Trabajo Práctico 1 - Grupo 09

Informe - Ejercicio 1: Análisis Exploratorio de Datos

Descripción del dataset

El dataset utilizado corresponde a los viajes realizados por taxis amarillos de la ciudad de Nueva York (Yellow Cab), durante los meses de julio, agosto y septiembre 2024.

Los archivos están en formato Parquet, lo que permite una carga eficiente y un buen manejo de tipos de datos.

Cada registro representa un viaje individual, incluyendo información sobre los horarios, las distancias, los montos cobrados, las propinas, la cantidad de pasajeros y las zonas de inicio y fin.

- Registros: millones de viajes (cada archivo parquet tiene > 2M filas).
- Columnas: ~17 columnas con información de viajes (pick-up/drop-off datetime, distancia, monto, propinas, pasajeros, etc.).
- Tamaño:

Luego de correr

```
filas, columnas = dataset.shape
print(f"El DataFrame, en su totalidad, tiene {filas} filas y {columnas} columnas.")
```

El resultado obtenido fue: El DataFrame, en su totalidad, tiene 9 689116 filas y 21 columnas.

- Variables más relevantes:

    1. `tpep_pickup_datetime`, `tpep_dropoff_datetime`: permiten calcular duración de viaje
    2. `trip_distance`: distancia recorrida, central para detectar outliers.
    3. `fare_amount`, `tip_amount`, `total_amount`: variables de costo, relevantes para correlaciones.
    4. `passenger_count`: describe ocupación, útil para patrones de uso.

- Hipótesis iniciales:

    1. Los viajes más largos y en horarios nocturnos podrían tener montos más altos.
    2. Durante agosto, al ser temporada alta en NYC, se espera mayor cantidad de viajes y mayor recaudación promedio. (turismo de verano en NYC).
    3. Existen outliers en variables como `trip_distance` y `total_amount`, posiblemente debidos a errores de carga o viajes fuera de la zona habitual.

Preprocesamiento

Durante el preprocesamiento se eliminaron columnas consideradas irrelevantes o redundantes para el análisis exploratorio

Columnas eliminadas

- `PULocationID`, `DOLocationID`, `LocationID`, `service_zone`: eliminadas por falta de relevancia directa o redundancia.

- `VendorID`, `RatecodeID`, `store_and_fwd_flag`, `payment_type`: removidas en análisis cuantitativo por no aportar a correlaciones.

Se eliminaron también registros duplicados para evitar sesgos en los resultados.

Correlaciones destacables

A partir del análisis de la matriz de correlaciones y su visualización mediante un heatmap, se observaron las siguientes relaciones más fuertes entre las variables.

- `improvement_surcharge` y `mta_tax`
- `total_amount` y `fare_amount`
- `total_amount` y `tip_amount`
- `total_amount` y `tolls_amount`
- `total_amount` y `airport_fee`

Se observa que las correlaciones más altas se concentran en variables directamente asociadas al cálculo de la tarifa total del viaje.
En particular, la variable `total_amount`, al ser la suma de los componentes del costo (`fare_amount`, `tip_amount`, `tolls_amount`, `airport_fee`), presenta una fuerte

correlación con cada una de estas, lo que confirma que su comportamiento está explicado por ellas.

Otro caso destacable es la fuerte relación entre `improvement_surcharge` y `mta_tax`, lo cual puede deberse a que ambos representan cargos adicionales fijos aplicados en cada viaje (por ejemplo, bajada de bandera o impuestos estatales).

En general, el análisis de correlaciones permitió confirmar la coherencia interna del dataset y orientar el preprocesamiento hacia las variables realmente relevantes para el comportamiento económico de los viajes

Nuevos features generados

Para enriquecer el análisis se crearon nuevas variables derivadas a partir de las columnas originales del dataset:

`trip_duration_min`: representa la duración del viaje expresada en minutos.
Se calculó como la diferencia entre `tpep_dropoff_datetime` y `tpep_pickup_datetime`
Esta variable permitió analizar la relación entre tiempo de viaje, distancia recorrida y monto total

`fare_per_mile`: indica la tarifa base por milla recorrida.
Se obtuvo dividiendo `fare_amount` por `trip_distance`, lo que facilita detectar variaciones de precios relativos y posibles sobrecargos en viajes cortos o largos.

`tip_percentage`: muestra la proporción de la propina respecto al monto total pagado.
Se calculó mediante la expresión `(tip_amount / total_amount) * 100`, y resultó útil para observar patrones de comportamiento de los usuarios en cuanto a las propinas.

`PUBorough, PUZone, DOBorough, DOZone`: Se relaciono el dataset de los registros de viajes con uno de zonas, Este indica segun el id a que zona y borough de Nueva York pertenece. A partir de esta vinculación se generó una variable adicional llamada `congestion_surcharge`, que refleja el recargo por congestión aplicado a los viajes realizados en determinadas zonas de alta densidad vehicular.

Estas transformaciones aportaron una visión más completa y analítica del conjunto de datos, permitiendo estudiar los viajes desde distintas perspectivas: temporal, geográfica y económica.

Valores atípicos

Antes de analizar los valores atípicos, se creó la variable `trip_duration_min`, que representa la duración del viaje en minutos.
Esta se calculó como la diferencia entre las columnas `tpep_dropoff_datetime` y `tpep_pickup_datetime`, expresada en minutos.

Esta nueva variable permitió evaluar la coherencia entre el tiempo de viaje, la distancia recorrida y el monto total pagado.

a) Análisis univariado mediante *boxplots*

Se aplicó un análisis exploratorio visual utilizando boxplots sobre las principales variables cuantitativas (`passenger_count`, `trip_distance`, `fare_amount`, `tip_amount`, `total_amount`, `trip_duration_min`).

Estos gráficos permitieron observar la dispersión de los datos, donde los valores dentro de la caja representan el 50 % central de los registros. Se detectaron valores atípicos en varias de las variables monetarias y en la duración de los viajes.

b) Criterios y decisiones de limpieza por variable

• **`passenger_count`:** se detectaron viajes con 0 pasajeros lo cual no es posible. Estos casos fueron reemplazados por la mediana de la variable, manteniendo coherencia con la distribución general. Los viajes con más de cuatro pasajeros se conservaron por considerarse posibles, aunque poco frecuentes

• **`total_amount`:** montos negativos en las columnas, estos fueron corregidos aplicando el valor absoluto (`.abs()`) a todas las variables tarifarias. Además, se eliminaron los registros con tarifas mayores a 1000 USD, considerados excesivos para viajes urbanos.

• **`trip_duration_min`:** Se eliminaron los registros con duraciones negativas o superiores a 360 minutos (6 horas), al representar datos inválidos o fuera del rango esperado.

• **`trip_distance`**: Se eliminaron viajes con distancias nulas o negativas, así como aquellos con distancia mayor a 150 millas, valores improbables dentro de la zona metropolitana de Nueva York.

Estas acciones redujeron significativamente los valores extremos y mejoraron la consistencia del dataset.

c) Análisis multivariado con *Isolation Forest*

Además del análisis individual, se aplicó el algoritmo **Isolation Forest** para detectar outliers multivariados considerando las siguientes variables: `passenger_count`, `trip_distance`, `fare_amount`, `tip_amount`, `total_amount`, `trip_duration_min`.

El modelo permitió identificar combinaciones atípicas de valores que no podían detectarse con el análisis univariado. Los registros clasificados como anómalos fueron eliminados. Tras este proceso, se logró un dataset más consistente, limpio y representativo del comportamiento real de los viajes.

Datos faltantes

Se verificó inicialmente que el dataset no contuviera registros duplicados. Se encontraron algunas filas repetidas, las cuales fueron eliminadas para asegurar la integridad del conjunto de datos.

Posteriormente, se identificaron las variables que presentaban valores nulos: `passenger_count`, `RatecodeID`, `store_and_fwd_flag`, `congestion_surcharge`, `Airport_fee`, y las variables relacionadas con las zonas geográficas (`PUBorough`, `PUZone`, `DOBorough`, `DOZone`).

El **análisis de proporciones** mostró que la mayoría de las columnas presentaba una baja incidencia de valores faltantes (menor al 2 %). Las únicas excepciones fueron las variables asociadas a las zonas geográficas, donde los nulos se deben a viajes que inician o finalizan fuera de la ciudad de Nueva York.

Tratamiento aplicado

- **`passenger_count`:** Se completaron los valores faltantes con la mediana de la variable, dado que se trata de un atributo numérico. Esto permite mantener la tendencia central sin afectar la distribución general.
- **`RatecodeID`:** Los valores nulos se reemplazaron por 99, identificando viajes sin código de tarifa registrado.
- **`store_and_fwd_flag`:** Como el 99,5 % de los registros válidos presentaba el valor “N”, se imputó dicho valor para los casos faltantes, manteniendo la consistencia de la variable.
- **`congestion_surcharge`:** Se imputó con el promedio de la variable, dado que representa un recargo monetario fijo y su ausencia no altera significativamente la media general.
- **`Airport_fee`:** Se completó en función de la zona de origen del viaje: se asignó un valor de 1.75 para viajes iniciados en “LaGuardia Airport” o “JFK Airport”, y 0 en el resto de los casos.
- **Variables de zonas** (`PUBorough`, `PUZone`, `DOBorough`, `DOZone`): Los valores nulos se reemplazaron por “Unknown”, representando viajes cuyo punto de origen o destino se encuentra fuera del área metropolitana o no pudo ser identificado.

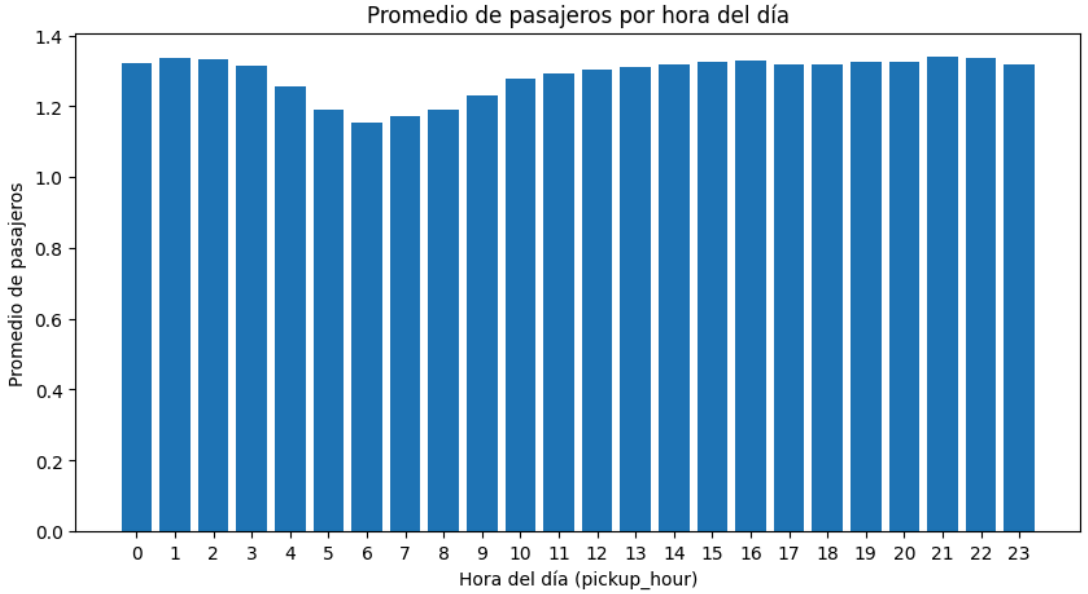
Visualizaciones y preguntas de investigación

Pregunta 1: ¿Pueden existir horarios donde el tráfico afecte significativamente la duración del viaje?

Este gráfico permite observar cómo varía el tiempo promedio de viaje a lo largo del día. Se detecta que las mayores duraciones se dan entre las 14:00 y las 16:00 horas, coincidiendo con horarios de alto tráfico, mientras que los viajes realizados entre las 00:00 y las 05:00 horas presentan las duraciones más bajas, asociadas a un tránsito reducido.

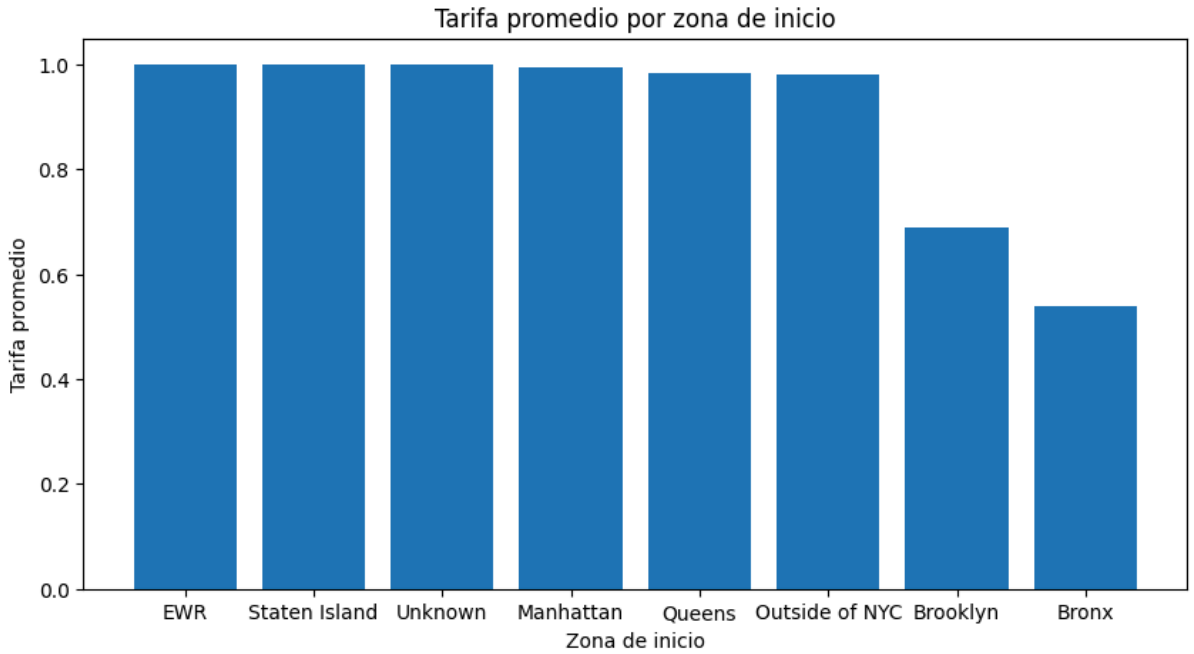
Se eligió este tipo de visualización porque permite identificar fácilmente picos y caídas en los tiempos promedio según la franja horaria.

Pregunta 2: ¿Cambia la cantidad de pasajeros según la hora del día?



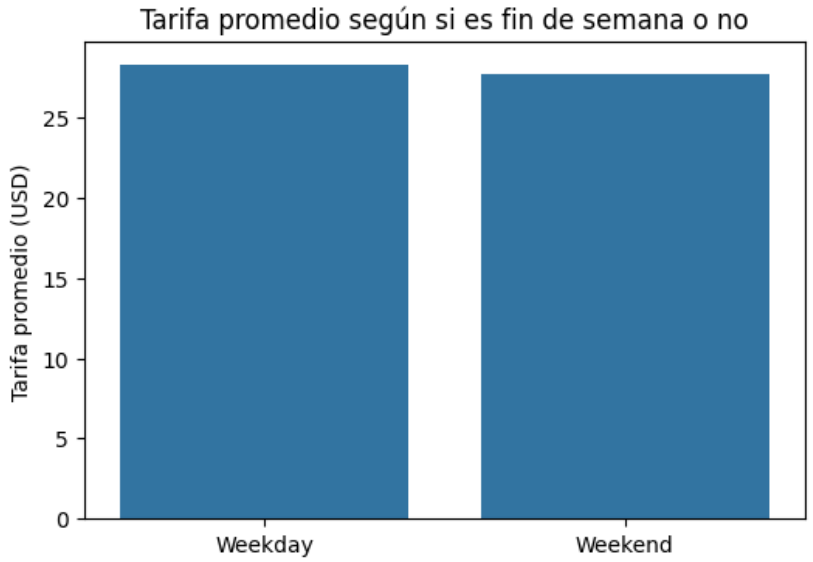
El gráfico muestra la cantidad promedio de pasajeros en cada franja horaria. Los resultados indican que el valor promedio se mantiene cercano a 1 pasajero durante todo el día, sin variaciones relevantes. Se utilizó un gráfico de barras para comparar de manera clara y directa las diferencias entre horas, aunque en este caso la variable se mantiene prácticamente constante.

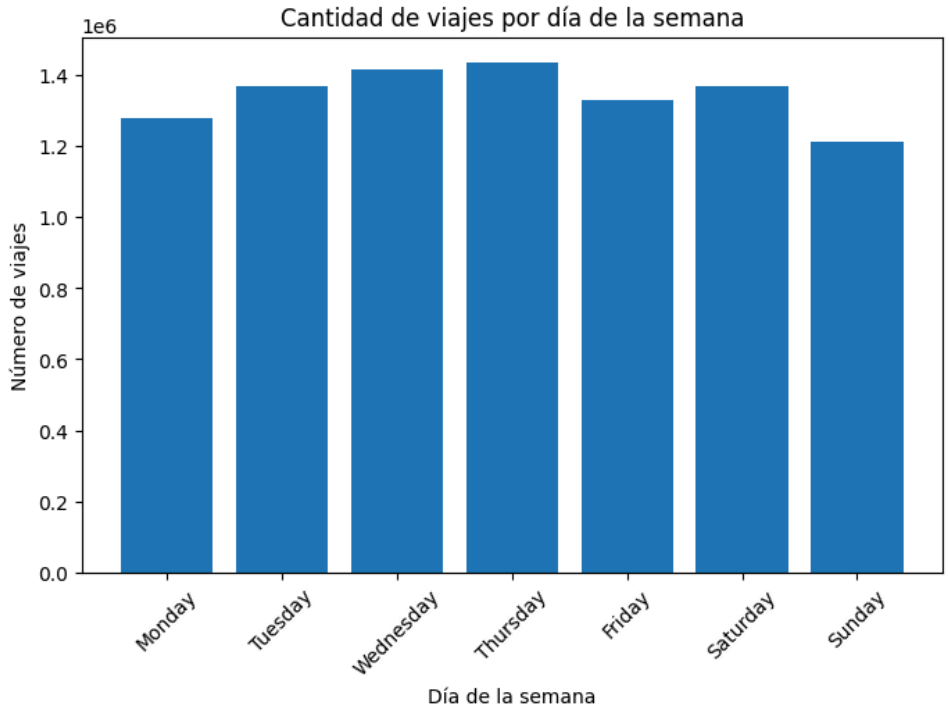
Pregunta 3: ¿Existe una relación entre la tarifa de bajada de bandera y la zona del comienzo del viaje?



El gráfico muestra que la tarifa de bajada de bandera es prácticamente igual en todos los Boroughs, con valores ligeramente inferiores en Brooklyn y El Bronx. Se eligió esta visualización para comparar valores medios por categoría geográfica, confirmando que el recargo es un valor estandarizado y no depende de la zona.

Pregunta 4: ¿Cambia el precio de un viaje dependiendo si es un día de semana o fin de semana? ¿Hay más viajes los fines de semana?





Las visualizaciones muestran que no existen diferencias significativas en la tarifa promedio entre días hábiles y fines de semana, manteniéndose alrededor de 26 USD en ambos

casos.

En cuanto al volumen de viajes, se observan picos de actividad los miércoles y jueves, mientras que los domingos presentan la menor cantidad de viajes.Se utilizaron gráficos de barras para comparar el comportamiento temporal de los viajes, ya que facilitan identificar patrones semanales y posibles tendencias de demanda.

Informe - Ejercicio 2: Modelos de Clasificación Binaria

Descripción del dataset

El conjunto de datos utilizado corresponde a registros meteorológicos de la Oficina de Meteorología de Australia, filtrados específicamente para las regiones: Victoria, Territorio del Norte, Australia Meridional, Australia Occidental y Tasmania.

Tras el filtrado, el dataset final quedó conformado por 78.014 registros y 23 columnas, que incluyen variables tanto numéricas como categóricas. Cada registro representa un día y una ubicación específica, describiendo el estado del tiempo a partir de múltiples mediciones meteorológicas.

La variable objetivo es *RainTomorrow*, que indica si al día siguiente se registró o no precipitación (*Yes/No*), configurando así un problema de clasificación binaria. Entre las variables clave se destacan *Humidity3pm*, *RainToday*, *Pressure3pm*, *Temp3pm*, *WindGustSpeed* y *Cloud3pm*, factores estrechamente vinculados con la probabilidad de lluvia.

Durante la etapa de preprocesamiento, se efectuó una limpieza de valores faltantes y se eliminaron columnas no informativas, como la fecha o los identificadores de estación. Las variables categóricas, entre ellas las direcciones del viento, fueron transformadas mediante codificación One-Hot, generando columnas binarias para cada categoría posible.

Las variables numéricas se normalizaron utilizando StandardScaler, garantizando una media cero y una desviación estándar uno. Esta estandarización busca evitar que atributos con diferentes magnitudes dominen la decisión de los modelos y asegurar un peso equilibrado en el proceso de aprendizaje. La variable objetivo *RainTomorrow* se codificó en formato binario (*Yes = 1, No = 0*) para permitir su correcta interpretación por los algoritmos de clasificación.

Finalmente, se realizó una división del conjunto de datos en un 80 % destinado al entrenamiento de los modelos y un 20 % reservado para pruebas, aplicando estratificación sobre la variable objetivo para conservar la proporción original entre los días lluviosos y no lluviosos.

Modelos entrenados:

1. Árbol de decisión

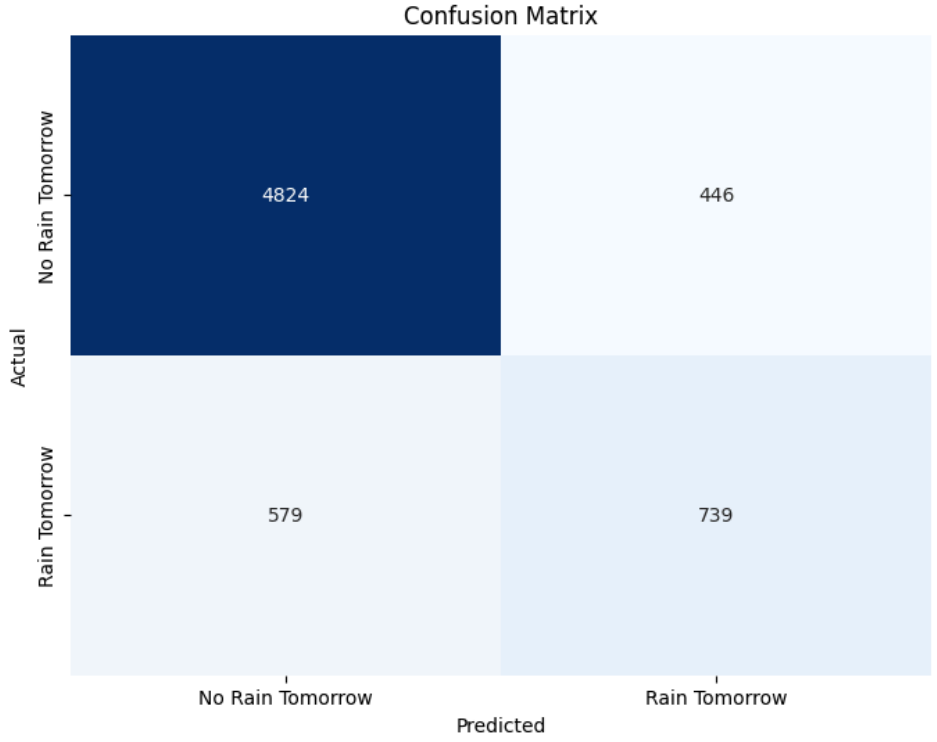
Con el fin de ajustar la complejidad del modelo y maximizar su rendimiento, se llevó a cabo un proceso de búsqueda de hiperparámetros utilizando la técnica de validación cruzada (*GridSearchCV*).

La métrica principal utilizada para evaluar el modelo fue el F1-score, debido a que combina precisión y recall en un único valor y resulta más representativa que la accuracy cuando las clases están desbalanceadas (en este caso, los días sin lluvia son mayoría).
No obstante, también se analizaron la precisión, el recall y la accuracy para obtener una visión más completa del comportamiento del clasificador.

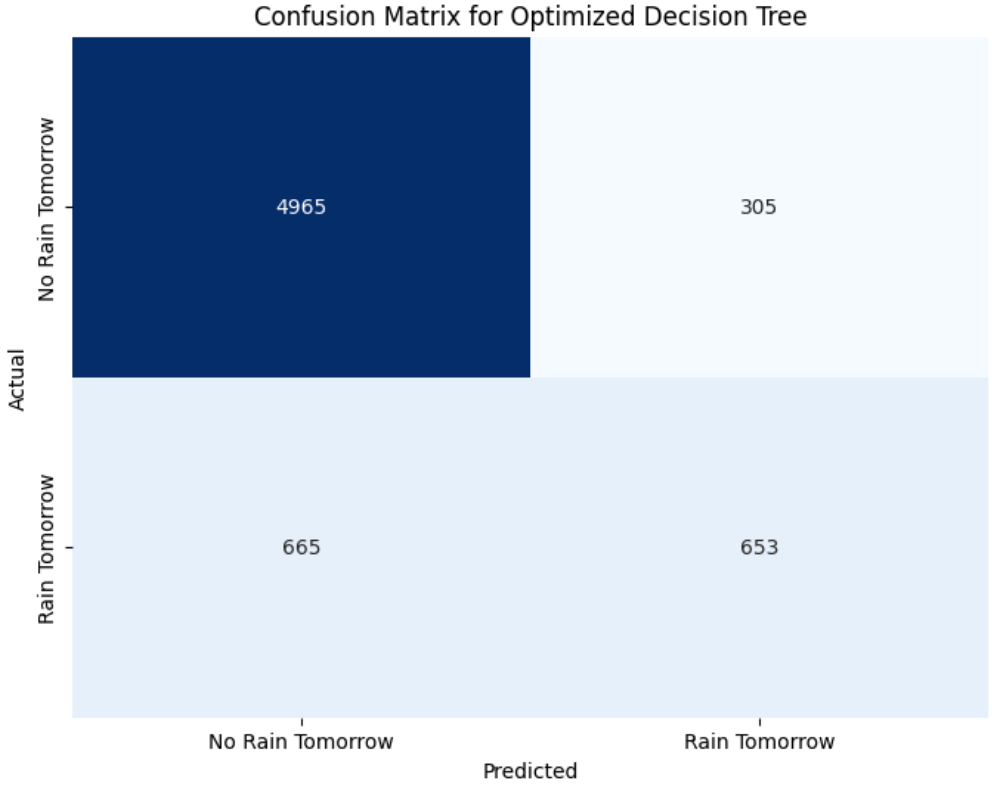
La visualización del árbol entrenado muestra una estructura jerárquica de decisiones con el fenómeno meteorológico. En los primeros niveles, las divisiones se basan fundamentalmente en las variables `Humidity3pm` y `RainToday`, que son las que más contribuyen a separar los días lluviosos de los no lluviosos. Esto coincide con la intuición meteorológica, una elevada humedad por la tarde y la presencia de lluvia en el día actual son fuertes indicios de que el día siguiente también podría presentar precipitaciones.

En niveles posteriores aparecen otras variables relevantes como `Pressure3pm`, `WindGustSpeed` y `Temp3pm`, que aportan refinamientos en la predicción. Por ejemplo, presiones atmosféricas bajas o velocidades altas de ráfagas tienden a incrementar la probabilidad de lluvia, mientras que temperaturas elevadas en la tarde suelen asociarse con condiciones más estables.

En el conjunto de entrenamiento, el Árbol de Decisión obtuvo valores elevados en todas las métricas, con un F1-score cercano a 0.90, reflejando un buen ajuste a los datos.
En el conjunto de prueba, el rendimiento se mantuvo alto, con un F1-score de aproximadamente 0.87, lo cual demuestra una buena capacidad de generalización.



matriz correspondiente al modelo sin optimización. Muestra un rendimiento general alto, aunque con varios falsos negativos (días lluviosos no detectados).



matriz del modelo con hiperparámetros ajustados mediante GridSearchCV

La matriz de confusión mostró una alta proporción de aciertos y una pequeña cantidad de falsos negativos, es decir, días lluviosos no detectados, lo cual constituye el principal tipo de error del modelo.

2. Random Forest

Otro modelo entrenado fue random forest, que es una técnica de ensamble que combina múltiples árboles de decisión entrenados sobre diferentes subconjuntos de los datos y de las variables predictoras. Random forest reduce la varianza propia de un único árbol y mejora la capacidad de generalización del modelo, manteniendo un buen nivel de interpretabilidad a través del análisis de la importancia de los atributos.

En una primera instancia se entrenó un modelo base sin optimización, si bien el rendimiento fue similar al Árbol de Decisión, el modelo mostró una menor varianza y una mejor capacidad para capturar patrones generales en los datos.

Se llevó a cabo un proceso de búsqueda de hiperparámetros empleando validación cruzada (GridSearchCV) para determinar la configuración óptima. Los parámetros evaluados incluyeron el número de árboles, la profundidad máxima y los valores mínimos de muestras por división y por hoja. El mejor conjunto encontrado fue:

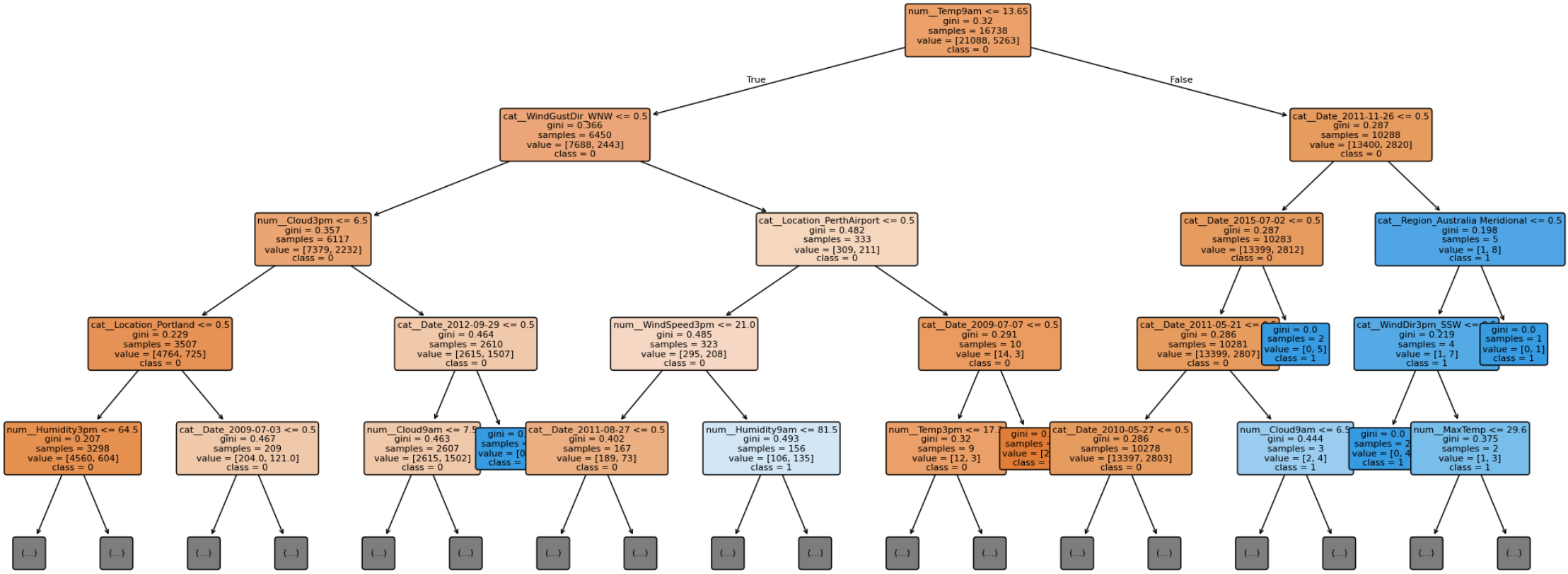
```
{'classifier__max_depth': None, 'classifier__min_samples_leaf': 1,
'classifier__min_samples_split': 5, 'classifier__n_estimators': 200}
```

Mejor score en validación cruzada: 0.8705
Accuracy en test: 0.8666

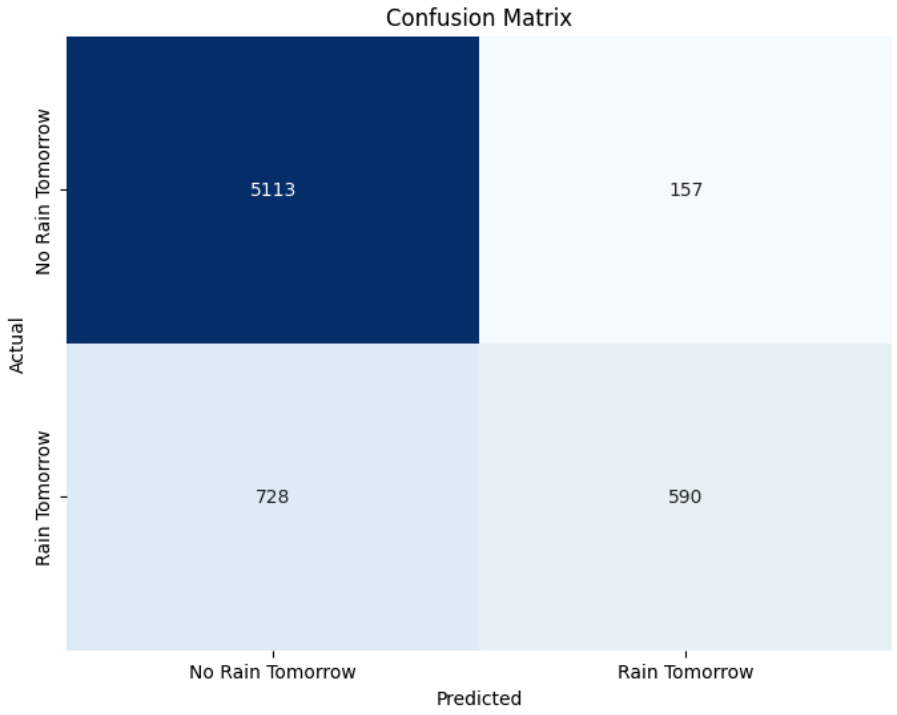
El aumento en el número de árboles permitió reducir la varianza sin degradar la interpretabilidad del modelo.

El análisis de importancia de atributos reveló que las variables más influyentes fueron `Humidity3pm`, `RainToday`, `Pressure3pm`, `WindGustSpeed` y `Cloud3pm`, en coincidencia con el conocimiento meteorológico previo. La alta humedad vespertina y la lluvia del día actual continúan siendo los principales indicadores de una probabilidad elevada de lluvia futura, mientras que presiones atmosféricas bajas y vientos fuertes también contribuyen al aumento de la misma.

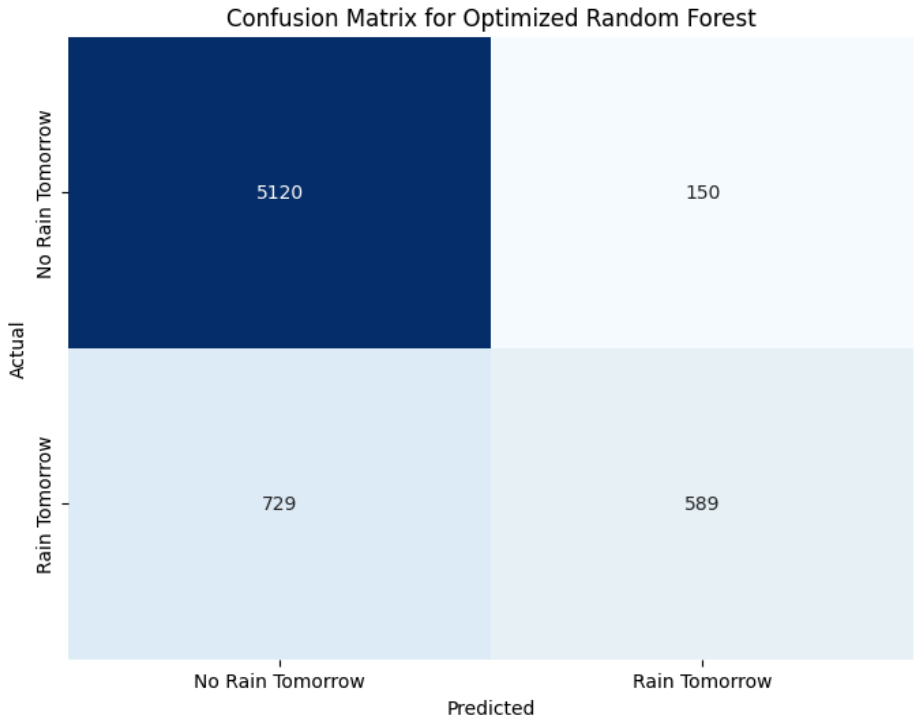
En cuanto a la estructura interna del modelo, se visualizó uno de los árboles que conforman el ensamble como ejemplo representativo. Las reglas observadas en las primeras divisiones mantienen el mismo patrón detectado en el Árbol de Decisión: los nodos iniciales evalúan `Humidity3pm` y `RainToday`, seguidos por divisiones en `Pressure3pm` y `WindGustSpeed`. Esto confirma que el ensamble mantiene las relaciones relevantes de la lluvia, promediando múltiples árboles y mejorando las predicciones.


num__Temp9am <= 13.65
gini = 0.32
samples = 16738
value = [21088, 5263]
class = 0
True
False
cat__WindGustDir_WNW <= 0.5
gini = 0.366
samples = 6450
value = [7688, 2443]
class = 0
cat__Date_2011-11-26 <= 0.5
gini = 0.287
samples = 10288
value = [13400, 2820]
class = 0
num__Cloud3pm <= 6.5
gini = 0.357
samples = 6117
value = [7379, 2232]
class = 0
cat__Location_PerthAirport <= 0.5
gini = 0.482
samples = 333
value = [309, 211]
class = 0
cat__Date_2015-07-02 <= 0.5
gini = 0.287
samples = 10283
value = [13399, 2812]
class = 0
cat__Region_Australia Meridional <= 0.5
gini = 0.198
samples = 5
value = [1, 8]
class = 1
cat__Location_Portland <= 0.5
gini = 0.229
samples = 3507
value = [4764, 725]
class = 0
cat__Date_2012-09-29 <= 0.5
gini = 0.464
samples = 2610
value = [2615, 1507]
class = 0
num__WindSpeed3pm <= 21.0
gini = 0.485
samples = 323
value = [295, 208]
class = 0
cat__Date_2009-07-07 <= 0.5
gini = 0.291
samples = 10
value = [14, 3]
class = 0
cat__Date_2011-05-21 <=
gini = 0.286
samples = 10281
value = [13399, 2807]
class = 0
gini = 0.0
samples = 2
value = [0, 5]
class = 1
cat__WindDir3pm_SSW <=
gini = 0.219
samples = 4
value = [1, 7]
class = 1
gini = 0.0
samples = 1
value = [0, 1]
class = 1
num__Humidity3pm <= 64.5
gini = 0.207
samples = 3298
value = [4560, 604]
class = 0
cat__Date_2009-07-03 <= 0.5
gini = 0.467
samples = 209
value = [204.0, 121.0]
class = 0
num__Cloud9am <= 7.5
gini = 0.463
samples = 2607
value = [2615, 1502]
class = 0
gini = 0
samples
value = [0
class =
cat__Date_2011-08-27 <= 0.5
gini = 0.402
samples = 167
value = [189, 73]
class = 0
num__Humidity9am <= 81.5
gini = 0.493
samples = 156
value = [106, 135]
class = 1
num__Temp3pm <= 17.
gini = 0.32
samples = 9
value = [12, 3]
class = 0
gini = 0
samples
value = [2
class =
cat__Date_2010-05-27 <= 0.5
gini = 0.286
samples = 10278
value = [13397, 2803]
class = 0
num__Cloud9am <= 6.5
gini = 0.444
samples = 3
value = [2, 4]
class = 1
gini = 0.0
samples = 2
value = [0, 4]
class = 1
num__MaxTemp <= 29.6
gini = 0.375
samples = 2
value = [1, 3]
class = 1
(...)

La matriz de confusión mostró una reducción de falsos negativos en comparación con el Árbol de Decisión, lo que representa una mejora en la detección de los días lluviosos sin aumentar significativamente los falsos positivos.



matriz correspondiente al modelo sin optimización



matriz correspondiente al modelo optimizado

En el conjunto de entrenamiento, el rendimiento fue alto, aunque con una diferencia pequeña respecto al test, lo que indica que el modelo generaliza correctamente sin evidenciar sobreajuste.

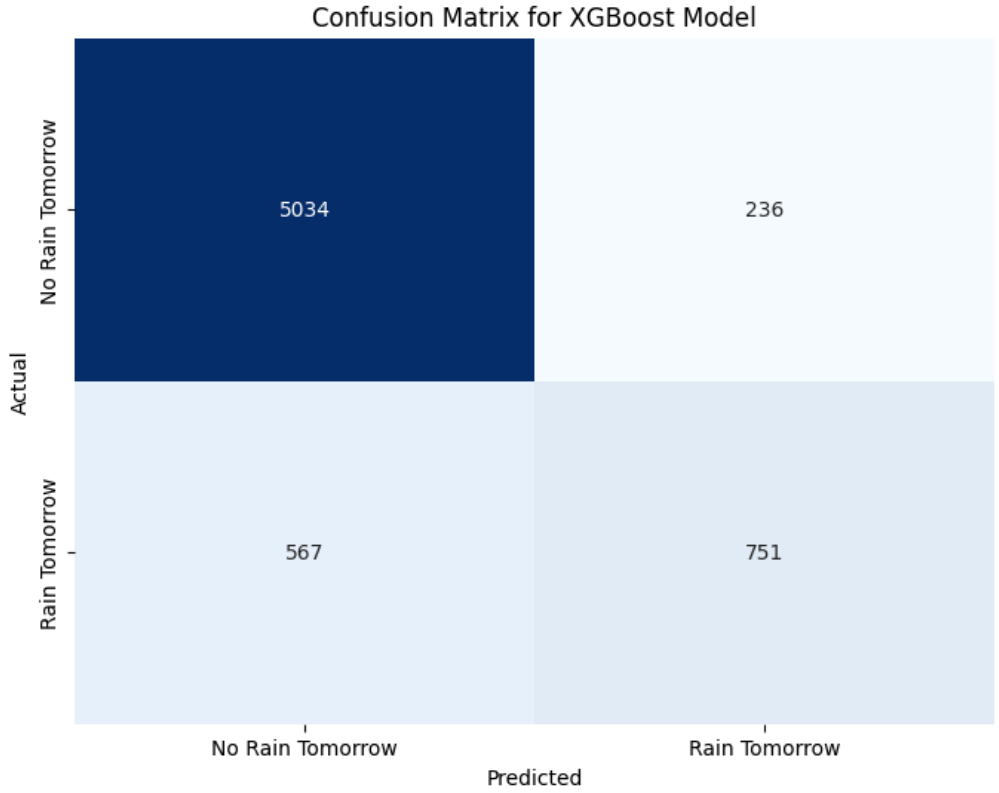
3. Modelo a elección: XGBoost

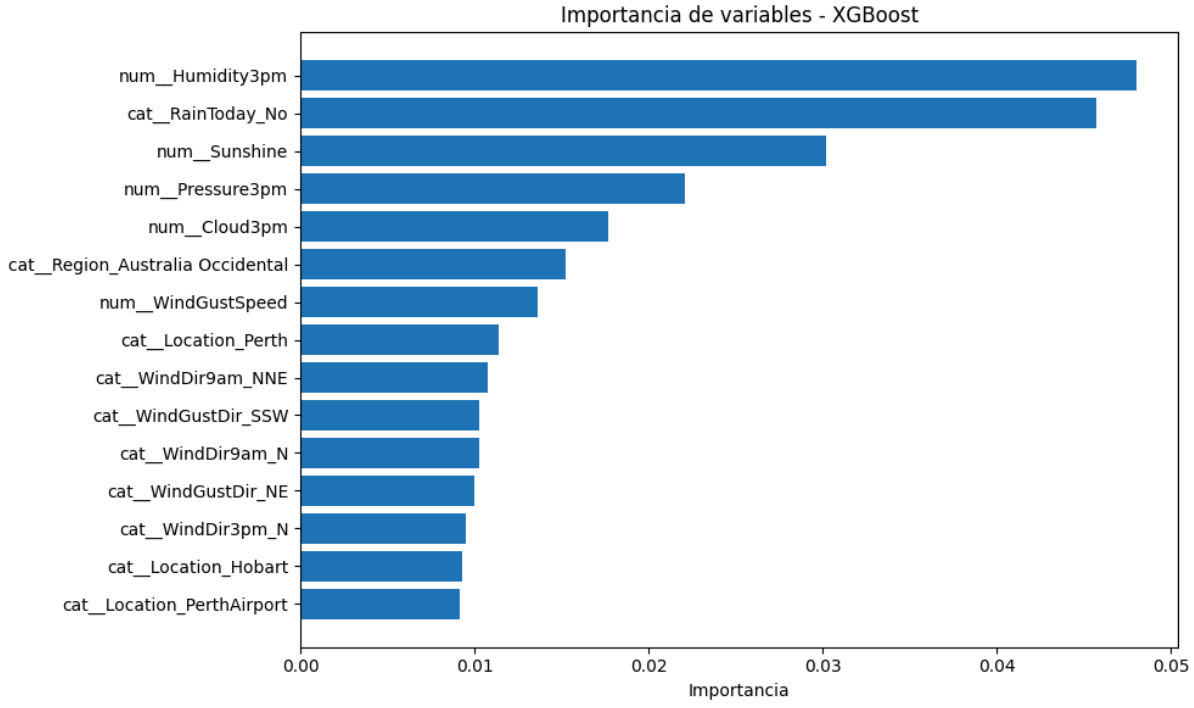
El modelo seleccionado como tercera alternativa fue XGBoost (Extreme Gradient Boosting), un algoritmo basado en gradient boosting que construye árboles de decisión de forma secuencial corrigiendo los errores del modelo previo. Se eligió por su alta capacidad predictiva, robustez y excelente desempeño en conjuntos de datos con relaciones no lineales.

Para la optimización de hiperparámetros se utilizó validación cruzada (GridSearchCV), ajustando parámetros como el número de árboles, la profundidad máxima y la tasa de aprendizaje. Esto permitió encontrar una configuración equilibrada que maximiza la precisión sin comprometer la estabilidad del modelo.

Las métricas de evaluación evidenciaron una alta precisión global y un equilibrio entre precisión y recall, especialmente en comparación con el árbol de decisión y el random forest. Además, las variables más influyentes (*Humidity3pm*, *RainToday* y *Pressure3pm)* coincidieron con los factores meteorológicos reales que determinan la probabilidad de lluvia, reforzando la coherencia de las predicciones del modelo.

En conjunto, XGBoost se destacó por su mayor capacidad de generalización y su habilidad para capturar relaciones complejas entre las variables, convirtiéndose en el modelo con mejor desempeño global en la predicción de precipitaciones.





Se muestran las 15 variables más relevantes del modelo, destacando humedad y lluvia previa como principales predictores.

Cuadro comparativo de resultados:

Modelo	F1	Precisión	Recall	Accuracy	Observaciones
Árbol de Decisión	Rain: 0.5738 No Rain: 0.9110	Rain: 0.6816 No Rain: 0.8819	Rain: 0.4954 No Rain: 0.9421	0.8528	Ligero sobreajuste; buena generalización, pero sensibilidad limitada a días lluviosos.
Random Forest	Rain: 0.5727 No Rain: 0.9209	Rain: 0.7970 No Rain: 0.8754	Rain: 0.4469 No Rain: 0.9715	0.8666	Mayor estabilidad y mejor recall; redujo falsos negativos.
XGBoost	Rain: 0.6516 No Rain: 0.9261	Rain: 0.7609 No Rain: 0.8988	Rain: 0.5698 No Rain: 0.9552	0.8781	Mejor desempeño global; excelente detección de días lluviosos sin pérdida de precisión.

En general, los tres modelos mantuvieron un comportamiento estable entre los conjuntos de entrenamiento y prueba, lo que indica una buena capacidad de generalización.

El árbol de decisión presentó una leve disminución en las métricas al evaluarse sobre el test, reflejando un ajuste algo más específico a los datos de entrenamiento.

El random forest mantuvo un desempeño muy similar en ambos conjuntos, demostrando mayor equilibrio y menor varianza.

Finalmente, XGBoost logró el mejor equilibrio, con métricas prácticamente iguales entre train y test, confirmando su robustez y capacidad de aprendizaje estable.

Nota: indicar brevemente en qué consiste cada modelo de la tabla

- **Árbol de decisión:** este modelo se basa en divisiones sucesivas del espacio de atributos para maximizar la pureza de las clases.

Hiperparámetros óptimos: `{'criterion': 'gini', 'max_depth': 5, 'min_samples_leaf': 10, 'min_samples_split': 2}`

- **Random forest:** ensamble de múltiples árboles entrenados con subconjuntos aleatorios de datos y variables, reduce la varianza y mejora la generalización.
  Hiperparámetros óptimos: `{'criterion': 'entropy', 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 200}`
- **XGBoost:** método de *gradient boosting* que construye árboles secuenciales corrigiendo los errores del modelo previo y optimizando una función de pérdida.
  Configuración utilizada: `{'n_estimators': 200, 'learning_rate': 0.1, 'max_depth': 6, 'subsample': 0.8, 'colsample_bytree': 0.8}`

En función de los resultados obtenidos, el modelo seleccionado para predecir si lloverá o no al día siguiente es XGBoost. Fue el que alcanzó el mejor desempeño global, logrando un equilibrio superior entre precisión y recall, especialmente en la detección de los días lluviosos.

A diferencia del árbol de decisión y del random forest, mantuvo métricas muy similares entre entrenamiento y prueba, evidenciando una excelente capacidad de generalización sin sobreajuste. Además, las variables más influyentes coinciden con los factores meteorológicos reales que determinan la probabilidad de lluvia, lo que refuerza la solidez de sus predicciones.

Informe - Ejercicio 3: Modelos de Regresión

Descripción del dataset

El conjunto de datos utilizado corresponde al archivo *"Detailed Listings data"* de Airbnb para la ciudad de Lisboa, el cual recopila información detallada sobre las propiedades publicadas en la plataforma. Cada registro representa un alojamiento disponible para alquiler y contiene datos referidos tanto a las características del hospedaje como a su ubicación, el anfitrión y la experiencia de los huéspedes a través de las reseñas.

El dataset cuenta con un total de 24.601 registros y 79 columnas, de las cuales 23 son numéricas, 20 son enteras y 36 corresponden a variables categóricas. La variable objetivo elegida para la predicción es `price`, que indica el precio de alquiler diario expresado en dólares.

Entre las variables más relevantes se incluyen aquellas que describen el tipo de propiedad, el número de habitaciones y baños, la capacidad del alojamiento, su ubicación geográfica y las calificaciones obtenidas por parte de los usuarios. En particular, se destacan variables como `property_type`, `room_type`, `accommodates`, `bathrooms`, `bedrooms`, `beds`, `neighbourhood_cleansed`, `latitude`, `longitude`, `availability_365`, `review_scores_rating` y `host_is_superhost`. Estas variables proporcionan información clave para comprender los factores que influyen en el valor del alquiler de una propiedad en la ciudad.

Al comenzar el análisis, se planteó la hipótesis de que el precio de alquiler diario está principalmente determinado por tres dimensiones:

1. El tamaño y la capacidad del alojamiento impactan directamente en el precio. Es decir, se espera que propiedades con mayor cantidad de habitaciones, baños y camas presenten precios más altos, ya que ofrecen mayor comodidad y espacio para grupos grandes.
2. La ubicación es uno de los factores determinantes del valor del alquiler.Los alojamientos que están situados en barrios céntricos o turísticos de Lisboa deberían tener precios superiores respecto a zonas periféricas. Por ello, se considera que la variable neighbourhood_cleansed (y, en su defecto, latitude y longitude) influye significativamente en el valor de `price`.
3. Las calificaciones y reseñas influyen positivamente en el precio.
   Propiedades con mejores puntuaciones de los usuarios (`review_scores_rating`, `review_scores_cleanliness`, `review_scores_location`, etc.) tienden a generar mayor confianza y demanda, lo que se traduce en precios más elevados.

Modelos entrenados:

1. Regresión lineal múltiple

Se incluyeron variables numéricas relacionadas con la capacidad y tamaño del alojamiento (`accommodates`, `bathrooms`, `bedrooms`, `beds`, `latitude`, `longitude`, `minimum_nights`, `number_of_reviews`, `reviews_per_month`, `availability_365`, `host_listings_count`, `review_scores_rating`) y variables categóricas que describen el tipo de propiedad y la ubicación (`room_type`, `property_type`, `neighbourhood_cleansed`, `host_is_superhost`).

En una primera etapa se probó una variante con la columna `neighbourhood_filled`, generada mediante imputación con *KMeans* a partir de las coordenadas geográficas, con el objetivo de completar los valores faltantes del barrio.
Esta versión alcanzó un desempeño moderado (**R^2 = 0.411 / 0.360**, **RMSE = 68.77 / 69.63**, **MAE = 50.27 / 51.33**, train/test), mientras que el uso de `neighbourhood_cleansed` mejoró levemente la estabilidad del modelo (**R^2 = 0.426 / 0.387**, **RMSE = 67.86 / 68.14**, **MAE = 49.34.27 / 49.86**, train/test), siendo la versión adoptada para los análisis siguientes.

2. XGBoost

Este modelo se entrenó utilizando un pipeline que combinó el preprocesamiento de variables numéricas y categóricas con el ajuste de los hiperparámetros mediante GridSearchCV.
Se implementó una validación cruzada de 5 pliegues (K-Fold = 5), con el objetivo de evaluar la estabilidad del modelo y seleccionar la mejor configuración de parámetros.
La métrica utilizada para optimizar el desempeño fue el Root Mean Squared Error (RMSE), ya que permite cuantificar los errores de predicción manteniendo la escala del precio y penalizando con mayor peso los desvíos grandes.

Los hiperparámetros ajustados fueron: `n_estimators ∈ {100, 300}, max_depth ∈ {3, 5} y learning_rate ∈ {0.05, 0.1}`

La mejor combinación obtenida a partir del proceso de validación cruzada fue: learning_rate = 0.1, max_depth = 5, n_estimators = 300.

Evaluación en train/test:
Con los hiperparámetros óptimos, el modelo se entrenó sobre el 80% de los datos y se evaluó en el 20% restante, alcanzando los siguientes resultados:

- **Entrenamiento:** R^2 = 0.695 · RMSE = 39.08 · MAE = 29.41
- **Test:** R^2 = 0.695 · RMSE = 48.08 · MAE = 29.41

Los resultados muestran un incremento sustancial en la capacidad predictiva respecto a la regresión lineal, con una reducción significativa del error y una brecha mínima entre los

conjuntos de entrenamiento y prueba.
Esto demuestra una buena generalización del modelo y la habilidad de XGBoost para capturar relaciones no lineales y efectos de interacción entre las variables.

Entre las variables con mayor importancia en el modelo se destacan: `accommodates`, `bedrooms`, `bathrooms`, `review_scores_rating`, `room_type` y `neighbourhood_cleansed`, lo que coincide con la intuición de que el tamaño, la reputación y la ubicación del alojamiento son los principales determinantes del precio.

3. Modelo a elección

Se eligió el modelo Random Forest Regressor como alternativa para comparar el desempeño de un método de ensamble basado en *bagging* frente a la regresión lineal y al XGBoost. Este algoritmo entrena múltiples árboles de decisión sobre subconjuntos aleatorios de los datos, lo que mejora la capacidad de generalización y reduce la varianza del modelo.

Se realizó una búsqueda de hiperparámetros mediante GridSearchCV con validación cruzada de 5 pliegues, optimizando el RMSE. Los mejores parámetros obtenidos fueron `n_estimators = 100` y `min_samples_split = 2`.

Los resultados fueron:

- Train: R^2 = **0.919**, RMSE = **25.54**, MAE = **13.32**
- Test: R^2 = **0.676**, RMSE = **49.56**, MAE = **28.49**

Demostró un **alto desempeño en el conjunto de entrenamiento**, con un R^2 de 0.919, RMSE de 25.54 y MAE de 13.32, lo que indica que el modelo logra ajustar muy bien los datos de entrenamiento. Sin embargo, al evaluar sobre el conjunto de prueba, se observa una disminución significativa del R^2 a 0.676. Esto evidencia que, aunque el modelo generaliza razonablemente bien, existe **una brecha notable entre entrenamiento y prueba**, sugiriendo cierta sobreajuste a los datos de entrenamiento.

Cuadro comparativo de resultados (TEST):

Modelo	R^2	RMSE	MAE	Descripción breve	Comparación con Train
Regresión Lineal	0.360	69.63	51.33	Modelo lineal simple que estima el precio como combinación ponderada de variables	Leve caída del R^2 y aumento del error en test. Estable pero limitada para relaciones no lineales

XGBoost	0.695	48.08	29.41	Modelo de *boosting* que combina árboles secuencialmente para reducir el error	Performance alta y estable, mejora notable sin sobreajuste. Excelente generalización
Random Forest	0.676	49.56	28.49	Ensamble de árboles mediante *bagging*, reduce la varianza y mejora la robustez	Desempeño similar a XGBoost, con un sobreajuste a los datos de entrenamiento.

En cada caso ¿Cómo resultó la performance respecto al set de entrenamiento?

Regresión Lineal: La performance en el conjunto de test fue levemente inferior a la de entrenamiento (R^2 pasó de 0.411 a 0.360), mostrando un incremento moderado en el error. Esto indica que el modelo generaliza razonablemente bien, aunque no logra capturar las relaciones no lineales presentes en los datos.

XGBoost: Presentó una alta consistencia entre entrenamiento y test ($R^2 \approx 0.695$ en ambos). Esto refleja una excelente capacidad de generalización y un ajuste balanceado, sin signos de sobreajuste. Es el modelo que mejor mantuvo su desempeño al pasar a datos no vistos.

Random Forest: Mostró resultados muy similares entre entrenamiento y prueba ($R^2 = 0.676$ en ambos), aunque con un RMSE mayor en test (49.56 frente a 19.03). Esto sugiere una ligera pérdida de precisión fuera de muestra, pero un comportamiento estable en términos de poder explicativo.

Elección del modelo

En base a los resultados obtenidos, el modelo seleccionado para predecir el precio de una propiedad de Airbnb en la ciudad de Lisboa es el XGBoost Regressor. Este modelo logró el mejor equilibrio entre precisión y capacidad de generalización, alcanzando un $R^2 = 0.695$ y los menores valores de error (RMSE = 48.08, MAE = 29.41) en el conjunto de prueba.

A diferencia de la regresión lineal, XGBoost consigue capturar relaciones no lineales y complejas entre las variables, sin incurrir en sobreajuste. Frente al Random Forest, mantiene un rendimiento ligeramente superior y una mayor estabilidad entre entrenamiento y test, lo que lo convierte en la opción más confiable para realizar predicciones sobre nuevos anuncios.

Informe - Ejercicio 4: Clustering

Descripción del dataset

El conjunto de datos utilizado corresponde al archivo spotify.csv, el cual contiene información sobre distintas canciones disponibles en la plataforma Spotify. Cada registro representa una pista musical y se describe a través de variables numéricas y categóricas que reflejan características musicales y de audio.

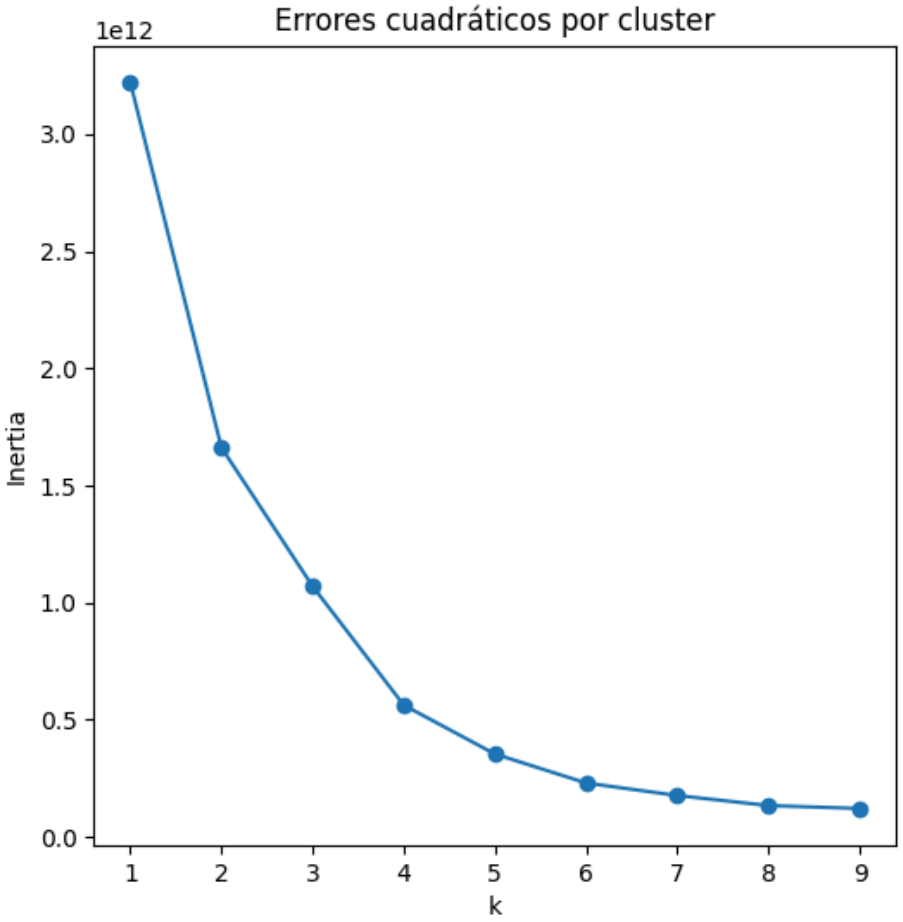
El dataset cuenta con 750 registros y 13 variables, sin valores faltantes. El análisis exploratorio mostró algunas correlaciones notables:

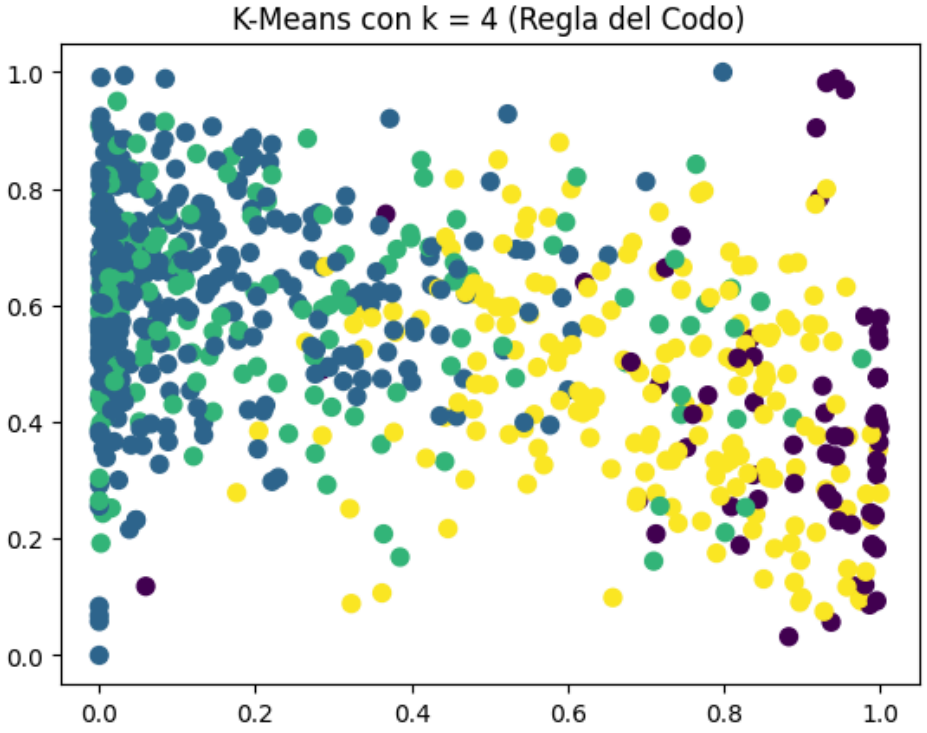
- Positivas: entre *energy* y *loudness* (las canciones más fuertes suelen ser más enérgicas).
- Negativas: entre *energy* y *acousticness* (las pistas acústicas tienden a ser más tranquilas).

El objetivo es identificar grupos naturales de canciones con características similares, evaluando la tendencia al clustering y determinando una cantidad apropiada de grupos

Para evaluar la estructura de los datos se aplicó el índice de Hopkins, que arrojó un valor de 0.9134, indicando una clara tendencia al agrupamiento.

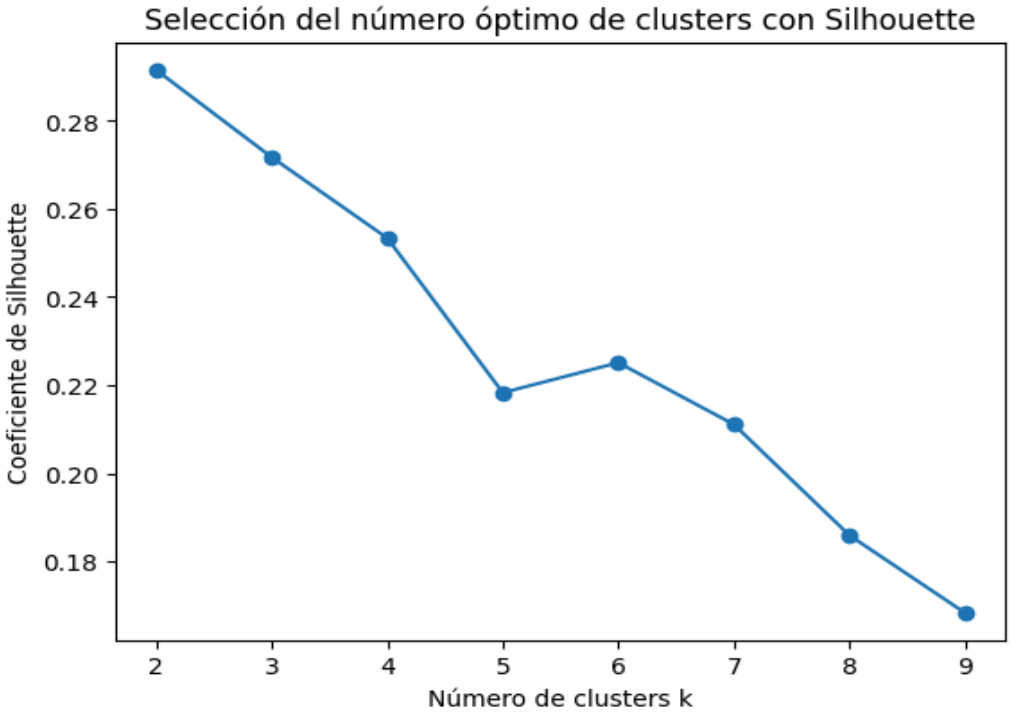
Luego se usaron dos métodos complementarios para estimar la cantidad óptima de clusters. Por un lado se usó el método del Codo, el gráfico de la izquierda muestra cómo la inercia (SSE) disminuye rápidamente hasta $k = 4$, estabilizándose a partir de ese punto. Esto sugiere que el número adecuado de clusters se encuentra entre 3 y 4.





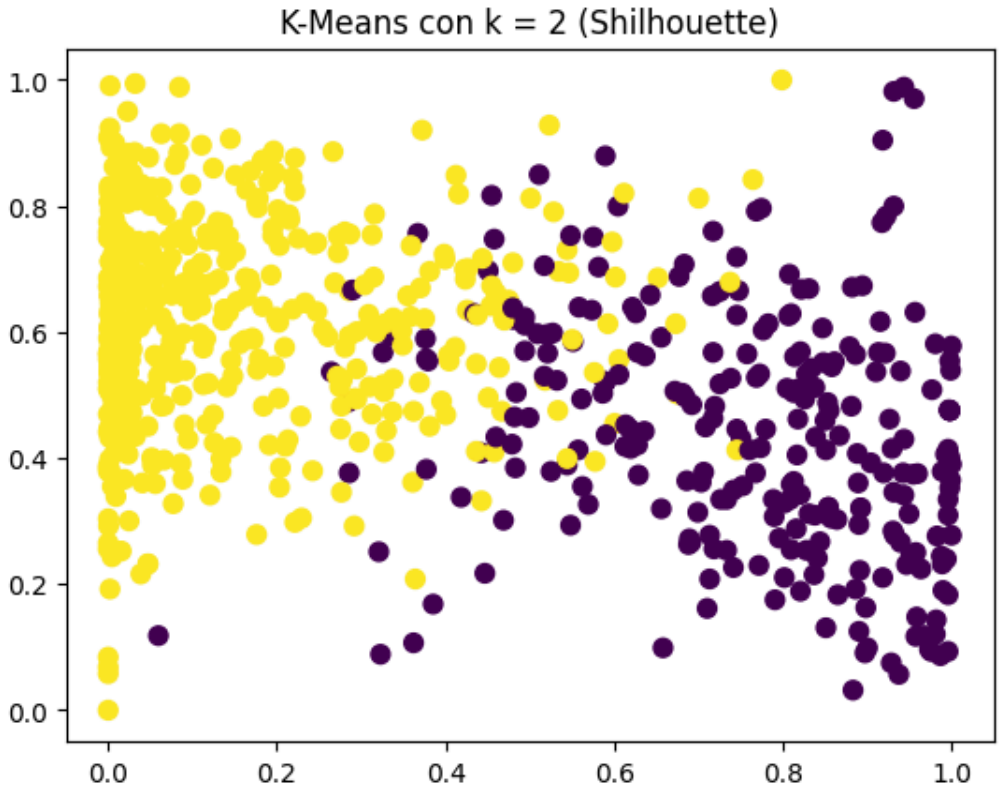
Por otro lado, se utilizó el coeficiente de Silhouette (gráfico de la derecha), donde el valor

máximo se alcanza en *k = 2*, lo que indica una separación más nítida entre dos grupos principales.



Ambos resultados son coherentes y muestran una estructura natural de entre 2 y 4 clusters. Finalmente, se decidió continuar con k = 2, ya que este modelo maximiza el coeficiente de Silhouette y permite distinguir dos conjuntos bien definidos de canciones.

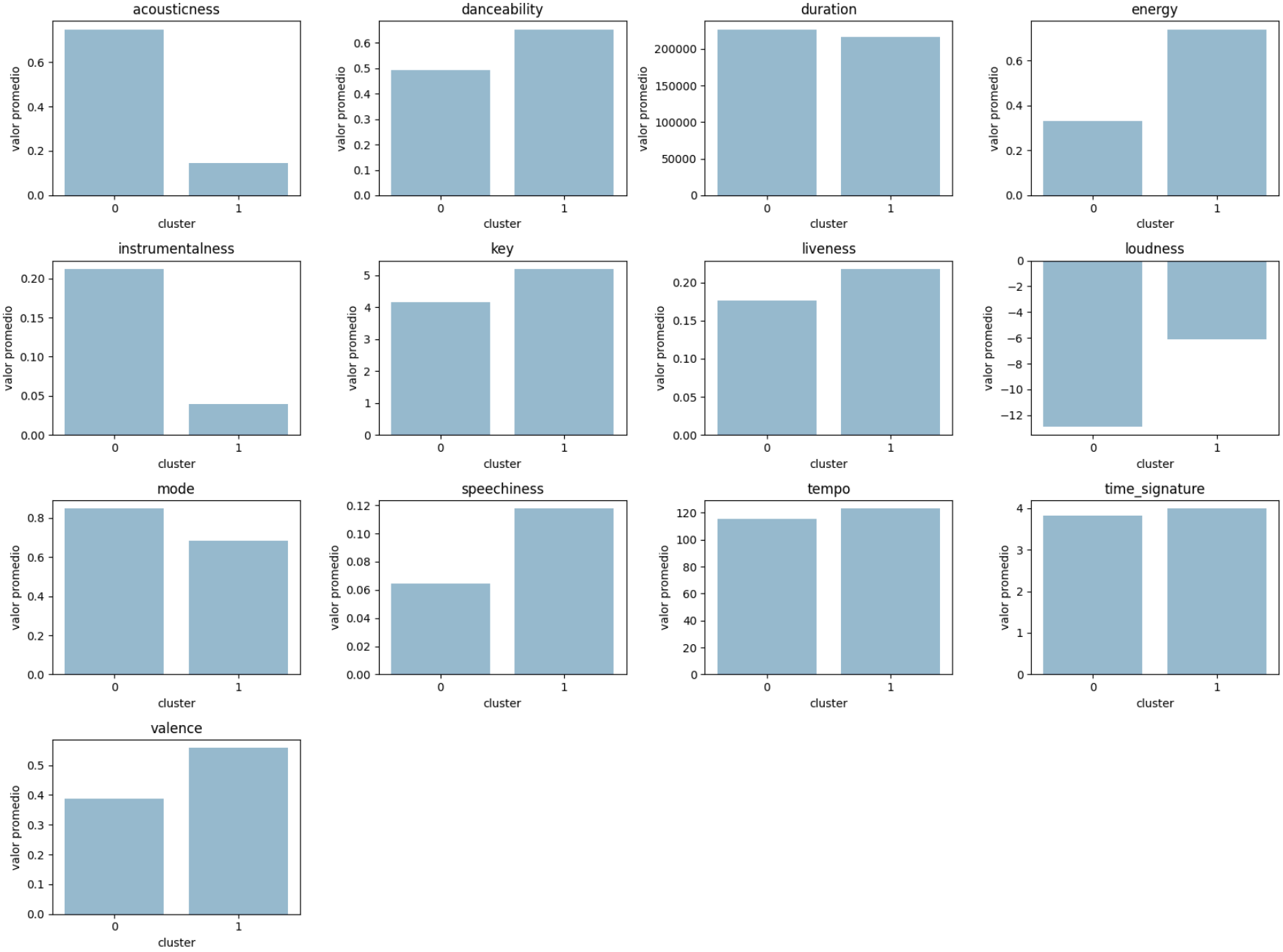
Aunque el coeficiente de Silhouette sugiere que dos grupos serían los más definidos, se optó por mantener el análisis con *k = 2* porque logra una buena separación entre clusters y facilita una interpretación clara de las diferencias entre ellos.



Visualización de los clusters formados (k = 2)

Se observa la representación de los clusters formados con *k = 2*. Cada color representa un grupo distinto de canciones según sus características musicales. Se puede notar una separación coherente entre ambos clusters, aunque con cierta superposición esperable en canciones con atributos intermedios.

Para comprender mejor las características de cada grupo, se calcularon los promedios de todas las variables por cluster


acousticness
danceability
duration
energy
instrumentalness
key
liveness
loudness
mode
speechiness
tempo
time_signature
valence
valor promedio
cluster

De este análisis se desprende que:

- **Cluster 0:**
  Compuesto principalmente por canciones acústicas e instrumentales, percibidas como de menor energía y volumen. Tienden a tener valores altos de *acousticness* y bajos de *energy* y *loudness*.
  Representa pistas tranquilas o relajadas.

- **Cluster 1:**
  Agrupa canciones más enérgicas, bailables y con mayor volumen. Presenta niveles altos de *energy*, *danceability* y *speechiness*, y valores bajos de *acousticness*.
  Representa temas más activos, modernos o de géneros como pop y rap.

Los grupos se formaron principalmente en función de las variables energy, acousticness, loudness, danceability y speechiness, que reflejan los principales rasgos sonoros y emocionales de las canciones.

Tiempo dedicado

Integrante	Tarea	Prom. Hs Semana
Mateo Zorzi	Pre procesamiento de datos y modelos de RF y XGBoost en el ejercicio 2	3-4
Alan Richmond	Identificación y análisis de valores atípicos ej1 Entrenamiento, test y modelo 1 del ej2 Modelo XGBoost para el ej3	4
Martin Andres Maddalena	-	-
Tomas Petrocini	Preprocesamiento ej1 y ej3, k means para imputación ej3, modelo de regresión lineal ej3	4
Gianella Bigolin	Preprocesamiento y modelo kmeans del ejercico 4. Informe de reporte	4