UNIVERSIDAD TORCUATO DI TELLA

TD VI: Inteligencia Artificial

Trabajo Práctico 2

Primer Cuatrimestre del 2025

Autores:

Chab Lopez, Catalina

Brusco, Catalina

Chen, Belén

Mayo del 2025

1 Introducción

El análisis de comportamiento de usuarios en plataformas de streaming es una herramienta clave para mejorar la experiencia de usuario y optimizar la recomendación de contenido. En este trabajo, se busca predecir si una sesión de usuario terminará en una reproducción o no, a partir de los eventos previos registrados. Para ello, se trabajó con datos reales provistos por Spotify de un determinado usuario.

En el presente informe, se describirá un modelo de aprendizaje supervisado basado en boosting (CatBoost) para predecir si un usuario saltará o no una canción en una plataforma de streaming, Spotify. Se seleccionó dicho modelo tras comparar varios algoritmos ya que superó otras alternativas como LightGBM y Gradient Boosting, con base en el rendimiento medido a través del AUC-ROC.

Algunas de las ventajas claves de CatBoost para mejorar la performance del modelo haciéndolo más robusto y dándole buena capacidad de generalización fueron:

- Manejo eficiente de variables categóricas.
- Tolerancia a valores faltantes sin necesidad de imputaciones complejas.
- Mejor desempeño en métricas de validación respecto a alternativas

Este informe detalla todas las etapas del trabajo: se incluye un análisis exploratorio de los datos provistos, la creación de nuevas variables relevantes y la definición de una estrategia de validación adecuada para entrenar y evaluar los modelos.

2 Modelo

El conjunto de datos utilizado proviene de interacciones reales de un usuario con una plataforma de streaming musical. Contiene **100.144 filas** y **12 columnas**. El porcentaje de valores faltantes es relativamente bajo, representando un **1,69%** del total. Además, se trata de un dataset balanceado: el **47%** de los registros tienen `TARGET = Sí`. En el preprocesamiento, las variables categóricas fueron completadas con el valor `'unknown'` cuando presentaban datos faltantes.

Se detectaron algunas columnas con una gran cantidad de valores nulos, como `user_agent_decrypted` (16.397 valores faltantes). Además, observamos que los 973 valores nulos presentes en las columnas `master_metadata_track_name`, `master_metadata_album_artist_name` y `master_metadata_album_album_name` corresponden a las mismas filas; es decir, en esas 973 filas faltan datos en las tres columnas simultáneamente.

A continuación, se presenta una tabla (Table 1) con el significado de cada variable, organizada en tres bloques: variables conservadas, eliminadas y nuevas agregadas. Se utiliza rojo para las eliminadas y verde para las agregadas.

Variable	Descripción
Variables conservadas	
master_metadata_track_name	Nombre de la canción
master_metadata_album_artist_name	Nombre del artista
reason_start	Motivo por el que empezó la canción
shuffle	Si estaba activado el modo shuffle
TARGET	Variable objetivo (si la canción fue saltada o no)
Variables eliminadas	
username	Identificador del usuario
ts	Timestamp de la reproducción
platform	Plataforma utilizada (ej. iOS, Android)
conn_country	País de conexión
user_agent_decrypted	Información del dispositivo (user-agent)
master_metadata_album_album_name	Nombre del álbum
spotify_track_uri	URI de la canción en Spotify
Variables agregadas	
mes	Mes extraído del timestamp
semana	Semana del año
dia_hora	Combinación día-hora
track_play_count	Frecuencia de aparición de la canción
artist_play_count	Frecuencia de aparición del artista
platform_by_year	Combinación plataforma y año

Table 1: Descripción de las variables del dataset

2.1 Separación de Datos

Los datos se dividieron en aproximadamente un 90% para entrenamiento y validación cruzada, y un 10% reservado exclusivamente para evaluar el desempeño final (*test*). Durante el entrenamiento, se buscó minimizar la pérdida logarítmica (*log loss*) en los datos de entrenamiento, mientras que la selección de hiperparámetros se realizó considerando su desempeño en validación (a partir de la métrica AUC ROC), con el objetivo de mejorar la generalización del modelo y evitar el sobreajuste.

Dividirlo inicialmente permitió evitar *data leakage* ya que se calcularon las variables, track play count y artist play count, luego de separar los datos.

Además, al utilizar stratify=y en el split, garantizamos que la proporción de clases del target se mantenga constante en ambos subconjuntos, preservando la distribución original y evitando sesgos en la evaluación.

2.2 Ingeniería de Datos

En este análisis, realizamos un filtro de atributos con dos enfoques. El primero consistió en evaluar las relaciones entre las variables, y el segundo en estudiar cómo estas variables se correlacionan con el objetivo o target.

Nuestro objetivo principal era reducir el sobreajuste (overfitting). Sabemos que agregar demasiadas variables puede introducir ruido en el modelo, lo que dificultaría su capacidad para generalizar correctamente.

2.2.1 Análisis de la Relación entre Atributos

En esta fase, analizamos cómo las variables se relacionaban entre sí. Esto nos permitió identificar posibles redundancias y variables altamente correlacionadas, lo cual es fundamental para reducir el ruido en el modelo.

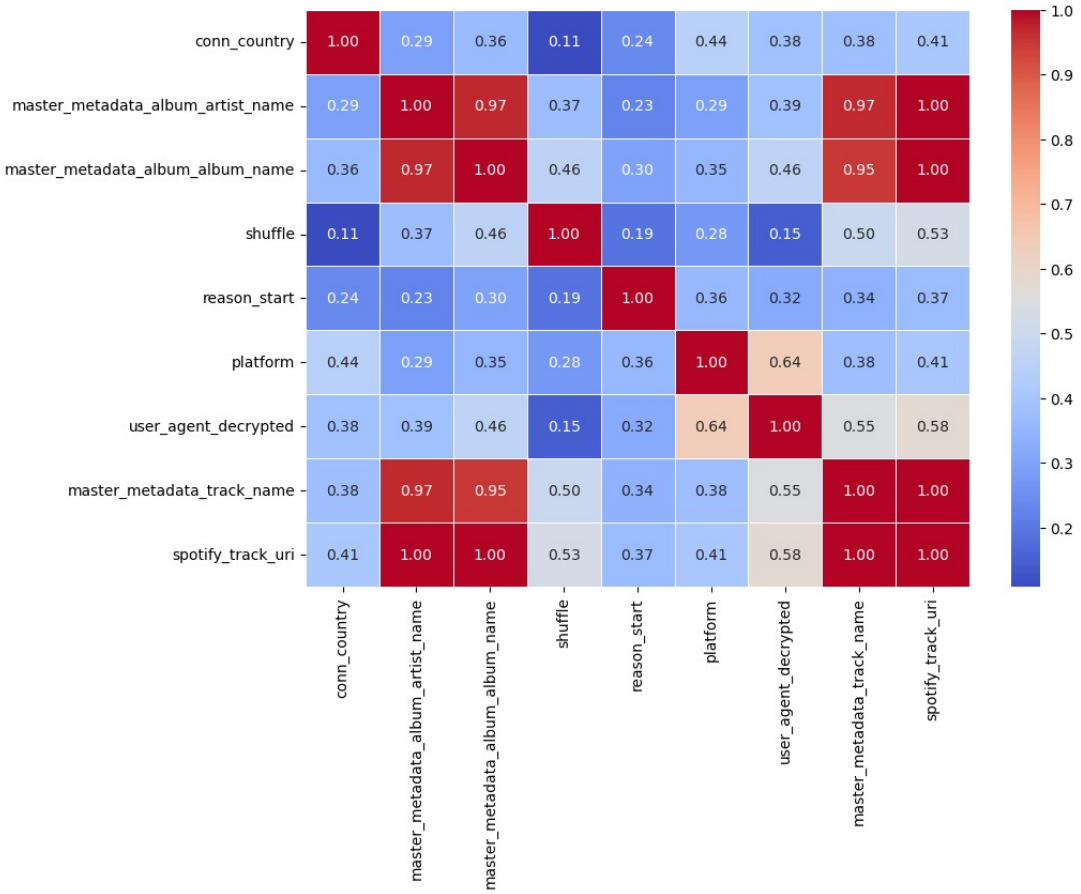


Figure 1: Matriz de correlación categórica entre atributos(versión original al dataset)

En la Figura 1 se presenta la matriz de correlación categórica entre los atributos seleccionados. Esta matriz se construyó utilizando una métrica de asociación diseñada para variables categóricas, lo que nos permitió identificar relaciones fuertes entre las columnas que podrían ser redundantes.

Una observación clave es la alta correlación entre los atributos relacionados con la metadata musical: `master_metadata_album_artist_name`, `master_metadata_track_name`, y `master_metadata_album_album_name`, todos con coeficientes cercanos o iguales a 1.0. Debido a esta alta correlación, decidimos eliminar el atributo `master_metadata_album_album_name`, ya que su eliminación no afectó negativamente el rendimiento del modelo durante la validación cruzada, mientras que eliminar alguna de las otras dos variables sí empeoraba el rendimiento.

Además, observamos que el campo `spotify_track_uri` presenta una correlación perfecta con estos atributos, lo que indica que contiene información redundante al representar una combinación única de los anteriores. Tras realizar pruebas de validación, encontramos que eliminar `spotify_track_uri` mejoraba el rendimiento del modelo.

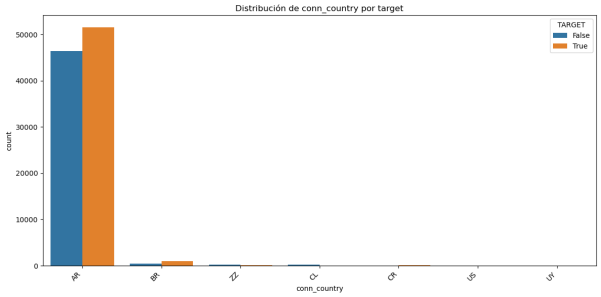
Por otro lado, los atributos `shuffle`, `reason_start`, `conn_country` y `user_agent_decrypted` muestran correlaciones moderadas entre sí. Esto sugiere que estas variables podrían aportar valor predictivo sin generar un exceso de colinealidad.

Este análisis de correlación nos permitió tomar decisiones más informadas durante la selección de variables, eliminando aquellas con alta correlación para reducir la multicolinealidad y mitigar el riesgo de sobreajuste, mejorando así la capacidad de generalización del modelo.

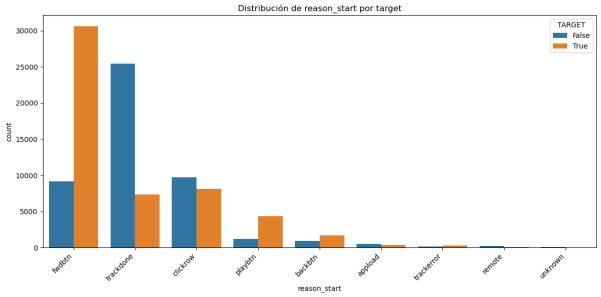
2.2.2 Relación con el Target

Para el análisis de las variables categóricas respecto al target, se decidió enfocarse únicamente en aquellas que mostraban patrones relevantes o poca influencia del target. Se conservaron los gráficos de las variables Conn Country, Reason Start y Shuffle. Por otro lado, se optó por no analizar en detalle variables como Master Metadata Album Album Name, Master Metadata Album Artist Name, Master Metadata Track Name y Spotify Track Uri, debido a su alta granularidad y cardinalidad, lo que dificulta extraer

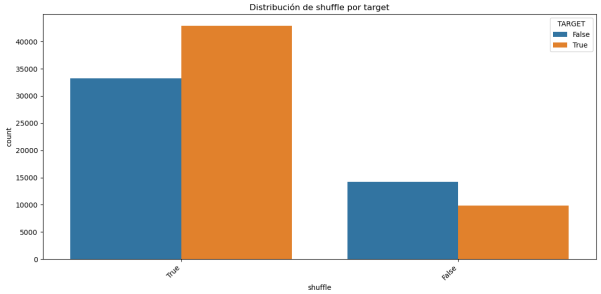
conclusiones generales. Además, la variable User Agent Decrypted fue descartada porque presenta una categoría predominante ("unknown"), limitando su valor explicativo. Esta selección permitió centrar el análisis en las variables con mayor potencial interpretativo, evitando sobrecargar el informe con gráficos poco informativos.



(a) Conn Country



(b) Reason Start



(c) Shuffle

Figure 2: Comparación de variables categóricas con el target

La Figura 2 compara la distribución de diversas variables categóricas con respecto al *target*. A continuación, se presenta un análisis detallado por gráfico:

- **(a) Conn Country:** Se observa una fuerte concentración de datos en un único país, mientras que los demás países presentan frecuencias muy bajas. Esto indica que el análisis estará dominado por ese país principal(Argentina), y las otras categorías tendrán un impacto estadístico reducido. La proporción entre True y False es muy similar en ese país predominante.
- **(b) Reason Start:** La mayoría de las reproducciones se explican por unas pocas razones principales (como autoplay, manual, etc.), mientras que las demás categorías son poco frecuentes. Existe un sesgo claro hacia ciertas razones, y las diferencias en True/False son notables en cada grupo o categoría, lo que sugiere que este predictor puede ser importante. Al intentar simplificar el modelo agrupando solo las razones principales (probando con la mejor, las dos mejores, las tres mejores y las cuatro mejores, y agrupando las restantes en "otras"), observamos una disminución en la performance en todos los casos. Por lo tanto, se decidió mantener la variable completa, ya que esta estrategia resultó ser la más efectiva.

- **(c) Shuffle:** Esta variable tiene solo dos categorías (True/False) y muestra una distribución desbalanceada entre las distintas categorías, pero parecida la distribución respecto al target en cada categoría. Aunque inicialmente pensamos que no aportaría información relevante al modelo, al probar excluirla observamos una disminución en la performance, por lo que decidimos conservarla.

Este análisis, sumado al anterior, nos permitió reducir el número de atributos categóricos a aquellos con mayor potencial predictivo, mejorando así la eficiencia y generalización del modelo `CatBoost`.

2.2.3 Análisis de Cardinalidad de Variables Categóricas

Como se mencionó en el punto anterior, la cardinalidad de las variables categóricas es un factor relevante. Algunas de estas variables poseen una alta cantidad de categorías únicas, lo cual puede representar un problema para el modelo, ya que aumenta la complejidad, ralentiza el proceso y puede llevar a un sobreajuste, es decir, a que el modelo aprenda patrones poco útiles. A continuación, se presentan algunas de estas variables, junto con su cardinalidad ordenada de manera decreciente:

- `spotify_track_uri`: 9644 categorías
- `master_metadata_track_name`: 8139 categorías
- `master_metadata_album_album_name`: 5994 categorías
- `master_metadata_album_artist_name`: 3042 categorías
- `user_agent_decrypted`: 20 categorías
- `reason_start`: 9 categorías
- `conn_country`: 7 categorías
- `shuffle`: 2 categorías

Es importante señalar que variables como `spotify_track_uri` y `master_metadata_album_album_name` fueron eliminadas debido a su alta correlación. Esta eliminación es beneficiosa para el modelo, ya que se trata de identificadores con alta cardinalidad que no contribuyen a identificar patrones generales, y su inclusión no aporta significativamente, a diferencia de la información contenida en el nombre del `track`. Por otro lado, la variable `user_agent_decrypted` presenta una considerable diversidad, pero, como se mencionó previamente, su contenido no tiene una relación directa con la decisión de salto de canción.

Sin embargo, se conservaron variables como `master_metadata_track_name` y `master_metadata_album_artist_name` debido a su potencial valor informativo. A pesar de su alta cardinalidad, se manejaron adecuadamente mediante el uso de contadores de frecuencia, como se mencionará en el próximo punto, lo que permite capturar información útil sin sobrecargar el modelo.

De esta manera, se equilibró la representatividad de los datos reduciendo la introducción de ruido o sobreajuste.

2.2.4 Nuevos Atributos

Se realizaron diversos experimentos que no resultaron beneficiosos:

- Incluir el género musical (`genre`) no mejoró la performance, ya que la distribución de saltos era similar entre géneros.
- Agrupar las razones de inicio (`reason_start`) en las tres más frecuentes según el cantante (`clickrow`, `fwdbtn`, `trackdone`) tampoco mejoró los resultados.
- Incorporar métricas basadas en tiempo y comportamiento secuencial del usuario (como *lags*, *skips previos*, *visto antes*) no generó mejoras significativas.
- Uso de franjas horarias (mañana, tarde, noche, madrugada) tampoco fue relevante de forma aislada.

- Se incorporaron nuevos datos descargados directamente desde la plataforma de Spotify, los cuales contenían ocho variables adicionales. En una primera instancia, dichas variables fueron integradas al conjunto original, completando con el valor `"unknown"` aquellos registros que no las poseían. También se evaluó el uso exclusivo de los datos nuevos, sin considerar las variables adicionales.

  Para estos registros recientes se generó una columna denominada `TARGET`, replicando el formato del dataset provisto por la cátedra. Esta variable fue construida a partir de los campos `reason_start` y `ms_played`. En particular, cuando `reason_start` era igual a `"fwdbtn"` o el valor de `ms_played` resultaba menor a 15.000, se establecía `TARGET` en `True`. Esta construcción tuvo como objetivo aproximar el criterio de la variable a predecir, dado que no se disponía de información precisa sobre la lógica de la columna `skipped` provista por Spotify.

  No obstante, esta estrategia no produjo mejoras en el rendimiento; por el contrario, se observó una disminución en la performance del modelo. Este deterioro podría atribuirse al hecho de que los datos analizados provenían únicamente de dos usuarios, lo cual restringe su capacidad de generalización. Para obtener patrones más representativos y robustos, habría sido necesario incorporar datos de una mayor cantidad de usuarios, permitiendo así capturar una diversidad más amplia de comportamientos.

Por otro lado, se desarrollaron transformaciones y nuevas variables que sí demostraron ser útiles para el modelo:

- A partir de la marca temporal original (`ts`), se derivaron atributos temporales como: **mes**, **semana del año**, **día de la semana** y **hora del día**. Luego, se combinaron en nuevas variables categóricas como `dia_hora`, que captura patrones de consumo específicos en determinadas franjas.
- Se creó una variable combinada `platform_by_year` que une plataforma de acceso y año de reproducción. Esta combinación permitió capturar posibles cambios en el comportamiento de los usuarios a lo largo del tiempo según la plataforma.
- Se incorporaron variables de **frecuencia personalizada**, como `track_play_count` (cantidad de veces que una canción aparece en el conjunto de entrenamiento) y `artist_play_count` (frecuencia del artista). Estas variables numéricas ayudaron a modelar la popularidad del contenido. Ambas se crearon luego de dividir el conjunto de datos para evitar data leakage.

Estas nuevas características enriquecieron la representación de los datos y contribuyeron a mejorar la capacidad predictiva del modelo `CatBoost`. A continuación, veremos el cambio entre las variables de correlación en la Figura 3.

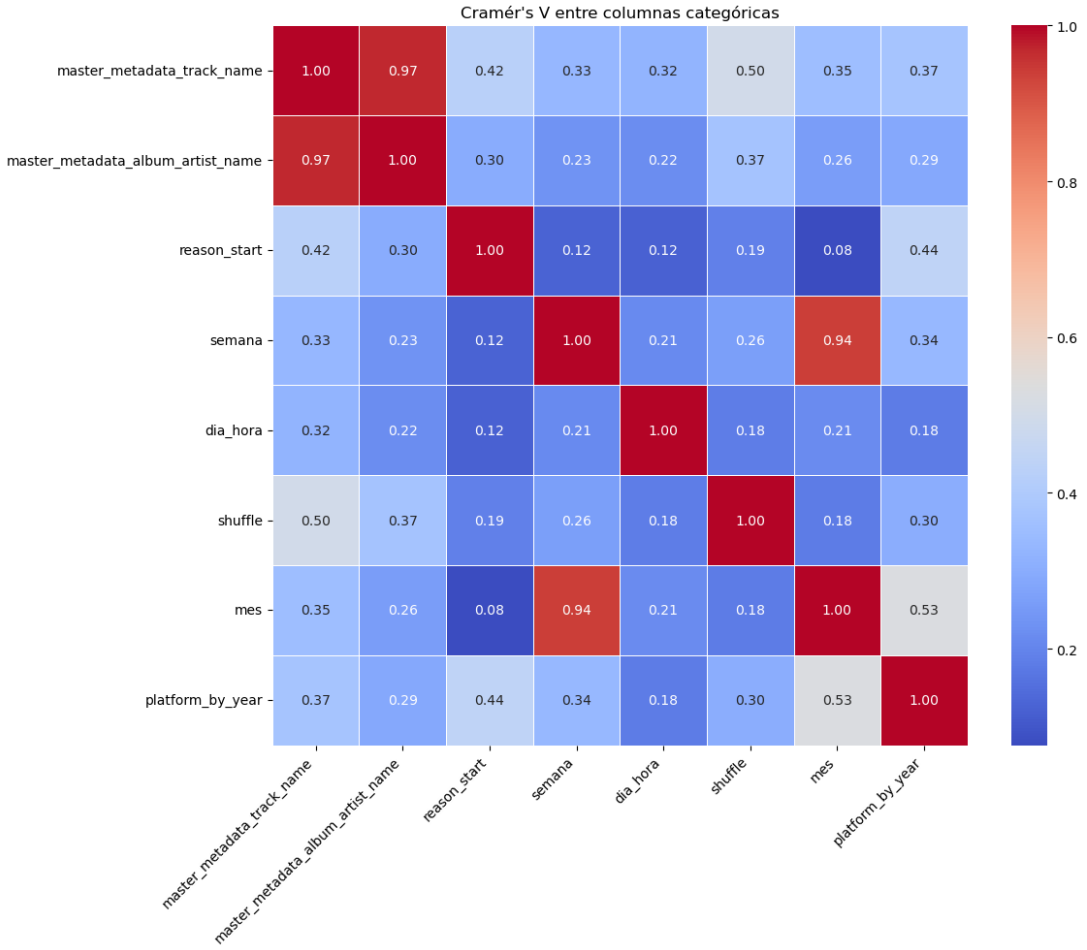


Figure 3: Matriz de correlación categórica entre atributos (versión final)

La inclusión de las nuevas variables temporales derivadas — `mes`, `semana`, `día_hora` — junto con `platform_by_year` permitió capturar patrones relevantes que no estaban presentes en los atributos originales. En la Figura 3, se observa cómo estas variables muestran correlaciones significativas, lo cual sugiere que aportan nueva información al modelo sin introducir redundancia excesiva.

En detalle:

- `mes`: Derivada de ts. Presenta una correlación alta con `semana`, aportando una perspectiva estacional que ayuda a capturar cambios en el comportamiento del usuario a lo largo del año. Si la quitamos, empeora la performance.
- `semana`: Derivada de ts. Refleja fluctuaciones semanales en el comportamiento, añadiendo granularidad temporal. Si bien está muy correlacionada con `mes`, quitarla perjudica la performance del modelo.
- `día_hora`: Derivada de ts. Esta variable combinada resultó especialmente relevante porque generó una mejora en la performance del modelo mayor que las modificaciones anteriores. Permitiendo al modelo identificar patrones específicos de consumo en ciertos días y franjas horarias (por ejemplo, horas pico de escucha).
- `platform_by_year`: Derivada de la combiación entre el año y la plataforma, ofreciendo una perspectiva combinada que ayudó a reflejar cambios en el comportamiento de los usuarios a lo largo del tiempo y diferencias en las dinámicas de uso entre plataformas.

Es importante resaltar que la incorporación de estas nuevas variables ha tenido un impacto positivo en el rendimiento predictivo del modelo `CatBoost`, evidenciado por la métrica de AUC-ROC. Estas variables no solo granularizan la información, sino que aportan un mayor entendimiento de las preferencias del usuario, mejorando así la capacidad de generalización al capturar variabilidad temporal y estructural que

no había sido considerada previamente. También eliminar y agregar dichas variables aportó mejoras la reducción de correlación entre las variables.

En la siguiente sección se detalla la validación, la selección de hiperparámetros y el análisis de importancia de variables.

3 Selección de hiperparámetros

3.1 Ajuste de hiperparámetros

Para optimizar el rendimiento del modelo, se llevó a cabo un ajuste de hiperparámetros mediante *grid search* aplicada al clasificador `CatBoostClassifier`. Esta técnica evalúa de forma exhaustiva todas las combinaciones posibles de un conjunto predefinido de valores para los hiperparámetros, lo cual permite identificar configuraciones óptimas.

3.1.1 Hiperparámetros explorados

Se definieron rangos adecuados para cada hiperparámetro, considerando un equilibrio entre flexibilidad y control del sobreajuste:

- `iterations`: [100, 200, 300]
  Este rango de iteraciones permite evaluar desde configuraciones más livianas hasta otras con mayor capacidad de aprendizaje, sin exceder tiempos de entrenamiento razonables; es adecuado considerando el tamaño del conjunto de datos, el uso de validación cruzada y las tasas de aprendizaje evaluadas.
- `learning_rate`: [0.01, 0.05, 0.1]
  Tasas de aprendizaje pequeñas y moderadas, para garantizar una optimización estable y favorecer la generalización.
- `depth`: [4, 6, 8]
  Profundidad de los árboles, que determina la complejidad del modelo y su capacidad para capturar interacciones no lineales.
- `l2_leaf_reg`: [1, 3, 5, 7]
  Parámetro de regularización L2, que ayuda a controlar la complejidad del modelo y mitigar el sobreajuste.

3.1.2 Validación

La selección de hiperparámetros se realizó utilizando validación cruzada de 5 particiones ($k = 5$) sobre los datos de entrenamiento. Esta técnica divide el conjunto de entrenamiento en cinco partes, entrenando el modelo en cuatro y validando en la restante de forma rotativa, permitiendo una evaluación robusta del desempeño.

Los valores de los mejores hiperparámetros encontrados son:

- 'depth': 8
- 'learning_rate': 0.1
- 'l2_leaf_reg': 5
- 'iterations': 300

3.2 Entrenamiento final

Una vez identificada la mejor combinación de hiperparámetros, se entrenó el modelo final utilizando exclusivamente los datos de entrenamiento completos. Posteriormente, se evaluó el desempeño del modelo en el conjunto de prueba mediante la métrica AUC-ROC, generando además la curva ROC para visualizar la capacidad discriminativa del modelo.

El valor de AUC se llevó a completar fue de **0.9379** en el conjunto de test que generamos para nuestra exploración, y **0.93186** en el conjunto privado de Kaggle.

4 Análisis de Feature Importance

Para comprender qué factores resultan más determinantes a la hora de predecir si un usuario saltará una canción, examinamos la "ganancia de información" que CatBoost asigna a cada variable. En la Figura 4 se muestran los veinte atributos más influyentes.

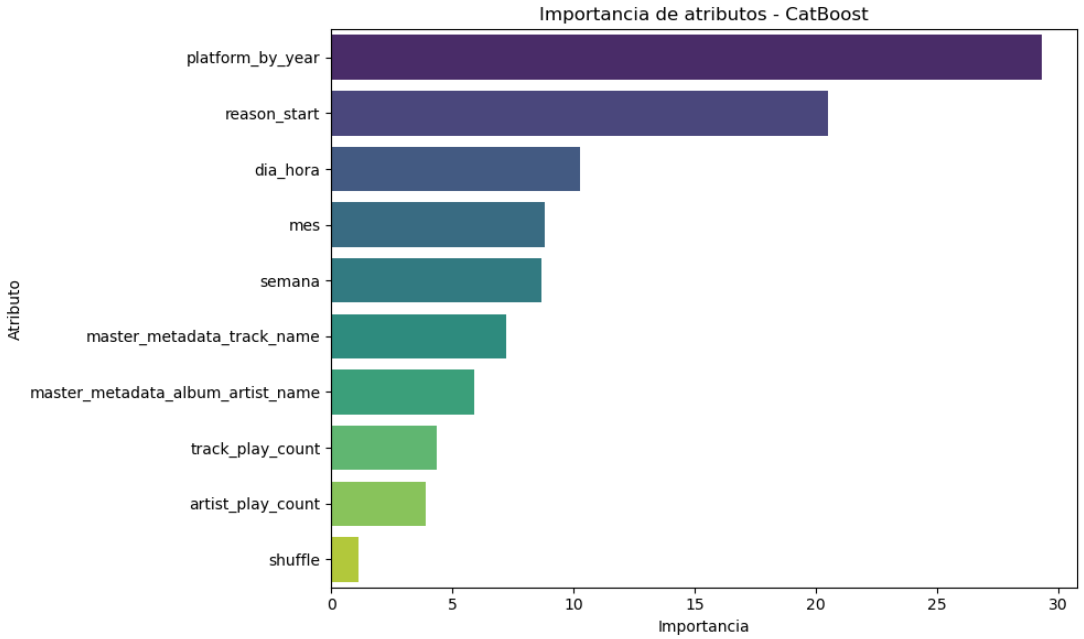


Figure 4: Gráfico de importancia de atributos

En particular, vemos que la variable `platform_by_year` es la de mayor ganancia informativa, lo que indica que los cambios en el comportamiento de escucha a lo largo del tiempo y según el dispositivo o la aplicación son determinantes para anticipar si un usuario saltará una canción. El segundo atributo más importante es `reason_start`, que añade información sobre el modo en que se inició la reproducción (autoplay, selección manual, recomendación), capturando de forma implícita aspectos de la intención del usuario al elegir contenido.

A continuación, las tres variables derivadas del timestamp (ts) -`día_hora`, `mes` y `semana`- contribuyen de manera conjunta a casi un tercio de la importancia total, lo que demuestra que existen patrones estacionales/diarios que influyen en la probabilidad de que el usuario skippee una canción.

En contraste, las características que solo refieren a la canción —el título del track y el nombre del artista—, así como los contadores de frecuencia de reproducción, presentan valores de ganancia más moderados. Esto sugiere que la popularidad o la familiaridad con la canción tienen un efecto real pero menos decisivo que el contexto en que se produce la escucha.

Finalmente, aunque `shuffle` aporta la menor ganancia individual, su inclusión optimiza mínimamente la performance, evidenciando que incluso elementos aparentemente triviales pueden capturar matices del comportamiento de usuario.

En resumen, este análisis parece indicar que para predecir si el usuario va a saltar una canción, resulta más relevante el contexto de la reproducción, que la canción en sí.

5 Conclusión

En este trabajo se diseñó y evaluó un modelo de aprendizaje supervisado utilizando CatBoost, con el objetivo de predecir si un usuario va a saltar una canción en Spotify. A partir de un análisis exploratorio profundo, se crearon variables temporales y de frecuencia, y se eliminaron atributos redundantes. Esto permitió mejorar considerablemente el rendimiento del modelo respecto al baseline inicial.

Lo que más aportó a la mejora fue la ingeniería de atributos: entender cómo, cuándo y por qué una canción es reproducida permitió construir variables más informativas que reflejan mejor el comportamiento del usuario. En especial, variables como platform_by_year, reason_start y el contexto temporal

(día, hora, mes) resultaron ser claves, ya que ayudaron a capturar rutinas, hábitos e intenciones de uso que van más allá de la canción en sí.

A pesar del buen desempeño del modelo, es importante señalar que el análisis se basó en datos de un único usuario. Esto limita la posibilidad de generalizar los resultados, ya que el comportamiento musical puede variar ampliamente entre personas. Como posible mejora, se podría ampliar el dataset con información de distintos perfiles, lo que permitiría entrenar modelos más robustos y menos dependientes de los hábitos individuales.

También se podrían probar otros algoritmos o combinar modelos para afinar las predicciones, así como ajustar más finamente los hiperparámetros y automatizar partes del preprocesamiento, lo que agilizaría futuros análisis.