2do Cuatrimestre 2024

Trabajo Práctico 1

Org. Datos 75.06/95.58

Gino Valentino Oggero - goggero@fi.uba.ar

Isabella Schneider - isabellaschneider1@gmx.de

Rita Castro Lobo - ritacastrolobo@tecnico.ulisboa.pt

1. EJ1: Análisis Exploratorio

Este ejercicio tiene como objetivo realizar un análisis exploratorio de un conjunto de datos de taxis. A través del análisis de las variables principales, la identificación de valores atípicos y la generación de nuevas *features*, buscamos comprender mejor el comportamiento de las carreras de taxis en relación con la distancia, el costo, el tiempo y otros factores relevantes.

Asimismo, se preprocesaron los datos eliminando valores faltantes, tratando outliers y generando variables adicionales que ayuden a mejorar los análisis posteriores.

Además, se han planteado hipótesis sobre la relación entre las diferentes variables, las cuales fueron verificadas a través de análisis estadísticos y gráficos. Las visualizaciones generadas ayudan a entender cómo se distribuyen los datos, permitiendo obtener conclusiones preliminares sobre los patrones de uso de los servicios de taxis.

1.1. Descripción del Dataset

A continuación, se describe brevemente la estructura del dataset y los resultados del análisis exploratorio de un conjunto de datos relacionados con viajes de taxis Yellow Cab en *New York* durante el año 2023. El análisis se hice para los meses de abril, mayo y junio.

El dataset utilizado contiene información detallada de las carreras de taxis, incluyendo el monto de las tarifas, la distancia y duración, el número de pasajeros hasta los métodos de pago utilizados.

Cantidad de Registros y Columnas:

- Cantidad de Registros: 10109133, que representan viajes de taxi individuales.
- Cantidad de Columnas: 24, que incluyen tanto variables numéricas como categóricas.

Las variables en este conjunto de datos incluyen tanto datos numéricos como categóricos, que requieren diferentes técnicas de preprocesamiento y análisis. Algunas de las columnas más destacables incluyen:

- **fare_amount (Monto de la tarifa):** Representa el coste del viaje en dólares. Este valor es numérico y es crucial para los análisis de costos y optimización de precios.

- **trip_distance (Distancia de la carrera):** Muestra la distancia recorrida en millas, lo cual es útil para evaluar la eficiencia y el comportamiento de los viajes.
- **pickup_datetime (Fecha y hora de recogida):** Esta columna contiene datos de fecha y hora, lo que nos permite extraer información temporal relevante como el día de la semana, la hora del día y la estación del año, que a su vez pueden influir en la duración o el coste de los viajes.
- **passenger_count (Cantidad de pasajeros):** Una variable numérica que nos ayuda a entender el comportamiento del servicio en función del número de personas transportadas.

A continuación se presentan los tipos de datos de cada columna.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10109133 entries, 0 to 10109132
Data columns (total 24 columns):
 #   Column                 Dtype
---  ------                 -----
 0   VendorID               int32
 1   tpep_pickup_datetime   datetime64[us]
 2   tpep_dropoff_datetime  datetime64[us]
 3   passenger_count        float64
 4   trip_distance          float64
 5   RatecodeID             float64
 6   store_and_fwd_flag     category
 7   PULocationID           category
 8   DOLocationID           category
 9   payment_type           int64
 10  fare_amount            float64
 11  extra                  float64
 12  mta_tax                float64
 13  tip_amount             float64
 14  tolls_amount           float64
 15  improvement_surcharge  float64
 16  total_amount           float64
 17  congestion_surcharge   float64
 18  Airport_fee            float64
 19  pickup_datetime        datetime64[us]
 20  day_of_week            object
 21  trip_duration          float64
 22  month                  int32
 23  pickup_hour            int32
dtypes: category(3), datetime64[us](3), float64(13), int32(3), int64(1), object(1)
memory usage: 1.5+ GB
```

Figura 1: Tipo de datos de las columnas del dataset

1.2. Hipótesis o Supuestos

En este análisis exploratorio del dataset de taxis, formulamos varias hipótesis clave basadas en la estructura de los datos y en comportamientos esperados en los patrones de viaje. Estas hipótesis buscan explicar las relaciones entre distintas variables y ayudar en la interpretación de los resultados.

Primero, se espera que exista una correlación positiva entre la distancia recorrida en la carrera y el monto de la tarifa. Es decir, cuanto mayor sea la distancia del viaje, mayor debería ser el monto a pagar. Esto es coherente con la forma en que normalmente se calculan las tarifas

de taxi, donde el precio aumenta a medida que la distancia recorrida es mayor. Esta relación será evaluada a través del análisis de correlación entre las variables *trip_distance* y *fare_amount*, con la expectativa de obtener un coeficiente de correlación positivo.

La segunda hipótesis plantea que los viajes realizados durante días festivos o fines de semana presentan diferencias significativas en cuanto a costo y duración en comparación con los días laborales. Se espera que en los días festivos haya menos disponibilidad de taxis, lo que podría resultar en un aumento de las tarifas debido a la mayor demanda. Para investigar esto, se creó una variable denominada *is_holiday*, que identifica si un viaje ocurrió en un día festivo, permitiendo así comparar los patrones de costo y duración en días festivos versus días normales.

Estas hipótesis reflejan supuestos basados en patrones de comportamiento urbano y estacional que pretendemos validar a través de un análisis detallado de las variables mencionadas, para así obtener una mejor comprensión de los factores que afectan los viajes en taxi.

1.3. Preprocesamiento de Datos

1.3.1. Eliminación de columnas

No se eliminaron columnas en esta etapa, ya que todas aportaban información relevante para el análisis y la construcción de nuevos *features*. Se mantuvieron las columnas originales como *PULocationID* y *DOLocationID* para estudiar los patrones de ubicación de recogida y destino de los viajes.

1.3.2. Correlaciones interesantes

Se detectaron algunas correlaciones interesantes, destacando:

- **Distancia del viaje (trip_distance) y la tarifa (fare_amount)**: Esta correlación resultó significativa, con un coeficiente de **0.68**, lo que muestra que a mayor distancia, mayor es la tarifa del viaje.
- **Duración del viaje (trip_duration) y tarifa total**: Se observó otra correlación positiva moderada con un coeficiente de **0.63**, lo que refleja que cuanto más larga es la duración del viaje, mayor suele ser el costo.

Estas correlaciones ayudaron a entender las relaciones más importantes entre las características cuantitativas.

1.3.3. Generación de nuevas *features*

Se generaron varios nuevas *features* a partir de las columnas originales:

- **price_per_mile**: Tarifa por milla recorrida, calculada como *fare_amount* dividido por *trip_distance*. Esta característica es útil para entender cómo varía el costo relativo en función de la distancia recorrida.
- **pickup_hour**: Se extrajo la hora de recogida del viaje a partir de la columna de fecha y hora (*pickup_datetime*), lo que permite analizar los patrones de viajes por hora del día.
- **distance_category**: Esta nueva variable categoriza los viajes en tres grupos: corto, mediano y largo, con base en los valores de la distancia del viaje (*trip_distance*), facilitando análisis específicos para cada grupo de distancia.
- **trip_duration**: Se calculó la duración del viaje en minutos utilizando las columnas de fecha y hora de inicio (*pickup_datetime*) y de finalización del viaje (*dropoff_datetime*).
- **time_of_day**: Clasificación del horario del viaje en intervalos: *morning*, *afternoon*, *evening* y *night*, basándose en la hora de recogida, permitiendo analizar cómo varían las tarifas y la duración de los viajes en función de la hora.
- **cost_per_passenger**: Calculado como la tarifa del viaje (*fare_amount*) dividida por el número de pasajeros, esta variable mide el costo por pasajero y ayuda a evaluar si hay diferencias significativas en los viajes compartidos.
- **is_holiday**: Columna binaria que indica si el viaje ocurrió en un día festivo, lo que permite analizar cómo los patrones de viaje varían entre días laborables y fines de semana.
- **trip_season**: Se creó una variable que clasifica los viajes según la estación del año en la que ocurrieron, lo cual es útil para identificar patrones estacionales en las tarifas y la duración de los viajes.

- **fare_per_minute**: Esta variable se calculó como la tarifa dividida por la duración del viaje, lo que proporciona una medida de la eficiencia del costo por minuto.

Estos nuevos *features* se crearon con el objetivo de mejorar el análisis y enriquecer los datos para futuros modelos predictivos.

1.4. Valores atípicos

La detección de valores atípicos o *outliers* es una parte fundamental del preprocesamiento de datos, ya que estos pueden afectar el rendimiento de los modelos y distorsionar los resultados del análisis. A continuación, se describen las técnicas empleadas para la identificación de outliers, así como las decisiones tomadas para su tratamiento.

```
Número de outliers na coluna VendorID: 1967
Número de outliers na coluna passenger_count: 2462244
Número de outliers na coluna trip_distance: 1353997
Número de outliers na coluna RatecodeID: 566747
Número de outliers na coluna payment_type: 2191082
Número de outliers na coluna fare_amount: 1136415
Número de outliers na coluna extra: 196918
Número de outliers na coluna mta_tax: 185331
Número de outliers na coluna tip_amount: 789323
Número de outliers na coluna tolls_amount: 852268
Número de outliers na coluna improvement_surcharge: 102162
Número de outliers na coluna total_amount: 1280592
Número de outliers na coluna congestion_surcharge: 840494
Número de outliers na coluna Airport_fee: 846534
Número de outliers na coluna trip_duration: 635352
Número de outliers na coluna month: 12
```

Figura 2: Número de *outliers* por columna

1. **Detección Univariada mediante el Método IQR**

   Para detectar outliers de manera univariada en las variables cuantitativas, se utilizó el método del *Rango Intercuartil*, IQR. Este método es ampliamente utilizado porque permite identificar valores que se encuentran significativamente alejados de la mayoría de los datos. El IQR se calcula como la diferencia entre el tercer cuartil, Q3, y el primer cuartil, Q1, de una distribución de datos. Un valor se considera un outlier si cae por debajo de $Q1 - 1{,}5 \times IQR$ o por encima de $Q3 + 1{,}5 \times IQR$.

   Decisión: Optamos por el método IQR debido a su simplicidad y eficacia para identificar outliers en distribuciones univariadas sin asumir una distribución normal en los datos. Los valores atípicos detectados mediante este método se ajustaron utilizando el *capping*, es

decir, fueron limitados a los valores límite de Q1 o Q3 para evitar la eliminación excesiva de datos.

2. **Detección Multivariada mediante PCA**

   Para la detección de outliers multivariados, utilizamos el método de *Análisis de Componentes Principales*, (PCA). Esta técnica se utilizó para reducir la dimensionalidad del conjunto de datos, manteniendo la mayor parte de la varianza. Al analizar los componentes principales, pudimos identificar puntos que se desviaban significativamente de las tendencias generales en múltiples variables.

   Decisión: El PCA fue elegido porque nos permite identificar outliers que podrían no ser visibles en un análisis univariado, ya que considera la correlación entre las variables. Los outliers detectados mediante PCA representan combinaciones inusuales de valores entre múltiples variables, lo que hace que esta técnica sea adecuada para detectar patrones complejos de datos atípicos.

3. **Decisiones sobre el Tratamiento de Outliers**

   Capping de Outliers Univariados: Los outliers detectados por el método IQR se limitaron al rango permitido (Q1 - 1.5 × IQR, Q3 + 1.5 × IQR) en lugar de eliminarlos completamente. Esta decisión se tomó para mantener la mayor cantidad de datos posible, pero sin dejar que los valores extremos influyeran en los resultados del análisis.

   Remoción de Outliers Multivariados: Los outliers identificados a través del análisis PCA, que mostraban una gran desviación en los componentes principales, fueron eliminados del conjunto de datos. La decisión de eliminar estos outliers se basó en la suposición de que podrían estar representando errores de registro o casos excepcionales que no reflejan el comportamiento general del sistema, tales como tarifas negativas.

El número de registros tras el tratamiento de los valores atipicos se muestra en la Fig3.

```
Registros no dataset original: 10109133
Registros após capping: 10109133
Registros após remoção de outliers: 9445786
```

Figura 3: Registros tras tratamiento

1.5. Datos Faltantes

En nuestro análisis inicial del dataset de taxis, identificamos varias columnas que contenían datos faltantes. A continuación, describimos las principales columnas afectadas, la proporción de datos faltantes y las decisiones tomadas respecto al tratamiento de estos valores.

A lo largo del análisis exploratorio, se identificaron varias columnas con valores faltantes en el conjunto de datos. En la Fig 4 se presenta la porcentage de datos faltantes por columna.

```
Análise de Dados Faltantes por Coluna:
Coluna: index - Percentagem de valores faltantes: 0.00%
Coluna: VendorID - Percentagem de valores faltantes: 0.00%
Coluna: tpep_pickup_datetime - Percentagem de valores faltantes: 0.00%
Coluna: tpep_dropoff_datetime - Percentagem de valores faltantes: 0.00%
Coluna: passenger_count - Percentagem de valores faltantes: 2.89%
Valores faltantes: 292373
Coluna: trip_distance - Percentagem de valores faltantes: 0.00%
Coluna: RatecodeID - Percentagem de valores faltantes: 2.89%
Valores faltantes: 292373
Coluna: store_and_fwd_flag - Percentagem de valores faltantes: 2.89%
Valores faltantes: 292373
Coluna: PULocationID - Percentagem de valores faltantes: 0.00%
Coluna: DOLocationID - Percentagem de valores faltantes: 0.00%
Coluna: payment_type - Percentagem de valores faltantes: 0.00%
Coluna: fare_amount - Percentagem de valores faltantes: 0.00%
Coluna: extra - Percentagem de valores faltantes: 0.00%
Coluna: mta_tax - Percentagem de valores faltantes: 0.00%
Coluna: tip_amount - Percentagem de valores faltantes: 0.00%
Coluna: tolls_amount - Percentagem de valores faltantes: 0.00%
Coluna: improvement_surcharge - Percentagem de valores faltantes: 0.00%
Coluna: total_amount - Percentagem de valores faltantes: 0.00%
Coluna: congestion_surcharge - Percentagem de valores faltantes: 2.89%
Valores faltantes: 292373
Coluna: Airport_fee - Percentagem de valores faltantes: 2.89%
Valores faltantes: 292373
Coluna: trip_duration - Percentagem de valores faltantes: 0.00%
Coluna: day_of_week - Percentagem de valores faltantes: 0.00%
Coluna: month - Percentagem de valores faltantes: 0.00%
Coluna: pickup_hour - Percentagem de valores faltantes: 0.00%
Coluna: dropoff_hour - Percentagem de valores faltantes: 0.00%
Coluna: hour - Percentagem de valores faltantes: 0.00%
```

Figura 4: Análisis de datos faltantes por columna

Si observamos la imagen, podemos concluir que las columnas **passenger_count**, **congestion_surcharge**, y **Airport_fee** presentaban un 2.89 % de valores faltantes y las columnas categóricas **RatecodeID** y **store_and_fwd_flag** también mostraban un 2.89 % de valores faltantes. Este porcentaje indica que alrededor del 2,89 % de los registros en cada una de estas columnas carecían de información completa.

Para resolver esta situación, se aplicaron técnicas de imputación:

- **Columnas Numéricas**

  En las columnas *passenger_count*, *congestion_surcharge* y *Airport_fee*, los valores faltantes se imputaron usando la mediana, lo cual es una técnica robusta contra valores atípicos y garantiza que los datos imputados no afecten significativamente la distribución de los datos.

- **Columnas categóricas**

  En *RatecodeID* y *store_and_fwd_flag*, se utilizó la moda para imputar los valores faltantes, ya que esto mantiene la categoría más frecuente y no introduce nuevas categorías artificiales.

El enfoque de imputación elegido asegura que no se pierdan registros importantes y que la estructura de los datos se mantenga coherente, lo que permite seguir adelante con el análisis y los modelos predictivos.

1.6. Visualizaciones

1.6.1. Lugares más recurrentes para inicio/fin de viaje

Para identificar los lugares más recurrentes donde se inician y terminan los viajes, se generaron gráficos de barras que se muestran en las Figuras 5 y 6). Estos gráficos muestran los 10 lugares más comunes tanto para inicio como para fin de viaje, en términos de frecuencia de viajes. Los lugares con más registros son áreas que probablemente concentran gran parte de la demanda de taxis, como estaciones de transporte, áreas turísticas y comerciales.

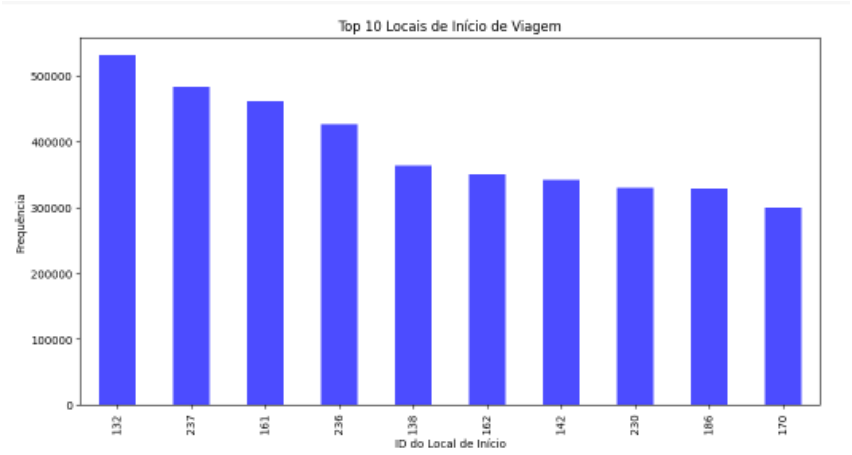

Figura 5: Top 10 lugares de inicio de viaje

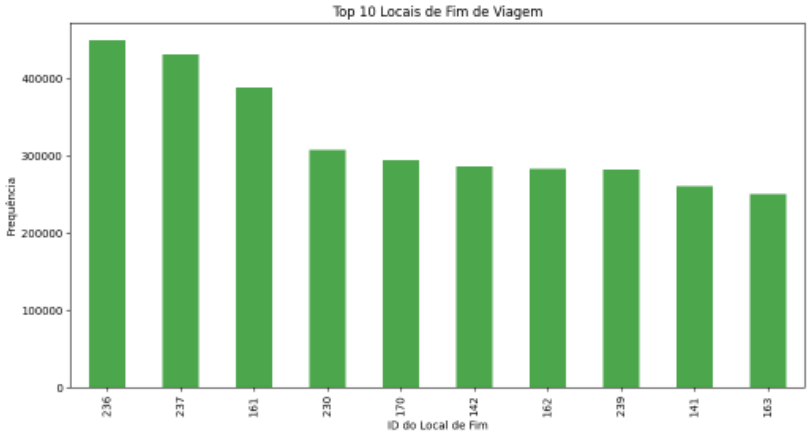

Figura 6: Top 10 lugares de fin de viaje

Estos resultados permiten caracterizar los patrones espaciales de inicio y fin de viaje, identificando zonas claves para la operación eficiente del servicio de taxis.

1.6.2. Viajes en distancia y tiempo

Para analizar las características típicas de los viajes en términos de distancia y tiempo, se generaron gráficos de caja (*boxplots*) que representan la distribución de estos dos aspectos.

La Figura 7 muestra que la mayoría de los viajes tienen una distancia corta, con algunos valores atípicos que representan viajes significativamente más largos. De manera similar, en la Figura 8 se observa que la duración de los viajes también sigue una tendencia a concentrarse en tiempos cortos, pero existen algunos viajes mucho más largos.

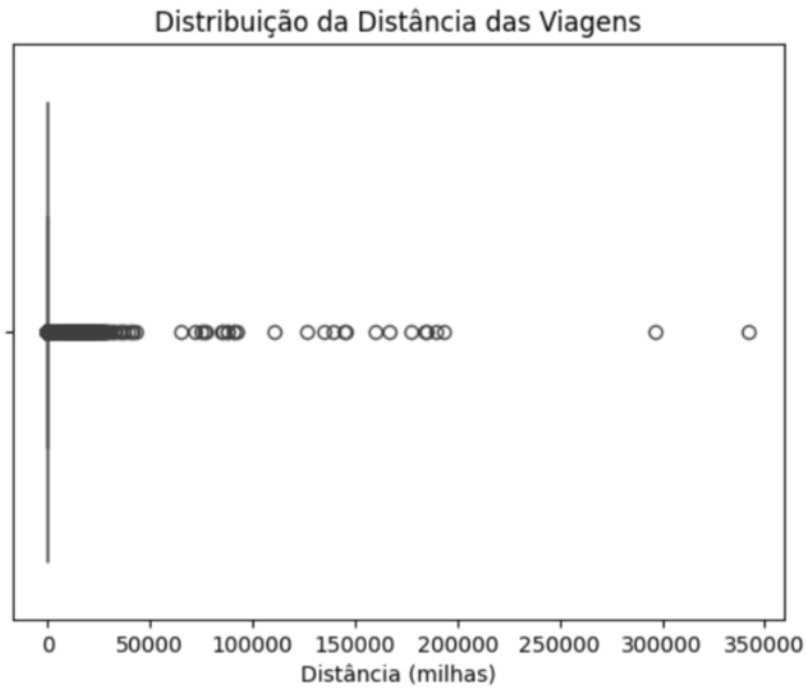

Figura 7: Distribución de la distancia de los viajes

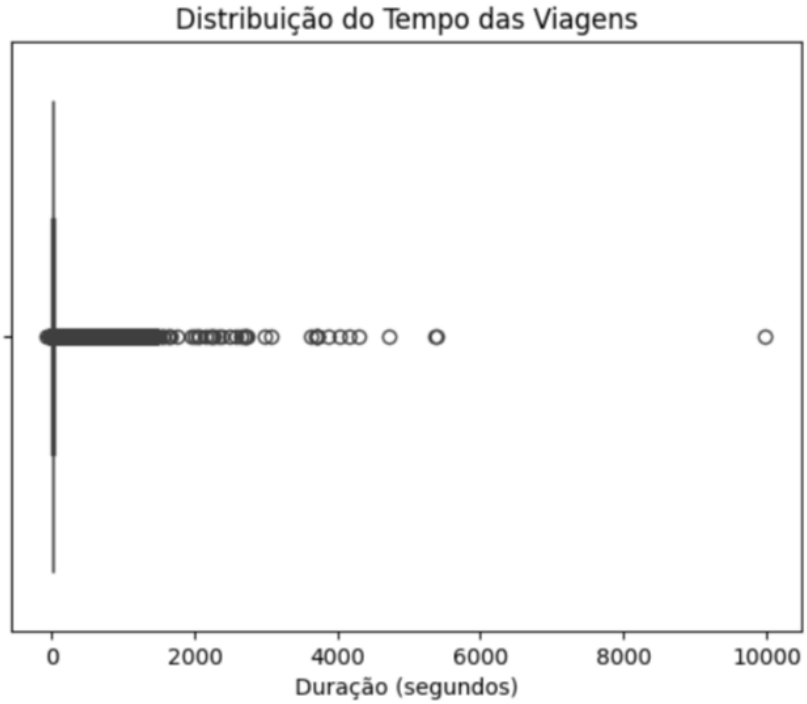

Figura 8: Distribución del tiempo de los viajes

1.6.3. Viajes en distancia y tiempo según el horario y/o el día de la semana

Para analizar cómo varían las distancias y tiempos de los viajes según el día de la semana y el horario, se generaron varios gráficos que muestran la relación entre estos aspectos. En la Figura 9, se puede observar la distancia media de los viajes para cada día de la semana. Los lunes y jueves parecen tener las mayores distancias medias.

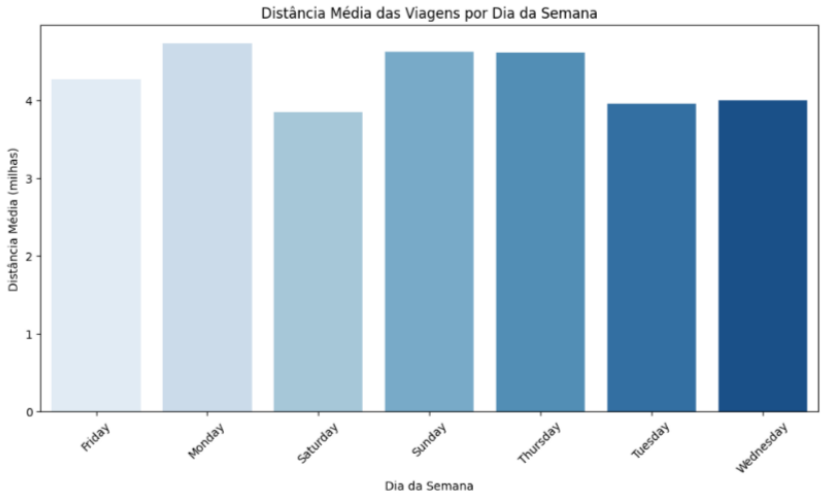

Figura 9: Distancia media de los viajes por día de la semana

De manera similar, en la Figura 10, se observa la variación de la distancia media de los viajes según la hora del día. Los viajes realizados entre las 4 y 5 de la mañana tienen una mayor distancia promedio, lo que puede reflejar viajes al aeropuerto o destinos lejanos en horarios poco usuales.

Figura 10: Distancia media de los viajes por hora del día

En cuanto a la duración, la Figura 11 muestra la duración media de los viajes según el día de

la semana. Se puede observar que los viajes del jueves tienen una duración media ligeramente mayor, lo cual podría estar relacionado con un mayor tráfico o patrones de viaje diferentes en ese día.

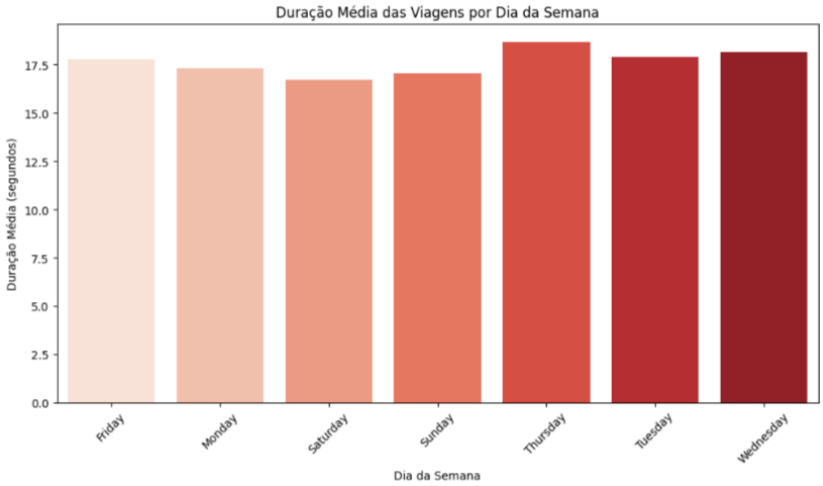

Figura 11: Duración media de los viajes por día de la semana

Finalmente, en la Figura 12, se representa la duración media de los viajes según la hora del día. Los viajes entre las 16 y 18 horas tienden a ser más largos, lo cual es probablemente el resultado de la congestión de tráfico durante las horas punta.

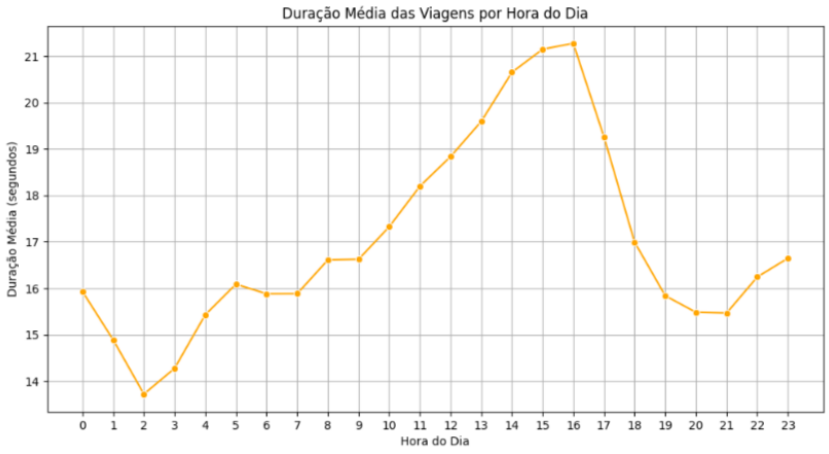

Figura 12: Duración media de los viajes por hora del día

Estos análisis permiten entender cómo varían los patrones de viaje a lo largo de la semana y del día, lo que puede ser útil para planificar la operación de taxis en función de la demanda y el comportamiento de los pasajeros.

2. EJ2: Modelos de Clasificación Binaria

Este ejercicio tiene como objetivo predecir si lloverá o no al día siguiente, basándonos en los datos meteorológicos del día actual. Para ello, se trabajará con información de diversas regiones de Australia, seleccionadas según las indicaciones del equipo docente y se entrenarán modelos que nos ayuden predecir lo que queremos.

Primero se hizo un merge entre un dataset con características meteorológicas de una locación y un dataset con las coordenadas y región de una locación. El dataset resultante contaba con:

- Cantidad de Registros: 145460, que representa las observaciones meteorológicas diarias de una estación específica.
- Cantidad de Columnas: 25, que incluyen tanto variables numéricas como categóricas.

Las columnas del dataset incluyen, pero no se limitan a, las siguientes variables:

- **MaxTemp**: Temperatura máxima durante un día particular (grados Celsius).
- **Evaporation**: Evaporación durante un día particular (milímetros).
- **Sunshine**: Horas de luz solar brillante durante un día particular (horas).
- **WindGusDir**: Dirección de la ráfaga de viento más fuerte durante un día particular (16 puntos cardinales).
- **WindGusSpeed**: Velocidad de la ráfaga de viento más fuerte durante un día particular (kilómetros por hora).
- **Humidity9am**: Humedad del aire a las 9 am (porcentaje).
- **Pressure9am**: Presión atmosférica a las 9 am (hectopascales).
- **Cloud9am**: Porciones del cielo cubiertas de nubes a las 9 am (octavos).
- **Temp9am**: Temperatura a las 9 am (grados Celsius).
- **RainToday**: Si llueve hoy, ”Yes”. Si no llueve hoy, ”No”.

- **RainTomorrow**: Si llueve mañana, ”Yes”. Si no llueve mañana, ”No”.

En el preprocesamiento de los datos, se utilizó la mediana para completar los valores faltantes de las variables numéricas, ya que este método minimiza el impacto de valores atípicos y mantiene la representatividad del conjunto de datos. La mediana es menos sensible a extremos en la distribución, lo que la convierte en una opción robusta en comparación con la media. Además, se implementó una función que limita los valores, conocida como çap”, para tratar los valores atípicos que aparecían en varias variables, como la precipitación y la temperatura. Este enfoque ayuda a suavizar la influencia de estas observaciones extremas en el análisis posterior y en el rendimiento de los modelos predictivos. Por otro lado, las variables categóricas, como las direcciones del viento y la indicación de lluvia, fueron convertidas en variables numéricas mediante la creación de variables dummy. Adicionalmente, se normalizaron las variables numéricas utilizando el MinMaxScaler, lo que permite que los datos se escalen a un rango específico. Esto es crucial para garantizar que las diferentes características tengan un impacto equitativo en los modelos, mejorando así la capacidad de estos para capturar patrones y relaciones en los datos.

2.1. Árbol de decisión

El árbol de decisión generado visualiza cómo se toman las decisiones basadas en las diferentes características de entrada. Cada nodo representa una pregunta sobre un atributo específico (por ejemplo, la temperatura o la humedad) que divide los datos en subconjuntos más homogéneos en relación con la variable objetivo, *RainTomorrow*. Las hojas del árbol indican la clase final asignada. Al interpretar el árbol, es fundamental observar la profundidad de los nodos y el número de muestras en cada hoja, ya que esto proporciona información sobre la confianza de las predicciones realizadas.

- **¿Optimizaron hiperparámetros? ¿Cuáles?** Sí, se optimizaron varios hiperparámetros del modelo de Árbol de Decisión. Los hiperparámetros que se optimizaron incluyen:

  - `criterion`: La función utilizada para medir la calidad de una división, con opciones `'gini'` y `'entropy'`.

- `splitter`: El método utilizado para elegir la división en cada nodo, en este caso, solo se utilizó `'best'`.
- `max_depth`: La profundidad máxima del árbol, que puede ser `None` o valores enteros como 10, 20, 30, 40 y 50.
- `min_samples_split`: El número mínimo de muestras necesarias para dividir un nodo, con valores posibles de 2, 5 y 10.
- `min_samples_leaf`: El número mínimo de muestras que debe haber en un nodo hoja, con opciones de 1, 2 y 4.
- `ccp_alpha`: Este hiperparámetro controla la poda del árbol al ajustar el costo de las hojas, y se optimizó utilizando un rango de valores generados con `np.linspace(0, 0.1, n_iter)`, lo que permite explorar diferentes niveles de complejidad del modelo.

- **¿Utilizaron K-fold Cross Validation? ¿Cuántos folds utilizaron?** Sí, se utilizó K-fold Cross Validation para evaluar el rendimiento del modelo. Se utilizaron 5 folds.
- **¿Qué métrica utilizaron para buscar los hiperparámetros?** Para buscar los hiperparámetros, se utilizó la **accuracy** como métrica principal.

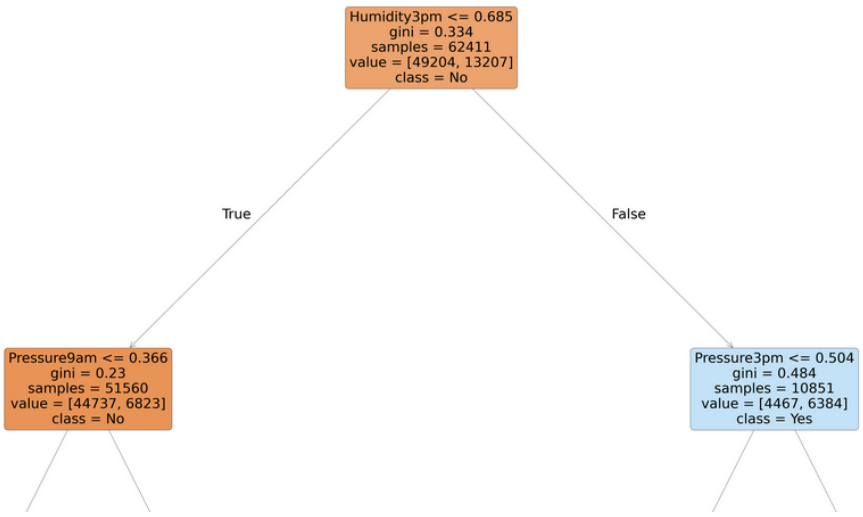

Figura 13: Árbol de decisión

Vemos que el nodo raíz es Humidity3pm. Si la humedad es mayor a 0.685, el árbol sigue por una rama indicando una mayor probabilidad de lluvia. Si la humedad es menor o igual a 0.685, se sigue por otra rama indicando una menor probabilidad de lluvia. Así se llega a los nodos intermedios. Después de dividir los datos con base en la humedad, el árbol buscará una segunda variable que pueda dividir aún más esos subconjuntos. En este caso Pressure9am y Pressure3pm. Hace esto sucesivamente hasta llegar a los nodos hoja donde se toma una decisión final. En los nodos hoja, el árbol ya no divide más los datos, sino que asigna una clase final (en este caso, ”Yes.º ”No”). Tambien se puede apreciar que predomina la clase ”No”, ya que es la que posee una mayor cantidad de muestras en comparación al ”Yes”, haciendo referencia a nuestra variable target.

2.2. Random Forest

- **¿Optimizaron hiperparámetros? ¿Cuáles?** Sí, se optimizaron varios hiperparámetros del modelo de Random Forest. Los hiperparámetros que se optimizaron incluyen:
  - `n_estimators`: Número de árboles en el bosque, se probaron 100 y 150.
  - `criterion`: La función utilizada para medir la calidad de una división, se utilizó solo `'gini'`.
  - `max_depth`: La profundidad máxima de los árboles, se probaron opciones como `None`, 10 y 15.
  - `min_samples_split`: El número mínimo de muestras necesarias para dividir un nodo, se probaron valores de 2 y 3.
  - `min_samples_leaf`: El número mínimo de muestras que debe haber en un nodo hoja, se probaron valores de 1 y 2.
  - `bootstrap`: Se utilizó la técnica de muestreo con reemplazo, en este caso, se fijó a `True`.
- **¿Utilizaron K-fold Cross Validation? ¿Cuántos folds utilizaron?** Sí, se utilizó K-fold Cross Validation para evaluar el rendimiento del modelo. Se utilizaron 5 folds.

- **¿Qué métrica utilizaron para buscar los hiperparámetros?** Para buscar los hiperparámetros, se utilizó la **accuracy** como métrica principal.

Representamos un árbol del forest

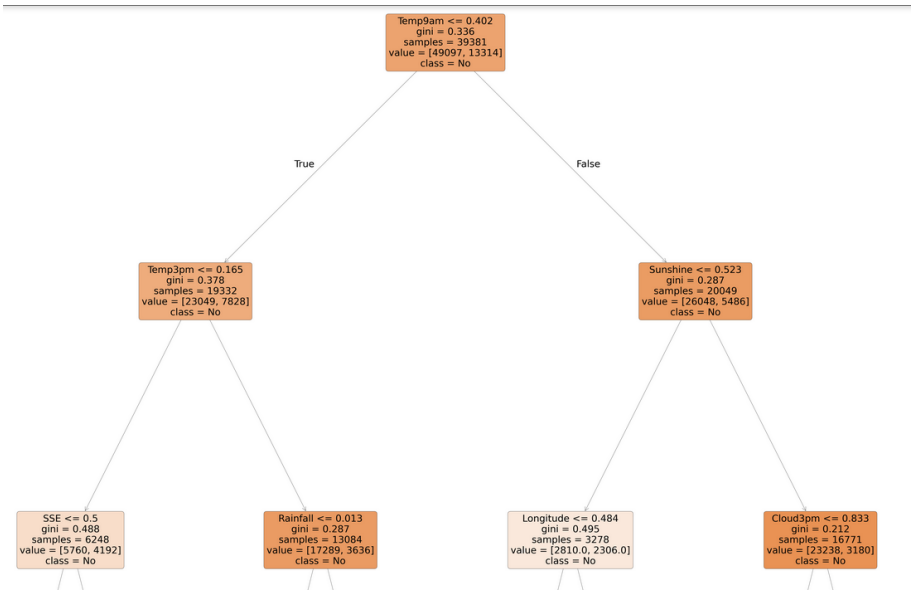

Figura 14: Random Forest

2.3. Regresión Logística

- **¿Optimizaron hiperparámetros? ¿Cuáles?** Sí, se optimizaron varios hiperparámetros del modelo de Regresión Logística. Los hiperparámetros que se optimizaron incluyen:

  - `penalty`: Tipo de regularización, se probaron opciones como `'l1'`, `'l2'` y `'elasticnet'`.
  - `C`: Parámetro de regularización inversa, se probaron valores como 0.001, 0.01, 0.1, 1, 10 y 100.
  - `solver`: Algoritmo utilizado para optimizar el modelo, se utilizaron `'liblinear'` para `'l1'` y `'l2'`, y `'newton-cg'`, `'lbfgs'`, y `'saga'` para `'l2'`.
  - `max_iter`: Número máximo de iteraciones para los algoritmos de optimización, se probaron 100, 200, 300, 400 y 500.
  - `l1_ratio`: Proporción de regularización L1, se utilizó con `'elasticnet'`.

- **¿Utilizaron K-fold Cross Validation? ¿Cuántos folds utilizaron?** Sí, se utilizó K-fold Cross Validation para evaluar el rendimiento del modelo. Se utilizaron 5 folds.

- **¿Qué métrica utilizaron para buscar los hiperparámetros?** Para buscar los hiperparámetros, se utilizó la **accuracy** como métrica principal.

2.4. Cuadro de Resultados

Comparamos las siguientes métricas y vemos que los valores obtenidos en el conjunto test se asemejan a los obtenidos en el conjunto train.

	Accuracies	F1 Scores	ROC AUC	Precision	Recall
LogisticReg	0.8629	0.8546	0.7453	0.8547	0.8629
DecisionTree	0.8490	0.8396	0.7234	0.8386	0.8490
RandomForest	0.8707	0.8611	0.7479	0.8641	0.8707

Figura 15: Medidas de rendimiento en el conjunto de Test

	Accuracies	F1 Scores	ROC AUC	Precision	Recall
LogisticReg	0.8633	0.8548	0.7446	0.8551	0.8633
DecisionTree	0.8762	0.8686	0.7646	0.8700	0.8762
RandomForest	0.9735	0.9730	0.9399	0.9741	0.9735

Figura 16: Medidas de rendimiento en el conjunto de Train

Los hiperparámetros utilizados en cada modelo de clasificación son:

- Árbol de decisión: {'splitter': 'best', 'min_samples_split': 2, 'min_samples_leaf': 4, 'max_depth': 10, 'criterion': 'entropy'}

- Random Forest: {'n_estimators': 150, 'min_samples_split': 2, 'min_samples_leaf': 2, 'max_depth': None, 'criterion': 'gini', 'bootstrap': True}

- Regresión Logística: {'solver': 'liblinear', 'penalty': 'l2', 'max_iter': 100, 'C': 1}

3. EJ3: Análisis Exploratorio

3.1. Descripción del Dataset

El dataset utilizado contiene 36,807 registros y 75 columnas que describen propiedades listadas en Airbnb, como el tipo de propiedad, amenidades, ubicación, precios, reseñas, entre otras características relacionadas con los hosts y las propiedades. Luego del preprocesamiento, se redujo la cantidad de columnas a las más relevantes tras eliminar las irrelevantes o duplicadas y realizar las transformaciones correspondientes.

3.2. Features más destacables

Algunas de las características más importantes y sus transformaciones son:

- **amenities**: Se codificó mediante *one-hot encoding* extrayendo las 150 amenidades más comunes y creando columnas binarias para cada una.
- **Variables temporales**: Se crearon nuevas características como *host_active_days* y *days_since_scraped*, que capturan la antigüedad de la propiedad y la actividad del host.
- **review_scores**: Las puntuaciones relacionadas con las reseñas de los huéspedes (*review_scores_rating*, *review_scores_cleanliness*, etc.) se imputaron con ceros en caso de ausencia de reseñas. Estas columnas reflejan la calidad percibida de la propiedad.
- **host_response_rate** y **host_acceptance_rate**: Ambas columnas, inicialmente en formato porcentual, se convirtieron a valores numéricos para representar las tasas de respuesta y aceptación de solicitudes de los anfitriones.
- **room_type** y **property_type**: Se codificaron usando *one-hot encoding* para capturar la variabilidad en los tipos de habitaciones y propiedades.
- **bedrooms** y **beds**: El número de habitaciones y camas se incluyó como variables numéricas importantes para predecir el precio. Ambas fueron analizadas para identificar y ajustar valores atípicos.

3.3. Preprocesamiento de Datos

3.3.1. Imputación y eliminación de columnas irrelevantes

Se eliminan columnas no útiles (como `id`, `license`, `calendar_updated`) y se imputan valores faltantes en variables categóricas con etiquetas como "unknown". Las variables binarias se completan con valores por defecto y las variables numéricas relacionadas con reseñas se imputan con ceros.

3.3.2. Transformación de fechas

Las columnas de fechas (`calendar_last_scraped`, `first_review`, etc.) se convierten a formato `datetime`. A partir de ellas, se generan nuevas características como `host_active_days` y `days_since_scraped`, capturando la antigüedad y actividad del listado.

3.3.3. Manejo de la columna `bathrooms_text`

La columna `bathrooms_text` se utiliza para rellenar valores faltantes en `bathrooms` y clasificar los baños en una nueva variable `bathroom_category` ("private", "shared" o "no bathroom").

3.3.4. Codificación de variables categóricas

Se implementa *one-hot encoding* en variables como `neighbourhood_cleansed` y `room_type`, mientras que el género del anfitrión (`host_gender`) se infiere a partir de su nombre. También se procesan amenidades (`amenities`) para crear variables binarias que representen las más comunes.

3.3.5. Imputación con KNN

Se imputan valores faltantes en variables numéricas utilizando el método KNN, asegurando la coherencia con observaciones similares sin introducir sesgos en la variable objetivo (`price`).

3.3.6. Tratamiento de valores atípicos (outliers)

Los valores atípicos en variables numéricas se ajustan utilizando el rango intercuartílico (IQR), minimizando su influencia en el modelo.

3.3.7. Selección de características

Se eliminan variables con baja correlación con el precio ($< 0{,}02$) y aquellas con alta multicolinealidad ($> 0{,}8$), reteniendo únicamente las más relevantes.

3.3.8. Escalado de características

Las variables numéricas se estandarizan utilizando `StandardScaler` para asegurar consistencia en la escala de las variables.

3.4. Modelos Entrenados

3.4.1. Regresión Lineal

¿Qué features seleccionaron para construir el modelo?

Para la construcción del modelo de regresión lineal, se seleccionaron una variedad de características tanto numéricas como categóricas. Algunas de las características más importantes incluyen:

- **Numéricas**: `bathrooms`, `calculated_host_listings_count_shared_rooms`, `calculated_host_listings_count_private_rooms`, `accommodates`, `days_since_last` `longitude`, y `latitude`.
- **Categóricas** (codificadas con one-hot encoding): `neighbourhood_cleansed` (ej., `Puerto Madero`, `Palermo`, `Recoleta`), `property_type` (ej., `Entire villa`, `Entire chalet`, `Casa particular`), `amenities` (ej., `Dishwasher`, `Free parking on premises`, `Gym`, `Hot tub`), `host_is_superhost`, y `bathroom_category`.

Evaluar la performance del modelo en el conjunto de evaluación, explicar todas las métricas. Comparar con la performance de entrenamiento.

Los resultados del modelo de regresión lineal en el conjunto de evaluación y en el conjunto de entrenamiento son los siguientes:

- **Performance en el conjunto de evaluación**:
  - **MSE** (Error cuadrático medio): 192,728,650.32
  - **RMSE** (Raíz del error cuadrático medio): 13,882.67
  - **R^2** (Coeficiente de determinación): 0.4928
- **Performance en el conjunto de entrenamiento**:
  - **MSE**: 195,730,124.83
  - **RMSE**: 13,990.36
  - **R^2**: 0.5020

3.4.2. XGBoost

¿Utilizaron K-fold Cross Validation? ¿Cuántos folds utilizaron?

Sí, se utilizó validación cruzada con K-folds para buscar los hiperparámetros óptimos del modelo. En este caso, se utilizó un esquema de validación cruzada con 5 folds (`cv=5`) dentro del proceso de `GridSearchCV`.

Evaluar la performance del modelo en el conjunto de evaluación, explicar todas las métricas. Comparar con la performance de entrenamiento.

Los resultados del modelo XGBoost en el conjunto de evaluación y en el conjunto de entrenamiento son los siguientes:

- **Performance en el conjunto de evaluación**:
  - **MSE** (Error Cuadrático Medio): 163,242,367.08
  - **RMSE** (Raíz del Error Cuadrático Medio): 12,776.63
  - **R^2** (Coeficiente de Determinación): 0.5704
- **Performance en el conjunto de entrenamiento**:

- **MSE**: 122,449,691.63
- **RMSE**: 11,065.70
- **R^2**: 0.6884

3.4.3. Modelo a Elección

¿Utilizaron K-fold Cross Validation? ¿Cuántos folds utilizaron?

Sí, se utilizó validación cruzada con K-folds para evaluar el modelo y ajustar los hiperparámetros. En este caso, se utilizó una validación cruzada con 2 folds debido a limitaciones de tiempo y recursos computacionales. Aunque un mayor número de folds generalmente produce resultados más robustos, 2-folds permite obtener una estimación razonable del rendimiento del modelo sin que el tiempo de ejecución sea excesivo.

¿Qué métrica utilizaron para buscar los hiperparámetros?

La métrica utilizada para optimizar los hiperparámetros fue el error cuadrático medio (MSE), que es apropiada para problemas de regresión, ya que penaliza los errores grandes, permitiendo seleccionar el modelo que minimiza este error.

Evaluar la performance del modelo en el conjunto de evaluación, explicar todas las métricas. Comparar con la performance de entrenamiento.

Los resultados del modelo a elección, evaluado con validación cruzada y en el conjunto de prueba, son los siguientes:

- **Performance con validación cruzada (5 folds)**:
  - **MSE validado**: [1.85485113e+08, 1.93179799e+08]
  - **RMSE validado**: [13,619.29, 13,898.91]
  - **Promedio del RMSE validado**: 13,759.10
- **Performance en el conjunto de evaluación**:
  - **MSE** (Error Cuadrático Medio): 173,455,692.77
  - **RMSE** (Raíz del Error Cuadrático Medio): 13,170.26
  - **R^2** (Coeficiente de Determinación): 0.5436

- **Performance en el conjunto de entrenamiento**:
    - **MSE**: 24,786,884.14
    - **RMSE**: 4,978.64
    - **R²**: 0.9369

Comparación con la performance de entrenamiento:

El modelo tiene un mejor rendimiento en el conjunto de entrenamiento (MSE: 24,786,884.14, RMSE: 4,978.64, R²: 0.9369) en comparación con el conjunto de evaluación (MSE: 173,455,692.77, RMSE: 13,170.26, R²: 0.5436). Esta diferencia significativa sugiere que el modelo puede estar sobreajustado a los datos de entrenamiento, lo cual podría deberse a una complejidad excesiva del modelo o a una falta de regularización adecuada. El modelo tiene un rendimiento decente en el conjunto de evaluación, pero muestra una gran diferencia en la capacidad de generalización, lo que sugiere que se puede optimizar aún más.

3.4.4. Cuadro de Resultados

A continuación, se presenta un cuadro comparativo con los tres modelos entrenados, incluyendo las métricas de rendimiento en el conjunto de *test*.

Modelo	MSE	RMSE	R² Score	RMSE vs Std (%)
Regresión Lineal	1.927287e+08	13,882.67	0.4928	71.21
XGBoost	1.632424e+08	12,776.63	0.5704	65.54
Random Forest	1.734557e+08	13,170.26	0.5436	67.56

Cuadro 1: Comparación de los Modelos Entrenados

RMSE vs Std (%) compara el error de predicción (RMSE) con la dispersión de los precios (desviación estándar del conjunto de prueba). Se calcula como:

$$\text{RMSE vs Std (\%)} = \left(\frac{\text{RMSE}}{\text{std\_y\_test}} \right) \times 100$$

Esta métrica indica qué porcentaje de la variabilidad de los precios es cubierto por el error del modelo. Un valor más bajo significa que el error del modelo es pequeño en relación con la dispersión natural de los datos.

3.4.5. Elección del Modelo

Con base en los resultados obtenidos, se recomendaría utilizar el modelo **XGBoost** para predecir el precio de una propiedad de AirBnB en la ciudad estudiada. Este modelo mostró el mejor rendimiento en el conjunto de prueba, con un **MSE** de 1.63e+08, un **RMSE** de $12,776.63, y un $\mathbf{R^2}$ de 0.5704. En comparación con la Regresión Lineal y el Random Forest, el XGBoost tiene una menor variabilidad en las predicciones y una mejor capacidad para explicar la varianza en los datos. Además, su capacidad para capturar relaciones complejas entre las características hace que sea el mejor predictor en este contexto, donde la relación entre las variables puede no ser estrictamente lineal.

4. EJ4: Clustering

Este ejercicio presenta un análisis detallado del proceso de clustering aplicado a un conjunto de datos de música. El objetivo es identificar la tendencia a la agrupación de canciones según sus características y determinar la cantidad adecuada de grupos que mejor representan la variabilidad en los datos. Además, se describirá en función de qué características se formaron dichos grupos y cómo llegaron a la conclusión sobre el número de clusters.

Para el análisis se utilizó el algoritmo K-Means, que es adecuado para este tipo de tareas no supervisadas, y métodos de evaluación como el *Silhouette Score* y el Método del Codo. Estos ayudaron a validar la cantidad óptima de grupos, que luego se visualizaron para una interpretación clara.

4.1. Preprocesamiento de datos

Antes de proceder con el análisis de clustering, se realizaron varios pasos de preprocesamiento para asegurar la calidad de los datos y optimizar los resultados del modelo. Los principales pasos realizados incluyen:

1. **Análisis de datos faltantes**:

   Se verificó la presencia de valores faltantes en todas las columnas del dataset. Afortunadamente,

no se encontraron valores faltantes en ninguna de las variables principales del conjunto de datos.

- energy: 0 % de valores faltantes
- acousticness: 0 % de valores faltantes
- loudness: 0 % de valores faltantes
- danceability: 0 % de valores faltantes

Esto indica que no fue necesario realizar imputación de datos ni eliminación de registros por valores faltantes.

2. **Detección y tratamiento de outliers**:

   Para identificar valores atípicos en las principales características numéricas, se utilizaron métodos como el IQR, Interquartile Range. Se detectaron algunos outliers en variables como loudness y tempo. Estos valores atípicos pueden distorsionar los resultados del clustering, por lo que se decidió aplicar el tratamiento de capping.

   Capping (limitación): Los outliers detectados se reemplazaron por los límites superior e inferior calculados mediante el IQR. Este método ayuda a reducir el impacto de valores extremadamente altos o bajos sin eliminar los datos completamente. Por ejemplo, los valores atípicos en la columna loudness fueron limitados al rango [Q1 - 1.5 * IQR, Q3 + 1.5 * IQR].

3. **Padronización de los datos**:

   Dado que las variables del dataset tienen escalas diferentes (por ejemplo, tempo puede variar considerablemente en comparación con energy), fue necesario aplicar padronización. Se utilizó la técnica de *Standard Scaler*, que transforma las características para que tengan una media de 0 y una desviación estándar de 1. Esto asegura que todas las características contribuyan de manera equitativa en el algoritmo de clustering, que es sensible a la escala de los datos.

La fórmula utilizada para la padronización fue:

$$z = \frac{x - \mu}{\sigma} \tag{1}$$

Donde x es el valor original, μ es la media de la columna, y σ es la desviación estándar.

4.2. Tendencia al clustering

Tras el preprocesamiento de los datos, se observó una tendencia clara hacia la agrupación de las canciones según sus características principales. Las variables que tuvieron una mayor influencia en la formación de los clusters fueron:

- **danceability**: Representa la capacidad de la canción para ser bailable. Es crucial en la formación de clusters porque las canciones con alta o baja bailabilidad se comportan de manera diferente en otros atributos como energía o tempo.
- **energy**: Esta variable cuantifica la intensidad y actividad percibida en la canción, lo que permite diferenciar grupos de canciones más enérgicas de las más suaves.

Estas variables contribuyeron significativamente a la separación entre clusters, ya que canciones con niveles similares de bailabilidad y energía tienden a agruparse, mientras que las diferencias importantes en estos valores tienden a generar separaciones naturales en los grupos.

4.3. Determinación de la cantidad adecuada de clusters

Para determinar la cantidad óptima de clusters, se emplearon dos métodos clave:

- **Método del Codo**:

  Este método gráfico muestra cómo la ïnercia.º suma de los errores cuadráticos dentro de los clusters disminuye a medida que aumenta el número de grupos. El codo del gráfico, que indica la cantidad apropiada de clusters, se encontró en 3 grupos.

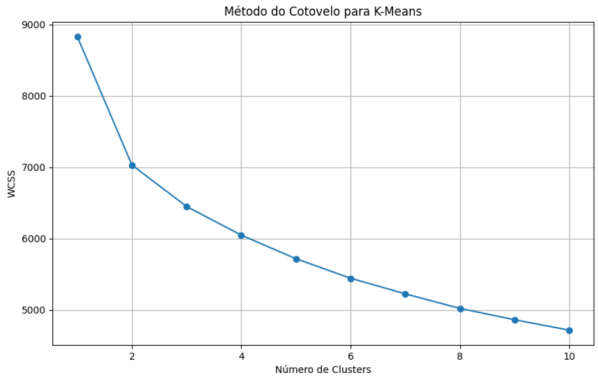

Figura 17: Gráfico del Método del Codo

Como se observa en el gráfico, el punto de codo más evidente se encuentra en 3 clusters. Esto indica que, a partir de esta cantidad de grupos, la mejora en la separación entre clusters disminuye significativamente.

- **Silhouette Score**:

  Este puntaje mide cuán separados están los grupos entre sí y cuán cohesionados están los puntos dentro de un mismo grupo. Los valores cercanos a 1 indican una buena clasificación dentro del cluster, mientras que los valores cercanos a -1 sugieren una mala asignación. En este caso, el puntaje más alto de Silhouette se obtuvo con 3 clusters, confirmando la elección del número adecuado de grupos.

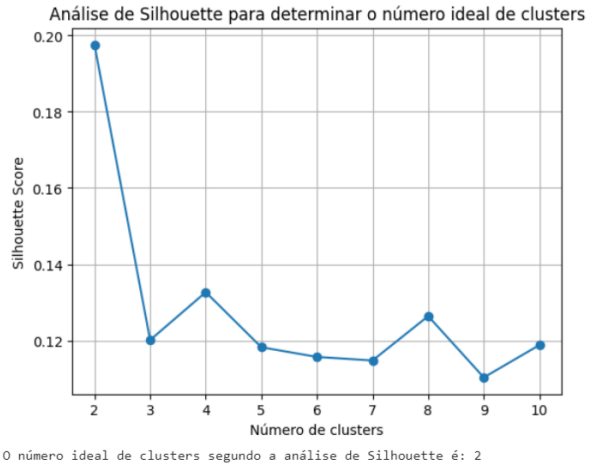

Figura 18: Analisis de Silhouette

4.4. Visualización de los grupos formados

A continuación, se presenta una visualización de los clusters formados, utilizando las principales características del dataset (bailabilidad, energía, y otras) para crear un gráfico de dispersión que muestra cómo se agrupan las canciones.

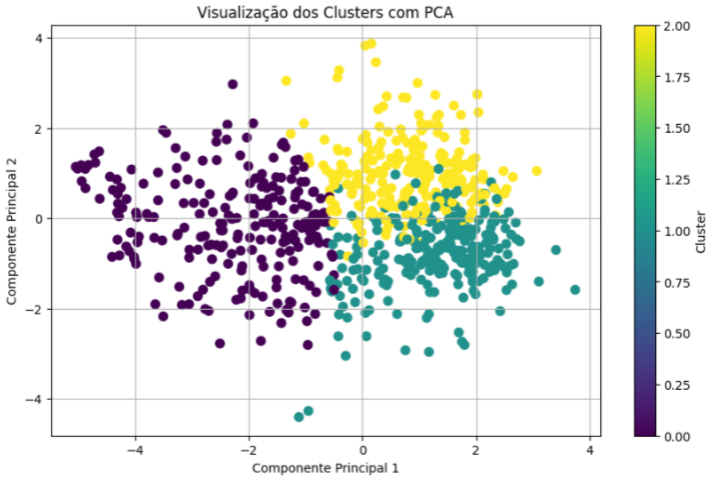

Figura 19: Gráfico de dispersión de los Clusters

En este gráfico, cada color representa un cluster distinto, permitiendo observar claramente la separación entre los grupos.

4.5. Características en función de las cuales se formaron los grupos

Los clusters se formaron principalmente con base en las siguientes características:

- **danceability**: Esta variable permitió distinguir grupos de canciones que son más bailables frente a aquellas con menos ritmo.
- **energy**: Su valor fue clave para agrupar canciones con alta energía en un mismo grupo, separándolas de las más suaves y calmadas.

  Estos atributos permitieron identificar patrones en la agrupación de las canciones, diferenciando claramente grupos con diferentes características rítmicas y energéticas.

El análisis de clustering realizado sobre el dataset de música sugiere que la cantidad óptima de grupos es 3, determinada por los métodos de validación aplicados. Las características principales

que impulsaron la formación de los clusters fueron *danceability* y *energy*, y la separación entre grupos fue clara y consistente en los análisis realizados.

5. Tiempo dedicado

Integrante	**Tarea**	**Prom. Hs Semana**
Rita Resende Casimiro de Castro Lobo	Ej 1, Ej 4	4
Gino Oggero	Ej 2	4
Isabella Schneider	Ej 3	4