



CIENCIA DE DATOS
CÁTEDRA RODRÍGUEZ

Trabajo Práctico 1

2C 2025

14 de octubre de 2025

Nombre	Padrón	Mail
Lorenzo Busato	110490	lbusato@fi.uba.ar
Tomas Caporaletti	108598	tcaporaletti@fi.uba.ar
Helen Elizabeth Chen	110195	hchen@fi.uba.ar
Blas Sebastián Chuc	110253	bchuc@fi.uba.ar
Franco Agustin Rodriguez	108799	frodriguez@fi.uba.ar

Índice

1. Análisis Exploratorio de Datos	3
1.1. Descripción del dataset	3
1.2. Preprocesamiento	3
1.3. Visualizaciones y preguntas de investigación	4
2. Modelos de Clasificación Binaria	7
2.1. Descripción del dataset	7
2.2. Modelos entrenados	12
2.2.1. Árbol de decisión	12
2.2.2. Random Forest	15
2.2.3. XGBoost (modelo a elección)	18
2.3. Comparación de resultados	21
2.4. Elección del modelo	22
3. Modelos de Regresión	22
3.1. Descripción del dataset	22
3.2. Modelos entrenados	23
3.2.1. Regresión lineal	23
3.2.2. XGBoost Regressor	25
3.2.3. KNN Regressor	27
3.3. Comparación de resultados	29
3.4. Elección del modelo	30
4. Clustering	30
4.1. Descripción y Análisis del dataset	30
4.2. Análisis de la tendencia al clustering del dataset	32
4.3. Estimación de la cantidad apropiada de grupos que se deben formar	32
4.4. Análisis de los clusters formados	34
4.5. Conclusión	36
5. Tiempo dedicado	36

Índice

1. Análisis Exploratorio de Datos

1.1. Descripción del dataset

El dataset utilizado corresponde a los registros de viajes en taxis *Yellow Cab* de la ciudad de Nueva York, publicados por la *New York City Taxi and Limousine Commission (TLC)*. Contiene información detallada de cada viaje, incluyendo fecha y hora de inicio y fin, duración, distancia recorrida, montos cobrados, propinas, tipo de pago y localización de origen y destino mediante códigos de zona geográfica (LocationIDs).

- **Cantidad de registros y columnas:** Se analizaron aproximadamente 10 millones de registros correspondientes a un período de tres meses (enero, febrero y marzo de 2025), considerando únicamente las columnas relevantes para el análisis, reduciendo así el uso de memoria. Se seleccionaron alrededor de 20 variables de las más de 40 disponibles en el dataset original.
- **Variables más relevantes:** Las principales variables cuantitativas utilizadas fueron `trip_distance` (distancia en millas), `ride_length` (duración del viaje en minutos), `total_amount` (monto total pagado), `fare_amount` (tarifa base), `tip_amount` (monto de propina) y `speed_mph` (velocidad promedio). Entre las variables cualitativas se destacan `payment_type` (medio de pago), `RatecodeID` (tipo de tarifa), `PULocationID` y `DOLocationID` (zonas de origen y destino) y `passenger_count` (cantidad de pasajeros).
- **Hipótesis iniciales o supuestos de trabajo:**
 - La duración y distancia de los viajes presentan patrones horarios y semanales marcados (horas pico).
 - Las propinas (`tip_rate`) se asocian positivamente con la distancia, duración y viajes hacia aeropuertos o entre zonas distintas.
 - Los viajes de un solo pasajero tienden a ser más cortos que los grupales.
 - Existen zonas de la ciudad que concentran la mayor proporción de viajes largos.

1.2. Preprocesamiento

El preprocesamiento se centró en la limpieza, validación y generación de variables necesarias para responder las preguntas de investigación, siguiendo criterios coherentes con la normativa de la TLC y las prácticas de análisis reproducibles.

- **Columnas eliminadas:** Se descartaron variables redundantes o irrelevantes para las preguntas planteadas, tales como `VendorID`, `store_and_fwd_flag`, `ehail_fee`, y otras asociadas a licencias o metadatos de carga. Esto redujo significativamente el tamaño en memoria del DataFrame.
- **Correlaciones destacables:** Se observó una alta correlación positiva entre `trip_distance` y `ride_length`, así como entre `fare_amount` y `total_amount`. La variable `speed_mph`, derivada de ambas, mostró variaciones consistentes con la congestión del tránsito en horas pico.
- **Nuevos features generados:**
 - `ride_length`: duración del viaje en minutos (calculada a partir de las fechas de inicio y fin).
 - `speed_mph`: velocidad promedio del viaje (millas por hora).

- **part_of_day**: franja horaria categorizada en *Madrugada [0-6)*, *Mañana [6-12)*, *Tarde [12-18)* y *Noche [18-24)*.
 - **fare_subtotal**: monto sobre el cual se calcula la propina, excluyendo impuestos y cargos no tippeables.
 - **tip_rate**: relación entre propina y tarifa base.
- **Valores atípicos**: Se eliminaron viajes con tiempos o distancias inconsistentes (por ejemplo, duración negativa o distancia 0), tarifas totales negativas y velocidades mayores a 60 mph, que se interpretan como errores de carga. Además, se acotaron las colas superiores de distancia y duración usando percentiles (P99.5) para evitar que unos pocos registros extremos distorsionaran los gráficos.
 - **Datos faltantes**: Las variables con pocos valores faltantes, como **Airport_fee** o **cbd_congestion_fee**, se completaron con 0. Los casos sin identificadores de zona (**PULocationID** o **DOLocationID**) fueron descartados. En **passenger_count**, los valores nulos o igual a 0 fueron reemplazados por la moda del conjunto (1 pasajero).

1.3. Visualizaciones y preguntas de investigación

Las visualizaciones se diseñaron para responder a las preguntas planteadas por el grupo, utilizando gráficos de calor, barras y boxplots. Todas las figuras fueron generadas con **matplotlib** y **seaborn**, asegurando una visualización clara y escalas apropiadas.

■ Preguntas planteadas:

1. ¿Cómo varían la distancia y duración por hora y día de semana?
2. ¿La velocidad promedio cae en horas pico?
3. ¿Qué factores se asocian con una mayor **tip_rate**?
4. ¿Qué zonas origen/destino concentran viajes largos?
5. ¿Las distancias más largas se realizan en viajes en solitario o con 2 o más pasajeros?
6. ¿El lugar de origen/ destino del viaje influye en la probabilidad de que el pasajero deje propina?

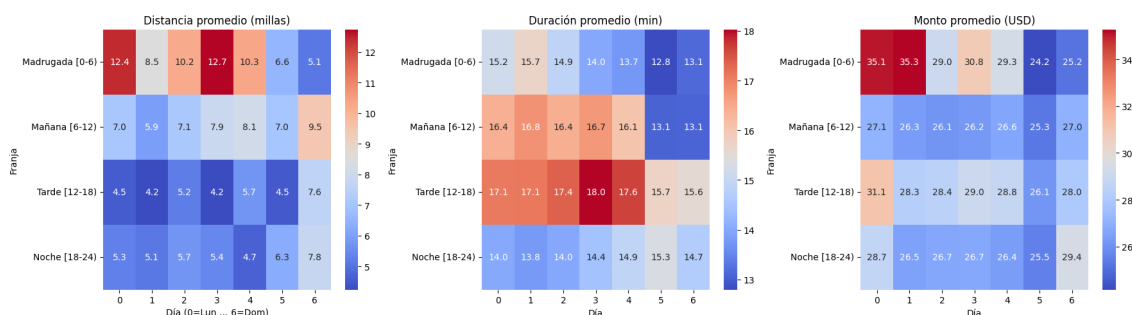


Figura 1: Distancia, duración y monto promedio por franja horaria y día de semana.

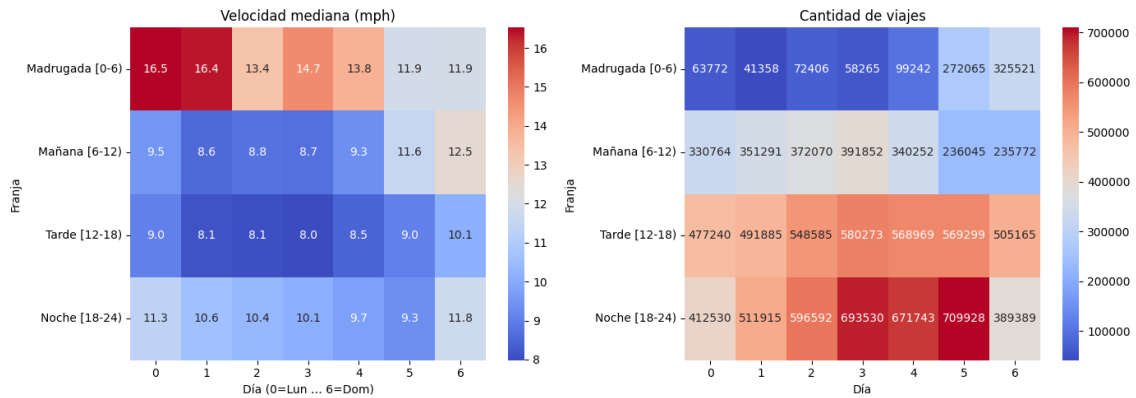


Figura 2: Velocidad mediana y cantidad de viajes por franja horaria: evidencia de horas pico.

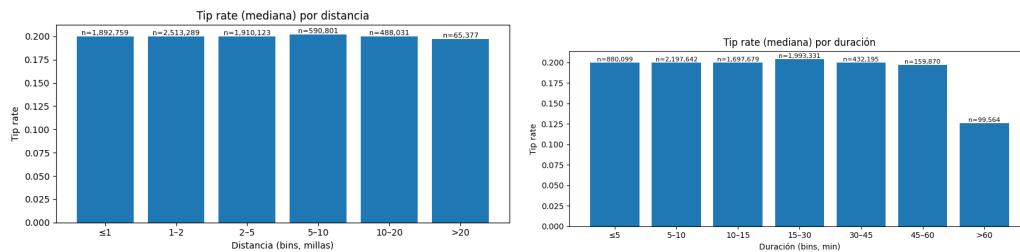


Figura 3: Propina relativa (`tip_rate`) según distancia y duración del viaje.

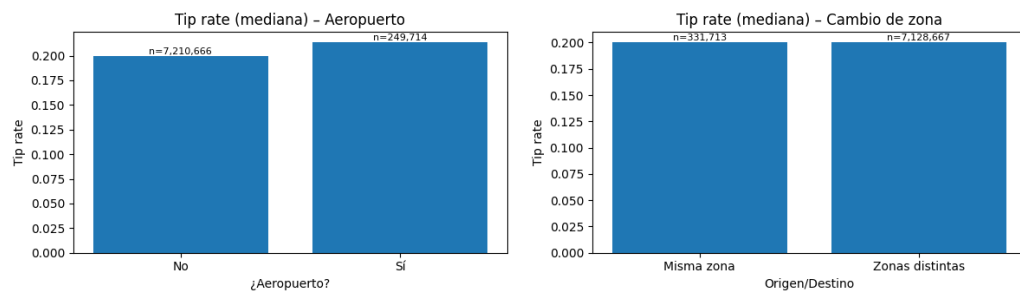


Figura 4: Propina relativa (`tip_rate`) según viajes hacia aeropuertos y entre zonas distintas.

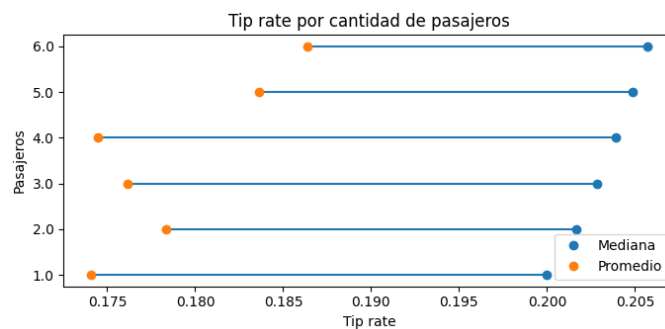


Figura 5: Propina relativa (`tip_rate`) en función de la cantidad de pasajeros.

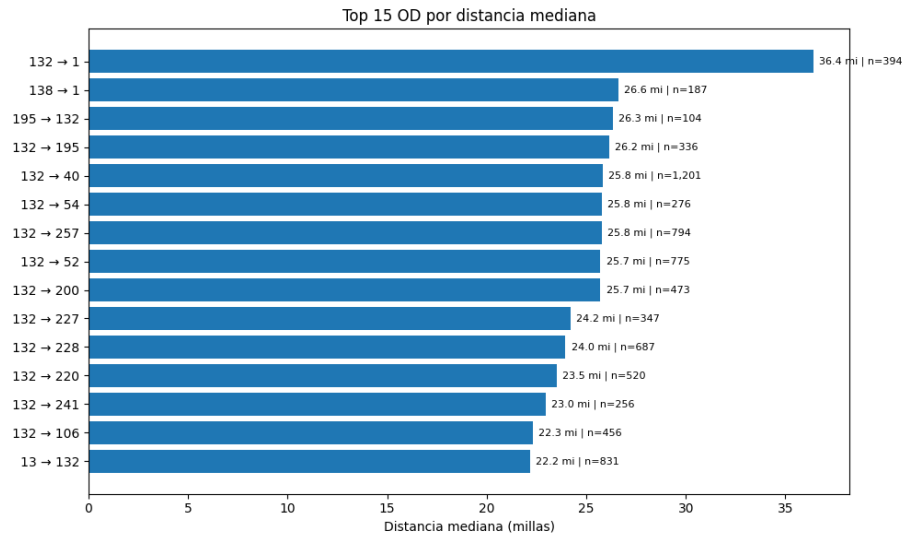


Figura 6: Zonas de origen y destino con mayor distancia mediana y proporción de viajes largos.

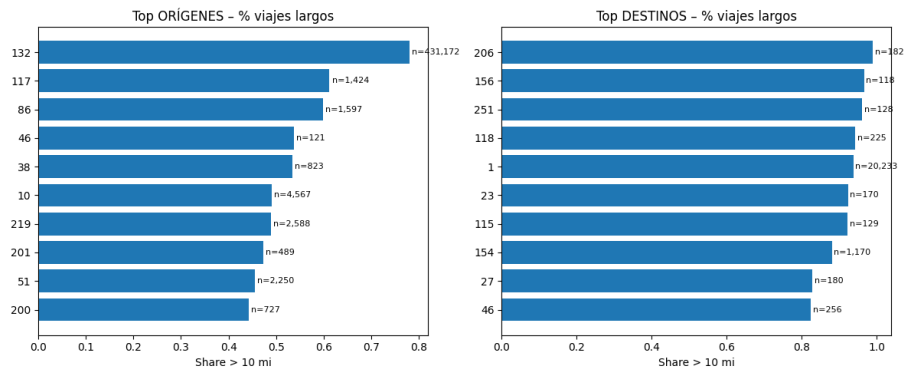


Figura 7: Zonas de origen con mayor distancia mediana y proporción de viajes largos.

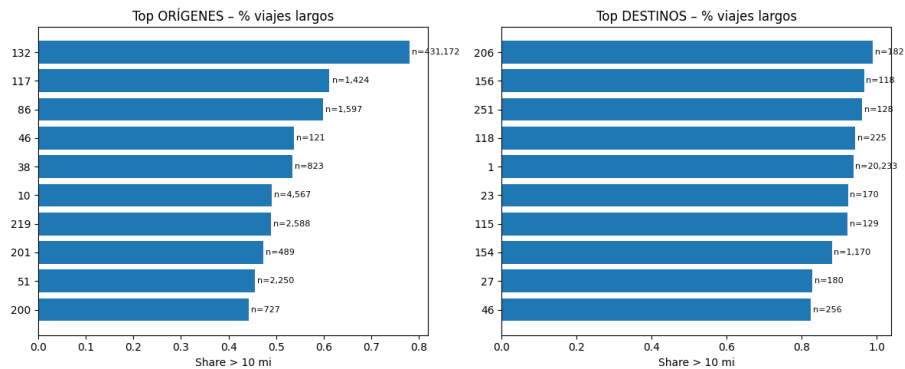


Figura 8: Zonas de destino con mayor distancia mediana y proporción de viajes largos.

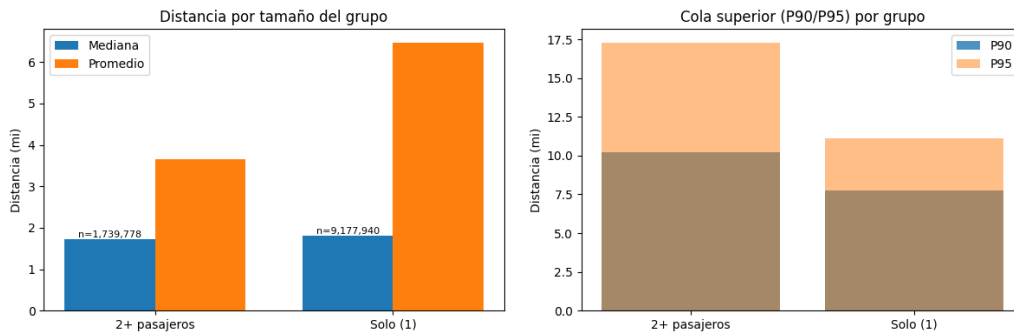


Figura 9: Distancia recorrida según tamaño del grupo: comparación entre viajes individuales y grupales. Los viajes más largos (el 5–10 % superior) se concentran en los grupos de 2 o más pasajeros, que alcanzan distancias de hasta 17 millas, mientras que los viajes individuales rara vez superan las 12 millas. Esto sugiere que los viajes compartidos o grupales son más comunes para trayectos largos (por ejemplo, traslados al aeropuerto o entre barrios distantes).

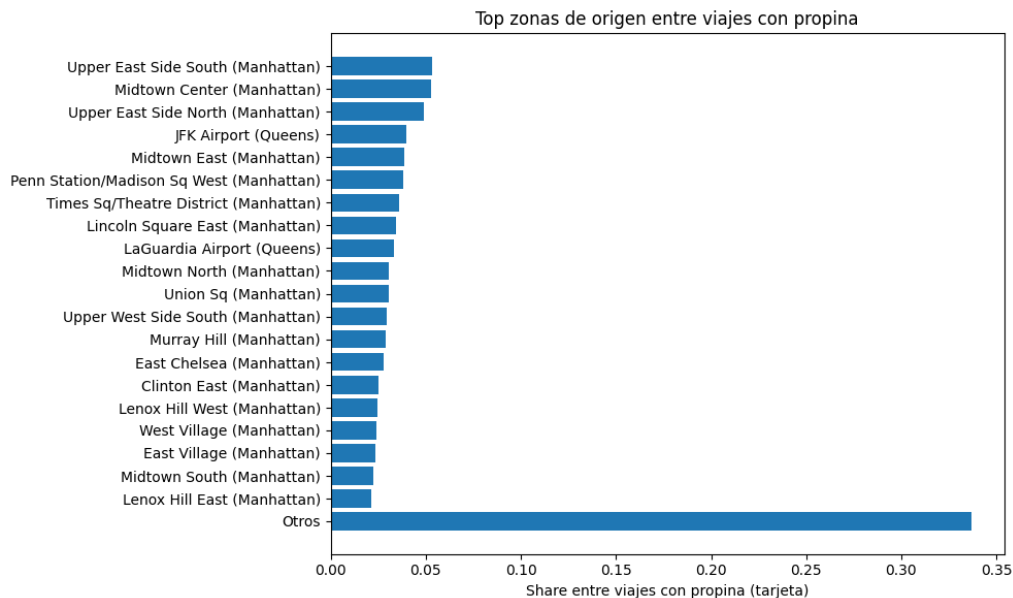


Figura 10: Proporción de viajes con propina según zona de origen.

2. Modelos de Clasificación Binaria

2.1. Descripción del dataset

Contiene registros diarios de variables meteorológicas recopiladas durante aproximadamente 10 años en múltiples estaciones distribuidas por toda Australia.

Este dataset es especialmente útil para entrenar modelos predictivos que anticipen eventos de lluvia, y permite explorar relaciones entre múltiples variables meteorológicas.

No trabajamos sobre la totalidad del dataset, las ciudades que nos fueron asignadas para entrenar los modelos fueron Nueva Gales del Sur, Queensland y Territorio de la Capital.

A continuación describimos más a detalle el contenido del dataset.

■ Cantidad de registros y columnas

El dataset original contiene 145460 registros y 23 columnas.

Nuestro dataset de trabajo, una vez filtradas las observaciones realizadas en las regiones de interés, quedó con una cantidad de 67446 registros, menos de la mitad que en el original.

■ Variables más relevantes

Breve descripción de cada una de las variables:

Información general	
Date	Fecha de la observación (YYYY-MM-DD).
Location	Nombre de la región para la estación meteorológica (e.g. Sydney, Melbourne, etc).
Temperatura	
MinTemp	Temperatura mínima medida en grados Celsius (°C).
MaxTemp	Temperatura máxima medida en grados Celsius (°C).
Temp9am	Temperatura registrada a las 9 a.m. (°C).
Temp3pm	Temperatura registrada a las 3 p.m. (°C).
Precipitación	
Rainfall	Cantidad de precipitación registrada en el día (mm).
RainToday	Valor binario: 1 si la precipitación registrada en las 24 horas previas a las 9 a.m. supera 1 mm, 0 en caso contrario.
RainTomorrow	Indicador binario que señala si se espera precipitación superior a 1 mm en el día siguiente. Es la variable objetivo.
Evaporación y sol	
Evaporation	Evaporación medida con el método 'Class A pan' (mm), correspondiente a las 24 horas previas a las 9 a.m.
Sunshine	Número de horas de sol brillante registradas durante el día.
Viento	
WindGustDir	Dirección de la ráfaga de viento más fuerte registrada en las 24 horas previas a la medianoche.
WindGustSpeed	Velocidad (km/h) de la ráfaga de viento más fuerte registrada.
WindDir9am	Dirección del viento registrada a las 9 a.m.
WindDir3pm	Dirección del viento registrada a las 3 p.m.
WindSpeed9am	Velocidad del viento a las 9 a.m. (km/h), promedio de los 10 minutos previos.
WindSpeed3pm	Velocidad del viento a las 3 p.m. (km/h).
Humedad	
Humidity9am	Humedad registrada a las 9 a.m. (%).
Humidity3pm	Humedad registrada a las 3 p.m. (%).
Presión atmosférica	
Pressure9am	Presión atmosférica registrada a las 9 a.m. (hPa), ajustada al nivel medio del mar.
Pressure3pm	Presión atmosférica registrada a las 3 p.m. (hPa), ajustada al nivel medio del mar.
Nubosidad	
Cloud9am	Fracción del cielo cubierta por nubes a las 9 a.m., medida en oktas (de 0 a 8).
Cloud3pm	Fracción del cielo cubierta por nubes a las 3 p.m., medida en oktas.

Cuadro 1: Descripción de las variables del conjunto de datos meteorológico

Se puede clasificar a cada una de estas variables como cuantitativa o cualitativa según sus características.

Tipo de variable	Variables
Cuantitativas continuas	MinTemp, MaxTemp, Rainfall, Evaporation, Sunshine, WindGustSpeed, WindSpeed9am, WindSpeed3pm, Pressure9am, Pressure3pm, Temp9am, Temp3pm
Cuantitativas discretas	Humidity9am, Humidity3pm, Cloud9am, Cloud3pm
Cualitativas nominales	Date, Location, WindGustDir, WindDir9am, WindDir3pm, RainToday, RainTomorrow

Cuadro 2: Clasificación de variables del conjunto de datos meteorológico

Consideramos que todas las variables descriptas eran importantes y aportaban valor a nuestro análisis, es por ello que no descartamos ninguna de las columnas inicialmente.

Lo único que decidimos eliminar previo a realizar un estudio sobre los datos, fueron los registros con valor nulo para el campo RainTomorrow, que es nuestra variable target. Tomamos esta decisión basándonos en que eran muy pocos registros en los que ocurría esto (menos del 3 %), y además, al tener un valor indefinido no iban a ayudar a nuestro modelo a aprender.

■ Transformaciones realizadas

- **Encoding por ubicación:** Para representar la información geográfica del dataset, se evitó aplicar directamente un encoding sobre la variable **Location**. En su lugar, se creó una nueva variable denominada **City**, que agrupa las observaciones según la ciudad correspondiente. Esta clasificación permitió aplicar un esquema de codificación *One-Hot Encoding* sobre **City**, tras lo cual se eliminó la columna original **Location** del conjunto de datos.

Esta estrategia se adoptó con el objetivo de reducir la dimensionalidad del problema, ya que el número de ciudades era considerablemente menor al de regiones, lo que permitió simplificar el análisis sin perder representatividad espacial.

- **Encoding para direcciones del viento:** Las variables que registran la dirección del viento (**WindGustDir**, **WindDir9am**, **WindDir3pm**) contenían una gran cantidad de categorías específicas (e.g. NNE, WSW, SSE). Para reducir esta complejidad, se reemplazaron todas las direcciones por su orientación cardinal principal: N, S, E o W. Esta simplificación permitió agrupar las observaciones según los puntos cardinales dominantes, facilitando el posterior *One-Hot Encoding* y mejorando la interpretabilidad del modelo.

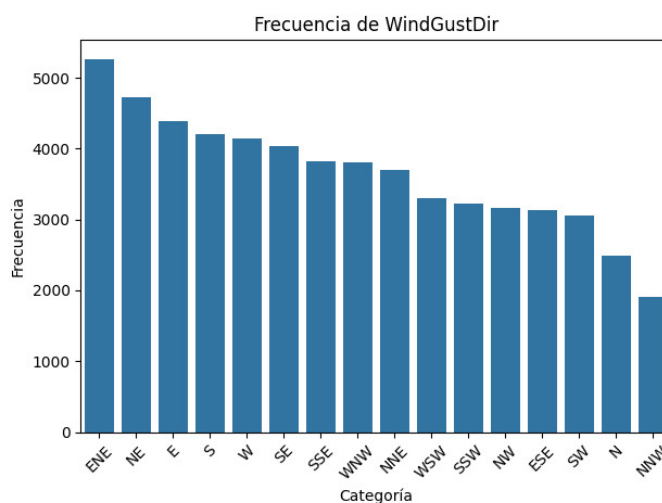


Figura 11: Agrupamiento de direcciones de viento por orientación cardinal

- **Conversión de variables binarias en texto:** Las variables `RainToday` y `RainTomorrow` estaban originalmente codificadas como texto, con valores "Yes" y "No". Para facilitar su tratamiento en modelos de aprendizaje automático, se transformaron a formato numérico, asignando 1 a "Yes" y 0 a "No".
- **Imputación de variables continuas con alta proporción de nulos:** Las variables `Sunshine` y `Evaporation` presentaban entre un 40 % y 50 % de valores faltantes. Para imputarlas, se aplicó un esquema basado en *K-Nearest Neighbors* utilizando como referencia aquellas variables con mayor correlación. Se consideró como correlación fuerte aquella con valor absoluto igual o superior al 40 %. Esta estrategia permitió completar la mayoría de los valores nulos; los casos restantes se imputaron utilizando el valor de la moda. Las imputaciones realizadas no alteraron significativamente la distribución original de las variables.

Antes de imputar:

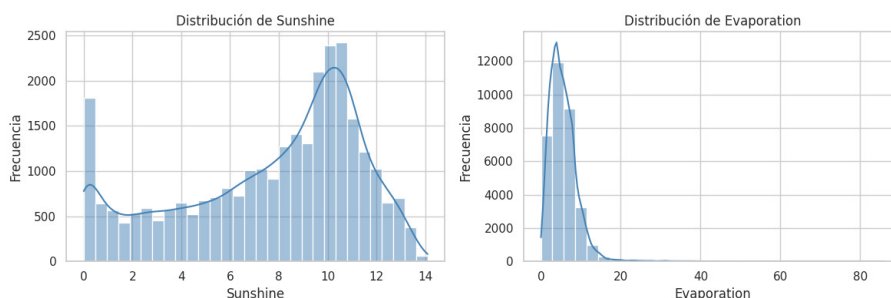


Figura 12: Distribución original de las variables `Sunshine` y `Evaporation`

Luego de imputar:

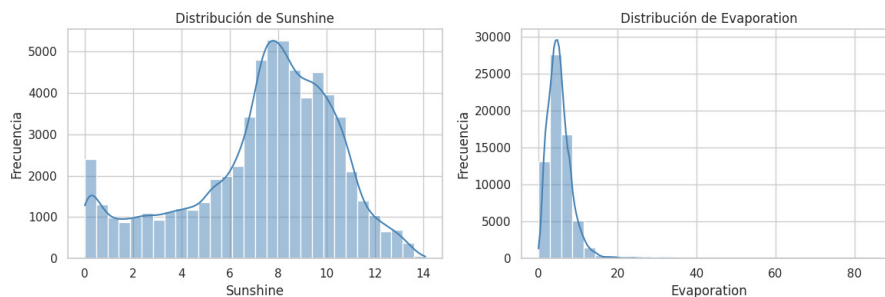


Figura 13: Distribución de las variables imputadas

- **Imputación de variables discretas con alta proporción de nulos:** Las variables `Cloud3pm` y `Cloud9am` presentaban una cantidad significativa de valores faltantes. Para imputarlas, se entrenó un modelo de *Random Forest* utilizando como predictores aquellas variables con mayor correlación. Esta estrategia permitió completar la mayoría de los valores nulos. Los casos restantes, que no pudieron ser imputados por el modelo, se completaron utilizando el valor de la moda. Esta combinación permitió preservar la estructura discreta de las variables sin introducir distorsiones significativas. Consideramos como proporciones grandes de valores faltantes y coeficientes importantes de correlación a los mismos rangos de valores descriptos en la imputación para variables continuas realizada anteriormente.

Antes de imputar:

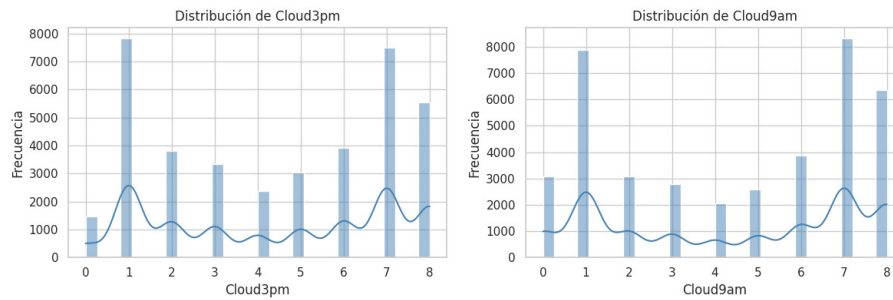


Figura 14: Distribución original de las variables Cloud

Luego de imputar:

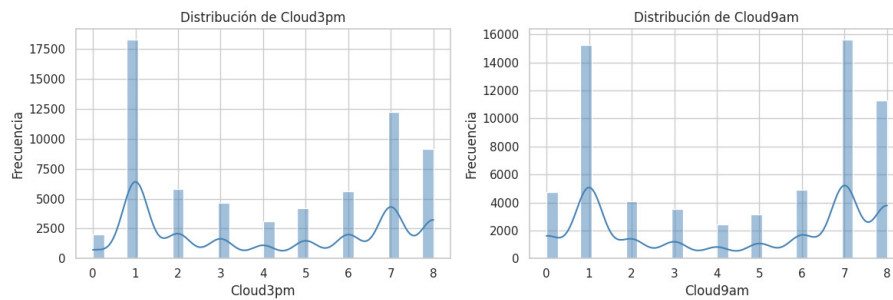


Figura 15: Distribución de las variables imputadas

Finalmente, se imputaron todas las variables cuantitativas restantes utilizando el valor de la mediana. Esta estrategia permitió completar los valores faltantes de forma robusta, preservando la forma general de las distribuciones originales y evitando distorsiones significativas en el análisis posterior.

■ Desproporción de la clase target

Se observó en el dataset una marcada desproporción en la variable objetivo **RainTomorrow**, con una mayoría significativa de casos negativos frente a los positivos. Este desbalance puede afectar directamente el rendimiento de los modelos de clasificación, ya que tienden a favorecer la clase mayoritaria, logrando métricas aparentemente altas pero con baja capacidad para detectar correctamente los eventos menos frecuentes (en este caso, los días con lluvia). Esto puede resultar problemático si el objetivo es anticipar correctamente los días con lluvia, seguro tengamos que, de alguna forma mejorar la representatividad de la clase minoritaria durante el entrenamiento.

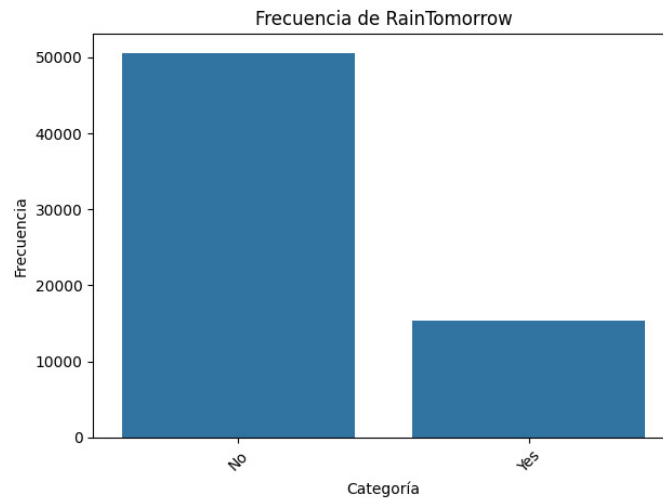


Figura 16: Distribución de la variable objetivo

2.2. Modelos entrenados

En esta parte describimos el proceso de búsqueda de hiperparámetros realizado para cada modelo, junto con los resultados obtenidos en los conjuntos de TRAIN y TEST utilizando los parámetros óptimos seleccionados. Las conclusiones sobre la performance de cada modelo y los valores de los hiperparámetros elegidos se presentan en la sección final del informe.

2.2.1. Árbol de decisión

■ Hiperparámetros optimizados y técnica de validación utilizada

Se utilizó `RandomizedSearchCV` para explorar un conjunto amplio de hiperparámetros del modelo `DecisionTreeClassifier`, incluyendo:

- `criterion`: función de impureza para la división de nodos (`gini` o `entropy`).
- `min_samples_leaf`: mínimo de muestras por hoja, entre 20 y 79.
- `min_samples_split`: mínimo de muestras para dividir un nodo, entre 20 y 79.
- `ccp_alpha`: parámetro de poda cost-complexity, evaluado entre 0 y 0.005.
- `max_depth`: profundidad máxima del árbol, entre 5 y 79.

La validación se realizó mediante **Stratified K-Fold Cross Validation** con $k = 5$ particiones, preservando la proporción original de clases en cada fold. Esta técnica es especialmente adecuada dado el desbalance en la variable objetivo `RainTomorrow`, donde los días sin lluvia son significativamente más frecuentes.

■ Métrica elegida

La métrica seleccionada para guiar la búsqueda de hiperparámetros fue el **F1-score**, calculado sobre la clase positiva (`RainTomorrow`). Esta elección responde al desbalance de clases, ya que el F1-score penaliza tanto los falsos positivos como los falsos negativos, logrando un equilibrio entre precisión y cobertura. Se descartó el uso de `recall` como métrica principal, dado que en pruebas preliminares se observó una disminución del rendimiento global del modelo.

■ Imagen del árbol

En esta sección se presenta el árbol de decisión representativo obtenido a partir del modelo entrenado, junto con la descripción e interpretación de las reglas que lo conforman. Este árbol

permite comprender de manera visual y lógica cómo el modelo realiza las clasificaciones a partir de las variables meteorológicas más relevantes.

La característica **Sunshine** (horas de sol) se posiciona como el principal criterio de división, indicando que la cantidad de insolación diaria constituye el factor más influyente para distinguir entre las dos clases. En el **nodo raíz**, el árbol divide los datos según el umbral $\text{Sunshine} \leq 6.23$. Los registros con menor cantidad de horas de sol se dirigen hacia la rama izquierda, mientras que aquellos con valores superiores se asignan a la rama derecha. Esta primera separación permite diferenciar contextos de baja y alta exposición solar.

En la **rama izquierda** ($\text{Sunshine} \leq 6.23$), las observaciones se caracterizan por condiciones de menor insolación. La siguiente división relevante corresponde a la variable **Humidity3pm**, con un umbral de 67.5. Cuando la humedad a las 15:00 hs es menor o igual a dicho valor, el modelo analiza nuevamente la insolación ($\text{Sunshine} \leq 5.31$), refinando la clasificación de días poco soleados. En cambio, cuando la humedad supera ese nivel, el modelo predice de manera consistente la **clase 1**, asociada a condiciones de alta humedad y baja radiación solar. En general, esta rama concentra la mayoría de los casos correspondientes a la clase 1, lo que sugiere una fuerte relación entre **alta humedad** y la ocurrencia del evento modelado.

Por otro lado, en la **rama derecha** ($\text{Sunshine} > 6.23$) se agrupan los días con mayor exposición solar. La variable **Cloud3pm** se convierte en el segundo criterio más relevante, dividiendo los casos según la cantidad de nubosidad a las 15:00 h ($\text{Cloud3pm} \leq 2.5$). Cuando el nivel de nubosidad es bajo y la **humedad a las 15:00 h** también es reducida ($\text{Humidity3pm} \leq 50.5$), el árbol tiende a predecir la **clase 0**, asociada a condiciones más secas y despejadas. En cambio, cuando la nubosidad o la presión atmosférica (**Pressure3pm**) son elevadas, el modelo mantiene una ligera mezcla entre ambas clases, aunque la clase 0 continúa siendo predominante.

En síntesis, el árbol representativo muestra que las **condiciones de baja insolación y alta humedad** incrementan la probabilidad del evento (clase 1), mientras que los **días con mayor radiación solar, menor nubosidad y baja humedad** se asocian con la clase 0. De esta manera, el árbol reducido proporciona una representación clara e interpretable del proceso de decisión del modelo, destacando la importancia de las variables **Sunshine**, **Humidity3pm**, **Cloud3pm** y **Pressure3pm** en la clasificación.

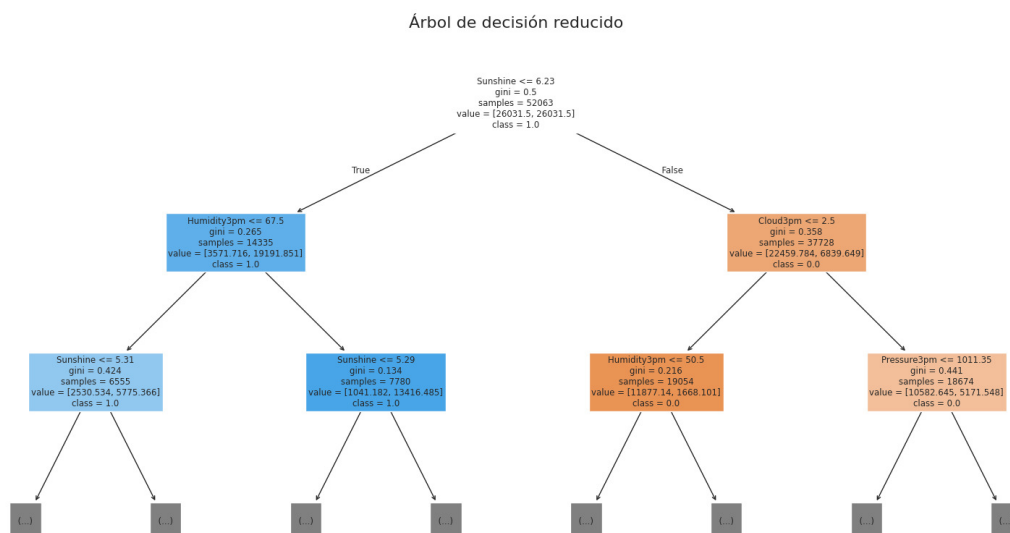


Figura 17: Sub-árbol de dos niveles

■ Evaluación en train/test

Class / Metric	Precision	Recall	F1-score	Support
TRAIN				
NoRainTomorrow	0.936	0.859	0.896	40078
RainTomorrow	0.630	0.803	0.706	11985
Accuracy	0.846		(on 52063 samples)	
Macro avg	0.783	0.831	0.801	52063
Weighted avg	0.865	0.846	0.852	52063
TEST				
NoRainTomorrow	0.932	0.853	0.891	10020
RainTomorrow	0.617	0.792	0.693	2996
Accuracy	0.839		(on 13016 samples)	
Macro avg	0.774	0.822	0.792	13016
Weighted avg	0.859	0.839	0.845	13016

Cuadro 3: Métricas Árbol de Decisión TRAIN - TEST

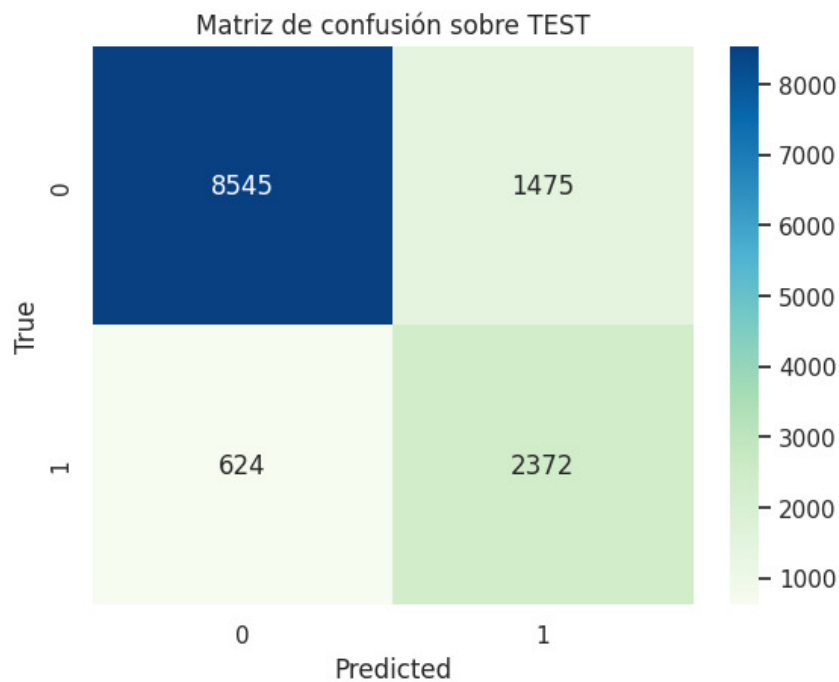


Figura 18: Árbol de decisión: Matriz de confusión en TEST

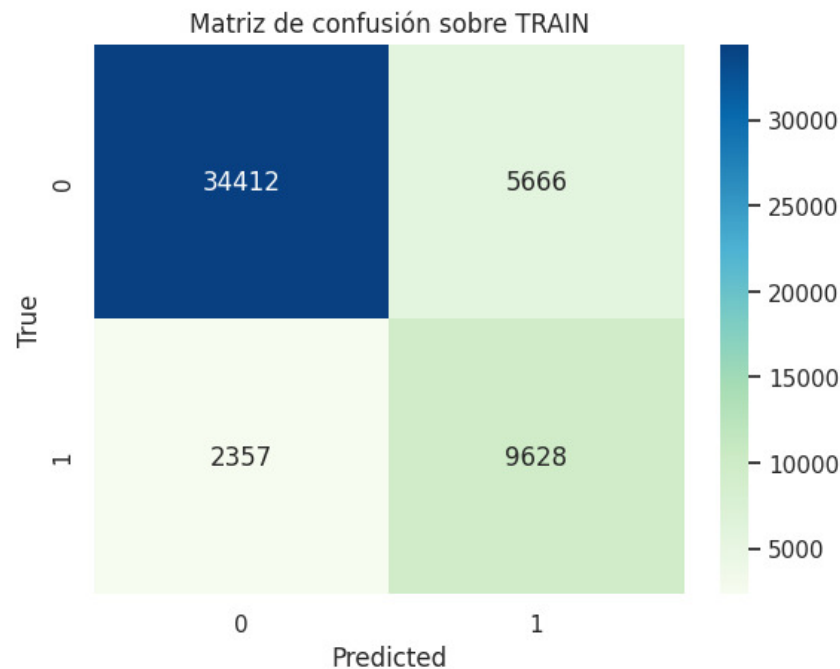


Figura 19: Árbol de decisión: Matriz de confusión en TRAIN

2.2.2. Random Forest

■ Hiperparámetros optimizados y técnica de validación usada

Para el modelo `RandomForestClassifier`, se realizó una búsqueda exhaustiva de hiperparámetros mediante `GridSearchCV`, evaluando múltiples combinaciones sobre un espacio acotado y controlado:

- **criterion**: se utilizó `gini` como medida de impureza.
- **min_samples_split** y **min_samples_leaf**: se exploraron valores altos (70–80 y 60–70 respectivamente) para limitar el crecimiento del árbol y reducir el overfitting.
- **max_depth**: se restringió a valores bajos (5–7), mucho menores que los del árbol individual, para controlar la complejidad del modelo.
- **n_estimators**: se fijó en 100 para acelerar el entrenamiento sin comprometer la estabilidad.
- **max_features**: se evaluaron proporciones parciales (0.3) y raíz cuadrada ("`sqrt`") del total de variables.
- **class_weight**: se utilizó "`balanced_subsample`" para ajustar el peso de las clases en cada árbol base.

La validación se realizó mediante **K-Fold Cross Validation estratificado** con $k = 5$ particiones, preservando la proporción de clases en cada fold. Esta técnica garantiza una evaluación robusta y representativa, especialmente relevante en contextos de desbalance como el presente.

■ Importancia de los atributos

Para comprender cómo el modelo Random Forest toma decisiones, se evaluó la *importancia de cada atributo*. Esta medida indica qué tan relevante es cada variable para reducir la impureza de los nodos en los árboles del bosque. A mayor valor, mayor influencia del atributo en la predicción.

Se consideraron los atributos cuya importancia supera el 1 %. Los resultados muestran que el modelo depende principalmente de variables meteorológicas directas:

- **Sunshine** (0.452): el atributo más influyente, indicando que la cantidad de sol es determinante en la predicción.
- **Humidity3pm** (0.180) y **Cloud3pm** (0.146): variables de la tarde que afectan significativamente el resultado.
- **Rainfall** (0.040), **Cloud9am** (0.040) y **RainToday** (0.034): contribuyen de manera moderada.
- **Humidity9am** (0.026) y **WindGustSpeed** (0.022): su efecto es menor pero aún relevante.
- **Pressure3pm** (0.014) y **Pressure9am** (0.012): factores de baja relevancia relativa.

El análisis muestra que el modelo se centra principalmente en variables meteorológicas relacionadas con sol, nubosidad y humedad, especialmente por la tarde. Las variables de fecha, ubicación, viento y temperatura tienen un impacto mucho menor (menos del 1 %), indicando que la predicción depende principalmente de las condiciones climáticas del día.

■ Métrica elegida

Se consideraron múltiples métricas durante la búsqueda: **f1**, **precision_1** y **recall_1**, todas calculadas sobre la clase positiva (**RainTomorrow**). Sin embargo, se definió **f1** como métrica principal para el ajuste (**refit="f1"**), dado que ofrece un balance entre precisión y recall, penalizando tanto los falsos positivos como los falsos negativos. Esta elección es especialmente adecuada en escenarios con clases desbalanceadas, donde optimizar una sola métrica puede generar sesgos.

■ Tratamiento del desbalance en el conjunto de entrenamiento

Dado que la variable objetivo presenta una proporción significativamente mayor de días sin lluvia (**RainTomorrow = 0**), se aplicó **RandomOverSampler** con una estrategia de muestreo del 30 % para la clase minoritaria. Esta técnica permitió aumentar la representatividad de los días con lluvia en el conjunto de entrenamiento, mejorando la capacidad del modelo para detectar eventos positivos sin comprometer la generalización. El oversampling fue aplicado antes del ajuste de hiperparámetros, asegurando que el modelo se entrenara sobre una distribución más equilibrada.

■ Árbol representativo y explicación de reglas

En esta sección se presenta un árbol de decisión representativo perteneciente al modelo **Random Forest** entrenado, junto con la descripción e interpretación de las reglas que lo componen. Dado que el Random Forest está constituido por un conjunto de múltiples árboles generados sobre distintas muestras del conjunto de datos, se seleccionó el **primer árbol del bosque** como ejemplo ilustrativo del funcionamiento interno del modelo.

En este árbol, la variable **Humidity3pm** (humedad a las 15:00 h) aparece como el **criterio principal de división** en el nodo raíz, con un umbral de **65.5**. Esta característica indica que la humedad vespertina tiene una influencia determinante en la clasificación: los valores inferiores o iguales a 65.5 se asocian con condiciones más secas, mientras que los valores superiores tienden a representar escenarios más húmedos.

En la **rama izquierda** ($\text{Humidity3pm} \leq 65.5$), el árbol continúa segmentando los datos en función de la variable **Sunshine** (horas de sol), con un umbral de **6.41**. Los días con menor insolación tienden a agruparse hacia la izquierda, donde la variable **Humidity9am** (humedad matinal) refina la decisión final. Por el contrario, cuando las horas de sol superan ese valor, la variable **WindGustSpeed** (velocidad de ráfaga de viento) se vuelve relevante, separando condiciones de mayor y menor viento. En esta rama, predominan los registros clasificados como **clase 0**, correspondientes a contextos secos y despejados.

En la **rama derecha** ($\text{Humidity3pm} > 65.5$), el modelo analiza primero la variable **Cloud3pm** (nubosidad a las 15:00 h), con un umbral de 5.5. Los valores bajos de nubosidad se asocian con mayor probabilidad de **clase 1**, mientras que en los casos de nubosidad elevada se evalúan variables complementarias como **Pressure3pm** (presión atmosférica a las 15:00 h) y nuevamente **Humidity3pm**, que contribuyen a afinar la predicción final. Esta rama agrupa la mayoría de los registros pertenecientes a la **clase 1**, caracterizados por alta humedad y mayor cobertura nubosa.

En síntesis, el árbol representativo del Random Forest sugiere que los **niveles altos de humedad y nubosidad** son los principales indicadores del evento modelado (clase 1), mientras que las condiciones **más secas, con mayor radiación solar y menor humedad** tienden hacia la clase 0. Aunque cada árbol del bosque difiere ligeramente en sus divisiones, el patrón general resalta la importancia de las variables **Humidity3pm**, **Cloud3pm**, **Sunshine** y **WindGustSpeed** en las decisiones del modelo.

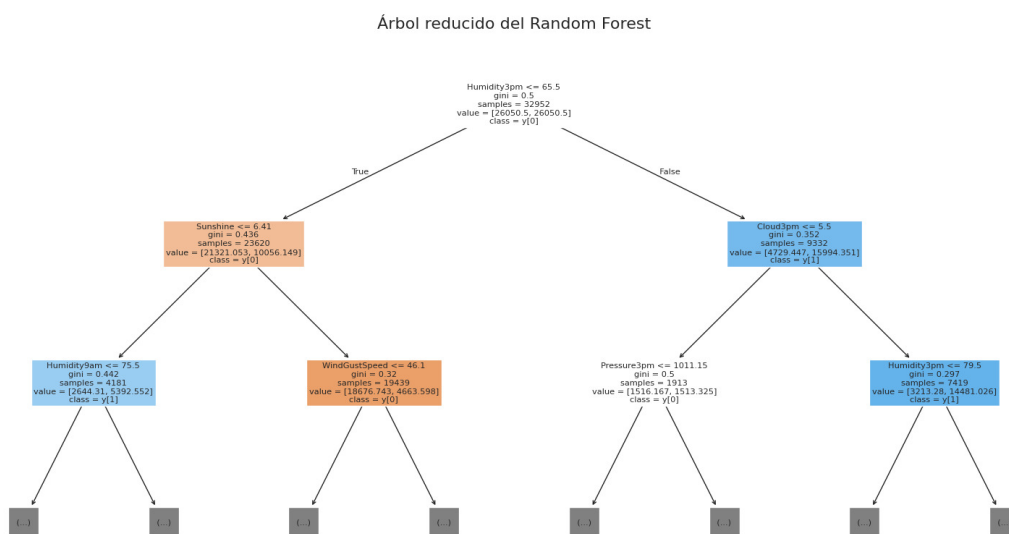


Figura 20: Sub-árbol de dos niveles del primer árbol del bosque

■ Evaluación en train/test

Class / Metric	Precision	Recall	F1-score	Support
TRAIN				
NoRainTomorrow	0.931	0.874	0.902	40078
RainTomorrow	0.651	0.786	0.712	12023
Accuracy	0.854		(on 52101 samples)	
Macro avg	0.791	0.830	0.807	52101
Weighted avg	0.867	0.854	0.858	52101
TEST				
NoRainTomorrow	0.929	0.870	0.899	10020
RainTomorrow	0.642	0.778	0.703	2996
Accuracy	0.849		(on 13016 samples)	
Macro avg	0.785	0.824	0.801	13016
Weighted avg	0.863	0.849	0.854	13016

Cuadro 4: Métricas Random Forest TRAIN - TEST

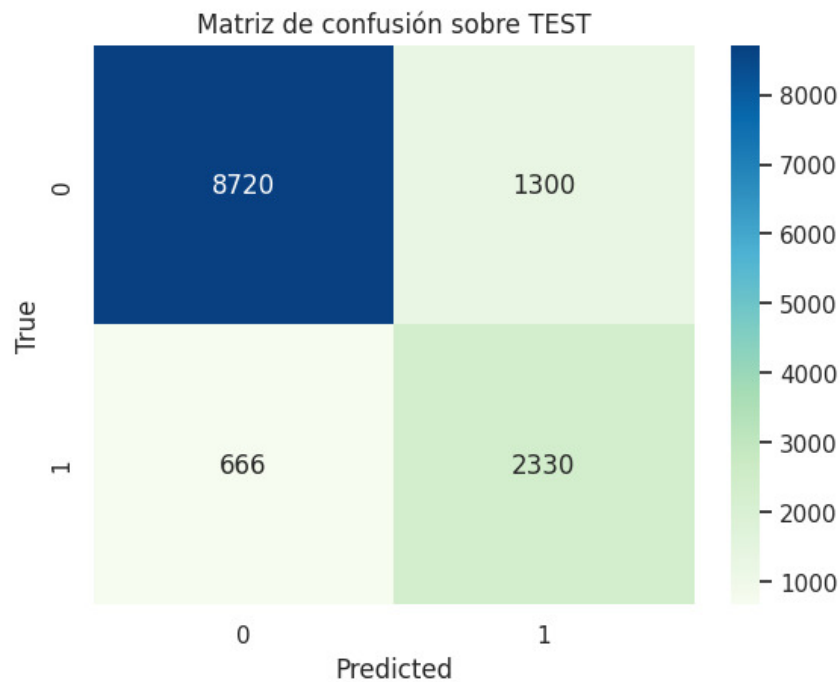


Figura 21: Random Forest: Matriz de confusión en TEST

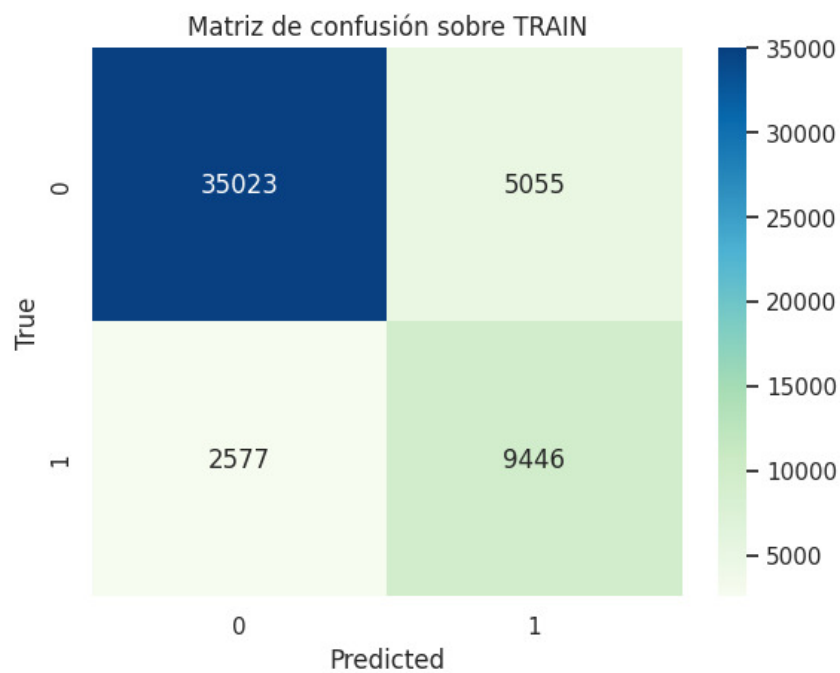


Figura 22: Random Forest: Matriz de confusión en TRAIN

2.2.3. XGBoost (modelo a elección)

■ Justificación de la elección del modelo

La elección de **XGBoost** como modelo para este problema se basó tanto en criterios técnicos como en una motivación del grupo por explorar un enfoque diferente a lo que veníamos

tratando en los modelos anteriores. Mientras que **Random Forest** representa una estrategia de **bagging**, basada en la agregación de múltiples árboles entrenados sobre subconjuntos aleatorios del conjunto de datos, **XGBoost** implementa una estrategia de **boosting**, donde los árboles se entrenan de forma secuencial corrigiendo los errores del modelo anterior.

Esta diferencia estructural permite que XGBoost se enfoque en ejemplos difíciles de clasificar, lo cual resulta muy valioso en contextos con clases desbalanceadas como con el que estamos tratando. Además, en comparación con modelos de bagging, XGBoost incorpora mecanismos avanzados de regularización, control de complejidad y manejo eficiente de atributos, lo que lo convierte en una opción robusta.

Desde una perspectiva exploratoria, se buscó comparar directamente el rendimiento de un modelo basado en bagging con uno basado en boosting, evaluando cómo cada enfoque responde ante el mismo conjunto de datos.

■ Hiperparámetros optimizados y técnica de validación usada

Para el modelo **XGBoostClassifier**, se realizó una búsqueda exhaustiva de hiperparámetros mediante **GridSearchCV**, evaluando combinaciones sobre un espacio cuidadosamente definido:

- **n_estimators**: número de árboles en el ensamble, evaluado en 100 y 200 para compensar learning rates más bajos.
- **max_depth**: profundidad máxima de cada árbol, restringida a 3 y 4 para evitar overfitting.
- **learning_rate**: tasa de aprendizaje, explorada en valores moderados (0.05 y 0.1) para evitar actualizaciones extremas.
- **subsample**: proporción de muestras utilizadas por árbol, evaluada en 0.8 y 0.9 para introducir regularización por muestreo.
- **colsample_bytree**: proporción de atributos considerados por árbol, también en 0.8 y 0.9.
- **reg_alpha**: regularización L1, probada en 0, 0.1 y 0.5 para fomentar sparsity (coeficientes cercanos a cero). No controlar el sparsity puede llevar a que algunas características tengan menor impacto o incluso se ignoren.
- **reg_lambda**: regularización L2, evaluada en 1, 5 y 10 para controlar la magnitud de los pesos. Ayuda a suavizar el modelo, evitando que los árboles se ajusten demasiado a los datos de entrenamiento.

La validación se realizó mediante **Stratified K-Fold Cross Validation** con $k = 5$ particiones, manteniendo la proporción de clases en cada fold. A diferencia de otros modelos, no se aplicaron técnicas de oversampling ni ajustes adicionales en los conjuntos de entrenamiento, ya que XGBoost maneja razonablemente bien el desbalance de clases mediante su estructura de boosting.

■ Métrica elegida

Durante la búsqueda de hiperparámetros se consideraron múltiples métricas: **f1**, **precision_1** y **recall_1**, todas calculadas sobre la clase positiva (**RainTomorrow**). Se definió **f1** como métrica principal para el ajuste (**refit="f1"**), ya que proporciona un balance entre precisión y recall, penalizando tanto los falsos positivos como los falsos negativos. Esta elección es especialmente relevante en problemas de clasificación binaria con clases desbalanceadas, como el presente caso.

■ Validación y métricas de performance

Class / Metric	Precision	Recall	F1-score	Support
TRAIN				
NoRainTomorrow	0.908	0.964	0.935	40078
RainTomorrow	0.848	0.672	0.750	11985
Accuracy	0.897		(on 52063 samples)	
Macro avg	0.878	0.818	0.842	52063
Weighted avg	0.894	0.897	0.892	52063
TEST				
NoRainTomorrow	0.901	0.952	0.926	10020
RainTomorrow	0.804	0.652	0.720	2996
Accuracy	0.883		(on 13016 samples)	
Macro avg	0.853	0.802	0.823	13016
Weighted avg	0.879	0.883	0.879	13016

Cuadro 5: Métricas XGBoost TRAIN - TEST

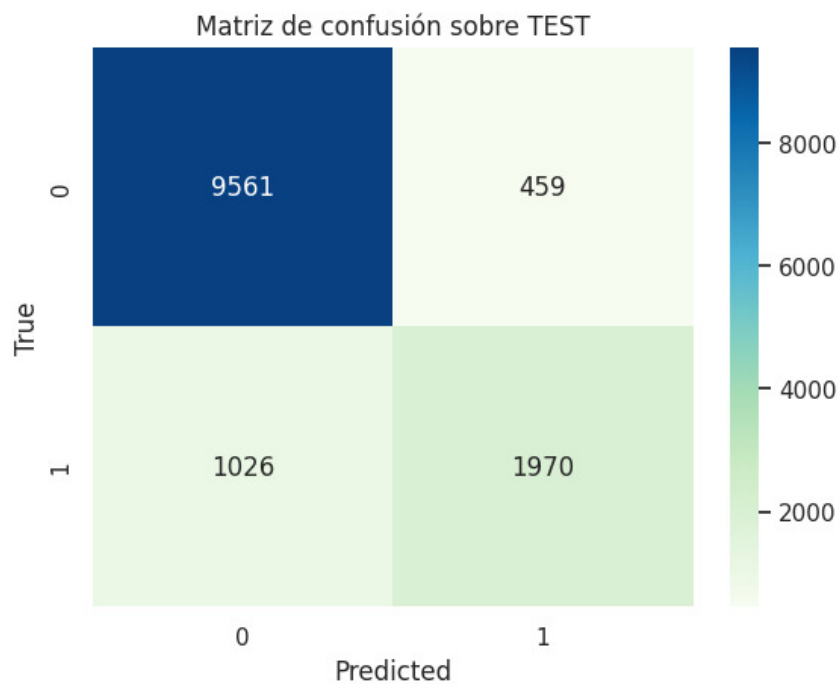


Figura 23: XGBoost: Matriz de confusión en TEST

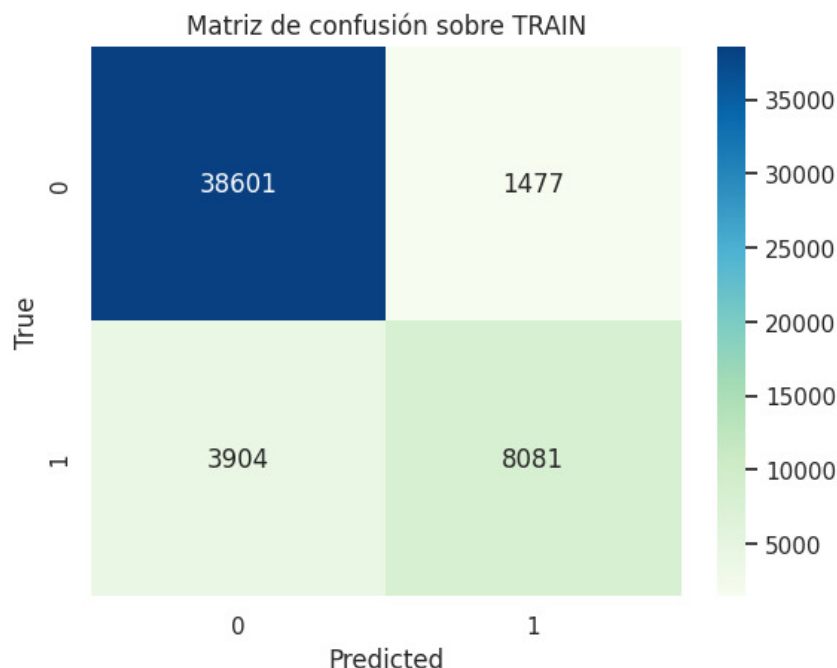


Figura 24: XGBoost: Matriz de confusión en TRAIN

2.3. Comparación de resultados

Métricas de rendimiento en TEST:

Modelo	F1	Precisión	Recall	Accuracy
Árbol de Decisión	0.693	0.617	0.792	0.839
Random Forest	0.703	0.642	0.778	0.849
XGBoost	0.720	0.804	0.652	0.883

Cuadro 6: Resultados de modelos de clasificación

Parámetros utilizados para el árbol de decisión: {min_samples_split: 72, min_samples_leaf: 65, max_depth: 55, criterion: 'gini', ccp_alpha: np.float64(0.00020408163265306123)}

Parámetros utilizados para el modelo Random Forest: {class_weight: 'balanced_subsample', criterion: 'gini', max_depth: 7, max_features: 0.3, min_samples_leaf: 70, min_samples_split: 70, n_estimators: 100}

Parámetros utilizados para el modelo XGBoost: {colsample_bytree: 0.8, learning_rate: 0.1, max_depth: 4, n_estimators: 200, reg_alpha: 0.1, reg_lambda: 5, subsample: 0.9}

Performance respecto al set de entrenamiento en cada modelo:

Performance del modelo Árbol de Decisión: El modelo mostró un rendimiento consistente entre entrenamiento y test, con diferencias mínimas en precisión (0,630 vs. 0,617), recall (0,803 vs. 0,792), F1-score (0,706 vs. 0,693) y accuracy (0,846 vs. 0,839). Estas variaciones son inferiores a 0,02, lo que indica una buena capacidad de generalización sin evidencia de overfitting.

Performance del modelo Random Forest: El modelo Random Forest mantuvo un com-

portamiento estable entre entrenamiento y test, con diferencias leves en precisión (0,651 vs. 0,642), recall (0,786 vs. 0,778), F1-score (0,712 vs. 0,703) y accuracy (0,854 vs. 0,849). Las métricas sugieren un modelo robusto, sin señales relevantes de overfitting.

Performance del modelo XGBoost: XGBoost presentó métricas más altas en entrenamiento, especialmente en precisión (0,848 vs. 0,804) y F1-score (0,750 vs. 0,720), con diferencias cercanas a 0,03. Aunque estas diferencias son moderadas, se mantienen dentro de un rango aceptable y no comprometen la generalización. El rendimiento en test sigue siendo competitivo, con buena precisión y accuracy (0,883).

2.4. Elección del modelo

En base a los resultados obtenidos ¿Qué modelo elegirían para predecir si lloverá o no el día siguiente? ¿Por qué?

Entre los tres modelos evaluados, elegiríamos **XGBoost** como el más adecuado para predecir si lloverá al día siguiente. Esta decisión se fundamenta en su **mayor precisión** (0,804), **mayor F1-score** (0,720) y **mayor accuracy** (0,883) en el conjunto de test, superando a los modelos de Árbol de Decisión y Random Forest en todas estas métricas claves.

Aunque XGBoost presenta un leve descenso de performance respecto al conjunto de entrenamiento (con diferencias de hasta 0,03), estas variaciones son moderadas y no comprometen su capacidad de generalización. En contraste, los modelos de Árbol de Decisión y Random Forest muestran métricas más bajas en test, especialmente en precisión y F1-score, lo que indica un rendimiento menos competitivo.

Además, XGBoost logra un mejor equilibrio entre precisión y recall, lo que lo convierte en una opción más confiable para un problema de clasificación binaria como este, donde es importante minimizar tanto los falsos positivos como los falsos negativos.

3. Modelos de Regresión

3.1. Descripción del dataset

El dataset utilizado corresponde a anuncios de alojamientos de Airbnb de la ciudad de México, el cual contiene información relacionada a las características de los alojamientos, su disponibilidad, métricas acerca del mismo y del propietario, y el precio de alquiler, el cual será nuestro objetivo predecir.

■ Cantidad de registros y columnas

Inicialmente, el conjunto de datos cuenta con 26.401 registros y 79 columnas, de las cuales algunas aportan información valiosa para realizar la predicción de un precio y muchas otras contienen información redundante o directamente inútil.

■ Variables más relevantes

Las variables más relevantes son aquellas que están directamente relacionadas con el alojamiento como objeto tangible y no como objeto de una publicación dentro de AirBnB. Estas son:

- El tipo de habitación, esto influye porque nos permite determinar la privacidad que tendremos como inquilinos.
- Un listado de todas las prestaciones del lugar de alojamiento.
- La posibilidad de reservar sin una aceptación previa del propietario, esto puede elevar el precio, ya que está dirigido a personas que desean realizar una transacción más rápida.

- La máxima cantidad de personas para el alojamiento.
 - La cantidad de baños disponibles dentro del alojamiento.
 - La cantidad de habitaciones con ubicaciones para descansar dentro del alojamiento.
 - La cantidad mínima de noches por la que se puede reservar el alojamiento.
 - La cantidad máxima de noches por la que se puede reservar el alojamiento.
 - El precio del alojamiento en cuestión. Nuestro objetivo.
- **Transformaciones realizadas**
- Dentro de las transformaciones que se realizaron, podemos mencionar, la eliminación de columnas poco o nada relacionadas con el alojamiento en cuestión, la quita de variables que en gran medida (más del 45 % del total) solo contienen valores nulos, la transformación en la manera de enunciar la información dentro de una columna (redondeo de valores numéricos, eliminación de símbolos, binarización de valores, etc.) y la agrupación y simplificación de las prestaciones que posee cada una de las viviendas.

3.2. Modelos entrenados

3.2.1. Regresión lineal

■ **Variables utilizadas**

Las variables utilizadas fueron las mismas que se mencionaron en el apartado de "Variables más relevantes", debido a que las consideramos importantes para predecir el precio de un alojamiento y porque su representación en el dataset nos permite utilizarla para ello.

■ **Evaluación en train/test**

Podemos visualizar cual fué el comportamiento del modelo con el conjunto de Train, en la siguiente figura.

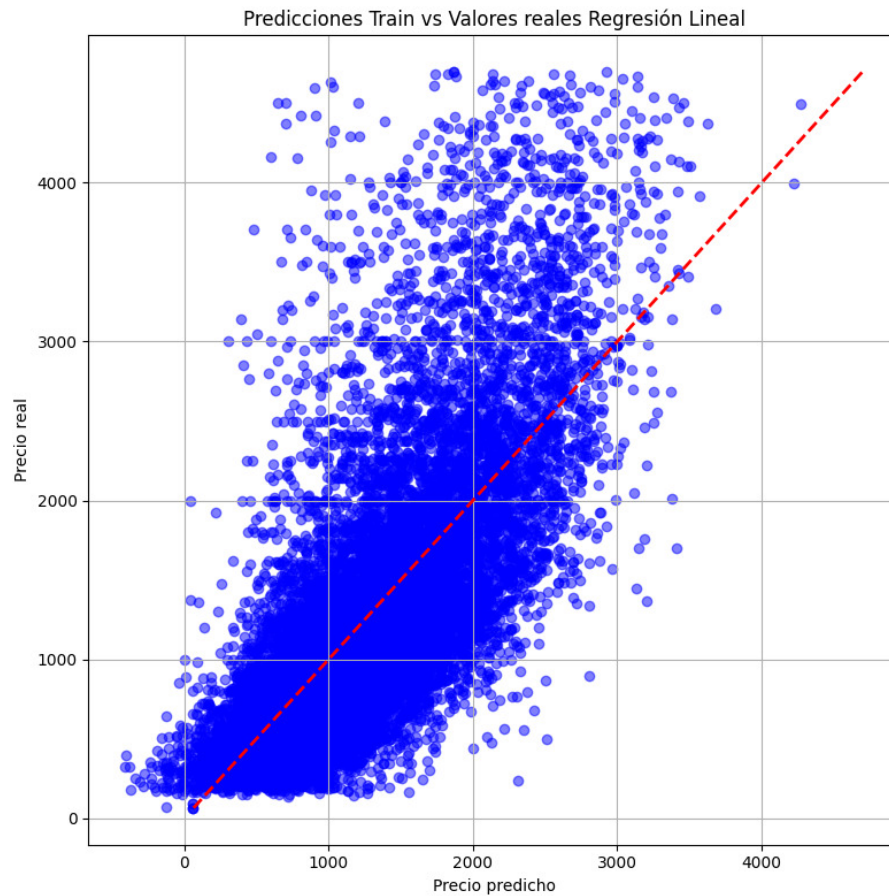


Figura 25: Predicciones Regresion Lineal Train

Posteriormente, realizamos predicciones sobre el conjunto de Test, y al igual que en el Train, predice precios inferiores a \$0, lo cual no tiene sentido en la realidad y ninguno de los precios con los que entrenó es inferior a dicho valor.

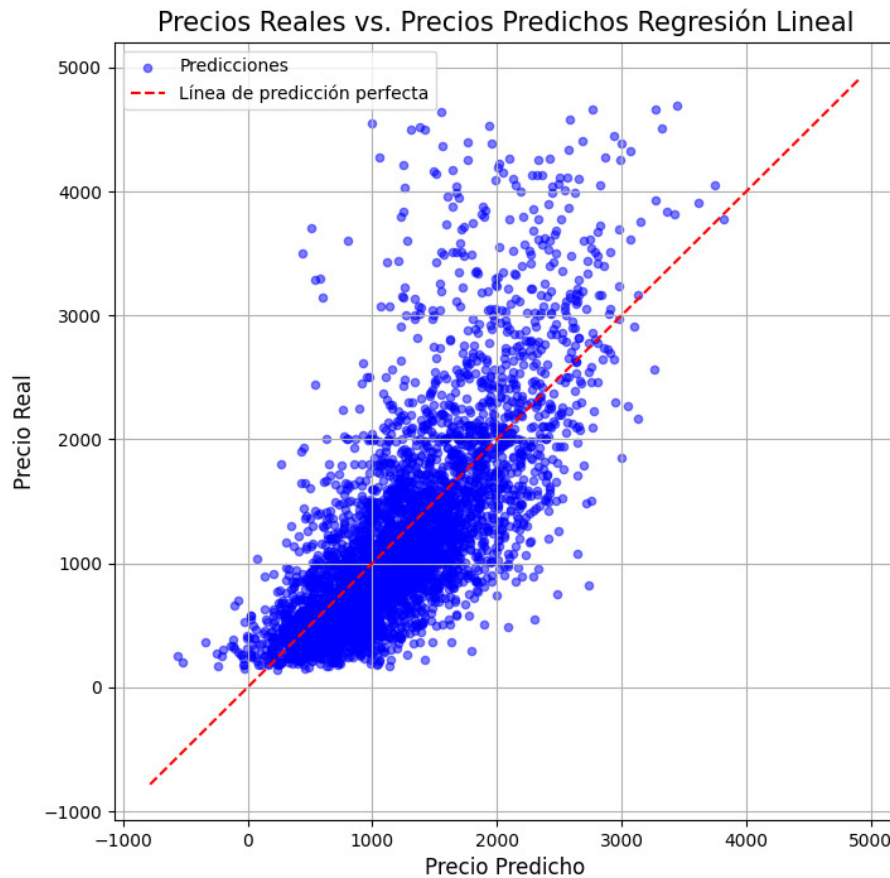


Figura 26: Predicciones Regresión Lineal Test

Lo que podemos observar en ambos casos, es que son muy similares (obviando el hecho de que la densidad del conjunto de Train es mucho mayor) en cuanto a su forma y a la desviación presente de los datos por sobre la diagonal.

Una comparativa útil la podemos tener con las métricas observadas en cada caso.

Tipo	MSE	RMSE	R^2	MAPE
Train	307440	554.47	0.548	0.401
Test	301333	548.93	0.538	0.407

Cuadro 7: Train vs. Test Regresión Lineal

Por lo tanto, no hay prácticamente diferencia en las métricas de las predicciones de ambos conjuntos.

3.2.2. XGBoost Regressor

■ Cross-validation, hiperparámetros y métrica optimizada.

Para entrenar este modelo, se realizó un Cross-validation de 5 folds, con una búsqueda randomizada, en la que se pretendía optimizar los siguientes hiperparámetros:

- Número de estimadores: controla cuántos de estos árboles individuales se añaden al modelo.
- Máxima profundidad: la profundidad máxima de un árbol. El aumento de este valor hará que el modelo sea más complejo y más propenso a sobreajustarse.

- Tasa de aprendizaje: Reducción del tamaño del paso utilizada en la actualización para evitar el sobreajuste.
- Mínimo peso por hijo: Suma mínima del peso de la instancia necesaria en un hijo. Si el paso de partición del árbol da como resultado un nodo hoja con una suma del peso de la instancia inferior a dicho valor, el proceso de construcción abandonará la partición posterior.

Con lo que se consiguió llegar a la combinación de:

- Número de estimadores = 58
- Máxima profundidad = 6
- Tasa de aprendizaje = 0.204
- Mínimo peso por hijo = 4

■ Evaluación en train/test

En la figura siguiente podremos observar como fué el desempeño del modelo entrenado con los parámetros previamente mencionados, en el conjunto de Train.

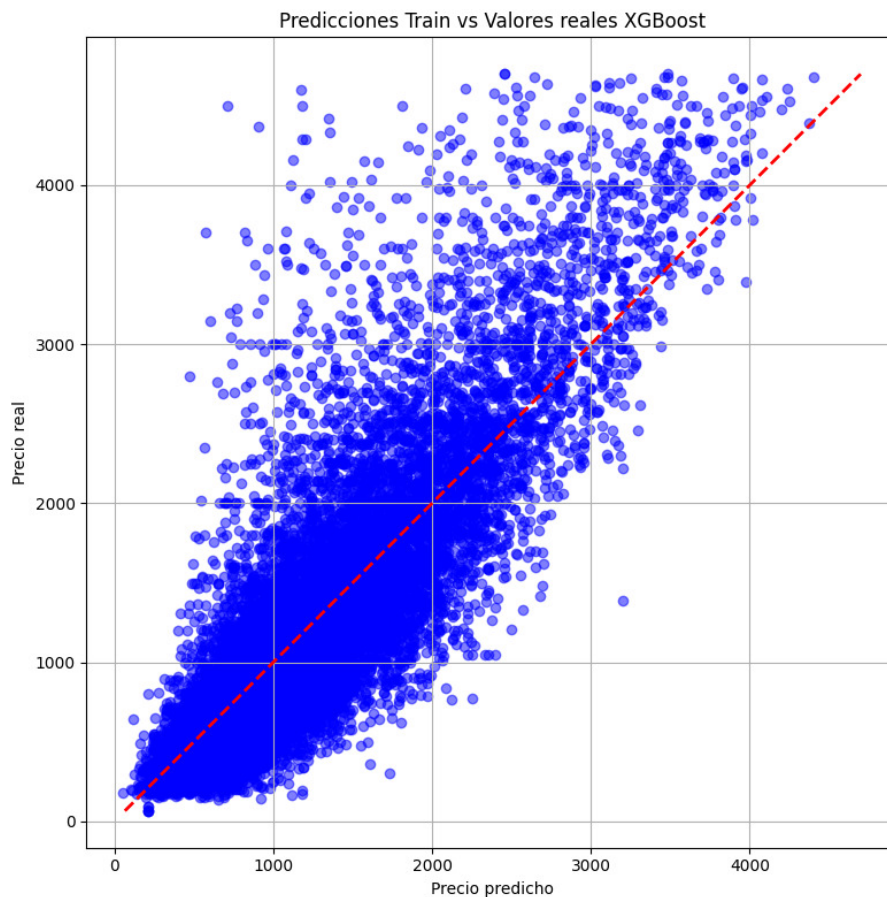


Figura 27: Predicciones XGBoost Regressor Train

A continuación, los resultados de las predicciones sobre el conjunto de Test.

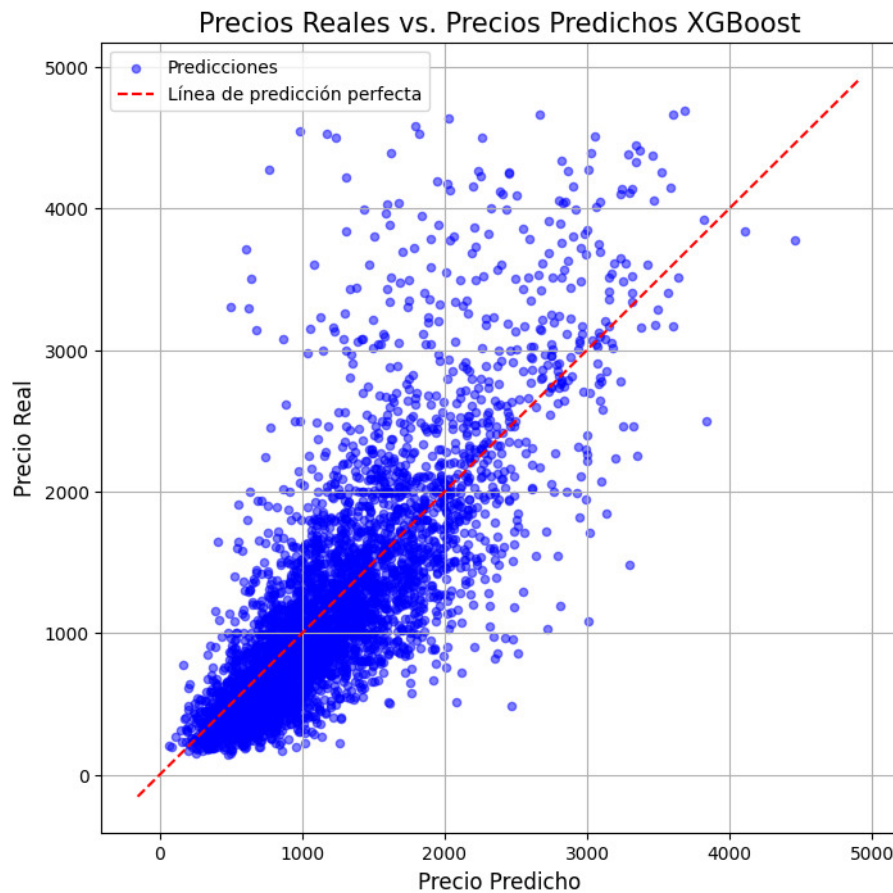


Figura 28: Predicciones XGBoost Regressor Test

Se puede observar, luego de mirar ambas figuras, que en el conjunto de Train se obtiene una mayor concentración sobre la diagonal principal, dando a entender que el desvío de los datos es mucho menor que en el del conjunto de Test.

Tipo	MSE	RMSE	R^2	MAPE
Train	301333	548.93	0.538	0.407
Test	248842	498.84	0.619	0.345

Cuadro 8: Train vs. Test XGBoost Regressor

Pese a lo observado en las figuras, el resultado en las métricas nos lleva a pensar que, aunque visualmente lo parezca, las predicciones en Train no son significativamente mejor que en Test.

3.2.3. KNN Regressor

■ Justificación de la elección

Se eligió este modelo por su conocimiento a raíz de los practicado en la cátedra y porque su variabilidad entre hiperparámetros no es tan extensa como para que eso implique una gran dedicación de tiempo, lo que nos permitió enfocarnos más en el preprocesamiento del dataset.

■ Validación y métricas de performance

Luego de haber buscado los mejores hiperparámetros, que consideramos más enriquecedores para este modelo, con cross-validation de 5 folds, llegamos a la conclusión de que los valores que mejoran nuestras predicciones son:

- Número de vecinos = 4
- Distribución del peso = Distancia (un peso mayor a más cerca se encuentre el vecino)
- Forma de calcular distancia = Euclideana

Con estos valores entrenamos el modelo con una relación de 80-20 de Train-Test y pudimos obtener los siguientes valores, representados en la siguiente figura. Aquí se puede observar cual fué el desempeño del modelo en Train.

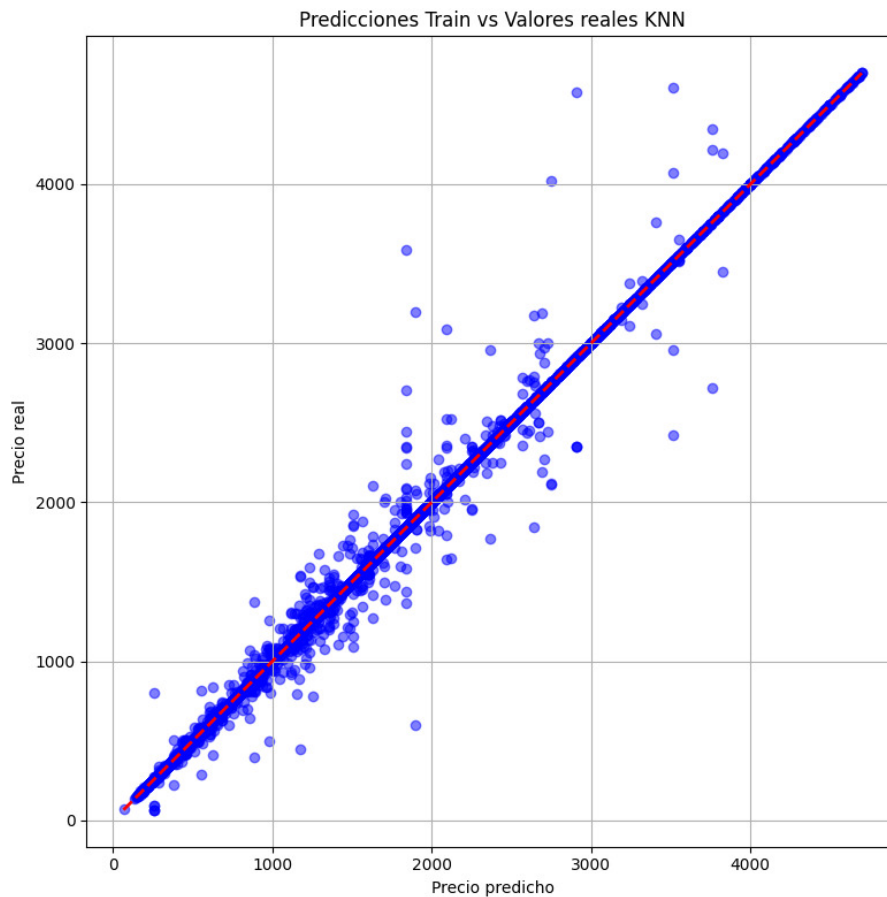


Figura 29: Predicciones KNN Regressor Train

A continuación se encuentra el desempeño del modelo entrenado con el conjunto de Test, con los mismos parámetros optimizados mencionados anteriormente.

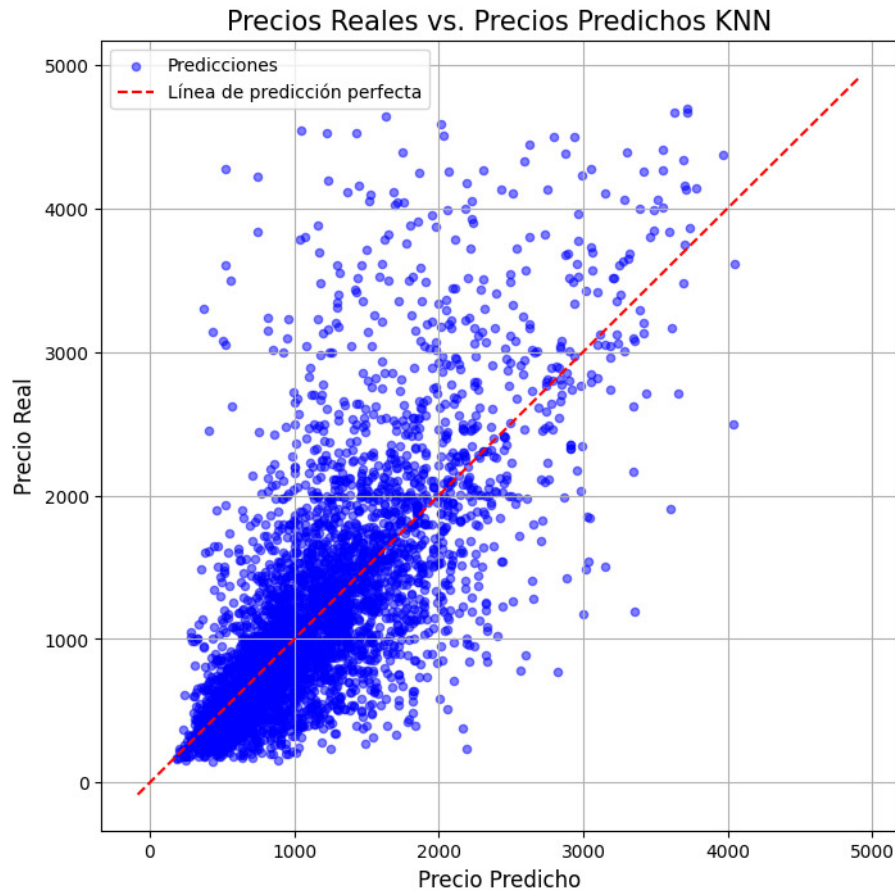


Figura 30: Predicciones KNN Regressor Test

Lo que se puede observar luego de apreciar ambas figuras es la existencia de overfitting durante el entrenamiento, mientras que en el conjunto de Test tenemos un desvalance notorio en la parte superior del gráfico.

Modelo	MSE	RMSE	R^2	MAPE
Train	1984	44.54	0.997	0.00431
Test	306135	553.29	0.531	0.359

Cuadro 9: Train vs. Test KNN Regressor

Con esta tabla, podemos confirmar la existencia de overfitting más allá de lo visual, por lo tanto llegamos a la conclusión de que fué costoso, para el modelo, generalizar con el conjunto de Test.

3.3. Comparación de resultados

Métricas de rendimiento en TEST:

Modelo	MSE	RMSE	R^2	MAPE
Regresión Lineal	301333	548.93	0.538	0.407
XGBoost Regressor	248842	498.84	0.619	0.345
KNN Regressor	306135	553.29	0.531	0.359

Cuadro 10: Resultados de modelos de Regresión

3.4. Elección del modelo

En base a los resultados obtenidos, la elección más razonable es XGBoost Regressor, ya que tiene mejores métricas (principalmente en R^2 y MAPE) y su tiempo de entrenamiento (incluyendo el de optimización de hiperparámetros) es ligeramente más lento que Regresión Lineal y sustancialmente más veloz que KNN Regressor.

4. Clustering

El objetivo de este ejercicio es analizar la posibilidad de agrupar los datos según determinados criterios mediante el algoritmo K-means.

Para ello, emplearemos un conjunto de datos que contiene información sobre diversos tracks (canciones) de Spotify.

4.1. Descripción y Análisis del dataset

Antes de comenzar a trabajar con los datos, primero dedicamos nuestro tiempo a entenderlos y analizarlos:

1. Análisis y clasificación de las variables

Nuestro dataset contiene aproximadamente 750 registros y 13 columnas con las siguientes variables:

Variable	Descripción
acousticness	Una medida de confianza entre 0.0 y 1.0 sobre si la pista es acústica. Un valor de 1.0 representa una alta confianza en que la pista es acústica.
danceability	Describe qué tan adecuada es una pista para bailar, basándose en una combinación de elementos musicales como el tempo, la estabilidad del ritmo, la fuerza del compás y la regularidad general.
duration	La duración de la pista en milisegundos.
energy	Es una medida perceptual de intensidad y actividad que va de 0.0 a 1.0. Típicamente, las pistas enérgicas se sienten rápidas, fuertes y ruidosas.
instrumentalness	Predice si una pista no contiene vocales. Cuanto más se acerca el valor a 1.0, mayor es la probabilidad de que la pista no tenga contenido vocal.
key	La tonalidad en la que se encuentra la pista. (Ej., 0 = Do, 1 = Do/Re, 2 = Re).
liveness	Detecta la presencia de una audiencia en la grabación, indicando si la pista fue grabada en vivo.
loudness	La sonoridad general de una pista medida en decibelios (dB).
mode	Indica si la melodía de la pista es mayor (1) o menor (0).
speechiness	Detecta la presencia de palabras habladas en una pista. Alto (>0.66): Es principalmente hablado (como un podcast). Medio (0.33 - 0.66): Es una mezcla de música y voz (como el rap). Bajo (<0.33): Es principalmente música.
tempo	La velocidad o el ritmo de la pista en pulsos por minuto (BPM).
time_signature	El compás de la pista, que especifica cuántos pulsos hay en cada medida.
valence	Describe la positividad musical transmitida por una pista (alegre vs. triste).

Cuadro 11: Descripción de las variables del dataset de Spotify.

Estas variables las clasificamos en:

- **Cuantitativas Continuas:** asociadas al proceso de medir; pueden tomar infinitos valores intermedios.

Variable	Justificación
acousticness	Es una medida de probabilidad en una escala de 0 a 1.

Variable	Justificación
danceability	Describe la aptitud para el baile en una escala continua de 0 a 1.
duration	Representa el tiempo, una magnitud continua.
energy	Mide la intensidad en una escala continua de 0 a 1.
instrumentalness	Mide la probabilidad (0 a 1) de ausencia de vocales.
liveness	Mide la probabilidad (0 a 1) de que la pista sea en vivo.
loudness	La sonoridad en decibelios (dB) es una escala de medición continua.
speechiness	Mide la proporción de palabras habladas en una escala continua.
tempo	La velocidad en BPM puede tener valores decimales.
valence	Mide la positividad musical en una escala continua de 0 a 1.

- **Cuantitativas Discretas:** asociadas al proceso de contar; no existen valores intermedios.

Variable	Justificación
time_signature	Es un conteo del número de pulsos por compás (valores enteros).

- **Cualitativas Ordinales (Cuasicuantitativas):** las categorías tienen un orden, pero no se puede medir la distancia entre ellas.

Variable	Justificación
key	Representa notas musicales con un orden, pero sin una distancia uniforme.

- **Cualitativas Nominales:** etiquetas distintas sin un ordenamiento lógico entre ellas.

Variable	Justificación
mode	Son dos categorías (1 = Mayor/0 = Menor) sin un orden intrínseco.

Luego, para las variables cuantitativas calculamos sus medidas de resumen, mientras que para las cualitativas visualizamos sus distintos valores existentes y la cantidad de filas para cada uno.

2. Análisis de datos faltantes

Debido a que nuestro dataset no contiene datos faltantes, no fue necesario imputar o eliminar datos.

3. Identificación de Outliers

Para realizar la detección de nuestros outliers, graficamos los Box-Plots y calculamos el z-score-modificado para todas nuestras Variables Cuantitativas.

Tras examinar los resultados, optamos por **mantener todos los valores atípicos identificados**. Esto se debe a que dichos valores no constituyen errores, sino que representan **casos extremos válidos** dentro de la diversidad del espectro musical. En particular, observamos estos outliers en las siguientes variables:

- **instrumentalness, liveness, speechiness:** los outliers reflejan categorías de audio menos frecuentes pero legítimas, que se diferencian de los patrones más comunes.
- **duration:** los valores extremos corresponden a canciones con estructuras no estándar, por ejemplo pistas muy largas o muy cortas, en contraste con la duración típica de 3-4 minutos orientada a radio.
- **loudness, energy, tempo:** los outliers son consecuencia de la amplia variedad de géneros musicales, desde estilos suaves y lentos hasta géneros rápidos e intensos.
- **time_signature:** los valores atípicos indican compases menos comunes pero correctos (como 3/4 o 5/4), característicos de ciertos géneros frente al predominante 4/4.

4.2. Análisis de la tendencia al clustering del dataset

Tras el análisis exploratorio inicial, procedimos a evaluar la tendencia al agrupamiento (clustering) del conjunto de datos. Este paso es fundamental, ya que la efectividad del algoritmo K-Means depende de la existencia de una estructura inherente en los datos. Si los datos no presentan esta tendencia, los clusters encontrados podrían no ser significativos.

Para llevar a cabo esta evaluación cuantitativa, el método que utilizamos es el Estadístico de Hopkins. Este nos proporcionará un valor numérico (entre 0 y 1) que indica si nuestro dataset es apto para ser agrupado o si su distribución es más bien aleatoria.

Dado que tanto el algoritmo K-Means como la prueba de Hopkins se basan en métricas de distancia, nos aseguramos de escalar las variables cuantitativas previamente. Este proceso asegura que todas las características contribuyan de manera equitativa al análisis, evitando sesgos por diferencias en sus escalas originales.

Al aplicar el Estadístico de Hopkins sobre el conjunto de datos escalado, se obtuvo un resultado de 0.7712. Dado que este valor es superior al umbral de 0.75, se concluye que el dataset presenta una fuerte tendencia al agrupamiento, lo que valida la aplicación de un algoritmo como K-Means para encontrar patrones significativos.

4.3. Estimación de la cantidad apropiada de grupos que se deben formar

Una vez confirmada la viabilidad del clustering, identificamos el número óptimo de clusters (K) a formar. Para ello, empleamos el Método del Codo, que busca identificar el punto de rendimientos decrecientes en la inercia del modelo a medida que aumenta el número de grupos. Adicionalmente, se utilizará el Análisis de Silueta para verificar esta elección desde una perspectiva de cohesión y separación de los clusters.

■ El método del codo

Al analizar el gráfico del Método del Codo, se observa un punto de inflexión pronunciado en $K=2$. Este *codorepresenta* el punto de rendimientos decrecientes, donde la reducción de la inercia (la suma de los errores al cuadrado dentro de cada cluster) comienza a aplanarse significativamente.

Si bien la inercia continúa disminuyendo para valores de K superiores a 2, la mejora es marginal. Esto significa que agregar clusters adicionales más allá de dos no aporta una mejora sustancial en la compactación de los grupos y podría simplemente estar dividiendo clusters que ya son cohesivos. Por lo tanto, según este método, se determina que el número óptimo de clusters para este dataset es 2.

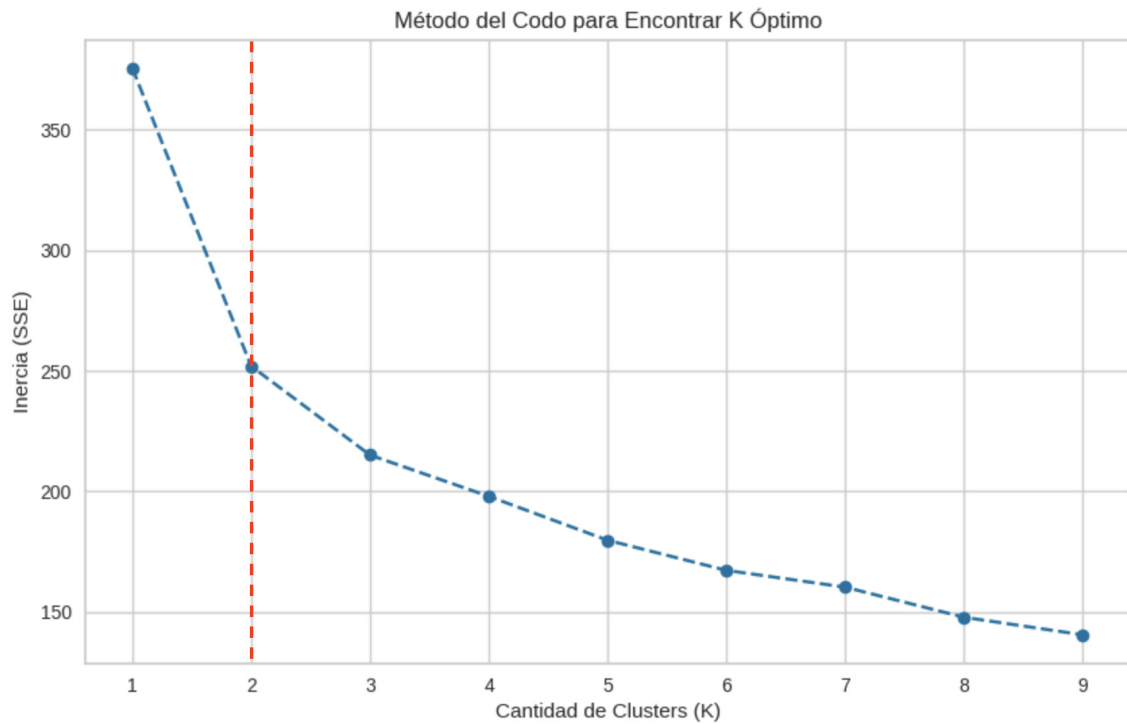


Figura 31: Resultado del método del codo

■ Análisis de Silhouette

El resultado de este análisis mostró que el puntaje más alto se alcanza con una configuración de $K=2$. Este valor máximo indica que $K=2$ es el número de clusters que logra el mejor equilibrio entre la cohesión (qué tan similares son los puntos dentro de un mismo cluster) y la separación (qué tan distintos son los clusters entre sí).

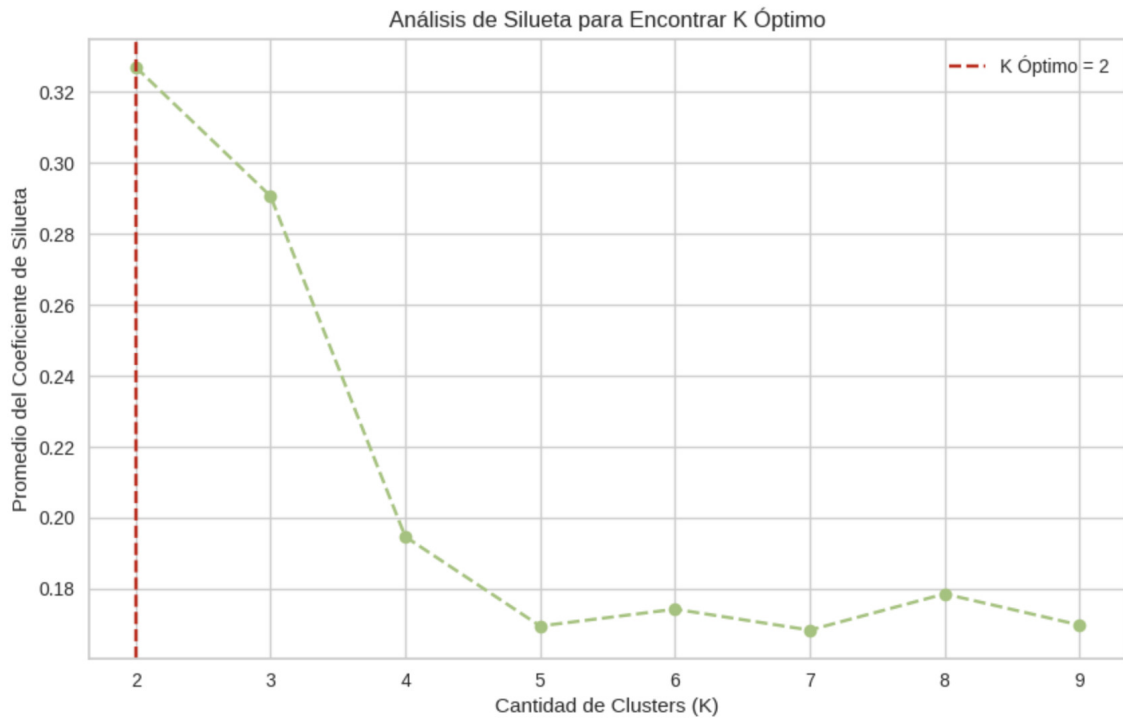


Figura 32: Resultado del análisis de Silhouette



Figura 33: Comparación de Silhouette Plots entre distintos K

Al comparar con los distintos K, si bien el Silhouette Plot de $K = 2$ muestra imperfecciones, como siluetas anchas (baja cohesión) y una pequeña cantidad de puntajes negativos, representa la estructura más equilibrada y con la mejor separación general.

Esto nos permite verificar el resultado obtenido previamente con el Método del Codo, proporcionando evidencia para seleccionar $K=2$ como el número óptimo de grupos para el análisis final.

4.4. Análisis de los clusters formados

Una vez determinado el número óptimo de clusters ($K = 2$), analizamos en detalle las características de los grupos formados para comprender su composición. Para ello, se asignó la etiqueta del cluster correspondiente a cada canción en el dataset original.

Luego, para las variables cuantitativas, agrupamos los datos por cluster y se calculó la media de cada característica. Estos promedios se compararon con la media general del dataset para identificar

las desviaciones más significativas que definen el “perfil” de cada grupo.

Para las variables cualitativas (**mode**, **key**), analizamos la distribución de frecuencias dentro de cada cluster para detectar si existían categorías dominantes que aportaran contexto musical a los perfiles.

Este proceso reveló la formación de dos clusters con perfiles musicales claros y opuestos:

- **Cluster 0 - “Pistas Acústicas y Relajadas”**: Este grupo se caracteriza por tener valores de **acousticness** e **instrumentalness** muy superiores a la media, y valores de **energy** y **loudness** muy inferiores. El análisis cualitativo mostró una fuerte preferencia por el Modo Mayor (82%). Este perfil corresponde a un arquetipo de música acústica, tranquila y a menudo instrumental.
- **Cluster 1 - “Pistas Energéticas y Fuertes”**: Por el contrario, este cluster presenta un perfil opuesto, con alta **energy** y **loudness** y baja **acousticness**, todos valores que se desvían significativamente de la media general. Este grupo define un arquetipo de música energética, bailable y con una producción más comercial, alineado con un sonido mainstream.

De esta forma, se concluye que el algoritmo K-Means logró segmentar exitosamente el dataset en dos macro-categorías musicales coherentes y significativas.

Finalmente, empleamos dos técnicas de reducción de dimensionalidad para validar visualmente la calidad de la segmentación:

1. **PCA**: Esta técnica nos proporciona una “vista honesta” de la estructura global de los datos, preservando la mayor cantidad de varianza posible.

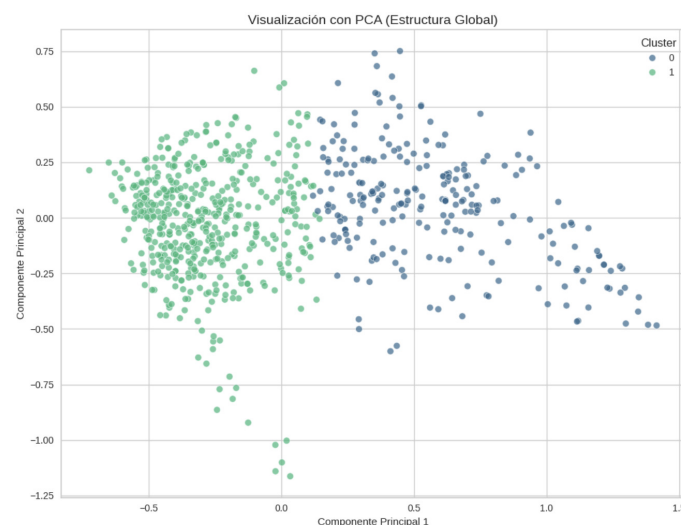


Figura 34: Visualización PCA de los clusters.

2. **t-SNE**: A diferencia de PCA, t-SNE se especializa en preservar la estructura local, es decir, mantiene juntos los puntos que son vecinos cercanos.

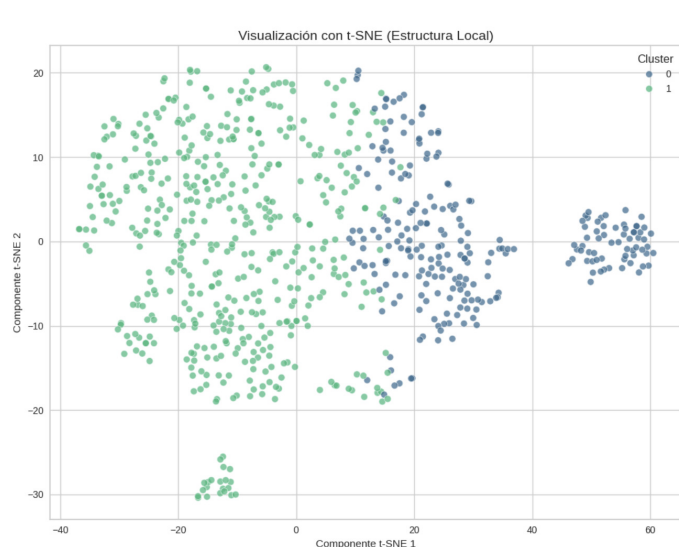


Figura 35: Visualización t-SNE de los clusters.

En estos gráficos se pueden identificar dos “polos” o centros de gravedad claramente definidos, correspondientes a cada cluster. A la vez, se observa un solapamiento entre ellos, lo que se puede interpretar como evidencia de que la música constituye un espectro con géneros híbridos situados en la zona de transición.

Esto refuerza los resultados del clustering, mostrando que K-Means ha encontrado la división más coherente posible dentro de un conjunto de datos complejo y continuo.

4.5. Conclusión

Si bien la variedad musical es vasta, descubrimos que es posible utilizar esta técnica de clustering para destilar esta complejidad en arquetipos comprensibles, revelando dos tipos de macroestructuras que definen la música que escuchamos.

5. Tiempo dedicado

A continuación se resumen las tareas asumidas por cada integrante del equipo y el tiempo promedio semanal invertido en estas.

Integrante	Tarea	Prom. Hs Semana
Franco Rodriguez	Realizar Ej2 e informe sobre dicho punto	6
Blas Chuc	Realizar Ej3 e informe sobre dicho punto	5
Helen Chen	Realizar Ej4 e informe sobre dicho punto	5
Tomas Caporaletti	Realizar Ej1 e informe sobre dicho punto	5
Lorenzo Busato	Trabajo sobre el Ej1	1

Cuadro 16: Distribución de tareas y carga semanal promedio