelo depende de dicha feature.



Vemos que la Feature que hemos creado is_entireHome influye mucho en el modelo.

MODELO ALL FEATURES

El modelo all features incluye todas las features numéricas incluidas las de las Reviews y las Features que hemos creado, así como las categóricas codificadas con el OneHotEncoding.

Dataset:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13115 entries, 0 to 13114
Data columns (total 63 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Latitude         13115 non-null   float64 
 1   Longitude        13115 non-null   float64 
 2   Accommodates     13115 non-null   int64  
 3   Bathrooms        13115 non-null   float64 
 4   Bedrooms         13115 non-null   float64 
 5   Beds              13115 non-null   float64 
 6   Guests Included  13115 non-null   int64  
 7   Extra People     13115 non-null   int64  
 8   Minimum Nights   13115 non-null   int64  
 9   Availability 30  13115 non-null   int64  
 10  Availability 365 13115 non-null   int64  
 11  Number of Reviews 13115 non-null   int64  
 12  Review Scores Rating 10315 non-null   float64 
 13  Review Scores Cleanliness 10303 non-null   float64 
 14  Review Scores Checkin 10287 non-null   float64 
 15  Review Scores Communication 10303 non-null   float64 
 16  Review Scores Location 10285 non-null   float64 
 17  Reviews per Month 10439 non-null   float64 
 18  Security Deposit 13115 non-null   float64 
 19  Cleaning Fee     13115 non-null   float64 
 20  Calculated host listings count 13115 non-null   float64 
 21  Num_Host_Verifications 13115 non-null   int64  
 22  Name_Len          13115 non-null   int64  
 23  Summary_Len       13115 non-null   int64  
 24  Space_Len          13115 non-null   int64  
 25  Description_Len   13115 non-null   int64  
 26  Summary_Lang      13115 non-null   bool   
 27  is_Thumbnail      13115 non-null   bool   
 28  is_HostAbout      13115 non-null   bool   
 29  is_ResponseInHours 13115 non-null   bool   
 30  is_EntireHome     13115 non-null   bool   
 31  is_Bed             13115 non-null   bool   
 32  is_UpdatedToday    13115 non-null   bool   
 33  Neighbourhood_Barajas 13024 non-null   float64 
 34  Neighbourhood_Carabanchel 13024 non-null   float64 
 35  Neighbourhood_Centro    13024 non-null   float64 
 36  Neighbourhood_Chamartin 13024 non-null   float64 
 37  Neighbourhood_Chamberi 13024 non-null   float64 
 38  Neighbourhood_CiudadLineal 13024 non-null   float64 
 39  Neighbourhood_Fuencarral 13024 non-null   float64 
 40  Neighbourhood_Hortaleza 13024 non-null   float64 
 41  Neighbourhood_Latina    13024 non-null   float64 
 42  Neighbourhood_Moncloa   13024 non-null   float64
```

```

43 Neighbourhood_Moratalaz      13024 non-null float64
44 Neighbourhood_PuenteVallecas  13024 non-null float64
45 Neighbourhood_Retiro         13024 non-null float64
46 Neighbourhood_Salamanca     13024 non-null float64
47 Neighbourhood_SanBlas        13024 non-null float64
48 Neighbourhood_Tetuán        13024 non-null float64
49 Neighbourhood_Usera         13024 non-null float64
50 Neighbourhood_Vicalvaro     13024 non-null float64
51 Neighbourhood_VillaVallecas 13024 non-null float64
52 Neighbourhood_Villaverde    13024 non-null float64
53 PropertyType_BedAndBreakfast 13024 non-null float64
54 PropertyType_Condominium     13024 non-null float64
55 PropertyType_House          13024 non-null float64
56 PropertyType_Loft            13024 non-null float64
57 PropertyType_Other          13024 non-null float64
58 BedType_PrivateRoom         13024 non-null float64
59 BedType_SharedRoom          13024 non-null float64
60 Cancellation_moderate      13024 non-null float64
61 Cancellation_strict         13024 non-null float64
62 Price                         13115 non-null float64
dtypes: bool(7), float64(44), int64(12)
memory usage: 5.7 MB

```

Train Test Split:

Separamos el dataset entre Train 80% y Test 20%

Imputar los NaN:

Imputamos los NaNs en las columnas de las Reviews, con la mediana del Train de cada columna en los NaN de esa columna tanto en Train como en Test.

Algoritmo:

Utilizamos el algoritmo LinearRegression de los linear models de sklearn

Métricas:

Como métricas de error utilizamos el MAE, el RMSE y el R2

Cross Validation:

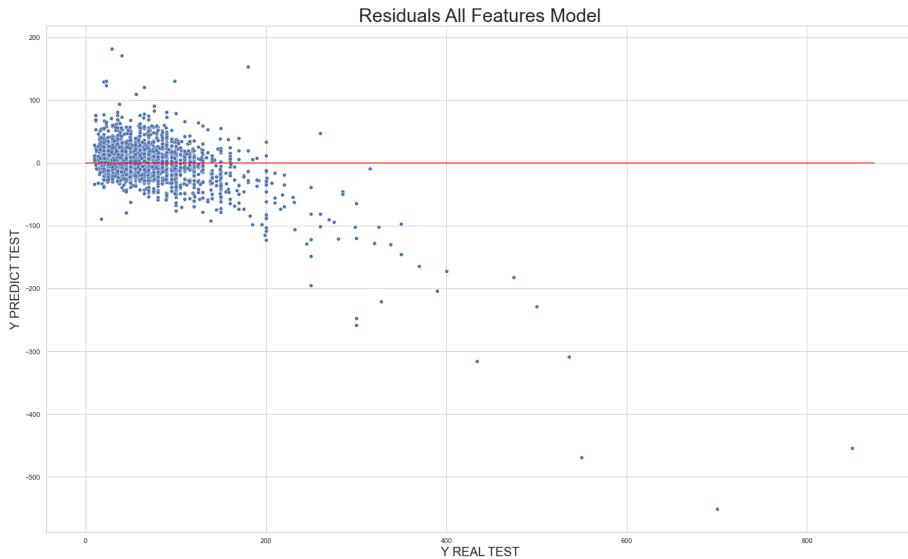
También utilizaremos la cross validation, para que el resultado no dependa del split.

Resultados:

Model	MAE	RMSE	R²
Train Modelo All Features	21,51	38,68	0,534
Test Modelo All Features	20,69	35,32	0,56
Cross Validation All Features	21,79	39,06	0,520
Cross Validation Numérico	21,64	38,98	0,520
Cross Validation Base Model	23,57	41,25	0,470

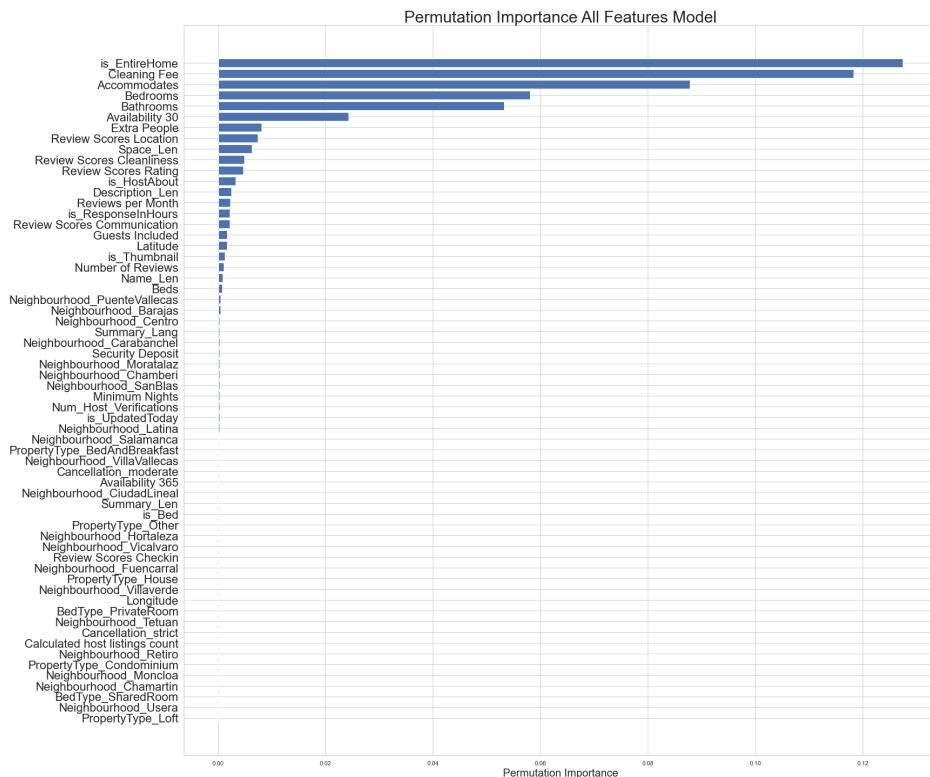
Residuales:

Vemos que los residuales aumentan a medida que aumenta el precio a predecir.



Permutation Importance:

En LinearRegression de sklearn no tenemos las feature importance como en otros modelos, pero si disponemos de la permutation importance, que nos dice cuánto del modelo depende de dicha feature.



Vemos que la Feature que hemos creado is_entireHome sigue influyendo mucho en el modelo. En cambio las codificadas, parece que no influyen tanto.

MODELO NUMÉRICO CON TARGET TRANSFORMADA

Como el modelo numérico de momento es el que mejores resultados nos ha dado, probaremos dicho modelo transformando la target, en vez de predecir 'Price', intentamos predecir log('Price').

Como en los anteriores modelos, utilizamos train test split 80/20, la imputación de NaN con la media de train y la cross validation. Y mantenemos las métricas MAE, RMSE y R² y el algoritmo utilizado es Linear Regression de sklearn.

Target Transformation

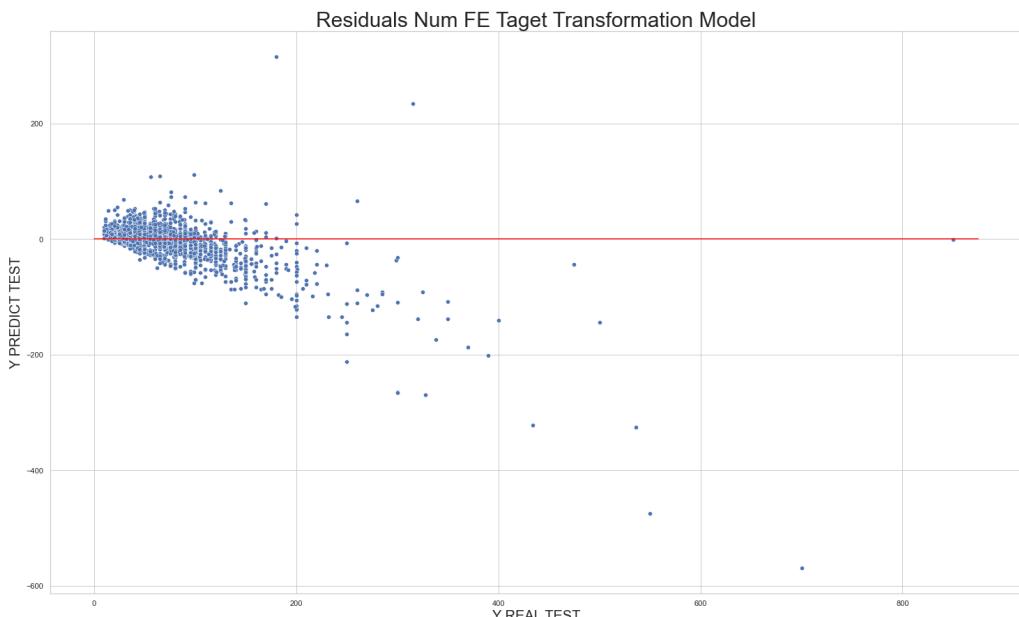
Transformamos la target con el log(target).

Resultados:

Model	MAE	RMSE	R²
Train Modelo All Features	18,83	41,30	0,469
Test Modelo All Features	17,48	33,15	0,612
Cross Validation Num log(target)	18,92	41,41	0,450
Cross Validation All Features	21,79	39,06	0,520
Cross Validation Numérico	21,64	38,98	0,520
Cross Validation Base Model	23,57	41,25	0,470

Residuales:

Vemos que los residuales aumentan a medida que aumenta el precio a predecir.



MODELO NUMÉRICO CON POLYNOMIAL FEATURES

Como el modelo numérico de momento es el que mejores resultados nos ha dado, probaremos dicho modelo transformando la features con polynomial features, de grado 2 para predecir Price.

Como en los anteriores modelos, utilizamos train test split 80/20, la imputación de NaN con la media de train y la cross validation. Y mantenemos las métricas MAE, RMSE y R2 y el algoritmo utilizado es Linear Regression de sklearn.

Polynomial Features

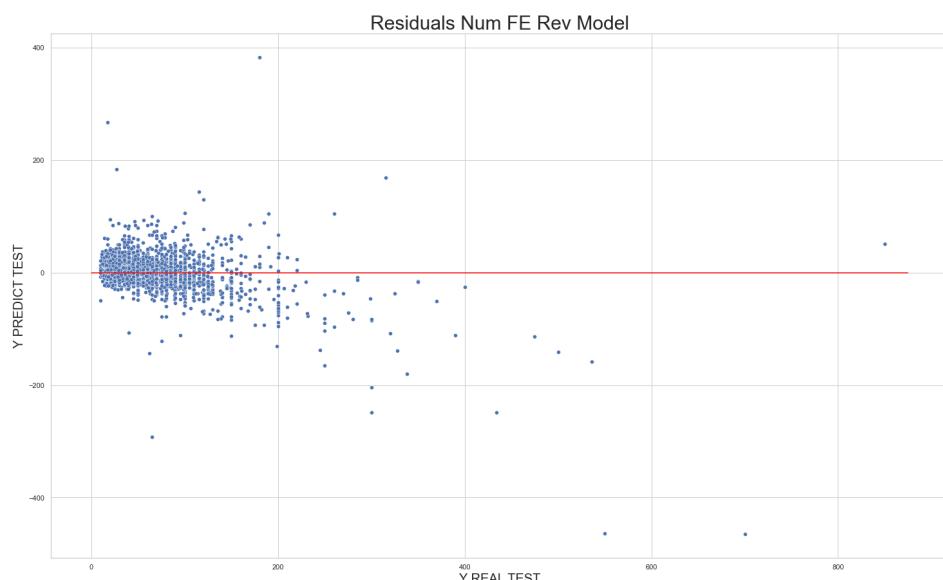
Creamos nuevas features con una Polynomial Features de grado 2.

Resultados:

Model	MAE	RMSE	R²
Train Modelo All Features	17,83	31,34	0,694
Test Modelo All Features	18,76	32,26	0,633
Cross Validation Polynomial F	19,97	36,48	0,58
Cross Validation Num log(target)	18,92	41,41	0,450
Cross Validation All Features	21,79	39,06	0,520
Cross Validation Numérico	21,64	38,98	0,520
Cross Validation Base Model	23,57	41,25	0,470

Residuales:

Estos residuales, parecen más centrados en el 0.



MODELO NUMÉRICO SIMPLIFICADO

Hasta ahora cada vez hemos creado modelos más complejos, para el fin que buscamos, que un usuario tenga una idea de que precio poner dado un alojamiento, creemos que mejor realizar un modelo menos complejo, así que hemos retirado las features que menos aportan al modelo, así como todas las de reviews que no dependen del propio alojamiento. Quedando en el dataset final.

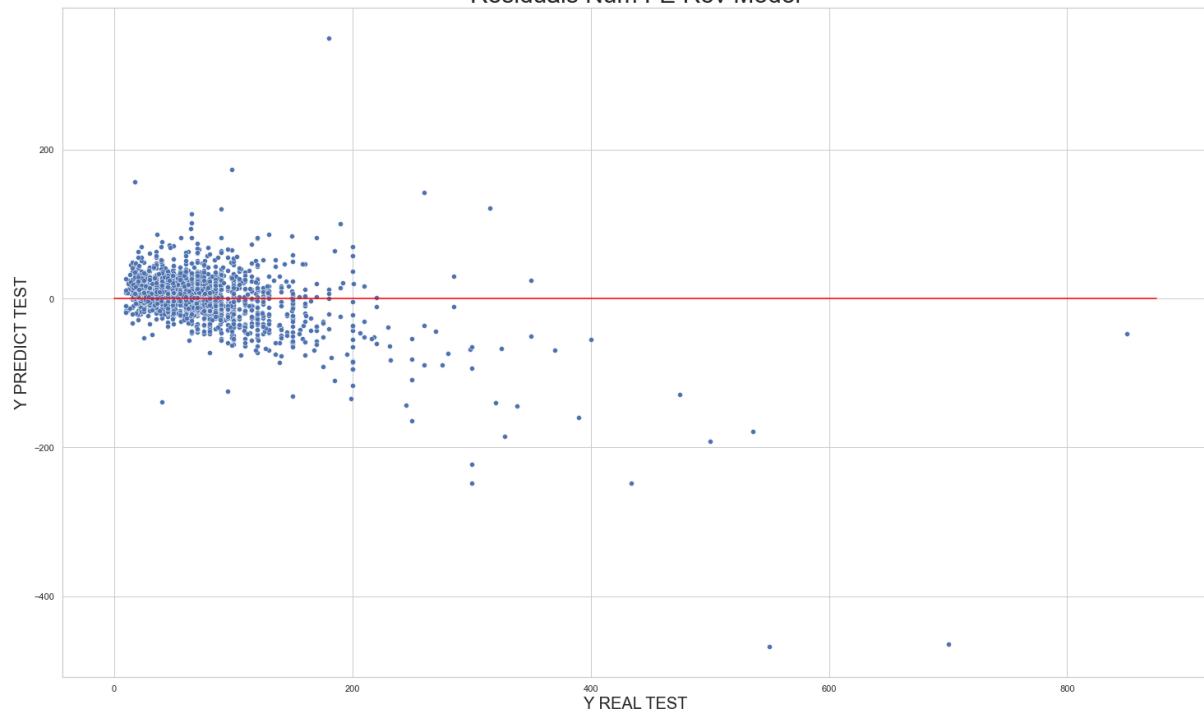
- Latitud
- Accomodates
- Bathrooms
- Bedrooms
- Beds
- Guests Included
- Extra People
- Availability 30
- Cleaning Fee
- Name Len
- Space Len
- is_thumbnail
- is_hostabout
- is_responseinhours
- is_entireHome
- Price

Como en los anteriores modelos, utilizamos train test split 80/20, el polynomial feature y la cross validation. Y mantenemos las métricas MAE, RMSE y R2 y el algoritmo utilizado es Linear Regression de sklearn.

Resultados:

Model	MAE	RMSE	R²
Train Modelo All Features	19,13	33,93	0,642
Test Modelo All Features	18,97	31,72	0,645
Cross Validation Num reduced	21,67	38,77	0,520
Cross Validation Num log(target)	18,92	41,41	0,450
Cross Validation All Features	21,79	39,06	0,520
Cross Validation Numérico	21,64	38,98	0,520
Cross Validation Base Model	23,57	41,25	0,470

Residuals Num FE Rev Model



6. Conclusiones

En este punto del proyecto, hemos creado un Data Warehouse con los datos, analizado las variables disponibles, creando las visualizaciones necesarias para entender las variables tanto en sí mismas como relacionadas con las otras. Además de un dashboard que nos permite visualizar los KPIs y filtrar por diversos parámetros.

Además, hemos creado algunas variables adicionales desde las disponibles para ayudarnos en la predicción. Finalmente, hemos realizado varios modelos de predicción del Precio con una Regresión Lineal, quedándonos con un modelo que a su vez nos da un buen resultado y no tiene una gran complejidad.

A partir de aquí, algunas cosas a realizar en el futuro podrían ser:

- Incorporar nuevas columnas a partir de las que tenemos, por ejemplo analizar lo que aparece en las fotos a partir de la url de la misma
- Incorporar nuevas columnas externas, por ejemplo, la distancia al metro más cercano desde el alojamiento, distancia a los principales museos o atracciones de Madrid (estadios de fútbol, parques de atracciones) desde el alojamiento.
- Crear modelos con otros algoritmos que se adapten mejor a la dispersión de precios: Ridge, Lasso, Elastic Net, RandomForest Regressor, SVR con parameter hypertunning.

Con este modelo mejorado, se podría crear una web accesible para todos los futuros anfitriones y posibles clientes donde pudiesen ver cómo estimar el precio de una noche en el apartamento en qué viven.

De esta manera, ofrecemos un servicio de valor añadido que permitirá a los propietarios sacar el máximo rendimiento a sus propiedades o estancias en una de las webs de referencia para el alquiler vacacional de inmuebles.