Introducción:

Se nos presentó el desafío de seleccionar y entrenar dos modelos de Machine Learning que pudieran predecir el precio de un alquiler en la aplicación Airbnb por ciudad en USA, en función de distintas variables que aportaba un dataset en particular. A continuación, veremos en detalle los inconvenientes que tuvimos que sortear en cada instancia del trabajo para lograr el objetivo.

Antes de implementar los modelos de ML, tuvimos que realizar los siguientes pasos.

Análisis exploratorio de los datos (EDA):

En primer lugar, debemos entender cómo está construido el dataset, qué dimensiones tiene, las variables que contiene y el tipo de datos que almacena en cada una de ellas. También es necesario corroborar que el dataset sea consistente y no presente 'huecos' o valores nulos en sus registros.

Nuestro dataset contaba con 19.309 registros (publicaciones en la app) y catalogaba un total de 29 variables. Muchas de esas variables albergaban datos del tipo numérico (int, float) pero la mayoría eran del tipo objeto. Estas ultimas eran en realidad strings que había que procesar y transformar en variables categóricas para poder abastecer a los distintos modelos de ML. De todas maneras, al explorar una muestra del contenido de esas variables, muchas fueron descartadas ya que no contenían información relevante para nuestro estudio: desde textos únicos como 'id' o 'nombre' hasta variables como 'review score rating' que suelen estar muy poco variadas y poco relacionadas con el precio ya que hay distintas expectativas, y por ende calificaciones, en función de cada rango de precio.

Habiendo eliminado las dimensiones irrelevantes, procedimos al tratamiento de los valores nulos dentro de las variables que quisimos conservar. Utilizamos dos criterios distintos para completar los registros dependiendo del tipo de variable. Para el caso de 'neighbourhood' (variable categórica) decidimos tratarla como string, la agrupamos por ciudad y decidimos completar los nulos con los valores más frecuentes de cada ciudad correspondiente. Para el resto de columnas con valores nulos, al ser todas variables numéricas, decidimos rellenar con la mediana de cada dimensión.

Habiendo realizado estas transformaciones al data frame, aun había que transformar las variables categóricas en numéricas. Para ello generamos las llamadas variables 'dummy' donde cada dimensión se transforma en una columna binaria. Una vez creadas, quitamos del data frame original las columnas categóricas y agregamos las dummies para obtener el data frame limpio que alimentará el modelo de ML en los siguientes pasos.

Manipulaciones del data frame y entrenamiento ML:

Importamos el data frame limpio producto del EDA y separamos las variables independientes 'x' de la dependiente 'y' a predecir (el precio de los Airbnb). Luego hicimos una partición de los datos tanto 'x' como 'y' entre train y test con una proporción 80-20, es decir que el tamaña de muestras de testeo es el 20% del total del data frame original.

PL1 – Regresión lineal

Para el primer modelo de machine learning utilizamos una regresión lineal simple. Es un modelo sencillo y poco demandante que suele fijar una base para comparar otros modelos de mayor complejidad [1][3]. Dada la dispersión entre los valores numéricos de las variables, sobre todo teniendo en cuenta las dummies, tuvimos que escalar los datos antes de correr el modelo.

PL2 – Gradient Boosting + PCA

En este caso, para el segundo modelo decidimos utilizar algo más complejo con la esperanza de obtener un mejor resultado. El modelo de Gradient Boosting es una técnica que entrena arboles de decisión de manera secuencial donde cada nuevo árbol corrige los errores del anterior [2]. Es capaz de captar relaciones no lineales entre las variables, aunque puede incurrir en el overfitting y es más demandante a nivel recursos. Fijamos un numero de 100 árboles con 3 niveles de profundidad, probando dos learning rates distintos y un threshold que filtra las variables que tienen una varianza menor a 0.01, es decir que son casi constantes. Para agilizar aun más el proceso, introducimos una reducción de dimensionalidad PCA. Esta ordena y selecciona los componentes del data frame que guardan mayor relación con la varianza de 'price', eliminando las variables correlacionadas entre sí (por ende, redundantes) y acelera el entrenamiento [4]. Esta técnica también puede ayudar a disminuir el riesgo de sobreajuste, al filtrar las variables que no son tan relevantes, reduciendo ruido.

Luego utilizamos grid search para encontrar la mejor combinación de hiperparamentros (todos fijos salvo el learning rate por una cuestión de recursos computacionales).

Predicciones y score de los modelos:

Una vez entrenados ambos modelos con el set de training, estamos en condiciones de correr las predicciones con la porción de datos separada previamente para el testeo.

Una vez obtenidos los resultados, queda calcular las métricas con las que evaluaremos la performance de ambos modelos. Creímos conveniente utilizar la raíz del error cuadrático medio o RMSE y el coeficiente de determinación o R2 [1][3]:

El RMSE mide cometido por el modelo penalizando las grandes deviaciones al elevar las diferencias al cuadrado. Nos permite comparar a nivel absoluto con la variable objetivo, ya que están en las mismas unidades.

El R2 mide la porción de la varianza que el modelo logra capturar de la variable objetivo. Resulta clave para comparara modelos de diferente complejidad, como es nuestro caso. Se mueve en un rango entre 0 y 1, siendo 1 indicativo de que el modelo es capaz de explicar el 100% de la varianza y 0 que el modelo no es mejor que directamente predecir la media de 'price'

En nuestro caso, obtuvimos los siguientes resultados:

PL1 modelo de Regresion Lineal -> RMSE= 163.74 -- R2= 0.355

PL2 modelo Gradient Boosting + PCA -> RMSE= 160.95 -- R2= 0.376

Conclusión:

Como supusimos en un principio, el modelo más sencillo de regresión lineal obtuvo un peor resultado en ambas mediciones. Gradient Boosting + PCA logra reducir el error absoluto en relación a los valores reales de la variable 'price' y también aumenta la porción de variabilidad que captura. De todas formas, aunque el PL2 generaliza mejor, es evidente que ambos modelos no logran hacer una buena predicción de la variable objetivo, capturando una porción baja de la varianza (R2<40%). Esto puede deberse a que el modelo esté sobre ajustando o que las variables relevadas en el dataset no puedan explicar en gran parte la varianza del precio de los Airbnb por ciudad en USA. Tal vez haya otro aspecto no contemplado en la base proporcionada que logre explicar de manera más contundente las variaciones en dicha variable.

Referencias:

1. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning: with applications in R. Springer.
2. Friedman, J. H. (2001). Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics, 29(5), 1189–1232.
3. James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning. Springer.
4. Jolliffe, I. T. (2002). Principal Component Analysis. Springer Series in Statistics.