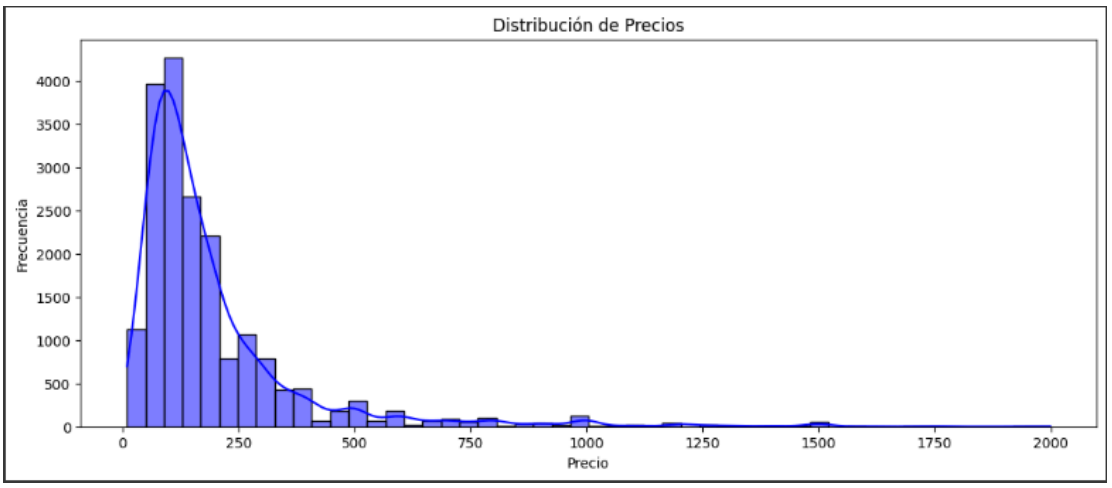
Introducción y objetivos:

El reporte a continuación fue realizado para predecir los precios de los hospedajes para algunas ciudades dentro de USA en base a la información de la plataforma Airbnb

Descripción del Dataset:

El dataset utilizado de se compone de 19.309 publicaciones y 29 variables, las cuales muestran algunas características de las propiedades, entre ellas podemos encontrar: el tipo de propiedad, el tipo de habitación disponible, el barrio donde se encuentran ubicadas, la cantidad de habitaciones, el tipo de cama e información sobre el alquiler como la política de cancelación, la opinión de los huéspedes, Fecha desde que el host se inició en airbnb, entre otros.

A continuación podemos observar la distribución de la variable objetivo:



Análisis exploratorio de datos:

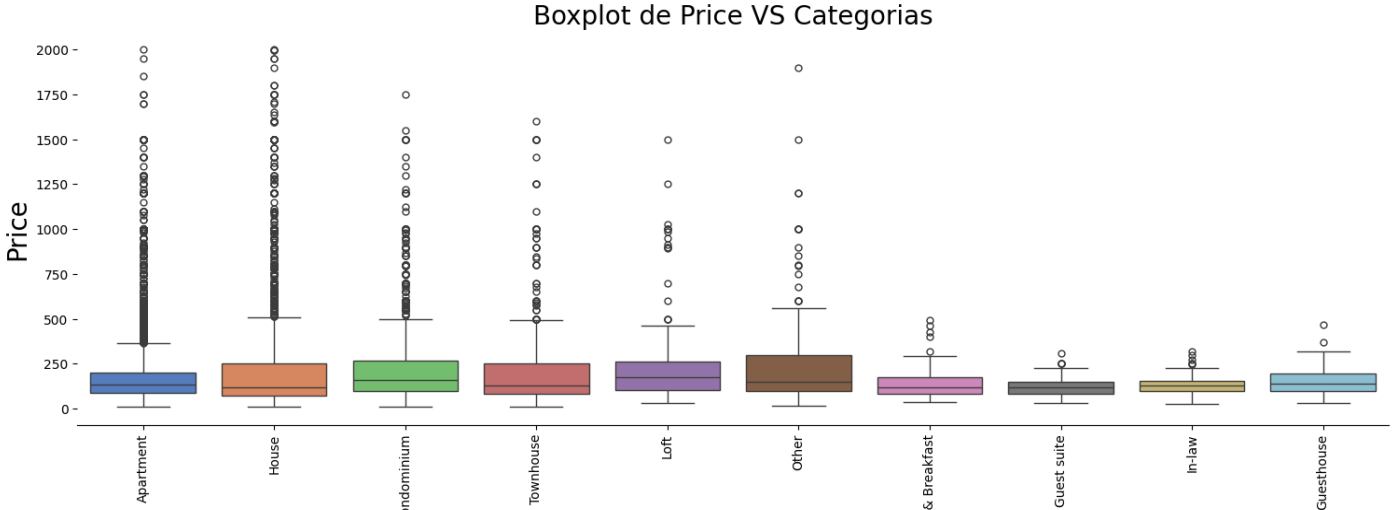
En base a la información del dataset notamos que son pocos los valores del tipo int o float, por lo que se deberá reemplazar las categorías en los features que correspondiere, convirtiendo las variables con el tipo de dato que les corresponde para su correcta manipulación:

- first_review, host_since, last_review → date
- cleaninig_fee, instant_bookable, host_identity_verified, host_has_profie_pic → boolean.
- host_response_rate → float.

A su vez, eliminamos las columnas ‘name’ y ‘thumbnail_url’ porque no son relevantes para el análisis.

En total hay 27 tipos de propiedades: ['House' 'Apartment' 'Loft' 'Townhouse' 'Condominium' 'Bungalow', 'Guesthouse' 'Dorm' 'Other' 'Bed & Breakfast' 'Boutique hotel' 'Hostel', 'In-law' 'Boat' 'Guest suite' 'Castle' 'Timeshare' 'Camper/RV' 'Cabin', 'Serviced apartment' 'Treehouse' 'Villa' 'Yurt' 'Tent' 'Train', 'Vacation home' 'Cave']

<u>Manipulación de Outliers:</u>
Para entender mejor cómo se distribuyen los datos, visualizamos cómo se distribuyen los precios según el tipo de propiedad:



El valor del percentil 97 es: $564.59 y el valor del percentil 1 es: $28.99
Analizamos la cantidad de outliers:

- Cantidad de valores por encima del percentil 95: 966
- Cantidad de valores por debajo del percentil 1: 172

Calculamos los límites inferior y superior:

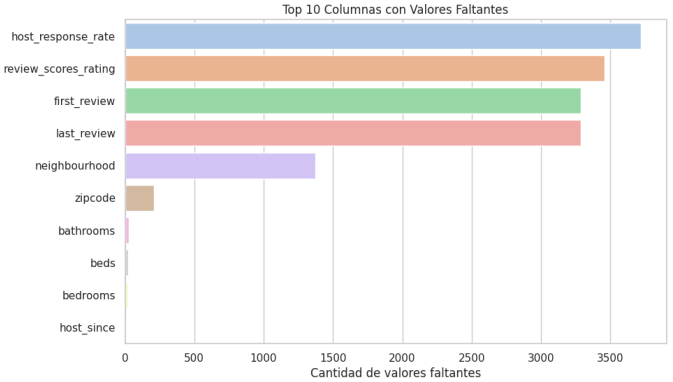
- Límite inferior: 20.49 (Q1- 0.5 * IQR, consideramos 0,5 dado que con 1.5 tomaba valores negativos)
- Límite superior: 419.5 (Q3 + 1.5 * IRQ)

Eliminamos los outliers para evitar sobre ajustar el modelo.

Manipulación de nulos:

Dado que las variables 'host_response_rate' y 'last_review' no representan información relevante al modelo como para eliminar samples debido a valores NaN, es por esto que estas feature no serán tomadas para el análisis futuro, eliminandolas del data frame.

En las variables 'bathroom', 'bedroom' y 'beds' debido a la baja proporción de variables NaN, eliminamos solo aquellos registros vacíos.



La variable host since la popularemos tomando como referencia la first review donde esté disponible, a su vez, en algunos registros, la fecha de 'first_review' puede ser anterior a 'host_since'. En esos casos, actualizamos la fecha de 'first_review' para que sea igual a 'host_since' en estos casos. Si las fechas fueron leídas en el orden incorrecto (es decir, first_review > host_since), intercambiar los valores de las columnas 'first_review' y 'host_since'. Luego de popular los valores de 'host_since', eliminamos la variable 'first_review'.

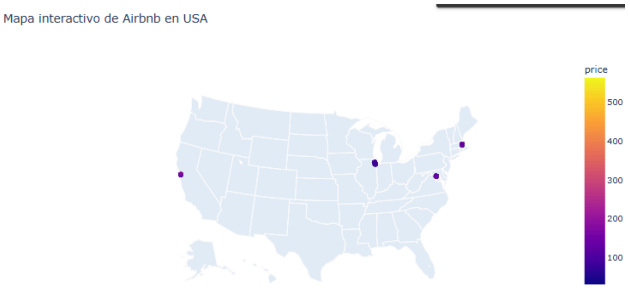
Para la variable 'reviews_score_rating', en lugar de eliminar aquellas filas que contengan nulos, se optará por reemplazarlo por el valor medio del feature. A su vez, en la variable 'neighbourhood' también popularemos los valores nulas, según el zipcode. Las variables que no posean 'zipcode' ni 'neighborhood' serán eliminadas.

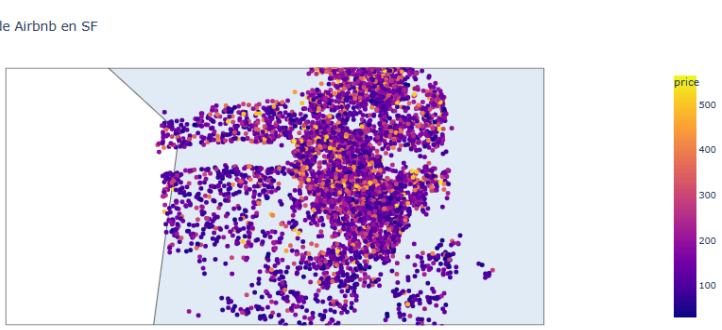
Codificación de variables categóricas:

Para la variable 'room_type' creamos una sola variable que codifica a cada categoría con un número secuencial. Este tipo de codificación se utiliza para categorías que tienen algún orden o que indican distancia entre sus clases. Hacemos lo mismo para la variable 'cancellation_policy_mapping', haciendo que tome el valor mínimo igual a 0 cuando la política es *flexible* y el valor máximo (5) cuando la política es *super_strict_60.*

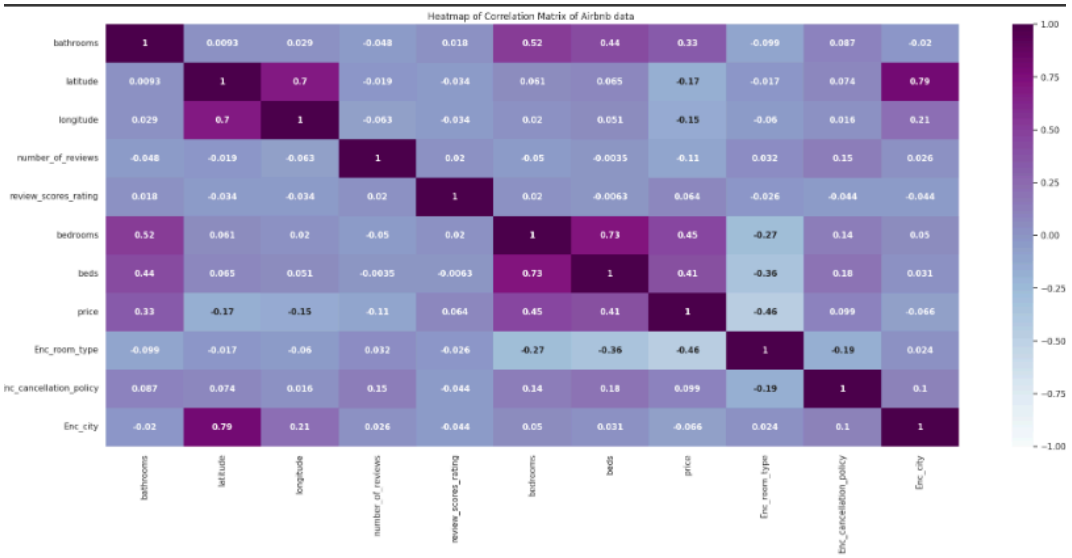
Para la variable 'property_type', aplicamos un Label Encoder, un proceso que transforma las categorías en valores numéricos binarios, generando una nueva característica para cada categoría existente. En el caso de la variable 'amenities', realizamos un procedimiento similar, pero debido a que es una variable de tipo cadena, fue necesario realizar un preprocesamiento previo para habilitar su codificación. Para ello, definimos un listado de 27 categorías de amenities, que sirvió como base para clasificar todas las posibles comodidades de las propiedades. Posteriormente, generamos una dummy para cada una de estas categorías. Finalmente, analizamos la variable 'amenities' de cada registro y establecimos su correspondencia con las dummies generadas.

Luego elaboramos unos heatmaps para poder visualizar cómo se distribuyen los datos en el plano:



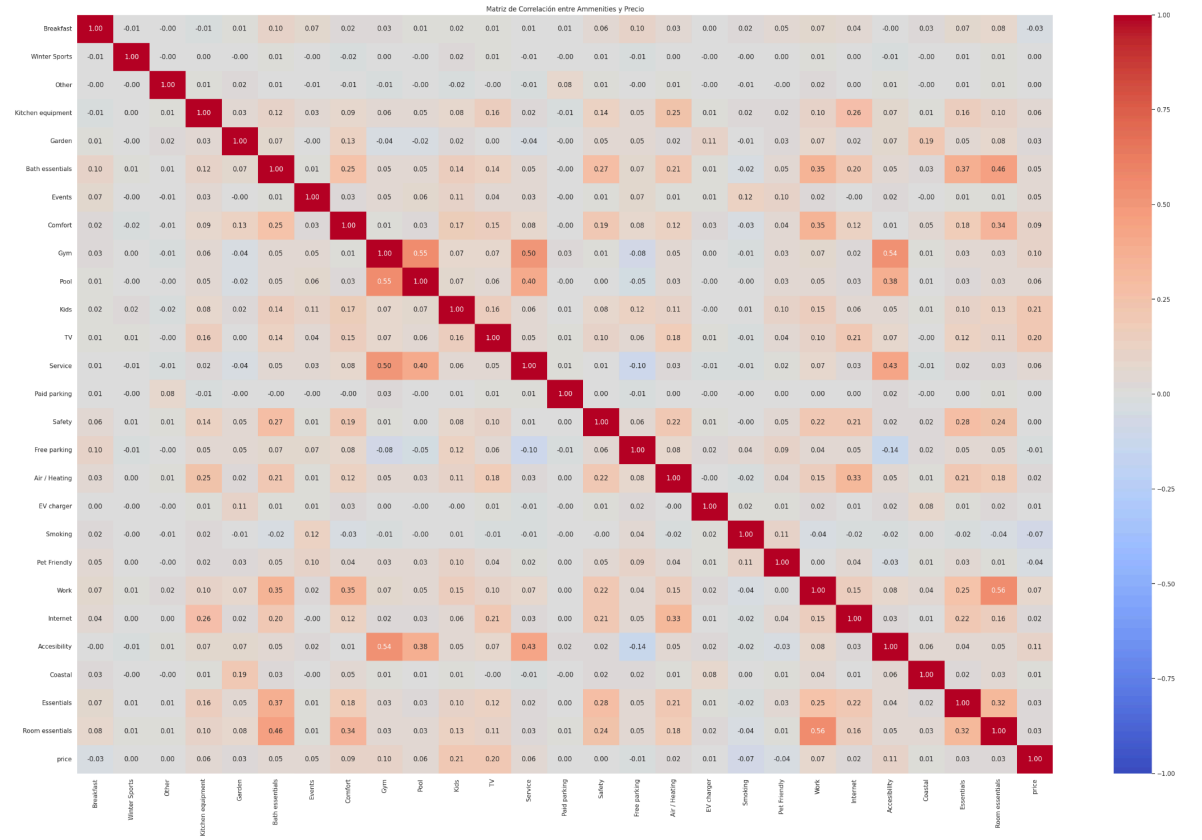


Por último, elaboramos una matriz de correlaciones para entender cómo se relacionan las variables:



De esta matriz podemos ver que las variables 'beds', 'bedrooms' y 'bathrooms' se encuentran altamente relacionadas con el precio y que variable 'room_type' tiene una relación inversa, es decir cuanto mayor valor tome la variable (shared rooms), menor será el precio.

Además decidimos llevar a cabo el mismo análisis con las amenities, para entender cuáles podrían ser aquellas que influenciaran el precio:



De esta matriz obtenemos que las amenities dentro de las categorías 'Kids', 'TV', 'Gym' and 'Accessibility' se encuentran relacionadas con el precio.

Materiales y métodos:

Las librerías utilizadas para llevar a cabo el análisis exploratorio fueron: Pandas, Numpy, Seaborn, Matplotlib y sklearn, gdown.

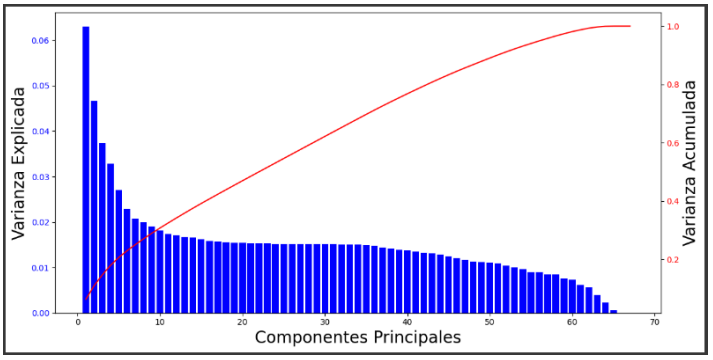
Experimentos y resultados:

Sobre el análisis realizado anteriormente, llevamos a cabo la predicción del precio de los hospedajes para algunas ciudades dentro de USA con el modelo de regresión lineal. Comenzamos separando las columnas numéricas de las categóricas. Luego, realizamos un proceso de preprocesamiento de datos, que incluyó varias etapa: en primer lugar, para manejar los valores nulos, los valores faltantes en las variables numéricas se rellenaron con la media de cada columna, mientras que en las variables categóricas se reemplazaron por el valor más frecuente de cada columna. A continuación, las variables numéricas fueron escaladas para asegurar que todas las características tuvieran una escala similar, lo que es esencial para mejorar la precisión y la estabilidad del modelo. Por otro lado, las variables categóricas fueron codificadas para transformarlas en un formato adecuado para el modelo de regresión.

Una vez procesadas y transformadas todas las variables, combinamos las variables numéricas y categóricas en un solo dataframe. Posteriormente, dividimos los datos en dos conjuntos: el 80% de los datos se utilizó para el entrenamiento del modelo, mientras que el 20% restante se reservó para la prueba.

Con los datos de entrenamiento listos, aplicamos un escalado automático para asegurar que todas las características estuvieran en la misma escala, lo cual optimiza el desempeño del modelo. Tras entrenar el modelo de regresión lineal con los datos de entrenamiento, evaluamos su rendimiento utilizando el conjunto de prueba. Las métricas obtenidas fueron un Error Cuadrático Medio (MSE) de 5941.5424 y una Raíz del Error Cuadrático Medio (RMSE) de 77.0814, lo que nos permitió cuantificar la precisión del modelo en las predicciones realizadas.

Luego, para mejorar las predicciones, llevamos a cabo un Análisis de Componentes Principales, una técnica estadística utilizada para reducir la dimensionalidad de un conjunto de datos, manteniendo la mayor cantidad posible de información en el proceso. Se definieron un total de 67 componentes principales para hacer el análisis y se ajustó el modelo PCA a los datos previamente escalados. Luego, se transformaron los datos para obtener el conjunto proyectado en los nuevos componentes.



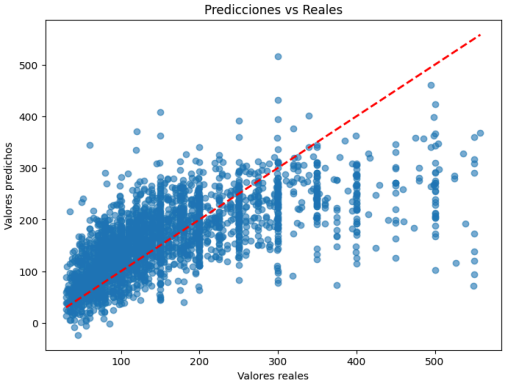
Se calcularon la varianza explicada y la varianza acumulada para entender cuánta información se conservaba.Visualizamos en las barras azules la varianza explicada por cada componente y la linea roja representa la varianza acumulada, la cual sirve para determinar cuantos componentes son necesarios para capturar la mayor parte de la variabilidad de los datos.

Luego, mediante `n_comps_95 = next(i for i, v in enumerate(eigenvalues_acum) if v >= 0.95) + 1` se determinó el número de componentes necesarios para explicar el 95% de las varianza, con lo cual obtuvimos un resultado igual a 57.

Una vez obtenidos estos resultados, calculamos nuevamente la regresión lineal donde obtuvimos los siguientes resultados:

- Error cuadrático medio (MSE): 5748.0991
- Coeficiente de determinación (R^2): 0.4419
- El RMSE en el conjunto de prueba es: 75.8162

Podemos notar que luego de aplicar el PCA, el MSE ha disminuido y el error absoluto medio en este caso es de 53.7983756



Representamos la relación entre las predicciones y los valores reales:

Con los resultados del PCA calculamos nuevamente la regresion lineal y obtenemos los siguientes resultados:

- Error Absoluto Medio (MAE): 53.7984
- Error Cuadrático Medio (MSE): 5748.0991
- Raíz del Error Cuadrático Medio (RMSE): 75.81621

Para mejorar las predicciones, aplicamos un modelo Random Forest donde obtuvimos los siguientes resultados:

- Error Absoluto Medio (MAE) - Random Forest: 52.8014
- Error Cuadrático Medio (MSE) - Random Forest: 5734.7246
- Raíz del Error Cuadrático Medio (RMSE) - Random Forest: 75.72796

Por último, aplicamos un GridSearch. Este es un proceso de optimización de hiperparámetros buscando mejorar el rendimiento del Radom Fores. El objetivo es encontrar la mejor combinación de hiperparámetros que minimiza el error de las predicciones y maximiza la precisión del modelo.

Discusión y conclusiones:

Podemos decir que a lo largo del proceso se observó una mejora en la precisión del modelo. La utilización del PCA tuvo un impacto positivo en el modelo, permitiendo reducir el número de variables pero manteniendo la mayor parte de la varianza explicada. La utilización del Random Forest resultó ser el mejor modelo para la predicción de precios. Por último, la implementación del GridSearch permitió encontrar la mejor combinación de hiperparámetros, lo que contribuyó a mejorar levemente las predicciones.
Para mejorar la predicción, se podría considerar la inclusión de variables externas como la estacionalidad o eventos en las ciudades, para entender mejor la variabilidad de los datos.

Referencias:

https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=8cf718c852722f641e13abebbc0f4bfa11b0170b#page=81