**TRABAJO FINAL**

**BOOTCAMP - IV - KEEPCODING**

**BIG DATA & MACHINE LEARNING**

# **Índice de Contenidos**

1. Índice de Contenidos · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 1
2. Datos y Propósito de este Trabajo Final · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 3
3. Definición del Proyecto · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 4
4. Dataset de Trabajo · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 4
   1. Estructura del Dataset · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 5
   2. Selección Inicial de Filas y Columnas · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 5
5. Preparación de los Datos · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 8
   1. Partición de Conjuntos de Training y Test · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 8
6. Análisis del Subset de Entrenamiento · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 8
   1. Columna Experiences Offered · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 9
   2. Columna Neighbourhood Group Cleansed · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 9
   3. Columna Neighbourhood Cleansed · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 10
   4. Columna City · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 11
   5. Columna Property Type · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 12
   6. Columna Room Type · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 13
   7. Columna Accommodates · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 14
   8. Columna Bathrooms · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 14
   9. Columna Bedrooms · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 15
   10. Columna Beds · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 16
   11. Columna Bed Type · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 17
   12. Columna Amenities · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 18
   13. Columna Square Feet · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 19
   14. Columna Price · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 19
   15. Columnas Weekly Price y Monthly Price · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 20
   16. Columna Security Deposit · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 20
   17. Columna Cleaning Fee · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 21
   18. Columna Guests Included · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 22
   19. Columna Extra People · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 22
   20. Columna Minimum Nights · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 23
   21. Columna Maximum Nights · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 24
   22. Columna License · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 24
   23. Columna Cancellation Policy · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 25
   24. Columna Reviews per Month · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 25
7. Exploración del Subset de Entrenamiento · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 26
8. Columna Objetivo · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 30
9. Reduciendo la Dimensionalidad · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 32
   1. Modelos f y mutual\_info · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 33
   2. Modelo Lasso · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 33
   3. Modelo DecissionTree · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 35
   4. Modelo RandomForest · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 36
   5. Modelo GradientBoosting · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 38
   6. Maldición de la Dimensionalidad · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 40
   7. Tratamiento de Outliers · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 41
10. Proceso del Conjunto de Test · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 41
11. Modelos Machine Learning para Clasificación · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 44
    1. Modelo GradientBoosting · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 44
    2. Modelo SVM · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 47
    3. Modelo XGBoost · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 48
12. El Modelo Clasificador Elegido · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 51
    1. El Modelo Prediciendo · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 51
    2. Portabilidad del Modelo · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 52
    3. Módulo de Ejecución · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 52
13. Presentación · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 56
    1. Gráfica de Inmuebles en Zona Cliente · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 57
    2. Gráfica de Éxito en el Alquiler · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 59
    3. Dashboard · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 60
    4. Un Ejemplo en Barcelona · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - 61
14. Comentario Final · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 62

ANEXO · - ·· - · - ·· - · - ·· - · - ·· - · - ·· - · - ·· 62

# **Datos y Propósito de este Trabajo Final**

|  |  |
| --- | --- |
| **Bootcamp IV Big Data & Machine Learning - KeepCoding** | |
| **Alumno** | Francisco Javier Gonzálvez Chico |
| **Título** | Recomendador de Precios para Anfitriones de AirBNB España |
| ***NOTA***: *En el ANEXO se incluyen los Notebooks conteniendo todas las celdas de código ejecutadas durante la práctica.*  *Asimismo, se adjuntará el modelo clasificador y el módulo de presentación.* | |

El objetivo principal de este trabajo, obviamente, es servir como colofón al bootcamp que he estado cursando durante estos últimos nueve meses.

Durante la tutoría orientativa para la confección del trabajo fin de bootcamp, se nos habló sobre realizar algún Proyecto o Prueba de Concepto (PoC) sobre alguna idea que pudiera ser de utilidad a empresas, algo que quizá pudiera llevarse a cabo en el mundo real.

Dándole vueltas a qué utilidad podría extraer de los datos de inmuebles de AirBNB se me ocurrió pensar en qué me haría falta a mí como cliente… entré en su web y estuve un rato buscando alojamientos en Madrid, Barcelona, Londres… y no se me ocurría nada que pudiera mejorar la experiencia.

Pero después me pregunté: ¿Quiénes son los verdaderos clientes de AirBNB? ¿Quiénes hacen posible que AirBNB pueda monetizar una página web que básicamente es un buscador de hospedajes por ciudad? La respuesta fue: los propietarios de los inmuebles. Los propietarios que quieren hospedar personas (y ganar algo de dinero) son los que ponen a disposición de AirBNB los que son sus principales activos, los inmuebles de su base de datos.

Acto seguido intenté ponerme en su piel y la siguiente cuestión que me vino a la cabeza fue: si yo quisiera alquilar mi casa, ¿por cuánto dinero la alquilaría? Y no me supe responder con total certeza.

Está claro que si elijo un precio que sea demasiado alto, mi oferta no será lo suficientemente interesante y no tendré un buen número de reservas/visitas de huéspedes por lo que mi monetización será baja y el negocio poco rentable. Por el contrario, si el precio al que alquilo la noche es muy barato, probablemente obtenga una alta demanda, lo cual derivaría en la obligación de dedicar mucho tiempo y recursos para mantener en buenas condiciones mi inmueble y en consecuencia el balance ingresos/gastos resultaría también en una monetización con poco margen de rentabilidad.

¿Y si hubiera algún sistema que me pudiera orientar, en función de las características de mi inmueble y de su ubicación, sobre cuál sería el precio más adecuado para su alquiler?

Y entonces llegué a la conclusión: ese sería mi trabajo final para este bootcamp.

La tarea que pretendo llevar a cabo en este trabajo será la implementación de un modelo recomendador de precios que pueda orientar a los propietarios de los hospedajes anunciados en AirBNB (o en cualquier otra plataforma intermediaria entre hospedantes y huéspedes).

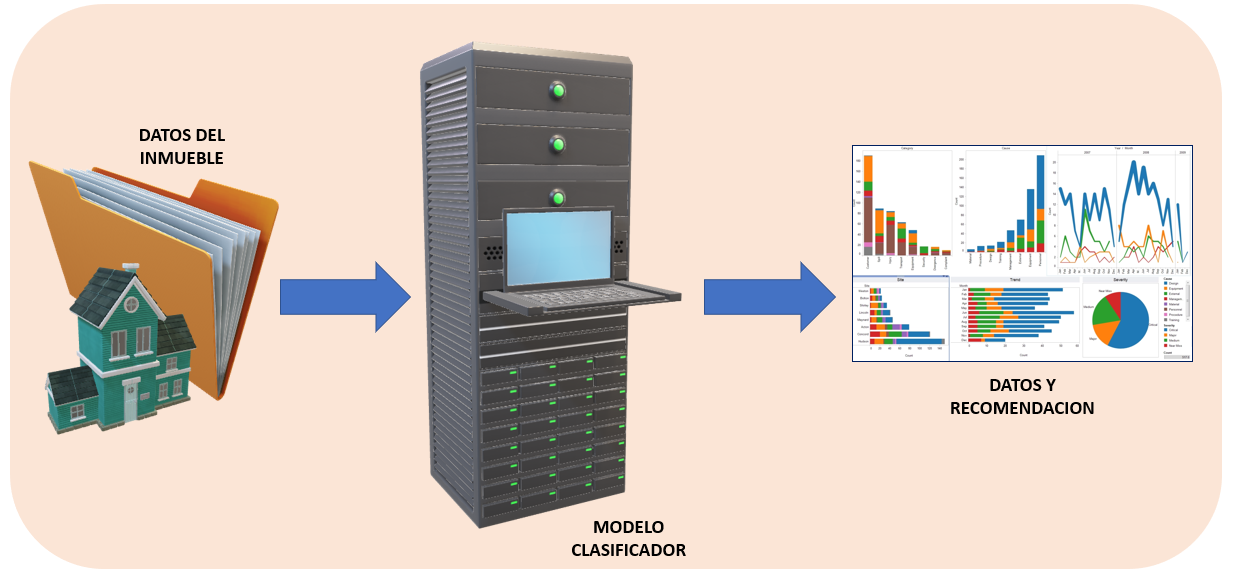
Este modelo recomendador podría ser un servicio que la plataforma ponga a disposición de sus clientes hospedantes como ayuda y orientación. Y podría tener un buen ROI ya que, si los precios de alquiler de todos sus inmuebles son más adecuados que los de otras plataformas, el número de contrataciones debería aumentar.

Implementaré algunos modelos de Machine Learning con el objetivo de llegar a proporcionar un rango de precios óptimo para las determinadas características y ubicación de los inmuebles.

Como PoC restringiré el modelo a las ubicaciones de Madrid y Barcelona por dos razones principalmente: primero y por causa evidente, por coste computacional y segundo por ser las dos principales ciudades de España.

# **Definición del Proyecto**

El proyecto se va a dividir en 2 zonas:

1. Un modelo clasificador que, a partir de los datos del inmueble introducidos por el hospedante en la web de la plataforma (AirBNB), obtendrá un rango óptimo para el precio por noche del inmueble.
2. La parte de presentación que, a partir de los datos del inmueble introducidos por el hospedante en la web de la plataforma (AirBNB) y del rango de precios obtenido por el modelo clasificador, proporcione los rangos de éxito de los inmuebles, análogos al del hospedante, en el rango de precios obtenido por el modelo clasificador y en los rangos adyacentes.

# **Dataset de Trabajo**

Utilizaré como datos aprovisionadores, los de un fichero de **AIRBNB** de obtención y uso libre.

Los descargaré manualmente desde este enlace web:

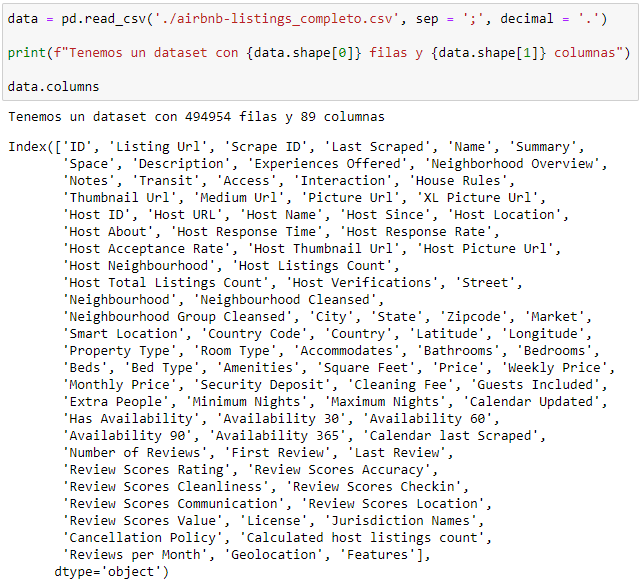
https://public.opendatasoft.com/explore/dataset/airbnb-listings/export/?disjunctive.host\_verifications&disjunctive.amenities&disjunctive.features&dataChart=%3D&location=16,41.38377,2.15774&basemap=jawg.streets

La opción de descarga será: formato plano CSV.

Según la página de descarga, el dataset contiene 494.954 (casi medio millón de) filas, lo cual exigiría un coste computacional bastante grande trabajando localmente o en máquinas compartidas de Google. Y por ello he decidido restringir los datos que utilizaremos a los de las dos mayores ciudades de nuestro país: Madrid y Barcelona.

Trabajaré este dataset en cuadernos **JupyterNotebook** sobre los que iré ejecutando celdas con el código oportuno en cada momento. Celdas que serán copiadas y aportadas en los distintos párrafos de este documento.

## **Estructura del Dataset**

Con el dataset de trabajo descargado en local, lo cargo en un **JupyterNotebook** con python/pandas.

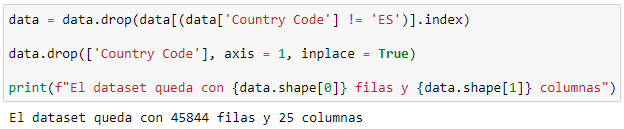
## **Selección Inicial de Filas y Columnas**

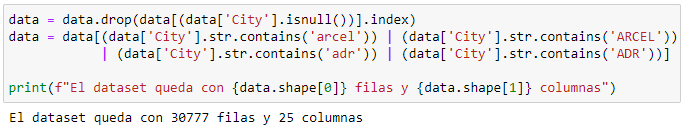
Para tener una primera toma de contacto con los datos, un vistazo inicial al dataset para identificar el contenido de las columnas y desechar las que no parezcan aprovechables para el trabajo.

| **COLUMNA** | **DESCRIPCIÓN DE CONTENIDO** | **TIPO** | **ACCIÓN** |
| --- | --- | --- | --- |
| ID | Código identificador del inmueble en AirBNB | Entero | Descarte |
| Listing Url | URL. Enlace al inmueble en AirBNB | Cadena | Descarte |
| Scrape ID | Timestamp de scrapeo del registro del inmueble | Entero | Descarte |
| Last Scraped | Fecha. Último scrapeo del registro del inmueble | Fecha | Descarte |
| Name | Texto libre. Nombre del inmueble | Cadena | Descarte |
| Summary | Texto libre. Resumen descriptivo del inmueble | Cadena | Descarte |
| Space | Texto libre. Descripción del entorno del inmueble | Cadena | Descarte |
| Description | Texto libre. Descripción detallada del inmueble | Cadena | Descarte |
| Experiences Offered | Tipo de experiencia o adecuada para el inmueble | Cadena | Se queda |
| Neighborhood Overview | Texto libre. Descripción del barrio de ubicación del inmueble | Cadena | Descarte |
| Notes | Texto libre. Notas adicionales sobre el inmueble | Cadena | Descarte |
| Transit | Texto libre. Medios de transporte cerca del inmueble | Cadena | Descarte |
| Access | Texto libre. Acceso al inmueble | Cadena | Descarte |
| Interaction | Texto libre. Interacción con el anfitrión | Cadena | Descarte |
| House Rules | Texto libre. Reglas del inmueble | Cadena | Descarte |
| Thumbnail Url | URL. Enlace a foto tamaño pequeño | Cadena | Descarte |
| Medium Url | URL. Enlace a foto tamaño mediano | Cadena | Descarte |
| Picture Url | URL. Enlace a foto tamaño grande | Cadena | Descarte |
| XL Picture Url | URL. Enlace a foto tamaño extra grande | Cadena | Descarte |
| Host ID | Código identificador del anfitrión en AirBNB | Entero | Descarte |
| Host URL | URL. Enlace al anfitrión en AirBNB | Cadena | Descarte |
| Host Name | Texto libre. Nombre del anfitrión | Cadena | Descarte |
| Host Since | Fecha. Registro del anfitrión en AirBNB | Fecha | Descarte |
| Host Location | Texto libre. Ubicación del anfitrión | Cadena | Descarte |
| Host About | Texto libre. Acerca del anfitrión | Cadena | Descarte |
| Host Response Time | Tasa temporal de respuesta del anfitrión | Cadena | Descarte |
| Host Response Rate | Tasa de respuesta del anfitrión | Entero | Descarte |
| Host Acceptance Rate | Tasa de aceptación de huéspedes del anfitrión | Cadena | Descarte |
| Host Thumbnail Url | URL. Enlace a fotografía tamaño pequeño del anfitrión | Cadena | Descarte |
| Host Picture Url | URL. Enlace a fotografía del anfitrión | Cadena | Descarte |
| Host Neighbourhood | Vecindario o barrio del anfitrión | Cadena | Descarte |
| Host Listings Count | Conteo de veces que se selecciona al anfitrión | Entero | Descarte |
| Host Total Listings Count | Cuenta total de veces que se selecciona al anfitrión | Entero | Descarte |
| Host Verifications | Verificaciones del anfitrión | Cadena | Descarte |
| Street | Texto libre. Dirección del inmueble | Cadena | Descarte |
| Neighbourhood | Texto libre. Vecindario o barrio en el que se ubica el inmueble | Cadena | Descarte |
| Neighbourhood Cleansed | Vecindario o barrio limpio en el que se ubica el inmueble | Cadena | Se queda |
| Neighbourhood Group Cleansed | Distrito en el que se ubica el inmueble | Cadena | Se queda |
| City | Ciudad en la que se ubica el inmueble | Cadena | Se queda |
| State | Provincia o estado en el que se ubica el inmueble | Cadena | Descarte |
| Zipcode | Código postal en el que se ubica el inmueble | Cadena | Descarte |
| Market | Mercado de alquiler del inmueble | Cadena | Descarte |
| Smart Location | Ubicación genérica del inmueble | Cadena | Descarte |
| Country Code | Código identificador del país en el que se ubica el inmueble | Cadena | Se queda |
| Country | País en el que se ubica el inmueble | Cadena | Descarte |
| Latitude | Coordenada geoposicional de latitud de ubicación del inmueble | Cadena | Descarte |
| Longitude | Coordenada geoposicional de longitud de ubicación del inmueble | Cadena | Descarte |
| Property Type | Tipo de inmueble | Cadena | Se queda |
| Room Type | Tipo de alquiler del inmueble | Cadena | Se queda |
| Accommodates | Cupo de inquilinos incluido en el alquiler del inmueble | Entero | Se queda |
| Bathrooms | Número de baños en el inmueble | Decimal | Se queda |
| Bedrooms | Número de habitaciones en el inmueble | Entero | Se queda |
| Beds | Número de camas en el inmueble | Entero | Se queda |
| Bed Type | Tipo de camas en el inmueble | Cadena | Se queda |
| Amenities | Comodidades del inmueble | Cadena | Se queda |
| Square Feet | Superficie del inmueble en pies cuadrados | Entero | Se queda |
| Price | Precio en euros de alquiler por día del inmueble | Entero | Se queda |
| Weekly Price | Precio en euros de alquiler por semana del inmueble | Entero | Se queda |
| Monthly Price | Precio en euros de alquiler por mes del inmueble | Entero | Se queda |
| Security Deposit | Depósito o fianza en euros para el alquiler del inmueble | Entero | Se queda |
| Cleaning Fee | Recargo o tasa en euros por la limpieza del inmueble | Entero | Se queda |
| Guests Included | Número máximo de invitados permitido en el inmueble | Entero | Se queda |
| Extra People | Recargo en euros por invitado | Entero | Se queda |
| Minimum Nights | Número mínimo de noches a alquilar el inmueble | Entero | Se queda |
| Maximum Nights | Número máximo de noches a alquilar el inmueble | Entero | Se queda |
| Calendar Updated | Calendario de disponibilidad del inmueble actualizado en AirBNB | Cadena | Descarte |
| Has Availability | Disponibilidad actual del inmueble | Cadena | Descarte |
| Availability 30 | Disponibilidad a 30 días del inmueble | Entero | Descarte |
| Availability 60 | Disponibilidad a 60 días del inmueble | Entero | Descarte |
| Availability 90 | Disponibilidad a 90 días del inmueble | Entero | Descarte |
| Availability 365 | Disponibilidad a 365 días del inmueble | Entero | Descarte |
| Calendar last Scraped | Fecha. Último scrapeo del calendario de disponibilidad del inmueble | Fecha | Descarte |
| Number of Reviews | Número de comentarios sobre el inmueble | Entero | Descarte |
| First Review | Fecha. Primer comentario sobre el inmueble | Fecha | Descarte |
| Last Review | Fecha. Último comentario sobre el inmueble | Fecha | Descarte |
| Review Scores Rating | Tasa de puntuación de los comentarios del inmueble | Entero | Descarte |
| Review Scores Accuracy | Puntuación sobre la veracidad de la información del inmueble | Entero | Descarte |
| Review Scores Cleanliness | Puntuación sobre la limpieza de los comentarios del inmueble | Entero | Descarte |
| Review Scores Checkin | Puntuación sobre el recibimiento de los comentarios del inmueble | Entero | Descarte |
| Review Scores Communication | Puntuación sobre la comunicación de los comentarios del inmueble | Entero | Descarte |
| Review Scores Location | Puntuación sobre la localización de los comentarios del inmueble | Entero | Descarte |
| Review Scores Value | Puntuación general de los comentarios del inmueble | Entero | Descarte |
| License | Código identificador de licencia | Cadena | Se queda |
| Jurisdiction Names | Jurisdicción bajo la que se ampara el alquiler del inmueble | Cadena | Descarte |
| Cancellation Policy | Política de cancelación de reservas del inmueble | Cadena | Se queda |
| Calculated host listings count | Cuenta calculada de veces que se selecciona al anfitrión | Entero | Descarte |
| Reviews per Month | Comentarios sobre el inmueble por mes | Decimal | Se queda |
| Geolocation | Coordenadas de geolocalización de la ubicación del inmueble | Cadena | Descarte |
| Features | Características del anfitrión y del inmueble | Cadena | Descarte |

En esta selección inicial he decidido mantener 26 columnas del dataset y, por tanto, descartar 63.

He descartado todas las columnas que contienen códigos identificadores propios de AirBNB, textos libres, URLs y datos propios de los perfiles de los anfitriones, ya que no tendrán valor para el modelo que implementaré. Asimismo, he desechado columnas con datos de ubicación de los inmuebles cuya información he creído redundante, o con formato poco adecuado. También prescindo de las columnas con información sobre opiniones y sobre la disponibilidad de los inmuebles porque tampoco aportarán valor al modelo. Básicamente selecciono los datos que se podrían tomar del proceso de alta del hospedaje (introducidos por el anfitrión) y la columna **Reviews per Month** que tomaré como referencia del éxito del alquiler del inmueble.

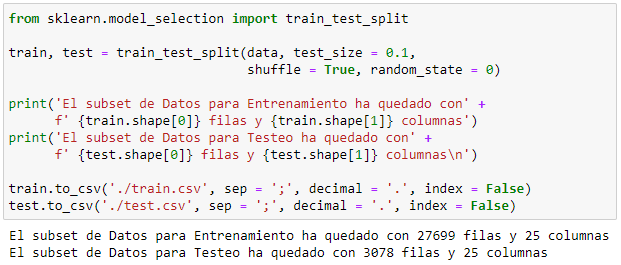
Reduzco el dataset a los registros de España y elimino la columna **Country Code**.

Reduzco de nuevo a los registros de Madrid y Barcelona.

# **Preparación de los Datos**

## **Partición de Conjuntos de Training y Test**

Previamente a la elaboración de los modelos, atendiendo a las buenas prácticas, particionaré el dataset.

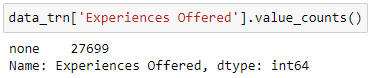
Como interesa tener un conjunto de datos de entrenamiento grande, dejaré para el testeo de los modelos únicamente el 10% de los registros. Para ello utilizo el método **train\_test\_split** de la librería **model\_selection** de **scikit-learn**.

Una vez partido el dataset de trabajo y obtenidos los subsets de datos de entrenamiento y testeo los guardo en dos archivos CSV y, desde este momento, la partición de test quedará apartada y aislada de la tarea de análisis hasta que le llegue su turno de uso.

# **Análisis del Subset de Datos de Entrenamiento**

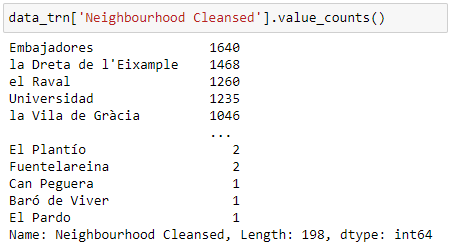
****Analizaré detalladamente el subset de datos de entrenamiento, columna a columna, aplicándoles los tratamientos que vaya decidiendo como adecuados. Pero lo primero es ver qué columnas contienen valores nulos (missings), ya que esto condicionará también de cierta forma su análisis.

## **Columna Experiences Offered**

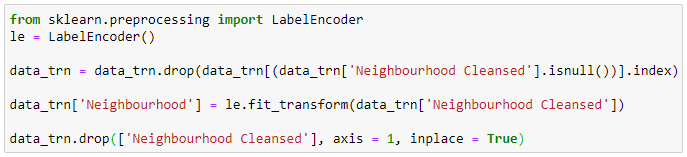
Esta columna indica el tipo de experiencia que ofrece el inmueble. Veamos que valores contiene.

En nuestra selección esta columna sólo contiene el valor **none**. La descarto por no aportar valor predictivo.

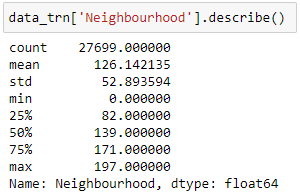
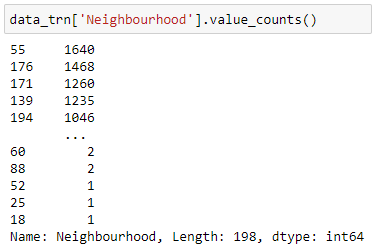
## **Columna Neighbourhood Cleansed**

Esta columna indica el barrio o vecindario en el que se ubica el inmueble. Veamos que valores contiene.

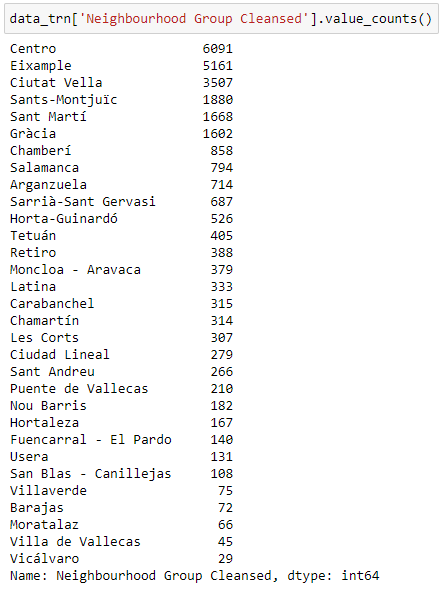
Esta columna tiene 198 valores únicos (\*) y no contiene nulos. Como es un número bastante alto de valores alfanuméricos únicos, opto por categorizarla con el método **LabelEncoder** de la librería **preprocessing** de **scikit-learn**.

Realizo un tratamiento preventivo para eliminar posibles futuros valores nulos en la columna, aprovecho el mapeo para crear la columna renombrada y termino descartando la columna original.

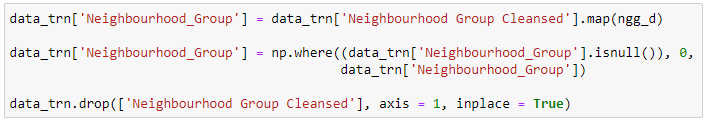
Veamos el aspecto de la nueva columna categorizada numéricamente al libre albedrío del método.



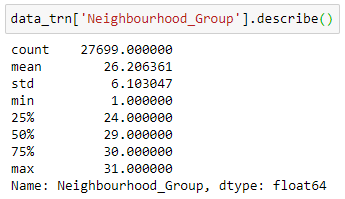
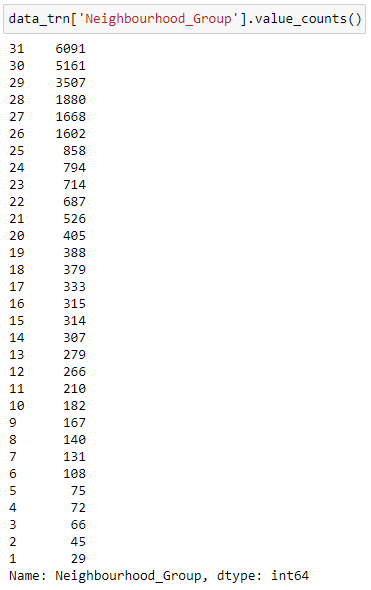
## **Columna Neighbourhood Group Cleansed**

Esta columna indica el distrito en el que se ubica el inmueble. Veamos que valores contiene.

La columna no contiene valores nulos. Al ser una columna alfanumérica debo categorizarla y para ello la mapearé de forma directamente proporcional al número de inmuebles que contiene, dando más valor a los distritos con más inmuebles.

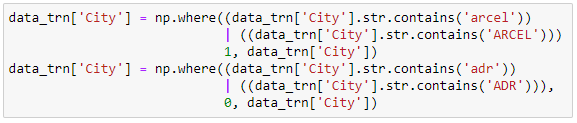
Aprovecho el mapeo para crear la columna renombrada, después realizo un tratamiento preventivo de posibles futuros valores nulos, asignándolos al valor **0** y termino descartando la columna original.

Veamos el aspecto de la nueva columna categorizada numéricamente.

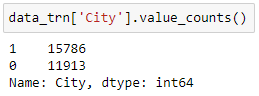


## **Columna City**

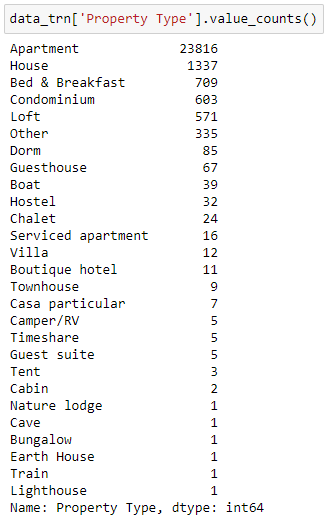
Esta columna indica la ciudad en la que se ubica el inmueble. En la selección previa quise incluir los registros del dataset de Madrid y Barcelona, pero lo hice con un operador de aproximación. Veamos que valores contiene realmente.

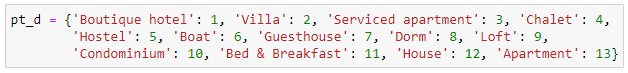
Aplicaré un tratamiento de limpieza para adecuar la columna a los dos valores válidos, aprovechando para categorizar directamente la variable.

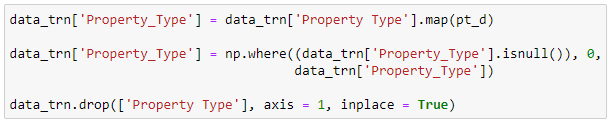
La columna ha quedado con dos valores (**0** y **1**, correspondiendo a Madrid y Barcelona).



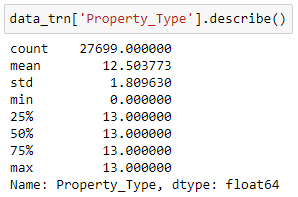
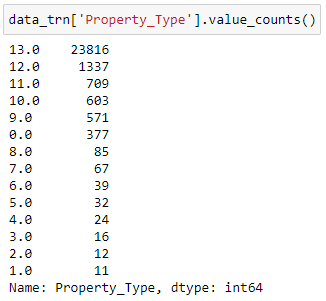
## **Columna Property Type**

Esta columna indica el tipo del inmueble.

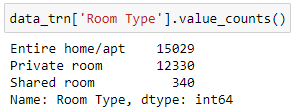
La columna no contiene valores nulos. Al ser una columna alfanumérica debo categorizarla y para ello la mapearé de forma directamente proporcional al número de inmuebles que contiene, dando más valor a los tipos con más inmuebles. Asimilaré los tipos minoritarios –con presencia menor a 10– al valor **Other** para reducir el número de clases de la columna.

Aprovecho el mapeo para crear la columna renombrada, después realizo el tratamiento de los valores no mapeados –que quedan nulos– asimilándolos al valor **0.0** y termino descartando la columna original.

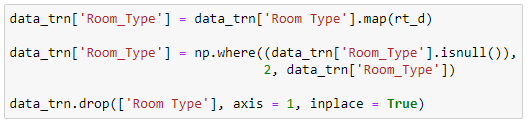
Veamos el aspecto de la nueva columna categorizada numéricamente.



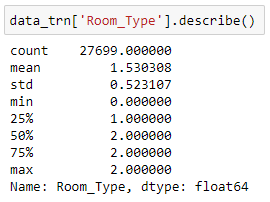
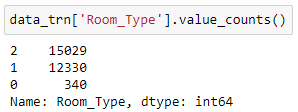
## **Columna Room Type**

Esta columna indica el tipo de alquiler ofertado por el inmueble.

La columna no contiene valores nulos. Al ser una columna alfanumérica debo categorizarla y para ello la mapearé de forma directamente proporcional al número de inmuebles que contiene, dando más valor a los tipos de alquiler con más inmuebles.

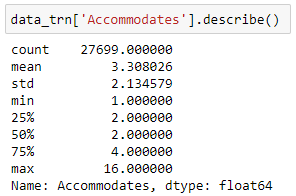
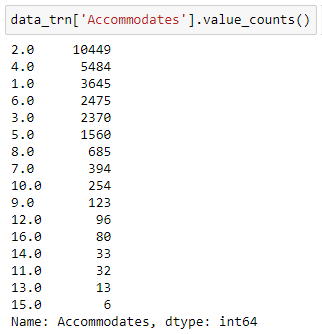
Aprovecho el mapeo para crear la columna renombrada, después realizo un tratamiento preventivo de posibles futuros valores nulos, asimilándolos al valor mayoritario y termino descartando la columna original.

Veamos el aspecto de la nueva columna categorizada numéricamente.



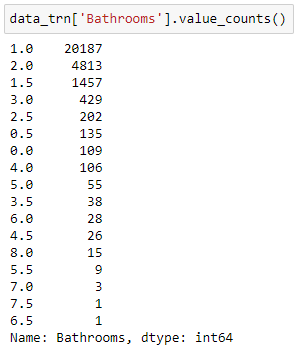
## **Columna Accommodates**

Esta columna indica el cupo de personas incluidas en el precio por noche de alquiler de un inmueble.

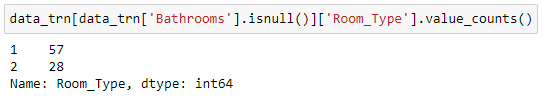


La columna es numérica por lo que no es necesaria su categorización. Y no contiene valores nulos, pero aplicaré un tratamiento preventivo por si en un futuro hubiera, asimilándolos al valor mayoritario.

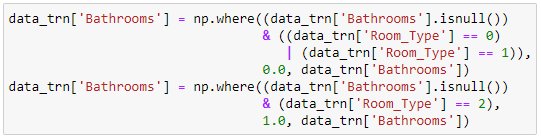
## **Columna Bathrooms**

Esta columna indica el número de cuartos de baño de cada inmueble.

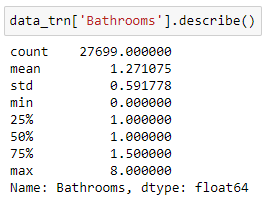
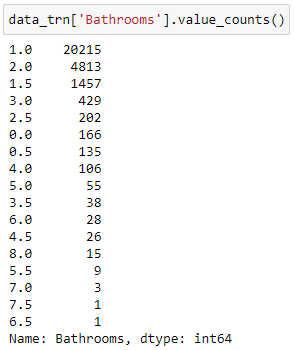
La columna es numérica por lo que no será necesaria su categorización, pero observo que hay valores con parte decimal. Investigando, descubro que esos valores “medios cuartos de baño” se corresponden con cuartos de aseo sin bañera o ducha.

La columna tiene valores nulos, analizo que valores tienen esos registros en la columna **Room\_Type**.

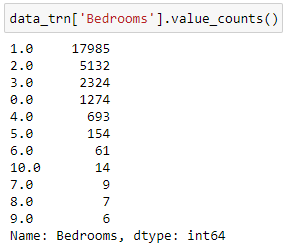
Los 85 nulos de la columna tienen un tipo de alquiler **1 - Private Room** o **2 - Entire home/apt**.

Decido que el tratamiento de limpieza a aplicar para los valores nulos de la columna dependerá del tipo de alquiler, si es **1 - Private Room** –o **0 - Shared Room**, para incluir también la opción– los nulos tomarán el valor **0.0** y si el tipo de alquiler es **2 - Entire home/apt** tomarán el valor **1.0**, ya que pienso que una habitación –sea privada o compartida– no tiene por qué tener cuarto de baño privado (o se indicaría), mientras que un inmueble completo tendrá al menos uno.

Veamos como ha quedado finalmente la columna.



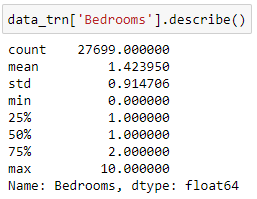
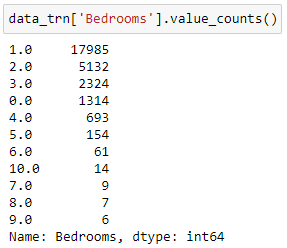
## **Columna Bedrooms**

Esta columna indica el número de habitaciones con las que cuenta el inmueble.

La columna es numérica por lo que no necesita categorización, pero observo que existen registros con el valor **0.0** que entiendo que deben corresponder a tipos de inmuebles sin cuartos como, por ejemplo, “tiendas de campaña”.

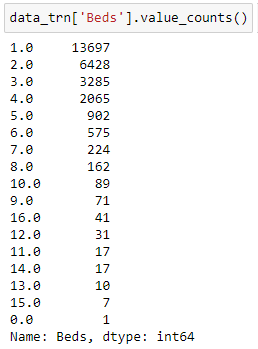
La columna tiene valores nulos a los que decido adjudicar el valor **0.0**, ya que entiendo que si el inmueble contara con habitaciones se indicaría.

Veamos como ha quedado finalmente la columna.



## **Columna Beds**

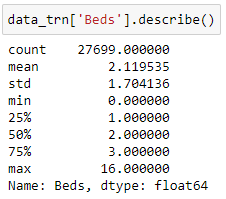
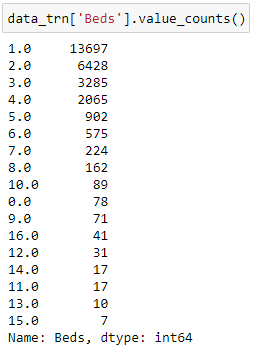
Esta columna indica el número de camas disponibles en el inmueble.



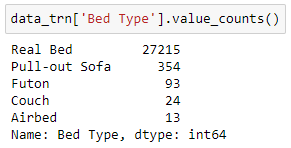
La columna es numérica por lo que no necesita categorización, pero vuelvo a observar registros con el valor **0.0** que entiendo que, igual que en la columna anterior, deben corresponder a tipos de inmuebles sin cuartos ni camas.

La columna tiene valores nulos a los que decido adjudicar el valor **0.0**, ya que entiendo que si el inmueble contara con camas se indicaría.

Veamos como ha quedado finalmente la columna.

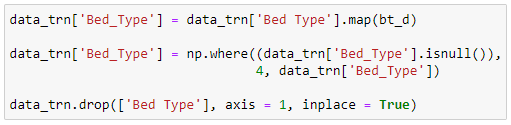


## **Columna Bed Type**

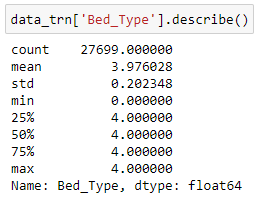
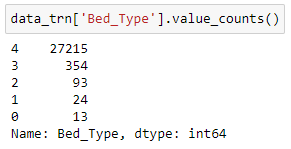
Esta columna indica el tipo de las camas disponibles en el inmueble.

La columna no contiene valores nulos y mayoritariamente apunta, en un 98,25% de los registros, al valor **Real Bed**. Este claro y enorme desbalanceo podría ser causa suficiente para su descarte, pero por ahora la mantendré en el dataset.

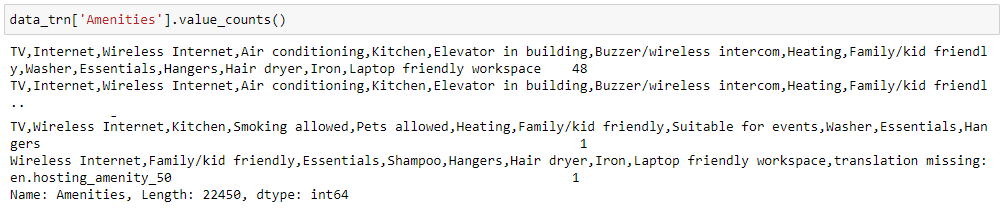
Al ser una columna alfanumérica debo categorizarla y para ello la mapearé de forma directamente proporcional al número de inmuebles que contiene, dando más valor a los tipos de cama más presentes en los inmuebles.

Aprovecho el mapeo para crear la columna renombrada, después realizo un tratamiento preventivo de posibles futuros valores nulos, asimilándolos al valor mayoritario y termino descartando la columna original.

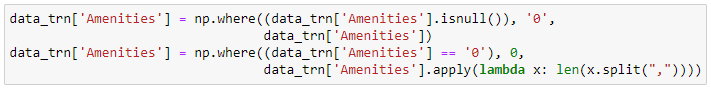
Veamos el aspecto de la nueva columna categorizada numéricamente.



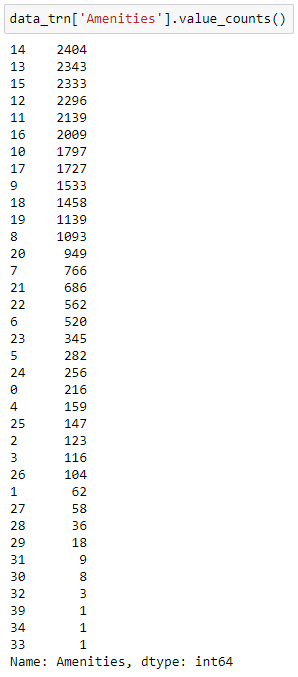
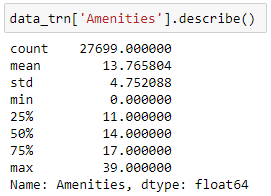
## **Columna Amenities**

Esta columna indica las comodidades disponibles en el inmueble.

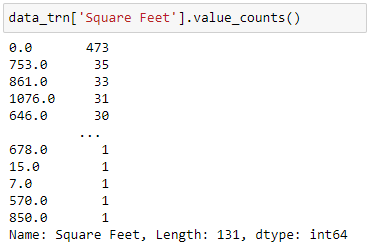
Esta columna contiene, o valores nulos, o una serie de objetos o servicios (comodidades), separados por comas, y decido categorizarla sumando las comodidades que contenga y asignando valor **0** a los registros con valor nulo. Esto otorga a la columna un peso directamente proporcional a los lujos del inmueble.

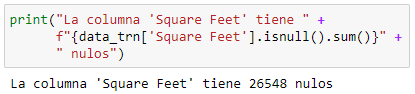
Bien es cierto que se puede valorar más un inmueble con “ascensor” o con “WiFi” frente a otro que provea “champú” o “televisión” y que se podría realizar un estudio de pesos más detallado o ajustado, pero entiendo que el coste computacional sería alto para el posible valor que aportaría.

Veamos el aspecto de la nueva columna categorizada numéricamente.



## **Columna Square Feet**

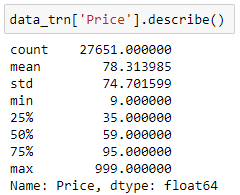
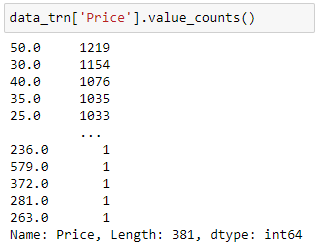
Esta columna indica la superficie de cada inmueble en pies cuadrados.

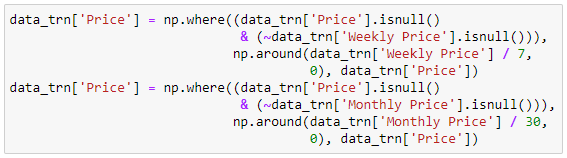
A la vista del conteo de sus valores, 473 registros con valor **0.0** y siendo este el valor predominante, no parece que la columna este demasiado bien informada. Veamos cuantos nulos tiene.

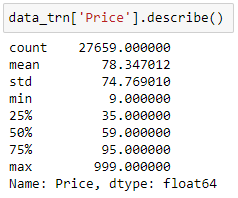
Confirmo que con 26.548 registros nulos y 473 con valor **0.0** –sobre un total de 27.699 registros–, la columna no va a aportar valor predictivo al modelo, por lo que decido descartarla.

## **Columna Price**

Esta columna indica el precio en euros de alquiler por noche del inmueble. Esta será la **columna objetivo** del modelo clasificador.



La descripción estadística muestra que la columna no contiene valores erróneos, ni valores **0.0**. Veo que hay 48 valores nulos que intentaré recuperar valiéndome de las columnas **Weekly Price** y **Monthly Price** ya que, si están informadas, contendrán respectivamente el precio semanal y mensual de alquiler del inmueble, en cuyo caso los dividiré entre 7 o entre 30 para obtener el precio de alquiler por noche.

Veamos cuantos registros se han recuperado de esta forma.

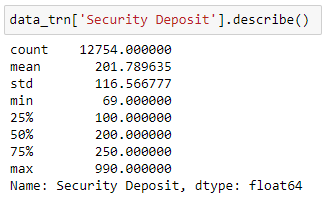
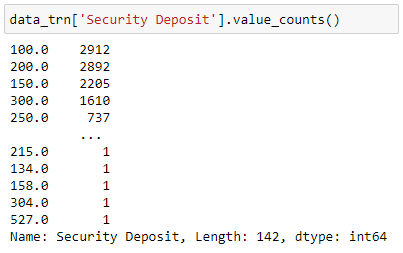
Tras la operación se han rescatado 8 registros nulos, el resto (40) decido considerarlos irrecuperables y los descarto definitivamente.

## **Columnas Weekly Price y Monthly Price**

Estas columnas no las descarté inicialmente porque pensé que podrían ser útiles a la hora de tratar la columna **Price**. Una vez han servido para tal fin, las descarto definitivamente.

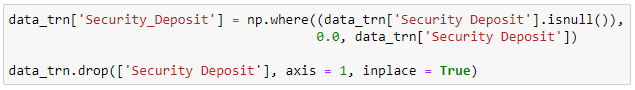
## **Columna Security Deposit**

Esta columna indica la cantidad en euros que se pide depositar como fianza para alquilar el inmueble.

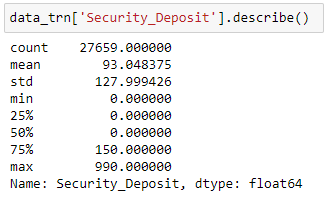
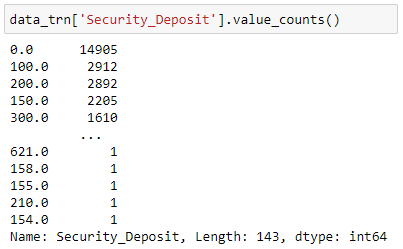


La columna es numérica por lo que no necesita categorización.

En la descripción estadística se aprecia un gran número de registros con valor nulo (14.905). Decido adjudicarles el valor **0.0** entendiendo que si no se especifica fianza para el alquiler del inmueble es porque no se cobrará.

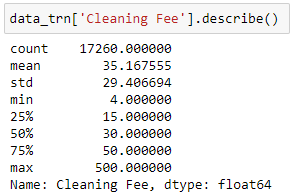
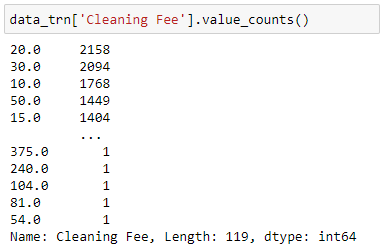
Aprovecho el tratamiento para crear la columna renombrada y termino descartando la columna original.

Veamos el aspecto final de la columna.



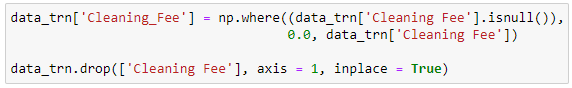
## **Columna Cleaning Fee**

Esta columna indica la cantidad en euros que se cobrará como recargo por el servicio de limpieza del inmueble.

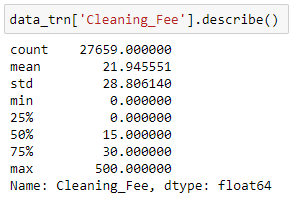
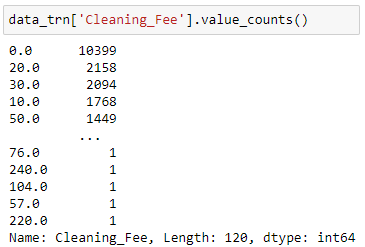


La columna es numérica por lo que no necesita categorización.

En la descripción estadística se aprecia un gran número de registros con valor nulo (10.399). Decido adjudicarles el valor **0.0** entendiendo que si no se especifica recargo por servicio de limpieza del inmueble es porque no se cobrará.

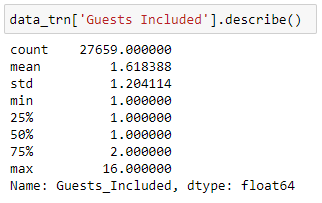
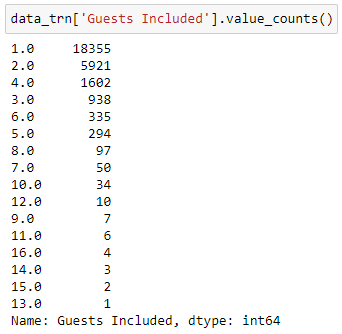
Aprovecho el tratamiento para crear la columna renombrada y termino descartando la columna original.

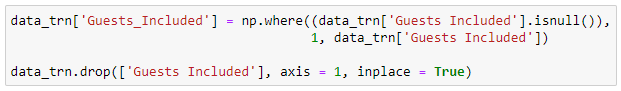
Veamos el aspecto final de la columna.



## **Columna Guests Included**

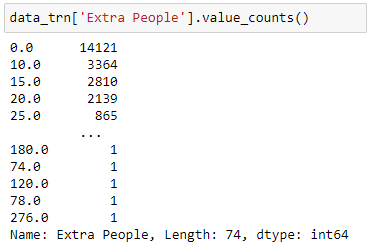
Esta columna indica la cantidad de invitados –por los que se deberá pagar un precio extra– permitidos en el inmueble.

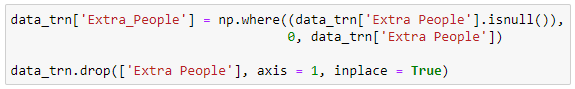


La columna es numérica por lo que no necesita categorización y no tiene valores nulos, pero aplicaré un tratamiento preventivo por si en un futuro hubiera, asimilándolos al valor mayoritario y aprovecharé para crear la columna renombrada y descartar la columna original.

## **Columna Extra People**

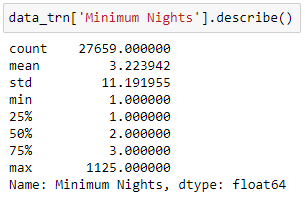
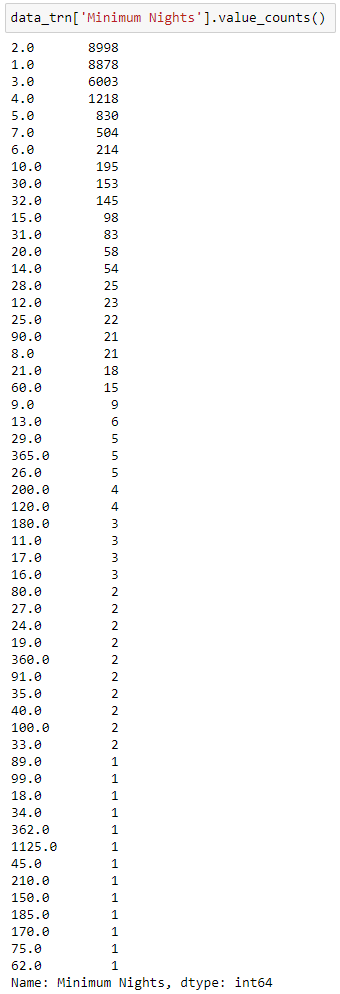
Esta columna indica la cantidad en euros especificada como recargo por cada invitado –permitido– en el inmueble.

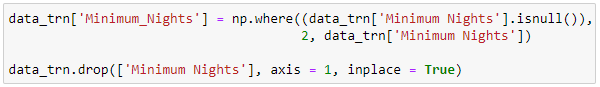


****La columna es numérica por lo que no necesita categorización y no tiene valores nulos, pero aplicaré un tratamiento preventivo por si en un futuro hubiera, asimilándolos al valor **0.0** entendiendo que si no se especifica precio extra por invitados es porque no se cobrará y aprovecharé para crear la columna renombrada y descartar la columna original.

## **Columna Minimum Nights**

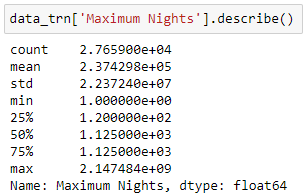
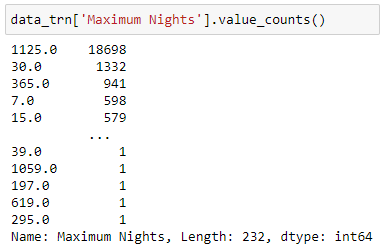
Esta columna indica la mínima cantidad de noches para poder alquilar el inmueble.

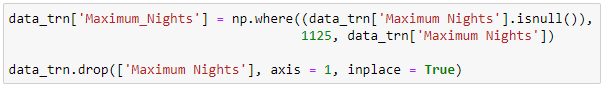


La columna es numérica por lo que no necesita categorización y no tiene valores nulos, pero aplicaré un tratamiento preventivo por si en un futuro hubiera, asimilándolos al valor mayoritarioy aprovecharé para crear la columna renombrada y descartar la columna original.

## **Columna Maximum Nights**

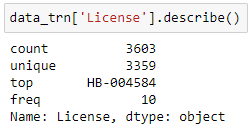
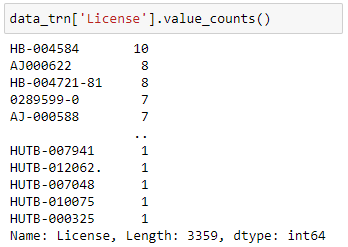
Esta columna indica la máxima cantidad consecutiva de noches aceptada para poder alquilar el inmueble.

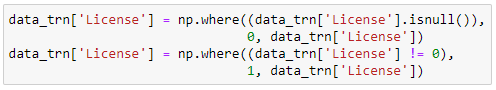


La columna es numérica por lo que no necesita categorización y no tiene valores nulos, pero aplicaré un tratamiento preventivo por si en un futuro hubiera, asimilándolos al valor mayoritarioy aprovecharé para crear la columna renombrada y descartar la columna original.

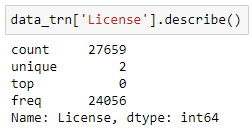
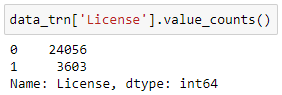
## **Columna License**

Esta columna indica si el hospedante cuenta con una licencia oficial para el arrendamiento del inmueble.

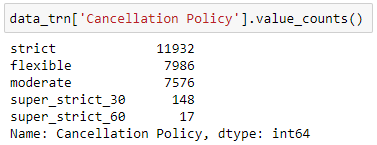


La columna es alfanumérica y necesitará ser categorizada. En la descripción estadística se aprecia un gran número de registros con valor nulo (24.056). Esta columna contiene o códigos de licencia o valores nulos. Decido categorizarla adjudicando el valor **0** para los nulos y el valor **1** para los registros informados.

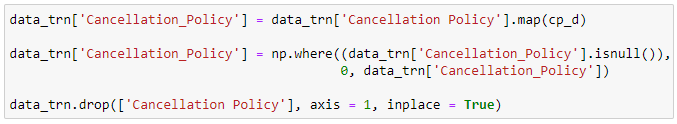
Veamos el aspecto de la columna categorizada numéricamente.



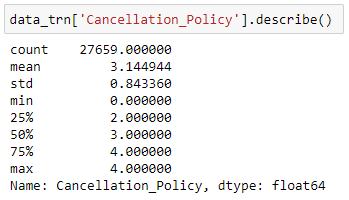
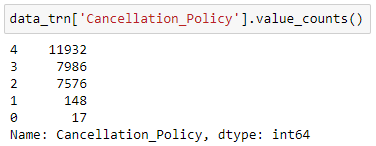
## **Columna Cancellation Policy**

Esta columna indica el tipo de política de cancelación del alquiler del inmueble.

La columna no contiene valores nulos. Al ser una columna alfanumérica debo categorizarla y para ello la mapearé de forma directamente proporcional al número de inmuebles que contiene, dando más valor a las políticas de cancelación más presentes en los inmuebles.

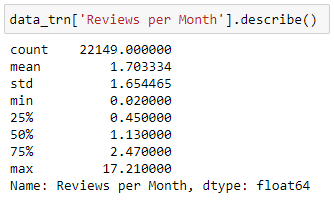
Aprovecho el mapeo para crear la columna renombrada, después realizo un tratamiento preventivo de posibles futuros valores nulos, asimilándolos al valor mayoritario y termino descartando la columna original.

Veamos el aspecto de la nueva columna categorizada numéricamente.

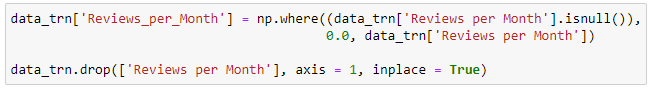


## **Columna** **Reviews per Month**

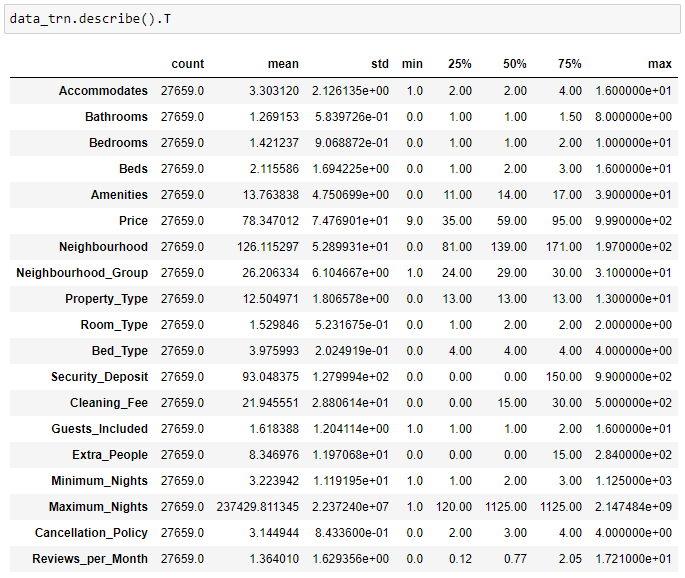
Esta columna indica el promedio mensual del número de comentarios recibidos por el inmueble.



Esta columna no se utilizará en el modelo clasificador. Esta columna se utilizará como referencia de éxito del alquiler de los inmuebles.

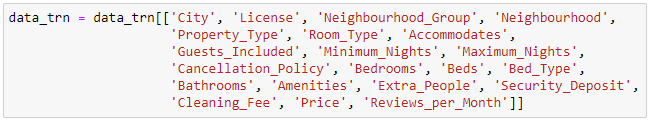
La columna es numérica no necesitará ser categorizada. En la descripción estadística se aprecia un gran número de registros con valor nulo (5.510) a los que decido adjudicar el valor **0.0**. Aprovecharé para crear la columna renombrada y descartar la columna original.

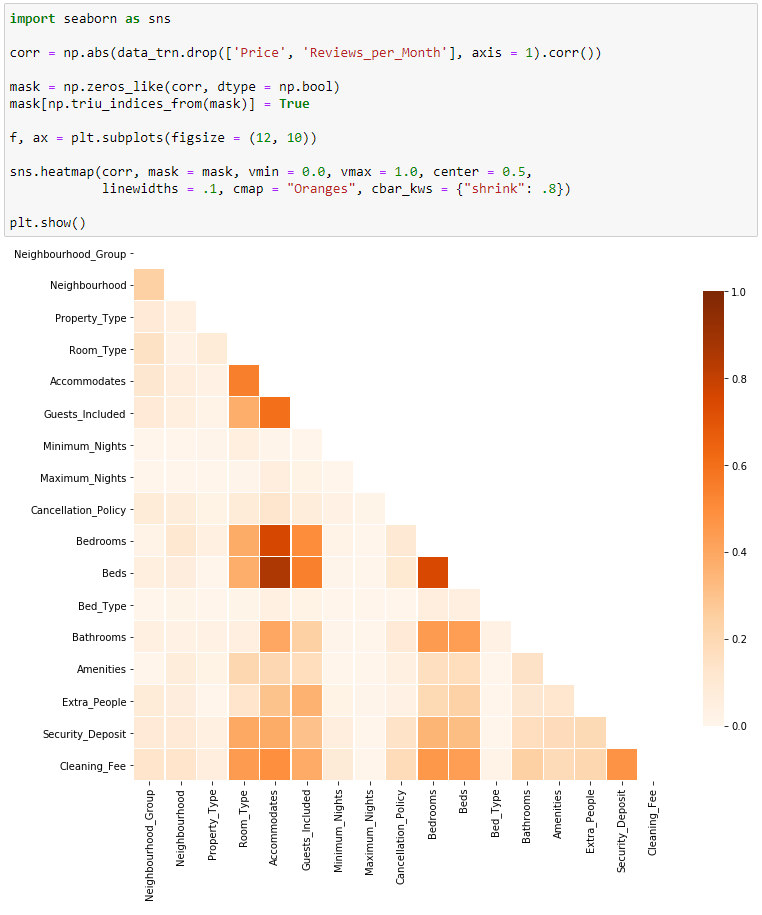
# **Análisis Exploratorio del Conjunto de Entrenamiento**

Una vez se han analizado todas las columnas seleccionadas inicialmente de forma individual y se ha procedido a su limpieza de nulos y/o categorización si ha sido necesario, veamos que aspecto estadístico tiene nuestro conjunto de entrenamiento.

Tenemos un conjunto de entrenamiento de 27.659 filas y 20 columnas (**Reviews\_per\_Month** no se utilizará para entrenar el modelo), todas ellas numéricas y sin valores nulos.

Las columnas tienen diferentes rangos y esto habré de tenerlo en cuenta al entrenar ciertos modelos que necesitarán una normalización.

Coloco el dataset con las columnas **Price** y **Reviews\_per\_Month** al final.

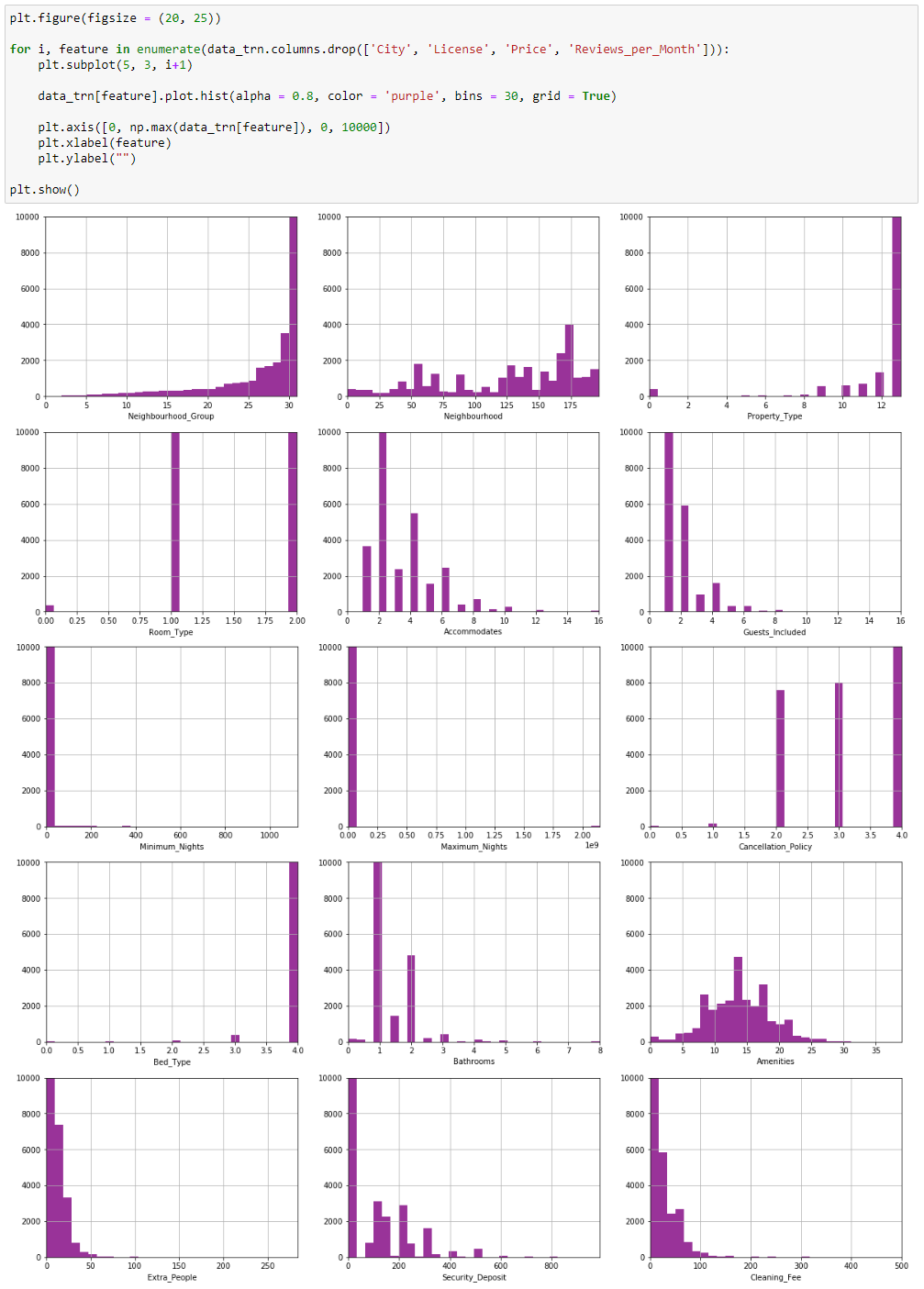
Ejecuto un gráfico de análisis de correlación para ver si existen relaciones directas entre ellas.

Observo una correlación muy alta entre las columnas **Beds**, **Bedrooms** y **Accommodates**, con todo el sentido del mundo puesto que parece lógico pensar qué si un inmueble está preparado para un determinado cupo de personas, cuente con un número de habitaciones y camas afín.

Decido quedarme sólo con la variable **Accommodates** por dos razones: la primera es que es la única columna de las tres que no contenía valores nulos, por lo que su información es muy probable que llegue siempre directamente desde el anfitrión, lo cual aporta fiabilidad; y la segunda porque indica el aforo incluido en el precio de alquiler por noche de cada inmueble, mientras que el número de habitaciones y de camas no.

Observo también cierta correlación entre varios pares de variables que, por ahora, no me hace decidir el descarte directo de ninguna de ellas:

* **Guests\_Included**, **Room\_Type** y **Cleaning\_Fee** frente a **Accommodates**
* **Cleaning\_Fee** y **Room\_Type** frente a **Security\_Deposit**
* **Cleaning\_Fee** frente a **Room\_Type**

Veamos los histogramas de las 15 columnas clasificadoras no binarias.

Observo distribuciones totalmente dispares a la forma normal.

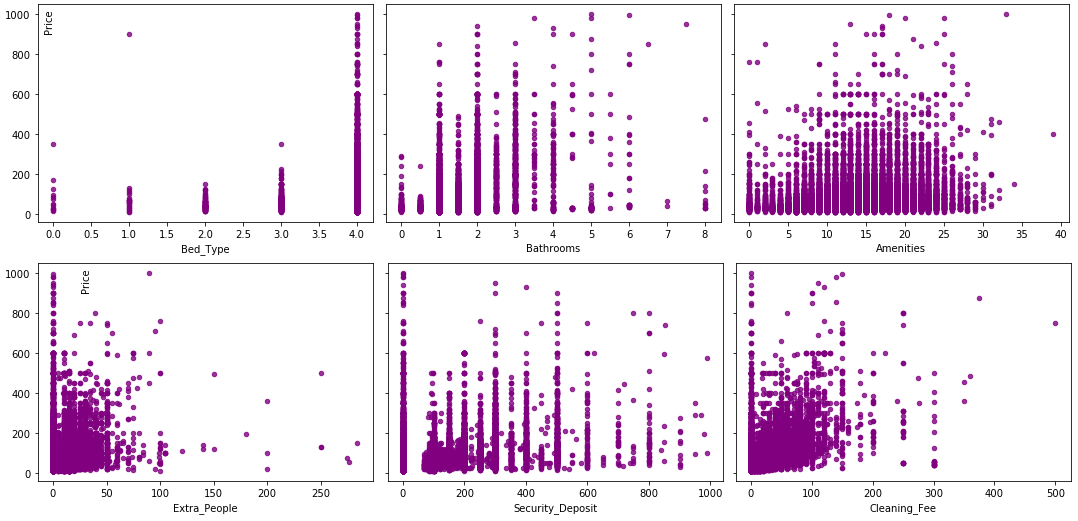
Llama la atención que las columnas **Minimum\_Nights** y **Maximum\_Nights** no presentan histogramas apreciables por la práctica unicidad de sus valores.

Las columnas **Extra\_People** y **Cleaning\_Fee** podrían ser candidatas para una transformación logarítmica ya que su histograma se escora totalmente hacia la izquierda, pero no es posible aplicársela porque en algunos registros tienen valor cero –y la transformación logarítmica no puede aplicarse sobre rangos con valores cero o negativos–.

También veo que las columnas **Property\_Type** y **Bed\_Type** tienen prácticamente un valor único mayoritario, lo que pudiera derivar en no aportar significación estadística y en consecuencia las desecharía del subset de datos.

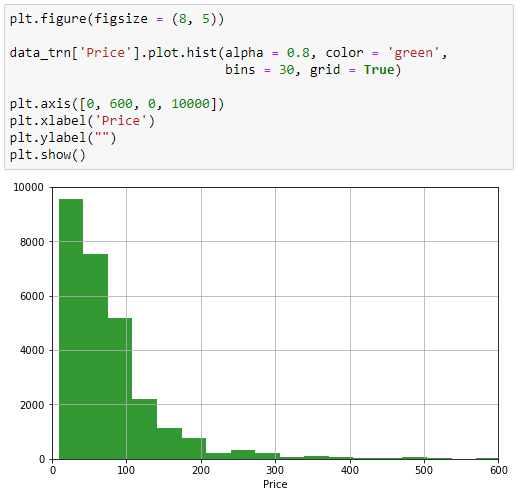
En principio estas gráficas no parecen indicar mucho más, pero analizándolas conjuntamente con el estadístico básico, aprecio cinco variables –**Security\_Deposit**, **Cleaning\_Fee**, **Extra\_People**, **Minimum\_Nights** y **Maximum\_Nights**– en las que su desviación típica supera a la media, lo que me hace sospechar una posible presencia de outliers en ellas.

Ejecuto un gráfico **scatter** para ver la relación de cada una de las 15 columnas frente a **Price**.

A la vista de las anteriores representaciones **scatter** aprecio claramente que las columnas **Guests\_Included**, **Minimum\_Nights**, **Maximum\_Nights,** **Bathrooms**, **Amenities**, **Extra\_People**, **Security\_Deposit** y **Cleaning\_Fee** presentan claros outliers en la parte derecha de sus gráficas.

Antes de acometer la tarea de eliminar filas, intentaré reducir en lo posible la dimensionalidad de las variables clasificadoras.

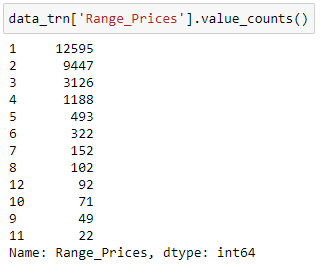
# **Columna Objetivo**

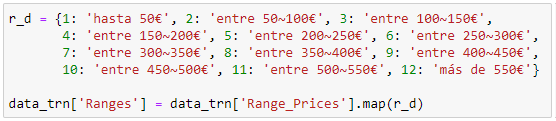
La columna **Price** es una variable continua con un número bastante elevado de valores únicos. Esto no es lo más adecuado para considerarla como columna objetivo de un modelo clasificador, por ello, crearé una nueva variable **Range\_Prices**, a partir de ella, con un número finito de clases para poder predecir.

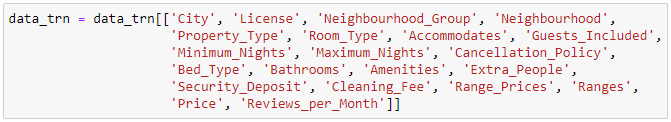
A la vista de su histograma, voy a definir una variable con 12 posibles clases que se corresponderían con los siguientes tramos de precios en euros:

* Clase **1**: **Range\_Prices** ≤ **50**
* Clase **2**: **50** < **Range\_Prices** ≤ **100**
* Clase **3**: **100** < **Range\_Prices** ≤ **150**
* Clase **4**: **150** < **Range\_Prices** ≤ **200**
* Clase **5**: **200** < **Range\_Prices** ≤ **250**
* Clase **6**: **250** < **Range\_Prices** ≤ **300**
* Clase **7**: **300** < **Range\_Prices** ≤ **350**
* Clase **8**: **350** < **Range\_Prices** ≤ **400**
* Clase **9**: **400** < **Range\_Prices** ≤ **450**
* Clase **10**: **450** < **Range\_Prices** ≤ **500**
* Clase **11**: **500** < **Range\_Prices** ≤ **550**
* Clase **12**: **550** < **Range\_Prices**

Veamos como ha quedado la nueva columna.

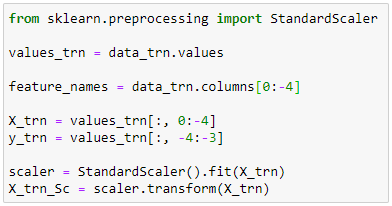


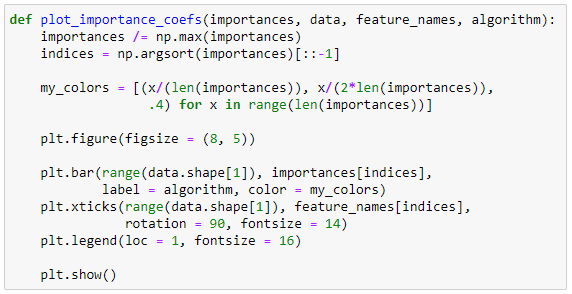
Una vez obtenida la variable objetivo **Range\_Prices** del clasificador, crearé una columna de tipo literal que me servirá en la parte de presentación junto a **Reviews per Month**.

Recolocaré de nuevo el dataset, ubicando al final del mismo las columnas que utilizaré en el área de presentación y no en el modelo clasificador.

# **Reduciendo la Dimensionalidad**

Para intentar descartar algunas columnas más y reducir así el número de variables clasificatorias, ejecutaré algunos modelos que ofrecen la posibilidad de medir la aportación predictiva de las columnas.

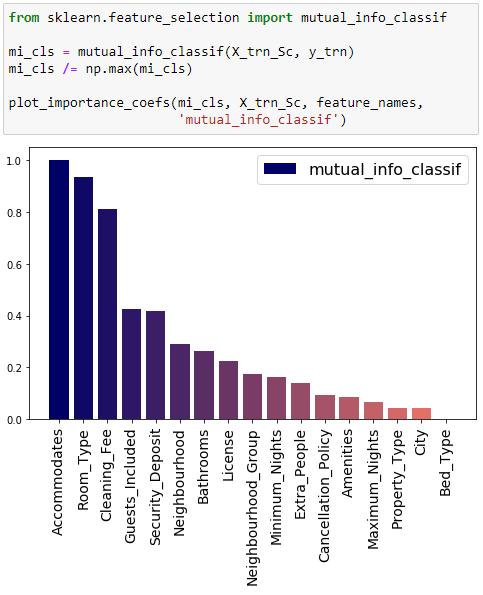
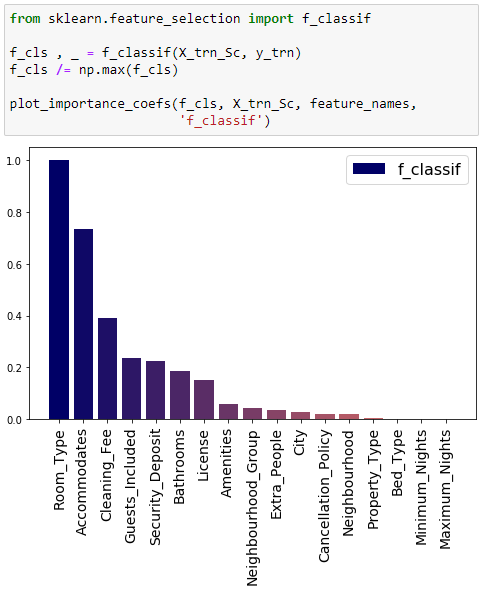
Lo primero que haré será normalizar los datos del conjunto de entrenamiento (para utilizarlos escalados en los modelos que lo necesiten) y para ello utilizaré el método **StandardScaler** de la librería **preprocessing** de **scikit-learn**.

A continuación, definiré una función para poder visualizar debidamente los coeficientes de importancia, de las variables, que se obtengan de cada modelo.

## **Modelos f y mutual\_info**

Ejecutaré un modelo **f\_classif** y un modelo **mutual\_info\_classif**, ambos de la librería **feature\_selection** de **scikit-learn**.

Estos no son modelos predictivos propiamente dichos, su utilidad radica en la selección de características de conjuntos de datos de entrada a modelos de clasificación.



El modelo **f\_classif** otorga nula importancia a las variables **Maximum\_Nights,** **Minimum\_Nights** y **Bed\_Type**.

El modelo **mutual\_info\_classif**, bastante menos restrictivo, únicamente otorga nula importancia a la variable **Bed\_Type**.

## **Modelo Lasso**

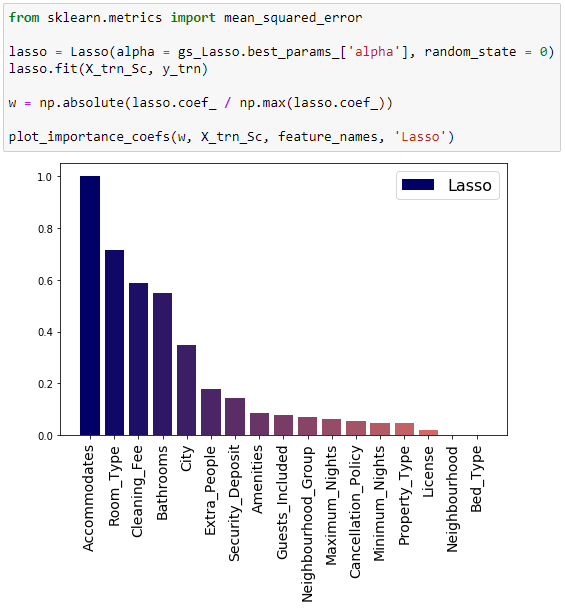
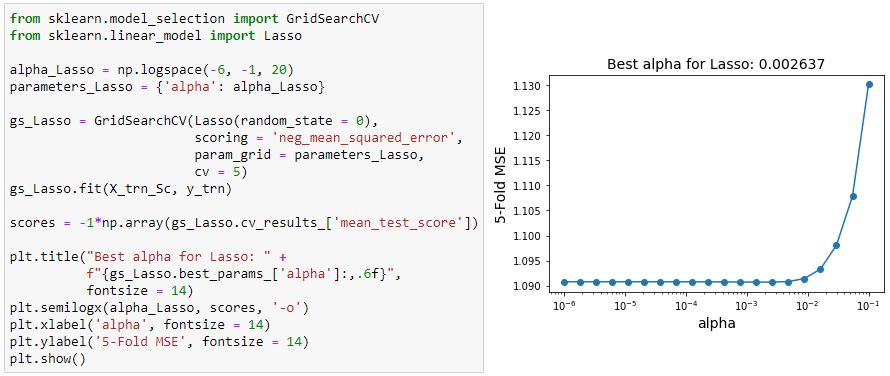
Ahora ejecutaré un modelo **Lasso** de la librería **linear\_model** de **scikit-learn.**

La utilidad de este modelo estriba en que regulariza (reduce) los coeficientes del modelo llegando incluso a anular variables no –o ínfimamente– relevantes.

Lo primero es ajustar el hiper-parámetro **alpha** a su mejor valor para que el modelo **Lasso** optimice su entrenamiento con nuestra muestra de datos.

Ejecutaré un algoritmo **GridSearchCV** de la librería **model\_selection** de **scikit-learn**, que implementará una búsqueda por fuerza bruta, sobre el modelo**Lasso**.

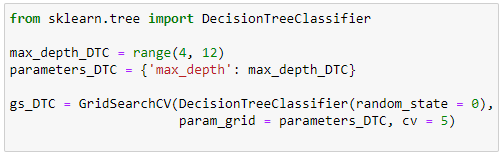
Así obtendré diferentes valores del error cuadrático medio (MSE) del modelo para diferentes valores del hiper-parámetro y elegiré el valor de este que llevé al mínimo el MSE.

Aplico el valor **alpha** óptimo en el modelo **Lasso** y lo entreno con la muestra de datos de entrenamiento.

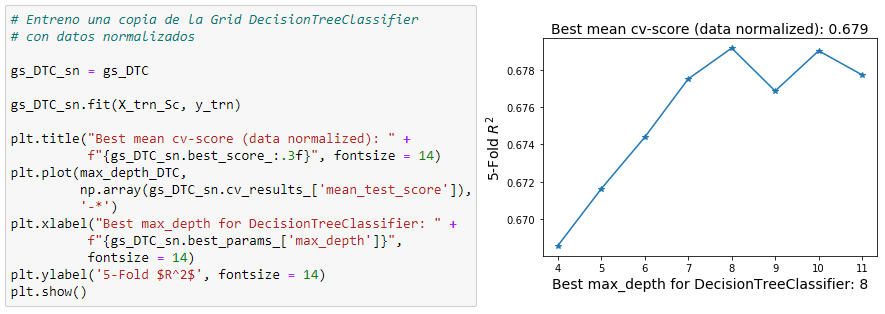
El modelo **Lasso** otorga nula importancia a las variables **Bed\_Type** y **Neighbourhood.**

## **Modelo DecissionTree**

Ejecutaré ahora un modelo **DecissionTreeClassifier** de la librería **tree** de **scikit-learn**, aprovechando que aporta coeficientes de importancia sobre las variables.

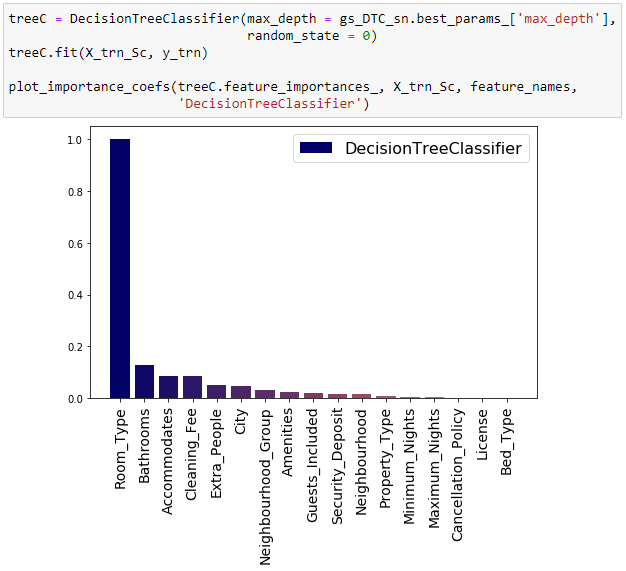
Buscaré el mejor valor de su hiper-parámetro **max\_depth**, ejecutando un algoritmo **GridSearchCV**, sobre el modelo**DecissionTreeClassifier**, para que éste optimice su entrenamiento con nuestra muestra de datos.

Para los modelos basados en árboles de decisión no es necesario que los datos estén normalizados. Entrenaré la **grid** con ambas opciones para ver si hay diferencias.



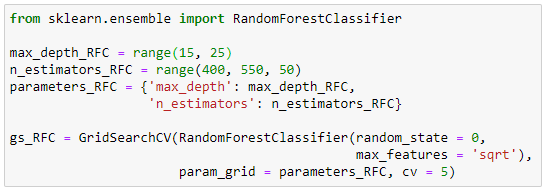
Ambos entrenamientos consiguen idéntico **cv-score** para su mejor valor de **max\_depth**, aunque con los datos normalizados parece que se optimiza la profundidad de ramas del modelo.

Aplicaré el valor **max\_depth** óptimo en el modelo **DecissionTreeClassifier** y lo entrenaré con la muestra de datos de entrenamiento normalizada, para ver que coeficientes de importancia de variables arroja.

El modelo **DecissionTreeClassifier** no da importancia a las variables **Bed\_Type**, **License** y **Cancellation\_policy**.

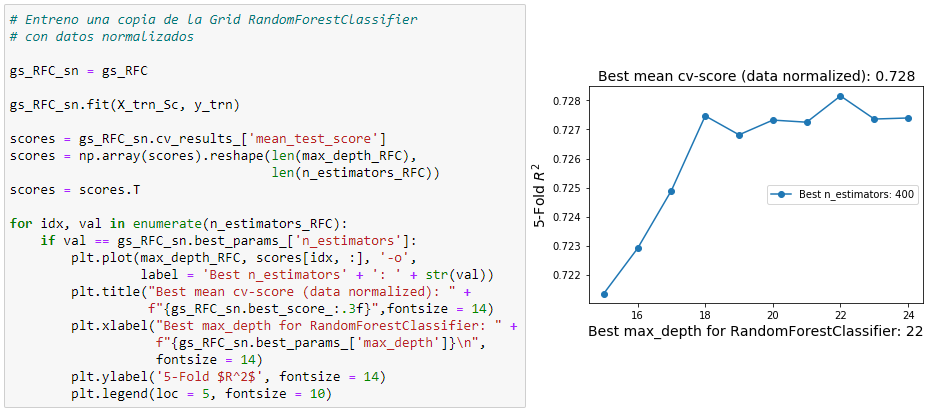
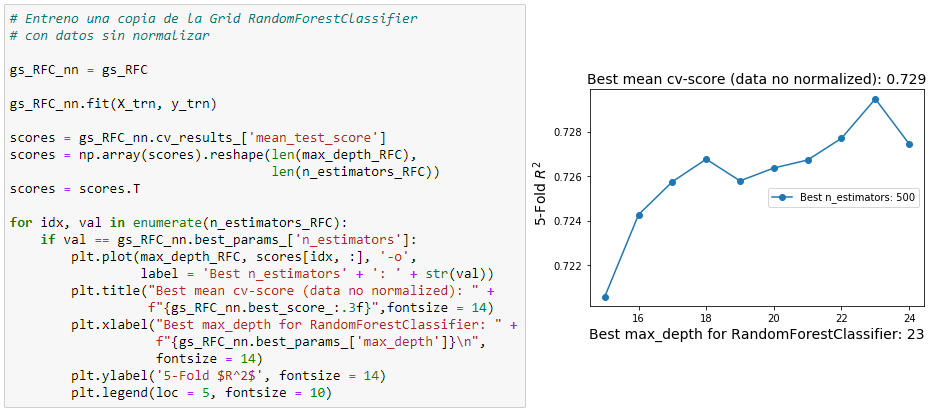
## **Modelo RandomForest**

Ejecutaré a continuación un modelo **RandomForestClassifier** de la librería **ensemble**de **scikit-learn**, que al igual que el anterior, aporta coeficientes de importancia sobre las variables.

Buscaré los mejores valores de sus hiper-parámetros **max\_depth** y **n\_estimators** –y dejaré su tercer hiper-parámetro **max\_features** fijado al valor **sqrt**–, ejecutando un algoritmo **GridSearchCV**, sobre el modelo**RandomForestClassifier**, para que éste optimice su entrenamiento con nuestra muestra de datos.

Al igual que en el apartado anterior, entrenaré la **grid** con datos sin normalizar y normalizados para ver si existen diferencias.

Como en el apartado anterior, ambos entrenamientos consiguen un **cv-score** prácticamente idéntico para sus mejores valores de **max\_depth** y **n\_estimators**, aunque con los datos normalizados también parece que se optimizan, tanto la profundidad de ramas, como el número de árboles del modelo.



Aplicaré los valores óptimos de **max\_depth** y **n\_estimators** en el modelo **RandomForestClassifier** y lo entrenaré con la muestra de datos de entrenamiento normalizada, para ver que coeficientes de importancia de variables arroja.

El modelo **RandomForestClassifier** no descarta directamente ninguna variable.

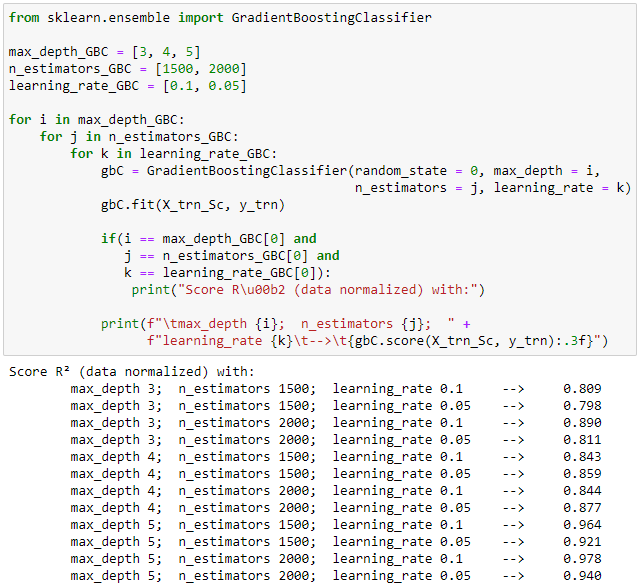
## **Modelo GradientBoosting**

Ejecutaré por último un modelo **GradientBoostingClassifier** de la librería **ensemble**de **scikit-learn**, que al igual que los demás modelos de la familia de árboles de decisión aporta coeficientes de importancia sobre las variables.

Buscaré los mejores valores de sus tres hiper-parámetros **max\_depth**, **n\_estimators** y **learning\_rate**, ejecutando un algoritmo **GridSearchCV**, sobre el modelo**GradientBoostingClassifier**, para que éste optimice su entrenamiento con nuestra muestra de datos.

*Tras varios intentos infructuosos de entrenamiento con* ***GridSearchCV*** *sobre* ***GradientBoostingClassifier****, tanto en* ***Jupyter Notebook****, como en* ***Google Colaboratory****, fijando uno, dos y hasta los tres hiper-parámetros e incluso rebajando el valor del* ***cross-validation****, decidí desistir y ejecutar una búsqueda por fuerza bruta manual.*

*A la vista de los resultados de los dos apartados anteriores y conociendo el coste computacional que conlleva su entrenamiento, para este modelo sólo utilizaré los datos normalizados.*

Aplicaré los valores de **max\_depth**, **n\_estimators** y **learning\_rate** con los que he obtenido la mejor bondad de ajuste [R2] en el modelo **GradientBoostingClassifier** y lo entrenaré con la muestra de datos de entrenamiento normalizada, para ver los coeficientes de importancia que asigna a las variables.

El modelo **GradientBoostingClassifier** tampoco descarta directamente ninguna variable.

## **Maldición de la Dimensionalidad**

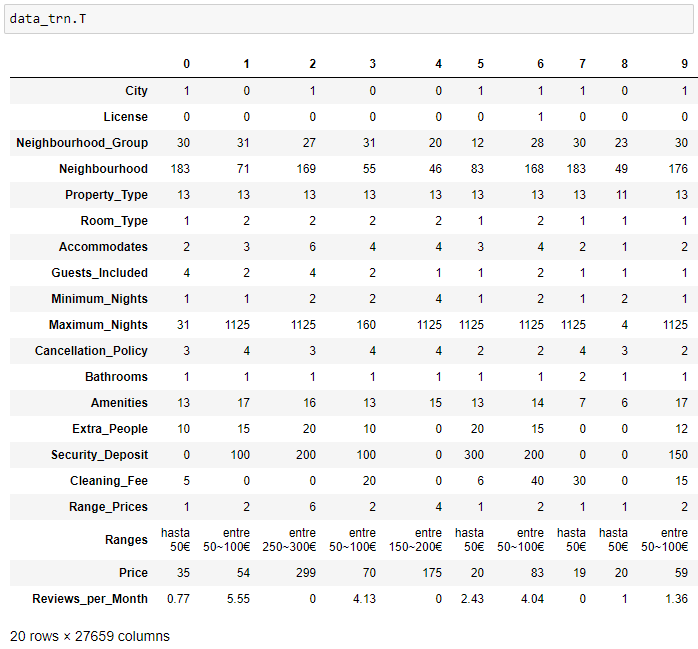
He entrenado seis modelos especialistas en selección de variables con el objetivo y la esperanza de poder descartar alguna o algunas de ellas y de esta forma paliar la, siempre presente en los datos, *Maldición de la Dimensionalidad*.

Recapitulando, cada modelo selector anulaba las variables:

* **f\_classif** **Maximum\_Nights,** **Minimum\_Nights** y **Bed\_Type**.
* **mutual\_info\_classif**  **Bed\_Type**.
* **Lasso** **Neighbourhood** y **Bed\_Type**.
* **DecissionTreeClassifier** **Bed\_Type**, **License** y **Cancellation\_policy**.

Los modelos **RandomForestClassifier** y **GradientBoostingClassifier** no descartaban directamente ninguna variable.

La única columna repetida en todos los modelos activos ha sido **Bed\_Type** y por este motivo –y porque pienso que tiene toda la lógica, como adelanté al inicio del análisis exploratorio, ya que su valor mayoritario, prácticamente único, es **Real Bed**–, la descartaré del dataset.

El conjunto de datos termina con esta apariencia (traspuesta).

Tengo 16 variables predictoras, todas ellas sin nulos y con formato numérico, la variable objetivo **Range\_Prices** y tres variables más, al final, que no entrarán al modelo clasificador.

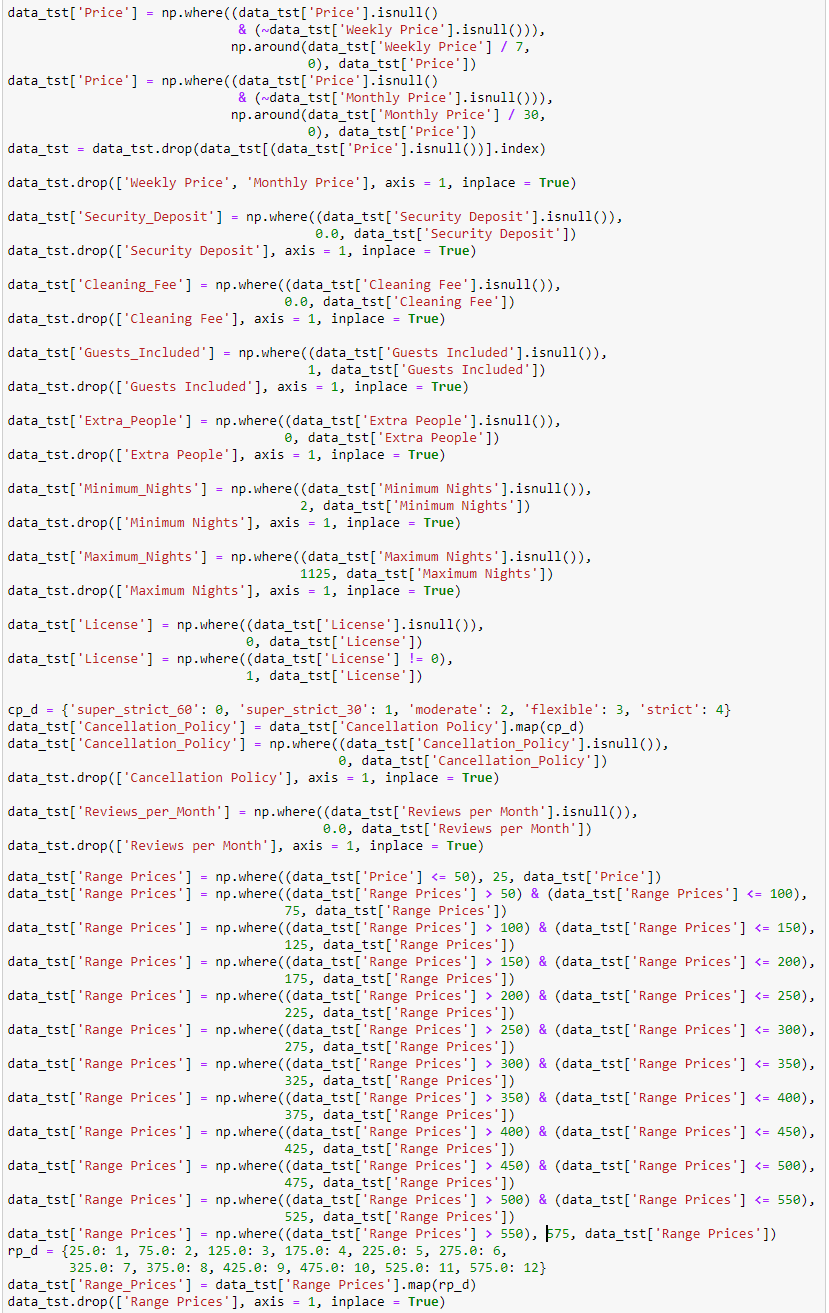
## **Tratamiento de Outliers**

Este sería el momento en el que procedería a analizar de nuevo las variables predictoras con el objetivo de detectar ciertos valores, en cada una de ellas, que pudieran marcar los límites tras los que consideraría los llamados valores anómalos u outliers.

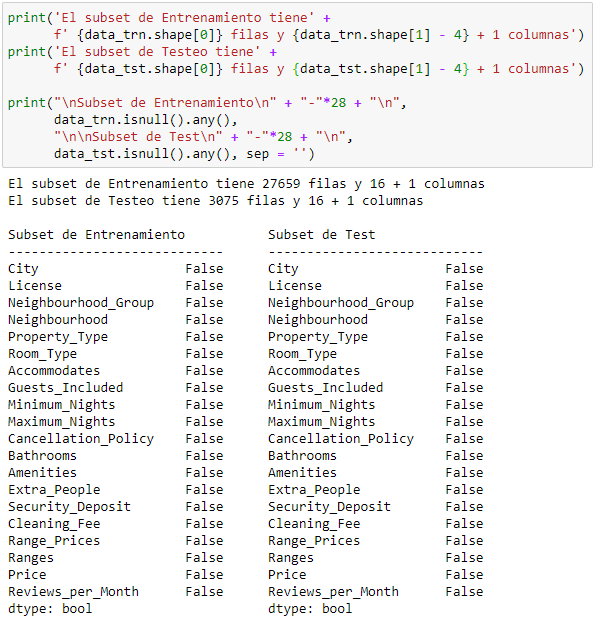
Pero como lo que pretendo implementar es un modelo recomendador y no quiero excluir ningún valor que pudiera llegarle al modelo directamente desde el usuario anfitrión, obviaré este paso aun sabiendo que quizá su precisión pueda verse algo afectada.

# **Proceso del Conjunto de Test**

Aplicaré sobre el conjunto de datos de test todos los tratamientos que hemos ido realizando al conjunto de entrenamiento, para que ambas muestras queden alineadas.

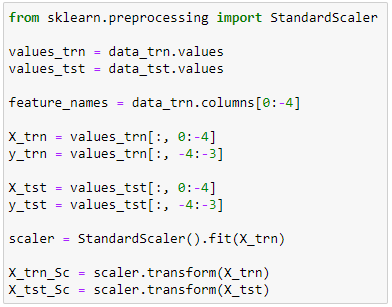




Tras el tratamiento aplicado, confirmo que ambos subsets de datos de test han quedado alineados y sin valores nulos en sus variables.

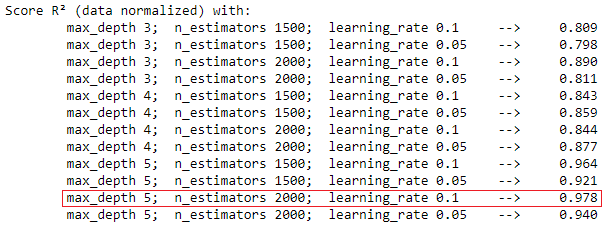
# **Modelos Machine Learning para Clasificación**

Tengo analizados, limpios y ajustados las muestras de datos de entrenamiento y test, voy ahora a implementar diferentes modelos de inteligencia artificial para el ejercicio de clasificación.

Lo primero será preparar las colecciones de datos para los procesos, separando la variable objetivo del resto y normalizando las variables predictoras de ambas muestras de trabajo.

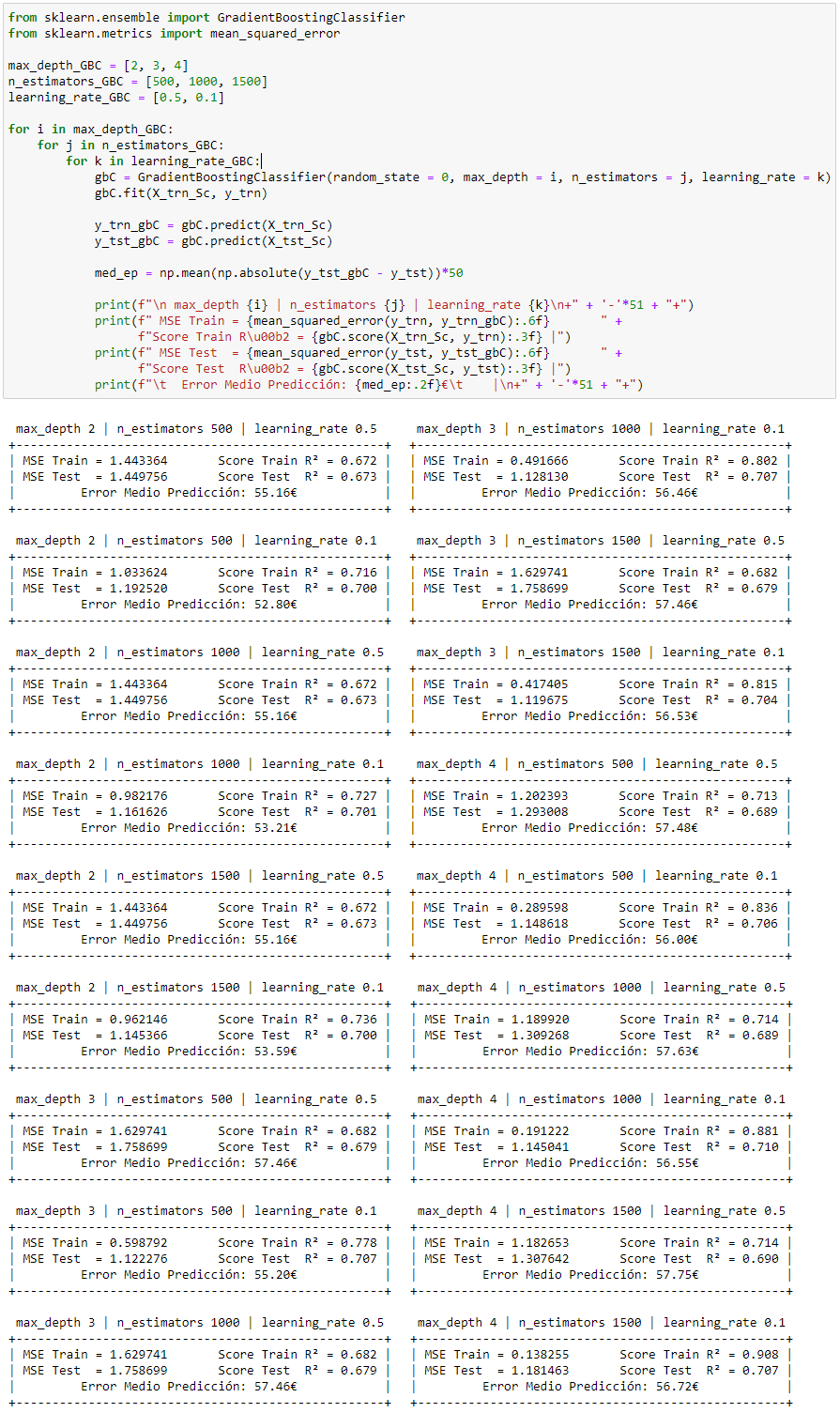
## **Modelo GradientBoosting**

El modelo selector de variables predictoras que mejores prestaciones ha presentado durante el análisis de coeficientes de importancia, para reducir la dimensionalidad, ha sido el **GradientBoostingClassifier** de la librería **ensemble**de **scikit-learn**.

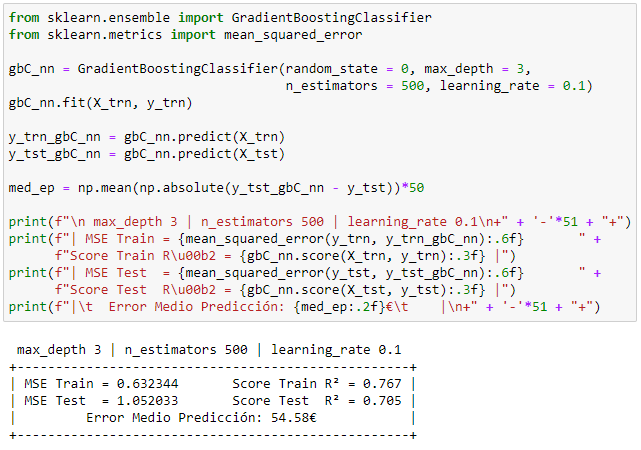
Además, el hecho de no haber sido posible entrenar una **grid** para este modelo, me llevó a buscar su mejor parametrización por fuerza bruta obteniendo de esta forma su precisión, o bondad de ajuste, con sus hiper-parámetros óptimos.

A mayor número de ramas la precisión del modelo parece aumentar, aunque con una profundidad de **5** el modelo tiende claramente al sobreajuste. El número de estimadores no parece ser claramente decisivo en la precisión final del modelo, sin casi diferencia de precisión en los entrenamientos en los que varía y coinciden profundidad y ratio de aprendizaje. Y con menor ratio de aprendizaje parece que la precisión tiende casi siempre a disminuir.

Voy a entrenarlo de nuevo con el subset de datos de entrenamiento normalizado, que ahora tiene una columna menos, fijando nuevos rangos de hiper-parámetros en función de mis observaciones anteriores. Al mismo tiempo testearé cada versión con el subset de datos de test.

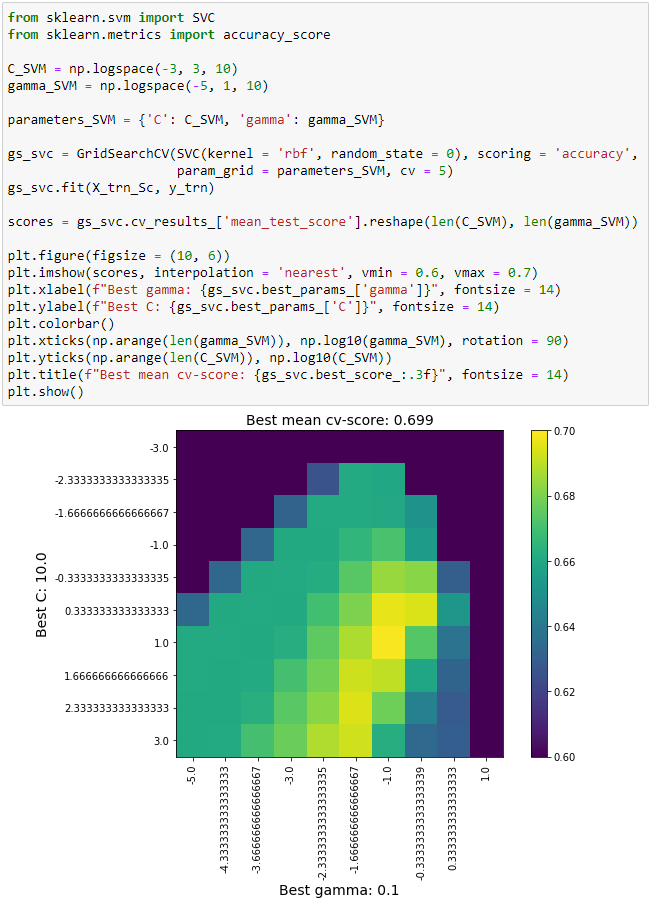


Tras observar y analizar los entrenamientos anteriores, decido que la mejor parametrización sería **max\_depth** = **3**, **n\_estimators** = **500** y **learning\_rate** = **0.1** ya que sus métricas, contra el subset de datos de test, con un error cuadrático medio (MSE) de **1.122276** y una bondad de ajuste (R2) de **0.707** son las mejores, obteniendo un error medio en predicción de **55.20€**.

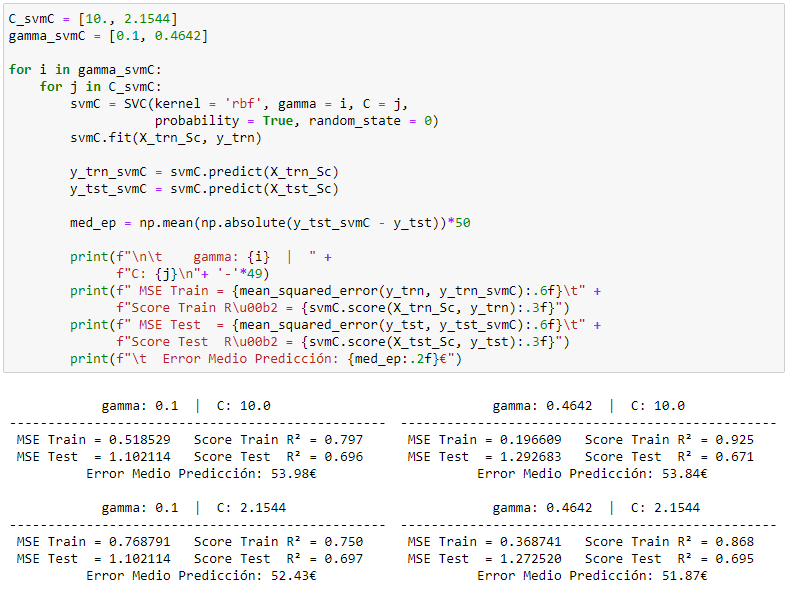
Probaré esta parametrización contra los datos no normalizados, por si acaso el modelo mejorara sus prestaciones y de paso nos ahorrara el coste computacional del escalado.

Pues por lo que parece, con datos no normalizados, el modelo aprende algo menos en entrenamiento, pero generaliza un poco mejor con la muestra de datos de test, baja dos milésimas su precisión, pero mejora siete centésimas el error cuadrático medio y presenta un error medio en predicción **0.62€** menor.

## **Modelo SVM**

Probaré a continuación un modelo Support Vector Machine (SVM) **SVC** de la librería **svm** de **scikit-learn**. Este modelo tiene dos hiper-parámetros **C** y **gamma** que obtendré ejecutando un algoritmo **GridSearchCV**, sobre el modelo, para que éste optimice su entrenamiento con nuestra muestra de datos.

Una vez obtenidos los valores óptimos de los hiper-parámetros **C** y **gamma**, aplicaré sobre el modelo, entrenando con la muestra de datos de entrenamiento, los rangos más cercanos a los valores óptimos para medir sus resultados contra el subset de datos de test.

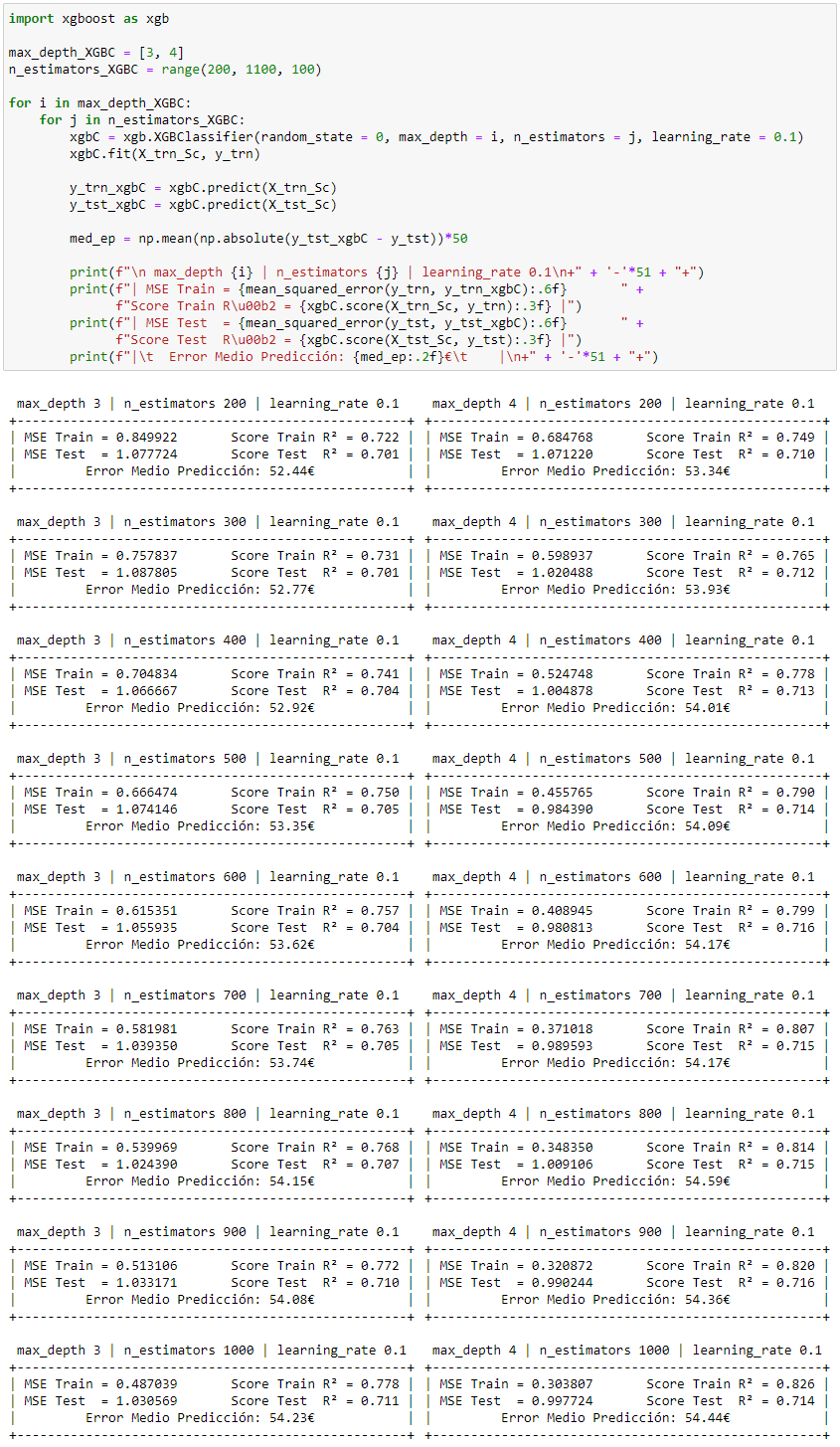
Las métricas sobre la muestra de test que obtengo con los valores óptimos (**C** = **10** y **gamma** = **0.1**) son muy parecidas a las del modelo **GradientBoostingClassifier**, con una precisión algo menor (una centésima) y mejorando el error cuadrático medio y el error medio en predicción.

Del resto de parametrizaciones me quedaría con (**C** = **2.1544** y **gamma** = **0.1**) ya que, aunque parece que con el subset de datos de entrenamiento obtiene unas métricas algo peores, después generaliza bastante bien en test. Las versiones con **gamma** = **0.4642** tienden al sobreajuste.

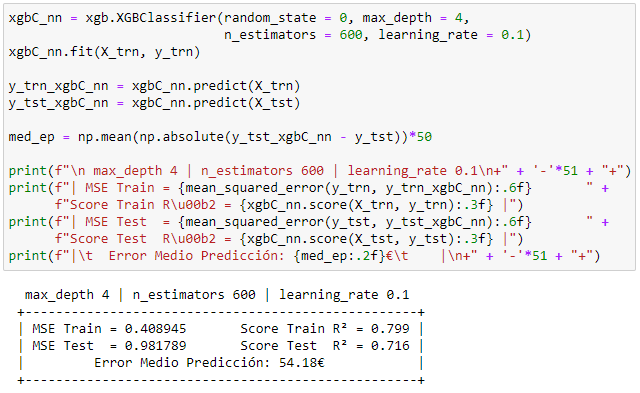
## **Modelo XGBoost**

Implementaré ahora un modelo **XGBoost**, que significa **eXtreme Gradient Boosting**. Este es el algoritmo que ha estado dominando recientemente los problemas de **machine learning** y las competiciones de **Kaggle** con datos estructurados o tabulares. **XGBoost** es una implementación del **GradientBoosting** diseñada para minimizar el tiempo de ejecución y maximizar el rendimiento.

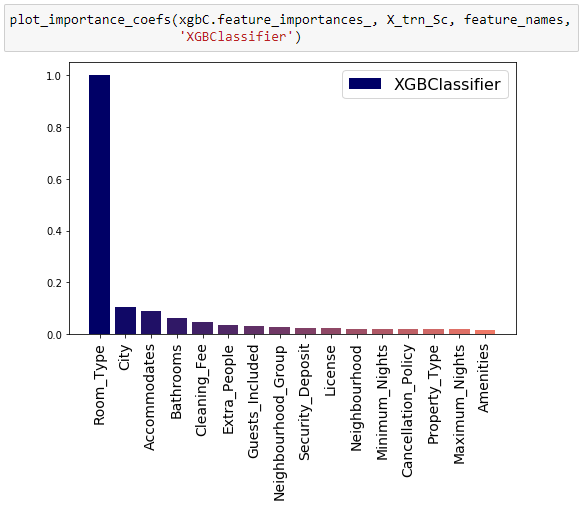
Al igual que hicimos en el párrafo 11.1 y sabiendo que en modelos basados en ‘Gradient Descending’ el algoritmo **GridSearchCV** precisa de un coste computacional excesivo para obtener los hiper-parámetros óptimos, ejecutaré entrenamientos y predicciones del modelo **XGBClassifier** alternando diferentes valores para sus hiper-parámetros**max\_depth**, **n\_estimators**, fijando **learning\_rate** al valor **0.1** y midiendo los resultados contra el subset de datos de test para obtener, por fuerza bruta, el modelo con mejor parametrización.



De nuevo, a mayor número de ramas la precisión del modelo parece aumentar, pero tendiendo al sobreajuste. Las métricas con los valores de **max\_depth** = **4**, **n\_estimators** = **600** y **learning\_rate** = **0.1** parecen las mejores y superan las de los dos modelos anteriores, disminuyendo el error cuadrático medio, aumentando la precisión contra el conjunto de test y con una buena generalización.

Probaré esta parametrización contra los datos no normalizados, por si acaso el modelo mejorara sus prestaciones y de paso nos ahorrara el coste computacional del escalado.

Por lo que observo, con datos no normalizados, el modelo proporciona idénticas métricas.

En su momento, no utilicé este modelo para medir los coeficientes de importancia de las variables predictoras. Como preveo que va a ser el elegido finalmente, por sus buenos resultados, voy a ejecutar la gráfica por si acaso descartara alguna columna del dataset.

Este modelo, al igual que el **RandomForestClassifier** y el **GradientBoostingClassifier**, tampoco descarta ninguna de las columnas predictoras.

# **El Modelo Clasificador Elegido**

Una vez implementados, entrenados y testeados los tres modelos llega el momento de elegir uno de ellos.

El mejor modelo **GradientBoostingClassifier**, parametrizado con **max\_depth** = **3**, **n\_estimators** = **500** y **learning\_rate** = **0.1**, proporcionó las siguientes métricas con datos normalizados:

* **MSE Train** = **0.598792** **Score Train R²** = **0.778**
* **MSE Test** = **1.122276** **Score Test R²** = **0.707**
* **Error Medio Predicción**: **55.20€**

El mejor modelo **SVC (SupportVectorMachineClassifier)**, parametrizado con **C** = **10** y **gamma** = **0.1**, proporcionó las siguientes métricas con datos normalizados:

* + **MSE Train** = **0.518529 Score Train R²** = **0.797**
  + **MSE Test** = **1.102114 Score Test R²** = **0.696**
  + **Error Medio Predicción**: **53.98€**

El mejor modelo **XGBClassifier**, parametrizado con **max\_depth** = **4**, **n\_estimators** = **600** y **learning\_rate** = **0.1**, proporcionó las siguientes métricas indistintamente con datos normalizados y sin normalizar:

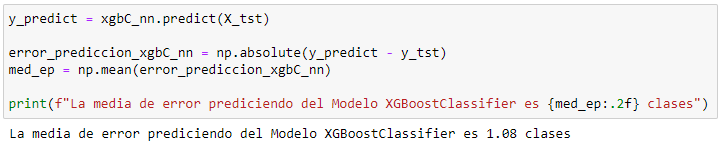
* + **MSE Train** = **0.408945 Score Train R²** = **0.799**
  + **MSE Test** = **0.980813 Score Test R²** = **0.716**
  + **Error Medio Predicción**: **54.17€**

Debo decantarme por el último y las razones son evidentes: mejora el error cuadrático medio y la bondad de ajuste de sus predecesores, tanto en entrenamiento como en testeo, no parece sufrir sobreajuste, su error medio en predicción está contenido y como media falla en una clase consecutiva.

Por lo que he visto, tanto en los entrenamientos de este trabajo, como en los realizados para las prácticas de las asignaturas del bootcamp, la calidad de este dataset no da para mucho más, por tanto, decido que el modelo **XGBClassifier** será mi algoritmo clasificador.

A lo anterior hay que añadir, tal y como comenté anteriormente, que esta implementación del descenso del gradiente **XGB**, está específicamente diseñada para minimizar el tiempo de ejecución y maximizar el rendimiento. Además, como hemos visto en sus pruebas, no nos exige la normalización de los datos de entrada, por lo que ahorra aún más en coste computacional.

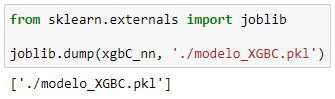
## **El Modelo en Predicción**

Creo que es de rigor, mostrar al modelo prediciendo. Para ello mediré la media de error por clases.

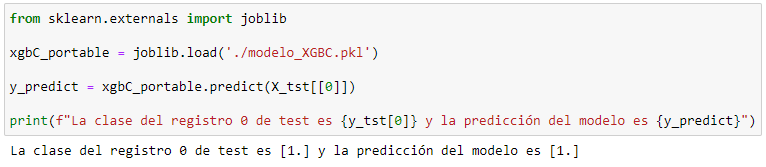
Como media, el modelo falla en 1.08 clases, es decir, sus predicciones pueden desviarse de media ±1.08 clases arriba o abajo.

## **Portabilidad del Modelo**

Para poder utilizar el modelo entrenado en un futuro, sin tener que volver a definirlo y entrenarlo, lo almacenaré en un fichero **pickle**.

Para ello utilizaremos la clase **joblib** de la librería **externals** de **scikit-learn**.

Esto guardará en nuestro repositorio local un archivo ‘**modelo\_XGBC.pkl**’ portable que podremos, en definitiva, llevar a cualquier sitio para el despliegue y uso del modelo clasificador.

Para su utilización posterior, únicamente tendríamos que extraer el modelo.

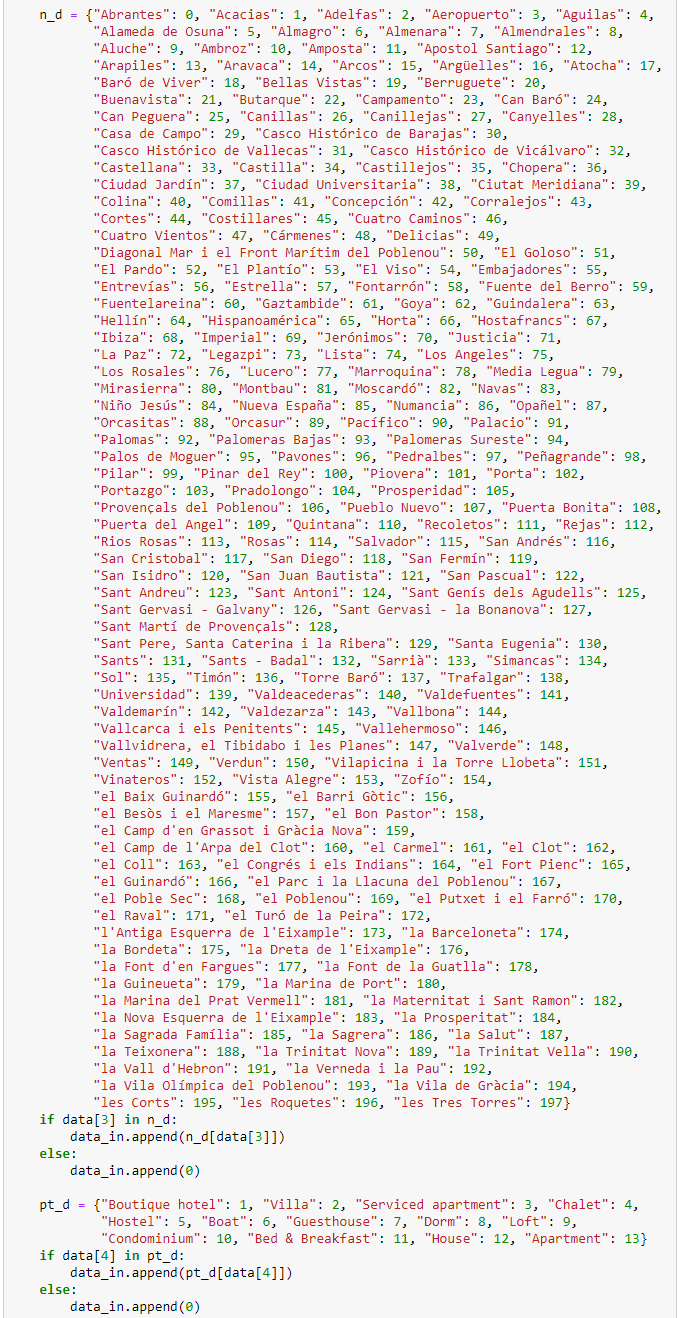
## **Módulo de Ejecución**

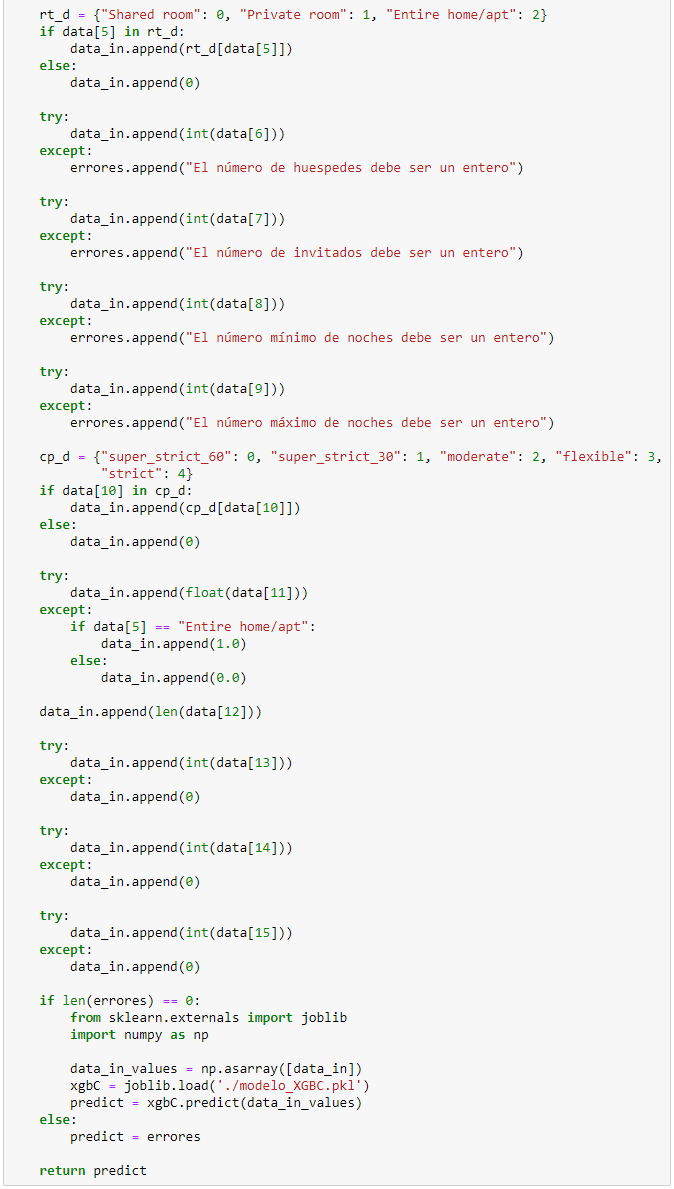
Implementaré ahora una función que contendrá el modelo predictor y que sería la que habría que invocar para ejecutarlo dentro del sistema recomendador de precios.

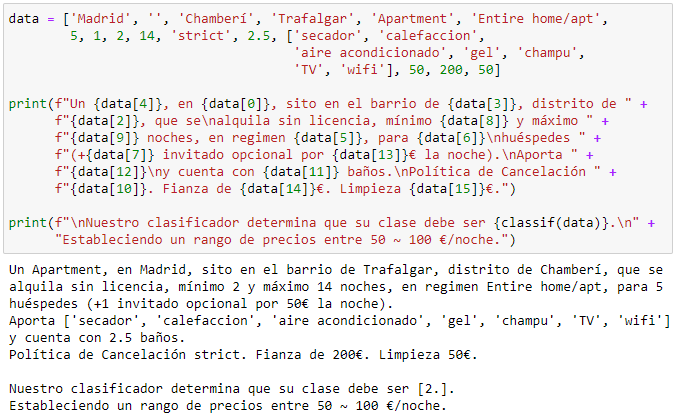
La función se debería invocar pasándole los datos adecuados, que el usuario hospedante ha introducido de su inmueble, ella se encargaría de transformarlos de forma adecuada para alimentar la entrada del modelo, a continuación, lo ejecutaría y finalmente devolvería la clase (rango de precios) predicha por él.

Los datos y el orden, que necesitará la función como parámetros de invocación, serían:

1. **City** -------------------------------------- “Barcelona” o “Madrid”
2. **License** ---------------------------------- Código de licencia o “” (vacío)
3. **Neighbourhood\_Group** ------------ Distrito
4. **Neighbourhood** ---------------------- Barrio
5. **Property\_Type** ----------------------- Tipo de propiedad
6. **Room\_Type** --------------------------- Tipo de alquiler
7. **Accommodates** ---------------------- Número de huéspedes incluido en el precio/noche
8. **Guests\_Included** --------------------- Número de invitados permitido
9. **Minimum\_Nights** -------------------- Número mínimo de noches para el alquiler
10. **Maximum\_Nights** ------------------- Número máximo de noches para el alquiler
11. **Cancellation\_Policy** ----------------- Tipo de política de cancelación
12. **Bathrooms** ---------------------------- Número de baños en el inmueble
13. **Amenities** ------------------------------ Lista con los servicios/lujos del inmueble
14. **Extra\_People** ------------------------- Precio/noche por cada invitado permitido
15. **Security\_Deposit** -------------------- Precio del depósito o fianza para el alquiler
16. **Cleaning\_Fee** ------------------------- Precio de la limpieza del inmueble





Un ejemplo de invocación y su resultado sería:

# **Presentación**

Para la parte de presentación de la PoC utilizaré Tableau.

No es mi pretensión montar un dashboard complicado con demasiadas gráficas o con representaciones de difícil interpretación. Mi idea es que, el cliente de AirBNB (el hospedante), reciba una información suficientemente clara de porqué el sistema le recomienda un determinado rango de precios para el alquiler de su inmueble, en función de, primero, los inmuebles que ya se alquilan en su zona y, segundo, del nivel de éxito que obtienen en función del precio/noche que ofertan.

Como fuente de datos para el módulo presentador tendremos el dataset completo de AirBNB, filtrado y limpio, con sus veintiuna variables.

Para mejor compresión, al ser la PoC una muestra sobre las ciudades de Madrid y Barcelona, modificaré el nombre de algunas columnas, que se utilizarán en las gráficas, traduciéndolas al castellano.

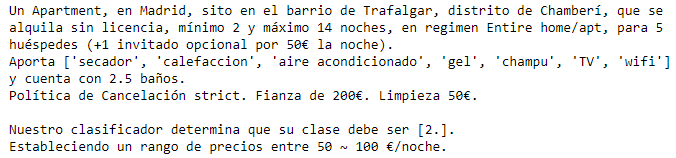
* City 🡪 Ciudad
* Neighbourhood Group 🡪 Distrito
* Neighbourhood 🡪 Barrio
* Property Type 🡪 Tipo de Inmueble
* Room Type 🡪 Tipo de Alquiler
* Accommodates 🡪 Huéspedes
* Range Prices 🡪 Rango Precios
* Ranges 🡪 Precios
* Reviews per Month 🡪 Alquiler por Mes

También crearé un par de jerarquías de variables que darán más sentido a las gráficas:

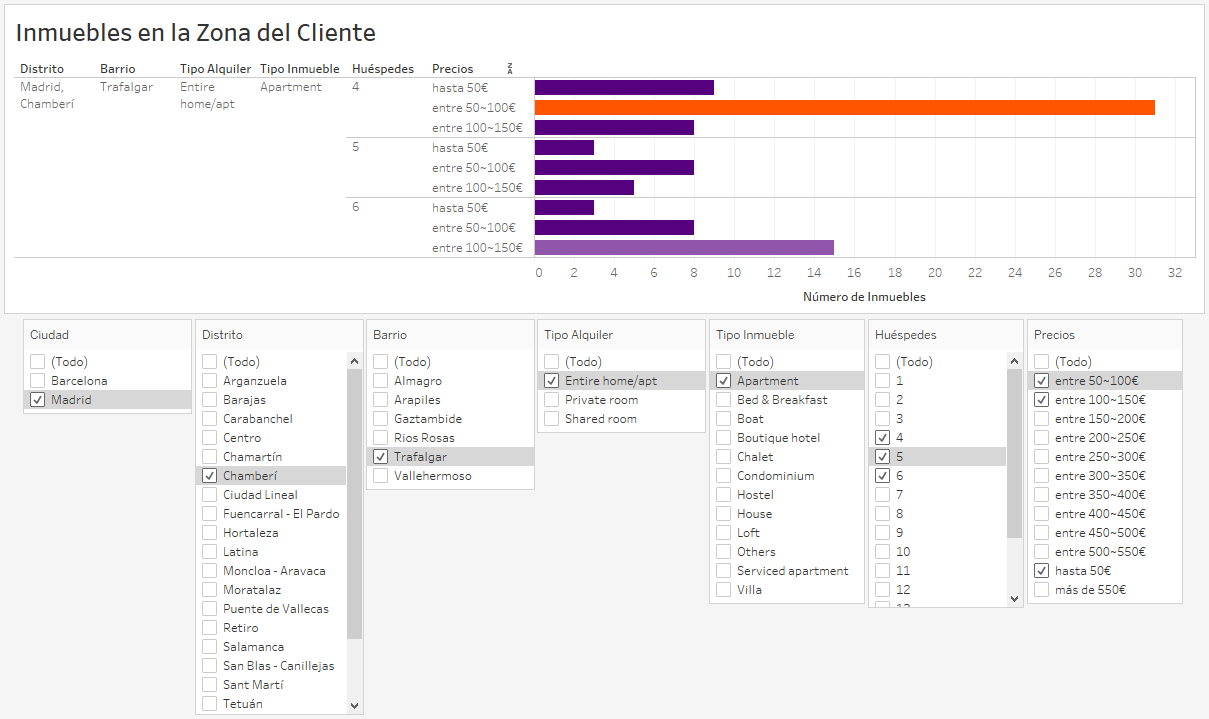
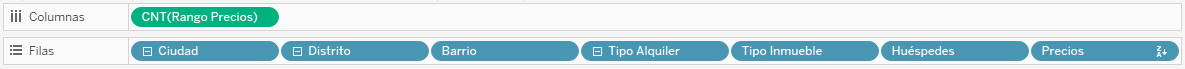
* Jerarquía Ubicación
  + Ciudad
  + Distrito
  + Barrio
* Jerarquía Tipo
  + Tipo de Alquiler
  + Tipo de Inmueble

## **Gráfica de Inmuebles en Zona Cliente**

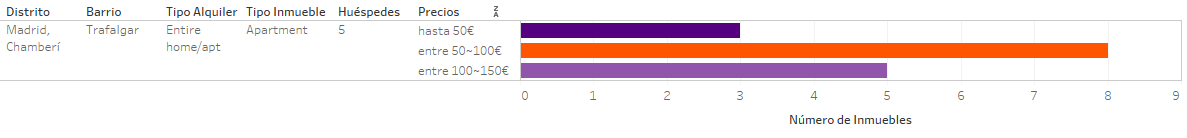
La primera comparativa que yo (como hospedante en AirBNB) quisiera ver sería cuántos inmuebles, con características similares al mío, se alquilan ya en la misma zona y a qué precios.

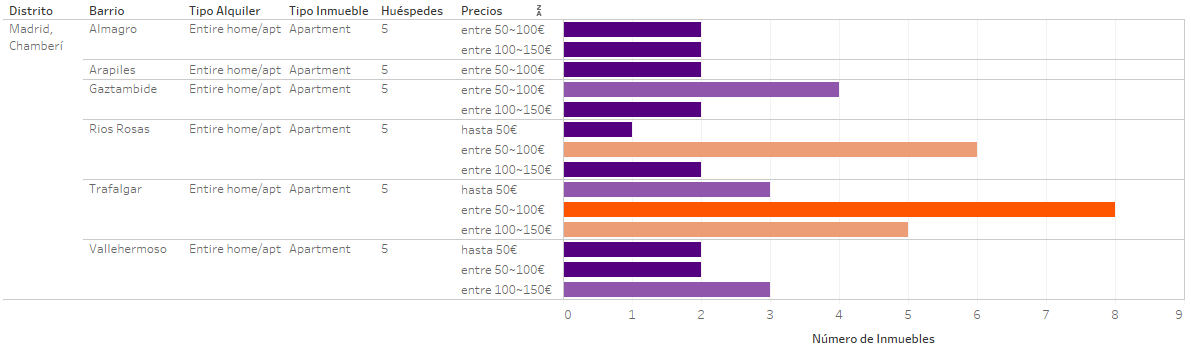
Utilizaré, para graficar con valores, el ejemplo del párrafo 12.3.

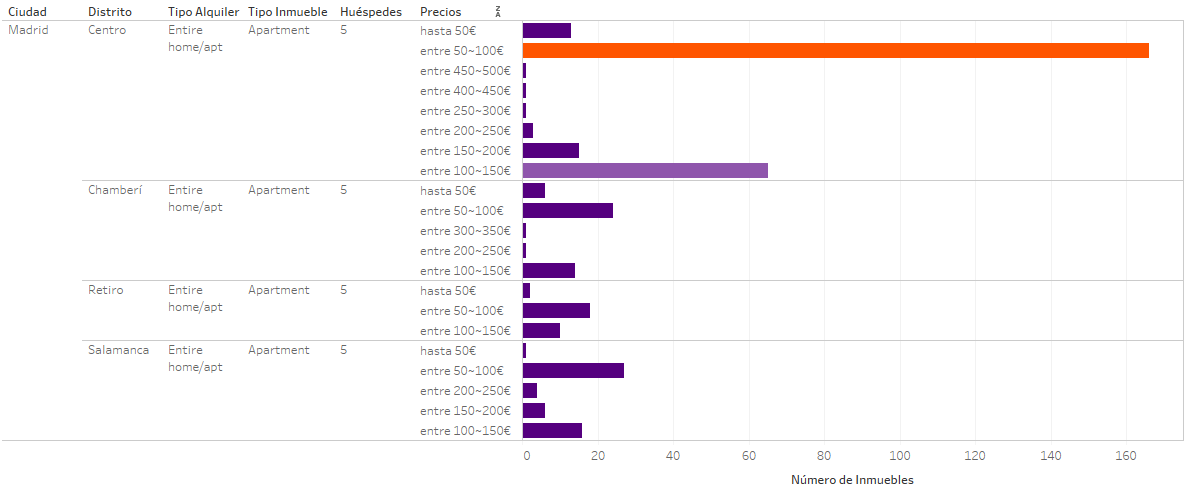
Para ello construiré una gráfica de barras horizontales con los siguientes desgloses (filtrados por defecto) sobre las que se medirá el número de inmuebles que responden a sus valores dimensionales.

1. Jerarquía Ubicación
   1. Ciudad (filtro = **Madrid**)
   2. Distrito (filtro = **Chamberí**)
   3. Barrio (filtro = **Trafalgar**)
2. Jerarquía Tipo
3. Tipo de Alquiler (filtro = **Entire home/apt**)
4. Tipo de Inmueble (filtro = **Apartment**)
5. Huéspedes (filtro = **4**, **5** y **6**)
6. Rangos de Precios (filtro = **hasta 50€**, **entre 50~100€** y **entre 100~150€**)

La gráfica aparecerá con los valores afines al inmueble del hospedante filtrados por defecto, pero será interactiva y permitirá al cliente seleccionar más opciones para poder ampliar o crear su propia comparativa.

En el ejemplo, el modelo clasificador asignó una categoría o clase 2 (precio/noche recomendado entre 50 y 100€) para el inmueble del cliente y, tal y como puede observarse en la gráfica, para cinco huéspedes, esta es la clase mayoritaria, con ocho inmuebles, en la zona para alquileres de su mismo tipo.

El cliente tendrá acceso a modificar los filtros por defecto y, por ejemplo, podría querer observar los valores, para inmuebles afines, en todos (o en algunos de) los barrios del distrito en el que se encuentra su inmueble.

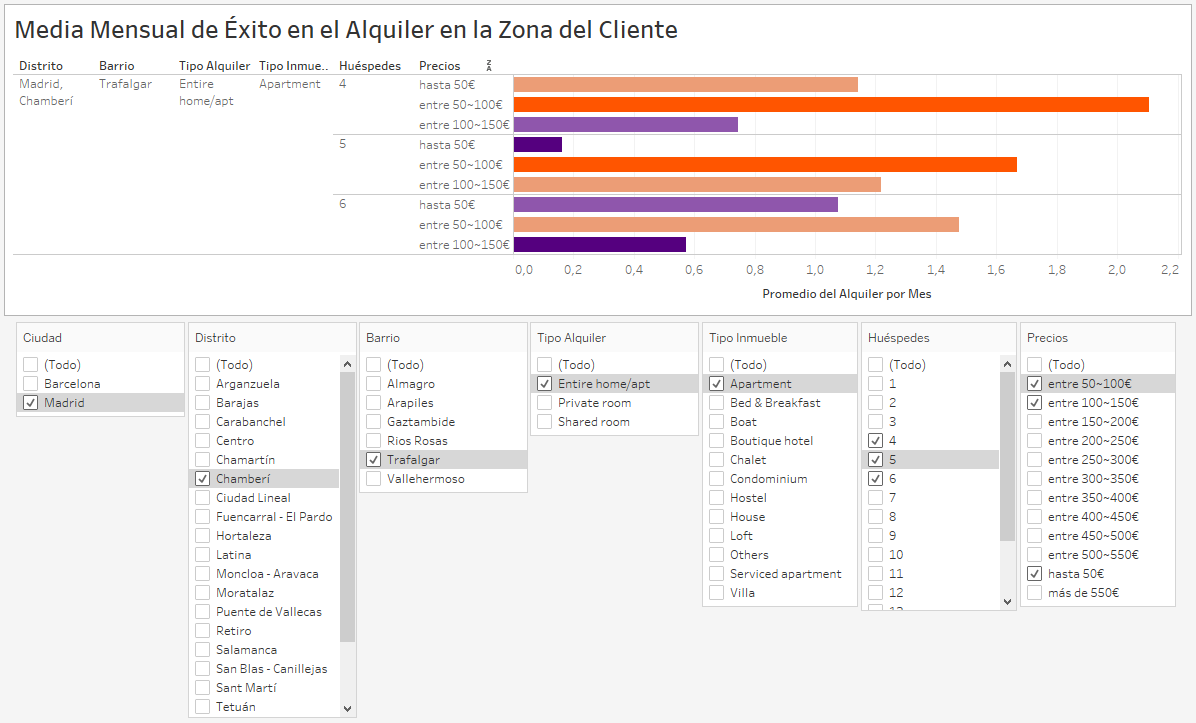
 También podría observar el número de inmuebles y sus precios de alquiler en los distritos cercanos.

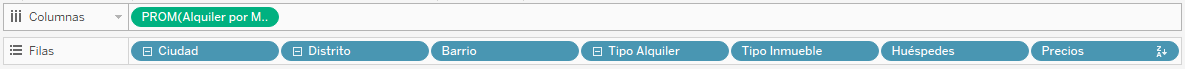
Como se puede ver, la gráfica permitirá que el usuario experimente con los filtros, ordenaciones y valores de una forma sencilla e intuitiva para saciar su curiosidad en cuanto al número de inmuebles que rodean al suyo.

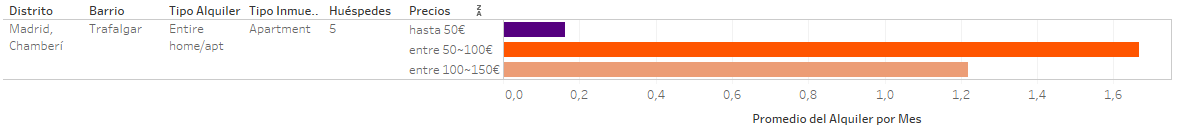
En definitiva, esta gráfica comparativa va a orientar e informar al potencial hospedante de la situación del alquiler en la ubicación de su inmueble justificando además la categoría o clase que el modelo clasificador ha elegido recomendarle.

## **Gráfica de Éxito en el Alquiler**

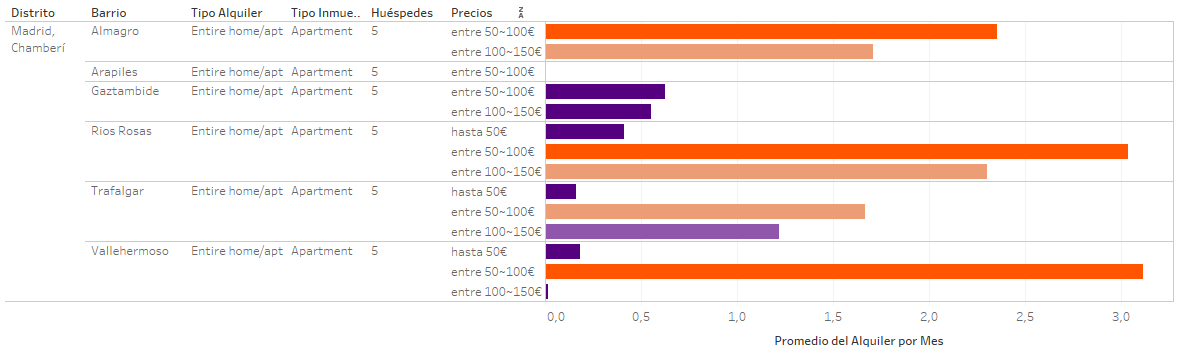
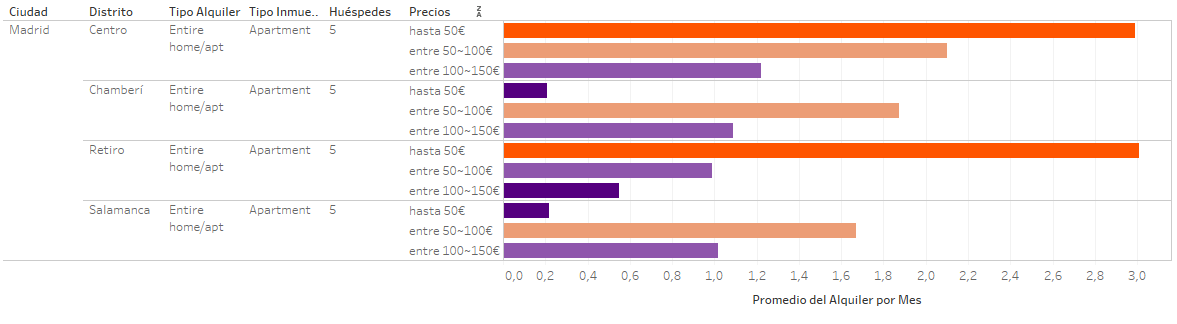
La segunda comparativa que yo quisiera ver sería qué media de éxito tienen, inmuebles con características similares al mío y en la misma zona, según su precio de alquiler por noche.

Siguiendo con los valores del ejemplo, construiré una gráfica análoga a la del párrafo anterior (para no romper el estilo de la presentación) de barras horizontales con los mismos desgloses (también filtrados por defecto), pero enfrentados al promedio de alquileres por mes.

La gráfica aparecerá filtrada por defecto con los valores del inmueble del hospedante, será también interactiva y permitirá al usuario seleccionar los valores que desee en los filtros.

En el ejemplo, el modelo clasificador asignó una categoría o clase 2 (precio/noche recomendado entre 50 y 100€) para el inmueble del cliente y, tal y como puede observarse en la gráfica, para cinco huéspedes, esta es la clase con mejor promedio, con 1,67 alquileres por mes, en la zona para alquileres de su mismo tipo.

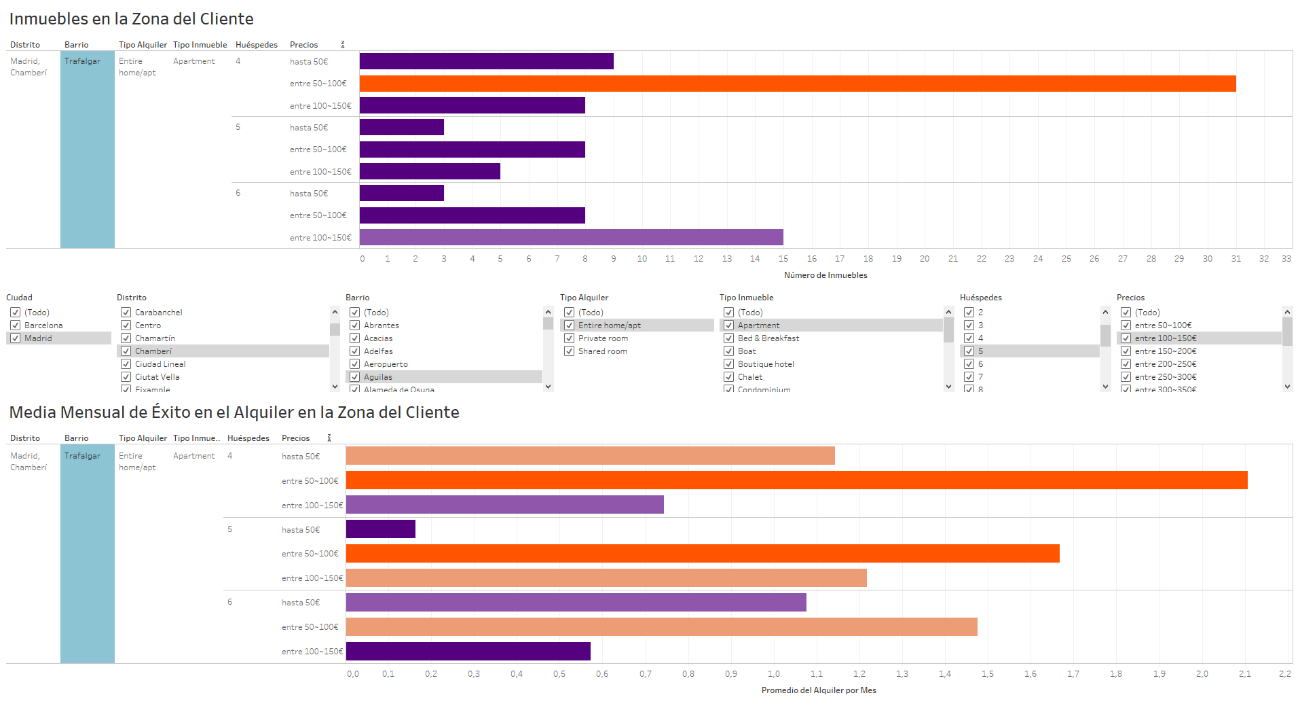
De idéntica forma a la gráfica del párrafo anterior, el cliente podrá modificar los filtros y ver, por ejemplo, el promedio de éxito en los barrios de su mismo distrito.

O en distritos cercanos al suyo.

Con esta gráfica se orientará al hospedante sobre el número de alquileres por mes que disfrutan otros inmuebles en su misma ubicación justificando también la recomendación del modelo clasificador.

## **Dashboard**

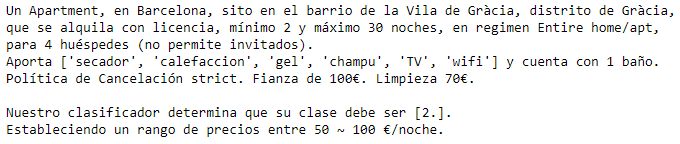
El dashboard de presentación sería sencillamente un montaje horizontal de las dos gráficas anteriores.

Entre ambas gráficas comparativas se sitúan los filtros sobre los que el usuario en puede interactuar, seleccionando/deseleccionando los diferentes valores de Ciudad, Distrito, Barrio, Tipo de Alquiler, Tipo de Inmueble, Número de Huéspedes y Rangos de Precios.

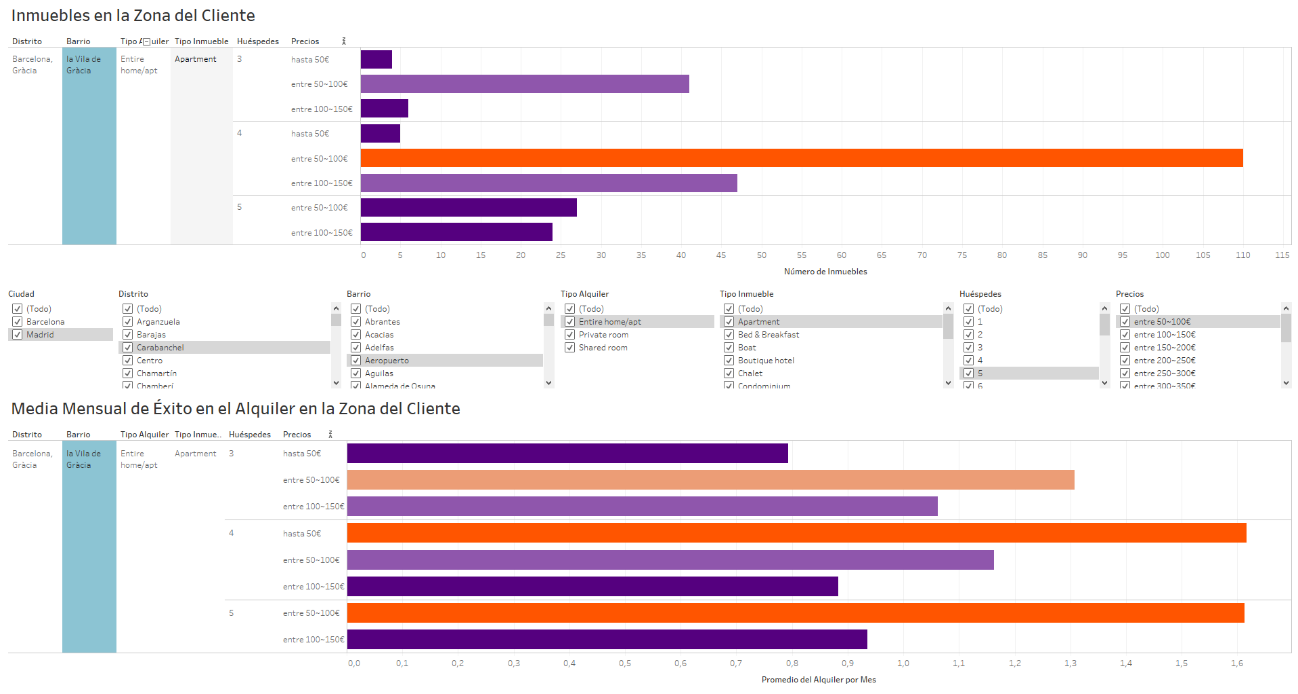
Los ejes de ambas gráficas también actúan como elementos de selección, haciendo que las comparativas sean totalmente interactivas entre ellas.

El objetivo es que el cliente (hospedante) pueda ver de forma clara la información y entienda porqué el sistema recomendador le ha indicado una determinada franja de precios como adecuada para el alquiler de su inmueble.

## **Un Ejemplo en Barcelona**

Para completar la PoC veamos un ejemplo en la ciudad de Barcelona.

En este caso el sistema recomendador también ha decidido que la clase adecuada es la 2.

Y el dashboard que se le presentaría al cliente sería:

En la presentación, en la primera gráfica se parecía que mayoritariamente en la zona hay 110 inmuebles análogos al del hospedante que se alquilan en un rango de precios entre 50 y 100€ (el recomendado).

En la segunda gráfica hay una pequeña divergencia y es que la franja de precios con mayor media de éxito en alquileres por mes, de inmuebles análogos al del hospedante, es la de hasta 50€, con 1,61 alquileres por mes (seguida eso si por la franja de precios recomendada con 1,16 alquileres por mes).

Esto viene también a dar sentido al pensamiento inicial de que no siempre el menor precio puede ser el más adecuado, ya que, aunque un precio más barato pueda tener un mayor éxito en número de alquileres, también esto exigirá una mayor dedicación del propietario hacia su inmueble.

Aún así parece que el modelo clasificador vuelve a acertar recomendando el rango de precios más adecuado para el alquiler óptimo del inmueble.

# **Comentario Final**

Como nota final de este trabajo fin de bootcamp me gustaría indicar que he querido realizar algo como esto porque aúna los conceptos tanto de Data Mining, como de Inteligencia Artificial, con algo de presentación en Tableau (una herramienta que en su día me pareció muy divertida y amena).

Este trabajo (y quizá esté mal que lo diga yo) no es un proyecto virtuoso o muy espectacular. Yo lo definiría como una solución sencilla, realizada con moderna tecnología, para un problema sencillo.

Quiero destacar que me decanté por los modelos de Machine Learning por dos razones básicamente, primero porque computacionalmente son más ligeros que los de Deep Learning y segundo porque tras probar algún modelo de red neuronal profunda (DAN) no me ofrecía una grandísima ventaja en términos de *accuracy* comparándolo con el modelo XGBoost.

Durante la elaboración de este proyecto he entendido porque estos modelos de gradiente extremo son los que dominan actualmente en inteligencia artificial (fuera del campo de las imágenes y del NLP). Son rápidos, ligeros y presentan unos niveles de precisión excelentes.

He querido dejar fuera del enfoque, o del *scope*, del proyecto todo lo que es integración. Y esto ha sido así porque realmente no tengo los conocimientos suficientes como para integrar un modelo predictor en un portal, ni tampoco para integrar su salida en el módulo presentador y éste después en el portal. Por tanto, mi trabajo se limita a aportar las piezas que conformarían o harían posible la solución.

Sin más, despedirme de este bootcamp y de todos los profesores.

Gracias.

**F. Javier Gonzálvez**

# **ANEXO**

Se incluyen aquí los notebooks, el modelo portable y el libro de Tableau.

