Escuela de Negocios. Master in Management + Analytics

Ingresos en Plataformas. Un enfoque de Machine Learning en los datos de la API pública de Spotify

Autoría: Ahedo, Agustín
Año: *2023*

UNIVERSIDAD TORCUATO DI TELLA

MASTER IN MANAGEMENT + ANALYTICS

INGRESOS EN PLATAFORMAS

Un enfoque de Machine Learning en los datos de la API pública de Spotify

Agustín Ahedo

Mayo de 2023

Tutor: Martín Ezequiel Masci

Resumen

Con la preponderancia de las empresas de plataformas en la última década y su expansión potencial, cada vez es mayor la oportunidad en descubrir maneras de tener y aumentar ingresos en las mismas como usuario. De esta forma, existe actualmente una gran variedad de indicadores o variables en estas compañías de plataformas las cuales pueden llegar a ser optimizadas para aumentar la ganancia potencial de parte del usuario.

En concreto, se utiliza como caso principal en este estudio datos proveídos por la API de la aplicación *Spotify.* Con estos datos se aplican modelos de aprendizaje automático para predecir la popularidad (basada en la cantidad de reproducciones) de las canciones. De esta forma, se determina si las variables técnicas de producción musical son suficientes para lograr una predicción correcta y, si lo son, cuáles variables son las más significativas. Estas variables terminaron siendo aptas para la predicción al tener los modelos un buen rendimiento en general, siendo el que obtuvo el mejor resultado el modelo *XGBoost* cuyas variables más significativas fueron tomadas como base para las conclusiones finales y las recomendaciones de negocio. A su vez, se desarrolla un repositorio virtual con los algoritmos y programas necesarios para lograr la trazabilidad del trabajo de investigación (https://github.com/Ag-Ah/SpotifyAPI). Por último, se efectúa una recomendación de negocio para aumentar ingresos en la plataforma basada en los resultados obtenidos por los modelos.

Abstract

With the prevalence of platform companies in the last decade and their potential expansion, there is an increasing opportunity to discover ways to gather and increase income as a user. Thus, there is currently a wide variety of indicators or variables within these platform companies that can be optimized to enhance the user's potential earnings.

Specifically, data provided by the Spotify API is used as the main case in this study. Machine learning models are applied to predict the popularity of songs (based on the number of streams). In this way, it is determined whether the audio features themselves are sufficient to achieve accurate predictions and, if so, which variables are the most

significant. These variables proved to be suitable for prediction as the models performed well overall, with the *XGBoost* model yielding the best results. The most significant variables from this model serve as the basis for the final conclusions and business recommendations. Additionally, a virtual repository is developed with the necessary algorithms and programs to ensure the traceability of the research work (https://github.com/Ag-Ah/SpotifyAPI). Finally, a business recommendation is made to increase revenue on the platform based on the results obtained from the models.

Índice

Índice 4
Índice de Tablas....5
Índice de Figuras 5
1. Introducción 7
1.1. Contexto 7
1.2. Problema 7
1.3. Objetivo 8
2. Monetización de Plataformas....10
2.1. KPIs 10
2.2. Monetización 11
3. Metodología y Datos 14
3.1. Tendencias Preliminares: Aprendizaje No Supervisado....14
3.2. Importancia de Variables: Aprendizaje Supervisado 15
3.3. Caso: Spotify API 16
3.4. Análisis Exploratorio 19
4. Resultados y Conclusiones....29
4.1. Resultados de Modelos....29
4.2. Análisis de Ingreso Potencial....38
5. Cierre y Mirada al Futuro....43
Referencias....46
Apéndice A. Scraping de Datos....48
Apéndice B. Misceláneo Datos y Modelos....52

Índice de Tablas

Tabla 1. Popularidad promedio de las canciones en base a cuartiles calculados por la variable *loudness*.....26
Tabla 2. Métricas de rendimiento entre los modelos de regresión lineal con aplicación de PCA y sin aplicación del mismo. 32
Tabla 3. Métricas de rendimiento entre los distintos modelos utilizados.....36
Tabla 4. Promedio de reproducciones en *Spotify* para las canciones por popularidad.....38
Tabla 5. Promedio de popularidad para las canciones según los cuartiles de la variable *loudness.* 39
Tabla 6. Promedio de popularidad para las canciones según los cuartiles de las distintas variables técnicas de audio. 40
Tabla 7. Promedio de popularidad para las canciones según los cuartiles de la variable *duration*. 40
Tabla 8. Promedio de popularidad para las canciones según el mes de lanzamiento. 41
Tabla 9. Rango o valor óptimo para cada variable significativa con su respectivo incremento potencial en ingresos. 42

Índice de Figuras

Figura 1. Pago por reproducción de las principales aplicaciones de streaming de música. Fuente: *Digital Music News* 13
Figura 2. Interacción entre la API de Spotify y el usuario mediante la utilización de la app. Fuente: *Spotify* API 16
Figura 3 Ejemplo de la lista de géneros para una canción específica. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.....21
Figura 4. Primera distribución del dataset en base a los distintos géneros. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.....22
Figura 5. Distribución del dataset en base a género después del balanceo de clases. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.....23
Figura 6. Popularidad promedio de las canciones por género. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*. 24
Figura 7. Mapa de calor con las correlaciones de las distintas variables numéricas del dataset. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.....25
Figura 8. Promedio de la variable *danceability* por género de las canciones. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify.....26
Figura 9. Promedio de la variable loudness por género de las canciones. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.....27
Figura 10. Promedio de la variable *loudness* por año de las canciones. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify. 28
Figura 11. Gráfico de "codo", es decir la variación *intra-cluster* por cantidad de segmentos usados en *k-means*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*. 30

Figura 12. Valores promedios de las variables técnicas mencionadas anteriormente para cada *cluster*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.....31
Figura 13. Porcentaje de la varianza explicada del dataset por la cantidad de dimensiones utilizados en *PCA*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*. 32
Figura 14. Resultados de la regresión lineal de las variables numéricas sobre la popularidad. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.....34
Figura 15. Hiperparámetros óptimos para el modelo XGBoost. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify. 36
Figura 16. Variables más significativas para la predicción para el modelo XGBoost. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify. 37
Figura 17. Error en el código por *timeout* de la conexión a la API de *Spotify*. Fuente: Consola del programa Visual Studio en base a código propio. 48
Figura 18. Código para correr indefinidamente el script para recolectar una canción aleatoriamente de la API de Spotify. Fuente: Elaboración propia.....49
Figura 19. Variación de la variable loudness por año divido por género musical. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify. 52
Figura 20. Distribución de la popularidad de las canciones por cuartiles en base a la variable loudness. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify. 53
Figura 21. Decisión del modelo para determinar el segmento para cada canción, visualizado con árboles de decisión. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify. 54
Figura 22. Decisión del modelo para determinar el segmento para cada canción y con las variables normalizadas, visualizado con árboles de decisión. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify. 54
Figura 23. Variación del rendimiento en el modelo de máquina de vector soporte al cambiar los hiperparámetros gamma y cost del mismo. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify. 55

1. Introducción

1.1. Contexto

En la última década, las compañías de plataformas han experimentado una gran expansión y se espera que sigan creciendo para la próxima. De hecho, se estima que el 70% del nuevo valor creado en la economía de los próximos diez años se basará en empresas de plataformas, explicado principalmente por la fuerte presencia de externalidades en ellas [1]. Además de su rápido crecimiento, otro aspecto distintivo de estas empresas de plataformas es su enfoque bilateral, lo que significa que operan en un mercado donde existen interacciones o transacciones entre dos grupos de usuarios distintos. Estos grupos suelen ser los proveedores de contenido o servicios y los consumidores que los utilizan. Esta dinámica bilateral crea un ecosistema único donde la generación de ingresos se vuelve más compleja pero también ofrece oportunidades innovadoras.

En respuesta a esta complejidad, han surgido diversas formas de monetización en estas plataformas, que van más allá de los modelos tradicionales. Los usuarios ahora tienen la posibilidad de participar activamente en la generación de ingresos a través de estrategias como publicidad, donaciones, suscripciones, comisiones por ventas o programas de afiliados. Estas nuevas formas de monetización ofrecen a los usuarios la oportunidad de aprovechar su contenido, audiencia o habilidades para obtener beneficios económicos.

1.2. Problema

El problema principal que aborda esta tesis se centra en la optimización de las variables de monetización en las plataformas digitales. Estas variables pueden ser cualquier forma por parte del usuario final de la plataforma de tener un ingreso monetario ya sea directa (como por ejemplo el pago por reproducción en una aplicación de *streaming* musical) como indirectamente (la cantidad de visitas en un video de *Youtube* puede llevar a

mayor publicidad en el canal y, por lo tanto, un mayor ingreso para el creador de contenido). A medida que estas plataformas continúan experimentando un crecimiento significativo, surge la necesidad de comprender y maximizar los ingresos para los usuarios que operan dentro de ellas.

La importancia de comprender y maximizar los ingresos para los usuarios de estas compañías radica en el hecho de que muchas de ellas se basan en modelos de negocio que involucran la participación activa de los usuarios en la generación de contenido, la promoción de productos o servicios, o la interacción con la audiencia. Estos usuarios son tanto creadores de contenido como consumidores, y su capacidad para obtener ingresos a partir de su participación en la plataforma es fundamental, en muchos casos, para su motivación, sostenibilidad y éxito a largo plazo.

Sin embargo, optimizar la variable de monetización en las plataformas presenta desafíos significativos. Las características y dinámicas propias de cada plataforma, así como las preferencias y comportamientos de los usuarios, influyen en la forma en que se pueden generar ingresos de manera efectiva. Además, la competencia en el entorno digital y la constante evolución de las tecnologías y estrategias de monetización requieren un enfoque integral y adaptativo para maximizar los beneficios para los usuarios.

Por tanto, el problema abordado en esta tesis se centra en desarrollar una metodología que permita comprender y optimizar las variables de monetización en las plataformas digitales, brindando a los usuarios herramientas y conocimientos para mejorar sus ingresos y aprovechar al máximo las oportunidades que estas plataformas ofrecen.

1.3. Objetivo

El objetivo principal de esta tesis es, específicamente, desarrollar una metodología que permita optimizar una variable de monetización específica en alguna plataforma digital, utilizando como caso de estudio información proveniente de la API de la plataforma de *streaming* musical *Spotify* y aplicando modelos de aprendizaje automático para predecir la popularidad de las canciones. Si bien el enfoque se centra en *Spotify*, este tipo de

análisis se podría replicar o adaptar a otras plataformas musicales o de contenidos audiovisuales con las modificaciones pertinentes.

Al lograr estos objetivos, se espera proporcionar a los usuarios de las plataformas digitales, en particular a aquellos involucrados en la producción y promoción de contenido musical, herramientas y conocimientos que les permitan tomar decisiones informadas e incrementar sus ingresos en base a la optimización de las variables más significativas para la predicción.

Esta tesis seguirá una estructura de una primera parte en la que se ahondará en el contexto de las plataformas hoy en día y se presentarán los distintos tipos de variables monetizables posibles a optimizar, así como también la metodología para realizar esto. Una segunda parte donde se presentará el caso de estudio mencionado anteriormente con la forma de recolección de estos datos y un análisis extensivo de los mismos, así como también la aplicación de los modelos probabilísticos para la predicción de la variable monetizable. Por último, habrá una tercera parte donde se analizarán los resultados de estos modelos y se efectuarán recomendaciones de negocio basados en los mismos para aumentar el ingreso en la plataforma.

2. Monetización de Plataformas

En este capítulo se presentarán primero los distintos *Key Performance Indicators* (KPIs) que pueden estar presentes en las plataformas para optimizar del punto de vista del usuario y después se ahondará en los distintos tipos de monetización presentes en las mismas.

2.1. KPIs

El crecimiento significativo de las compañías de plataformas en los últimos tiempos ha hecho que las mismas se desempeñen en diversas industrias con características muy diferentes entre ellas. Por lo tanto, la forma y el indicador que se desea optimizar (*KPI*) para maximizar los ingresos dentro de la plataforma dependerán de las características propias de dicha plataforma.

En esta tesis se centrará en aplicaciones de streaming de música (*Spotify, Tidal, Apple Music*, entre otros) donde la variable que mayormente explica el ingreso para el usuario es la cantidad de reproducciones de la canción (o podcast, álbum, etc.). Sin embargo, este mismo análisis se puede replicar para cualquier tipo de KPI que provea un ingreso ya sea directa o indirectamente para el usuario mientras se tengan disponibles los datos relevantes. Algunos ejemplos pueden ser los siguientes:

- Números de suscriptores/seguidores: Medir la cantidad de usuarios que siguen o se suscriben al contenido de la persona en la plataforma, lo cual puede llevar a una mayor visibilidad y, por consiguiente, potencialmente llevar también a un ingreso mayor mediante publicidades o canjes.
- Engagement rate: Medir el nivel de interacción que tienen los otros usuarios con el contenido con métricas como cantidad de “me gusta”, visitas, comentarios o compartidos.

- Click-through rate (CTR): Métrica que indica el porcentaje de usuarios que hacen click en la publicidad del contenido sobre la cantidad total de usuarios que lo consumieron.
- Conversion rate: Métrica similar a CTR pero en vez de medir el porcentaje de usuarios que hacen click sobre la publicidad, la misma indica la proporción de usuarios que realizaron una acción deseada como comprar un producto.

La métrica relevante a analizar y optimizar entonces va a depender fuertemente de cómo se estructura la plataforma a analizar y cómo es la monetización en la misma. Se verán varios casos y ejemplos en el apartado siguiente.

2.2. Monetización

En los denominados mercados bilaterales, la forma de rentabilizar el contenido por parte del usuario puede variar de gran manera según la naturaleza de la plataforma en sí. Algunas de las más comunes son:

- Publicidad: los usuarios pueden monetizar su contenido al permitir que se muestren anuncios junto a él, lo que les permite ganar una parte de los ingresos publicitarios generados. Este modelo es común en plataformas como *YouTube, Instagram, TikTok* y *Twitch*.
- Donaciones: los usuarios también pueden aceptar donaciones de sus seguidores o espectadores en plataformas como *Patreon, Ko-fi* y *PayPal*. Esto es comúnmente utilizado por creadores de contenido, artistas y músicos.
- Suscripciones: se ofrece contenido exclusivo a los seguidores mediante suscripciones mensuales. Este modelo se utiliza en plataformas como *Twitch* y *Patreon*.
- Comisiones de venta: se gana una comisión por la venta de productos o servicios en plataformas de comercio electrónico como *Amazon* y *eBay*.

- Programas de afiliados: los usuarios pueden ganar una comisión por referir a otros usuarios o clientes a una plataforma o producto específico en programas de afiliados. Este modelo es utilizado por muchas empresas de comercio electrónico y marketing de afiliación como *AirBnB*.

- Interacción/engagement con el contenido: se puede también cobrar directamente un monto por la interacción de los usuarios con el contenido ofrecido como por ejemplo la cantidad de visitas en el video en *Youtube* o la cantidad de reproducciones en las aplicaciones de streaming musicales. Esta tesis se centrará en este tipo de monetización para los usuarios en la plataforma.

Incluso dentro de un mismo tipo de monetización puede haber grandes diferencias en los montos que ofrecen las distintas compañías. Por ejemplo, *Youtube* paga alrededor de 18 dólares por cada mil reproducciones de un video mientras que la plataforma china *TikTok* tiene una bonificación de entre 2 y 4 centavos por cada mil reproducciones por video [2].

Como se mencionó anteriormente, esta tesis tomará como ejemplo concreto el caso de las plataformas de streaming musicales donde esto también ocurre. A modo de ilustración, se puede notar en la Figura 1 que hay mucha dispersión en el monto de pago por reproducción en las principales aplicaciones. *Spotify*, aplicación cuya información será utilizada como base para el estudio, por ejemplo, paga alrededor de 0,0033 dólares por reproducción mientras que *TIDAL* lo hace por 0,013 dólares. [3]

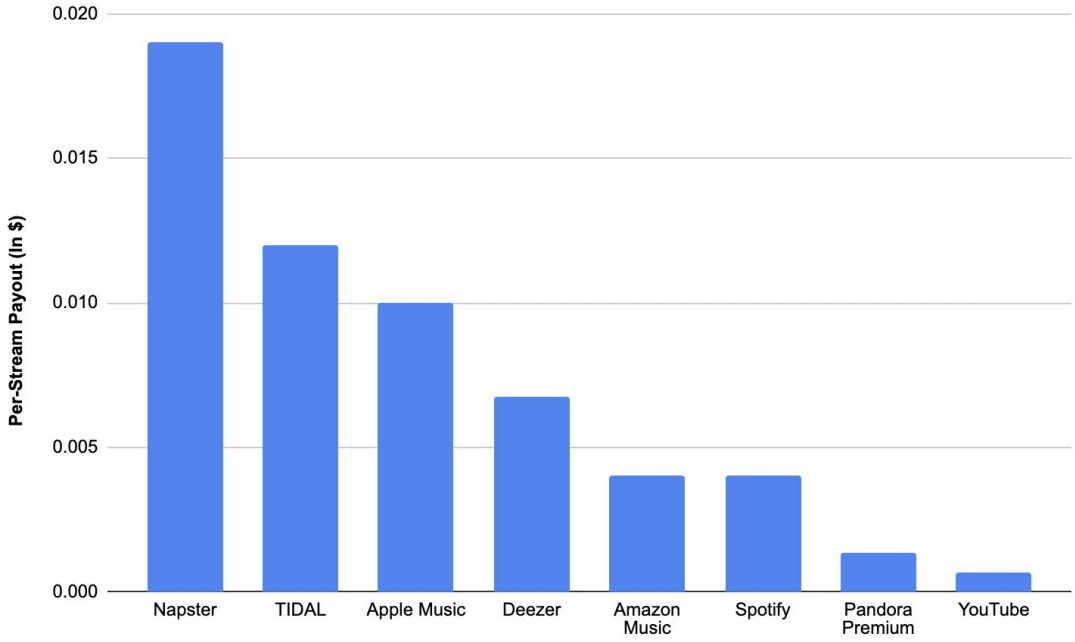

Figura 1. Pago por reproducción de las principales aplicaciones de streaming de música. Fuente: *Digital Music News*

Existe, entonces, una gran variedad de KPIs posibles a optimizar presentes en las plataformas y, a la vez, las mismas pueden tener distintas formas de monetización. En esta tesis se centrará, sin embargo, específicamente en la optimización de cantidad de reproducciones cuya monetización viene por un monto fijo por reproducción. En los siguientes capítulos se presentará la metodología utilizada para lograr esto y se presentará un caso de estudio especifico. Sin embargo, esta misma metodología podría ser replicada para otros casos basados en otros tipos de KPIs y formas de monetización que fueron abarcados anteriormente.

3. Metodología y Datos

Profundizando en lo visto en las secciones anteriores, este capítulo muestra la metodología de esta tesis con un ejemplo basado en un estudio de los datos de la API pública de *Spotify*. Se aplicarán métodos tanto de aprendizaje no supervisado para encontrar tendencias preliminares en los datos como de aprendizaje supervisado para poder determinar cuán determinantes son las variables disponibles (y si son determinantes cuáles son las más importantes) para poder predecir el KPI deseado. No se abordarán en esta tesis modelos basados en refuerzos debido a que solo se busca predecir una sola variable y no viene al caso cómo el modelo interactúa con el ambiente.

En este capítulo también se abordará el proceso de recolección de datos para el estudio, así como también una explicación de las variables finales obtenidas y un análisis exploratorio preliminar del dataset.

3.1. Tendencias Preliminares: Aprendizaje No Supervisado

Como se mencionó anteriormente, se aplicarán técnicas de aprendizaje no supervisado por dos motivos principales:

Debido a la cantidad de variables que pueden llegar a explicar la predicción del KPI deseado, es probable que en el dataset haya varios features para pasar a los modelos. Esto puede desencadenar en el denominado *curse of dimensionality* donde se encuentra una cantidad alta de dimensiones en el dataset y termina impactando en el rendimiento del modelo en los datos de validación [I].

De esta manera, se aplicará *Principal Component Analysis* para determinar si la varianza del dataset está concentrada en pocas componentes y si lo están entonces utilizar los datos con menores dimensiones para correr el modelo y observar si existe una mejora en términos de performance.

También se aplicará el método de *k-means* para calcular los centroides de cada cluster y ver cómo se divide la población en el espacio de atributos y poder encontrar segmentaciones con potencial interés para el análisis.

3.2. Importancia de Variables: Aprendizaje Supervisado

A su vez, se aplicarán técnicas de modelos probabilísticos para la predicción del KPI deseado (aprendizaje supervisado). De esta manera, se podrá determinar si las variables disponibles en el dataset son lo suficientemente significativas para realizar la predicción y en el caso de que lo fueran cuáles son las que más impacto tienen.

Si bien es deseable tener una buena performance a la hora de predecir la variable deseada, esta tesis se va a centrar para la predicción de los modelos solamente en variables donde la persona que quiera maximizar su ingreso en la plataforma pueda tener incidencia. Por ejemplo, más adelante se introducirá el estudio hecho a base de los datos de la API pública de *Spotify*. En ese caso, no se incluirán para la predicción variables categóricas como el nombre del artista, aunque puedan ser bastante incisivas a la hora de predecir en el modelo (una variable categórica basada en el nombre del artista por ejemplo sería sumamente importante para predecir la popularidad de una canción, pero no es una variable accionable en términos de negocio).

Para la predicción final se utilizarán diversos modelos de aprendizaje automático y se compararán los rendimientos de los mismos posteriormente:

- **Boosting (XGBoost)**
- **Random Forest**
- **Redes Neuronales**
- **Support Vector Machine**
- **Regresión Lineal**
- **Regresión Lineal con dimensionalidad reducida mediante PCA**

Esos modelos fueron elegidos por la naturaleza del problema a abarcar en esta tesis y su eficiencia va a variar de gran manera según cómo está compuesto el dataset y cuál es

la variable a predecir. En esta tesis se verá en el inciso siguiente que se aplicará a un problema de regresión con una variable numérica para predecir, los resultados y la eficiencia de los modelos muy probablemente varíen si la variable a predecir es una categórica, por ejemplo.

3.3. Caso: Spotify API

Para ejemplificar lo anterior se va a desarrollar un análisis sobre la popularidad de las canciones con datos proveídos por la API pública de *Spotify* para ver qué tan potentes son las variables de la misma en la predicción de la popularidad de la canción y cuáles de las variables son las más significativas al momento de predecir la popularidad y poder aumentar los ingresos. Dicha API puede ser accedida por cualquier usuario con una cuenta premium mediante la página de *Spotify for Developers* [4]. Para ello se creó una app que tiene su respectivo ID del cliente y permite la comunicación directa con la API:

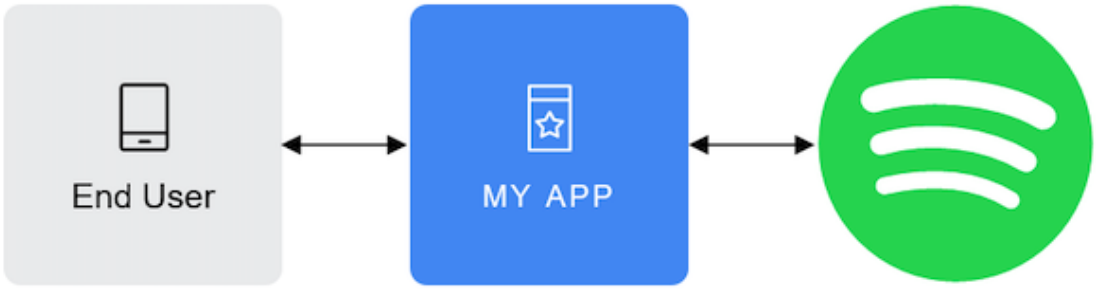

Figura 2. Interacción entre la API de Spotify y el usuario mediante la utilización de la app. Fuente: *Spotify* API

Se decidió utilizar la API de *Spotify* por su fácil acceso y simpleza de uso, como además de la información que se encuentra disponible en la misma. Esta API tiene información técnica pertinente a la producción musical de las canciones que va a ser muy útil para el análisis en cuestión. Sin embargo, este mismo análisis se podría replicar con información de otras plataformas de streaming musicales como *Tidal* o *Apple Music*.

Dicho esto, se desarrolló un código en lenguaje *Python*[1] para poder acceder a ella mediante la utilización de la librería *Spotipy* [5]. Tanto el código completo como el desarrollo del mismo y las distintas librerías utilizadas para la construcción final del dataset son profundizados en el Apéndice A de esta tesis.

De esta manera, se pudo construir un dataset final con 210.926 canciones, cada una con las siguientes variables:

- **artist_name:** Nombre del artista.
- **album_name:** Nombre del álbum al cual pertenece la canción.
- **track_id:** ID de Spotify de la canción, esta variable se va a utilizar como un índice.
- **track_name:** Nombre de la canción.
- **duration:** Duración de la canción en milisegundos.
- **genre:** Géneros correspondiente a la canción.
- **release_date:** La API de Spotify no provee información acerca de la fecha de lanzamiento de la canción en sí, sino que solo provee la fecha de lanzamiento del álbum, para simplificar el análisis se va a asumir que ambas son equivalentes.
- **danceability:** Indicador con valor de 0 a 1 que indica cuán "bailable" es la canción.
- **key:** Indicador con valor de 0 a 11 que indica la escala en la que está compuesta la canción.
- **loudness:** Nivel del ruido de la canción en términos de decibeles, generalmente fluctúa en valores de -60 a 0.
- **mode:** Indicador binario que determina la modalidad de la escala de la canción (0 es escala menor y 1 es escala mayor).
- **time_signature:** Indicador que determina la signatura de compás de la canción. La misma fluctúa entre rangos de 3 a 7 indicando por ejemplo un tiempo de "3/4" o "7/4" para la canción.
- **speechiness:** Indicador con valor de 0 a 1 que indica cuán "a capela" es la canción.

[1] Disponible en el repositorio de *Github* mencionado anteriormente (https://github.com/Ag-Ah/SpotifyAPI)

- **acousticness:** Indicador con valor de 0 a 1 que indica cuán "acústica" es la canción.
- **instrumentalness:** Indicador con valor de 0 a 1 que indica cuán "instrumental" es la canción.
- **liveness:** Indicador con valor de 0 a 1 que indica cuán "en vivo" es la canción.
- **valence:** Indicador con valor de 0 a 1 que indica cuán "positiva" es la canción.
- **tempo:** El tempo de la canción en *beats per minute*, generalmente fluctúa en valores de 50 a 150.
- **primary_color:** Color principal de la tapa del álbum de la canción en formato hex.
- **color_palette:** Paleta de colores de la tapa del álbum de la canción en formato hex.
- **popularity:** Variable que tiene un rango de 0 a 100 y está basada en la cantidad de reproducciones totales que tiene la canción ponderando con mayor importancia a aquellas que fueron recientes. Mientras más alta sea esta variable, mayor número de reproducciones tiene la canción. Por ejemplo, todas las canciones en el top 10 global tienen una *popularity* de 100. Esta va a ser la variable que se va a utilizar para la predicción.

La variable que se va a utilizar para la predicción final y que va a ser parámetro para los resultados de los modelos no va a ser, entonces, la cantidad de reproducciones absolutas de la canción sino otra variable que se encuentra disponible en la API de *Spotify* (*popularity*) y que está correlacionada con la cantidad de reproducciones de la misma. Como se mencionó anteriormente, esta variable pondera con mayor importancia aquellas reproducciones más recientes (la aplicación no especifica en el glosario de la API cómo está realizada la ponderación exactamente), lo cual es sumamente útil para el análisis al poder controlar por el factor temporal (una canción podría tener más reproducciones que otra simplemente porque se encuentra disponible para escuchar en la aplicación desde antes).

Todas estas variables provienen directamente desde la API de Spotify salvo las variables *primary_color* y *color_palette.* Para obtener ambas variables se utilizó la librería *colorthief* [6] de Python que permite de manera simple conseguir mediante la función *get_color()* el color principal y mediante la función *get_palette()* la paleta de colores en

formato hex de una imagen que se le provea (en este caso la URL de la tapa del álbum de la canción que fue extraída de la API).

Cabe destacar que ya existen trabajos académicos que utilizaron datos de la API pública de Spotify o similares aplicados a modelos de aprendizaje automático. En el paper [II] se utilizaron datos de la API pública de la página *Last.fm* para poder predecir datos demográficos de las personas en base a su historial de música escuchada. Para los papers [III] y [IV] sí se utilizaron datos provenientes de la API de Spotify (específicamente las variables técnicas de audio) de las canciones para predecir si la misma iba a ser un hit (en base a si aparece en el ranking de la revista *Billboard* o no).

También se utilizaron datos de la misma API para el paper [V] donde se hizo un análisis sobre la relación de las variables técnicas con la cantidad de reproducciones de la canción mediante una regresión lineal. Ese mismo análisis fue expandido mediante la utilización de otros modelos *(SVM, Random Forest, Redes Neuronales*, etc.) al abarcar al problema tanto como clasificación en los papers [VI], [VII], [VIII] y [IX] como con regresión en el paper [X].

Sin embargo, en ninguno de los estudios académicos mencionados anteriormente se efectuó una recomendación de negocio en base a los resultados de los modelos, sino que todos se concentran meramente en si las variables técnicas de audio son suficientes como variables para explicar el éxito de una canción.

3.4. Análisis Exploratorio

Una vez ya teniendo el dataset completo, se procedió a analizar los tipos de variables y sus respectivas categorías. A la vez, se crearon nuevas variables donde se consideró necesario:

Variables Numéricas:

- **duration**
- **danceability**
- **loudness**

- **speechiness**
- **acousticness**
- **instrumentalness**
- **liveness**
- **valence**
- **tempo**
- **time_signature**
- **popularity:** Variable de predicción.

Variables Binarias:

- **mode**

Variables Categóricas:

- **genre_final:** Versión simplificada de la variable *genre*.
- **key:** La escala está representada por un número por lo cual fue automáticamente identificada como una variable numérica al importar el dataset. Sin embargo, esta es una variable discreta, por lo cual fue transformada para ser categórica para determinar si las canciones de cierta escala son más determinantes para determinar la popularidad de las mismas.
- **month**: mes de la variable *release_date*.
- **month_day:** fecha del año de la variable *release_date*.
- **primary_color**
- **color_palette**

Variables de caracteres y fecha (no utilizados para la predicción):

- **artist_name**
- **album_name**
- **track_id**
- **track_name**
- **release_date**
- **genre**

Con lo que respecta a la variable *genre*, la información proveída por la API representa una lista de subgéneros para la misma canción:

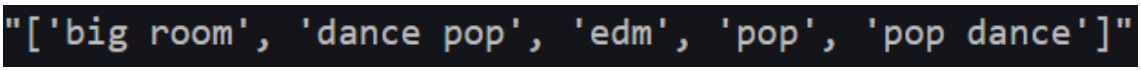

Figura 3 Ejemplo de la lista de géneros para una canción específica. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Debido a la gran cantidad existente de estos subgéneros y para simplificar el análisis, se va a resumir estos géneros en uno de los siguientes:

- Pop
- Rock
- Metal
- Electrónica/Dance
- Hip-Hop/Rap
- Alternativo/misceláneo

Para esto se introdujo en el código una búsqueda de palabras en la lista de subgéneros para clasificar la canción en alguno de los géneros mencionados anteriormente. La lógica se hizo de la siguiente manera, empezando por aquellos géneros que considero menos populares para tenerlo equilibrado lo más que se pueda (una lista de subgéneros puede tener más de una de estas palabras):

- “Electrónica” o “EDM” = “Electrónica”
- “Metal” = “Metal”
- “Hip-Hop” o “Rap” = “Hip-Hop”
- “Rock” = “Rock”
- “Pop” = “Pop”

Esta lógica se cumple si la palabra aparece completa o parcialmente, por ejemplo, los subgéneros como *dream pop* o *k-pop* caerían bajo el umbral de *Pop*. Si una lista no contiene ninguna de las palabras anteriores entonces la canción pertenece a la categoría de *Alternativo*.

La distribución final en cuanto a los géneros terminó siendo la siguiente:

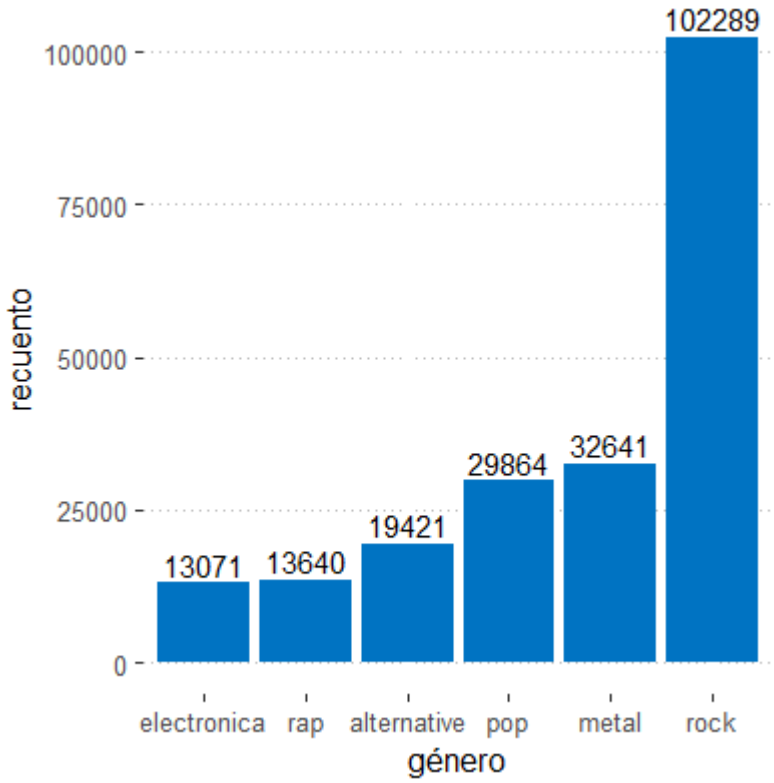

Figura 4. Primera distribución del dataset en base a los distintos géneros. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Como se puede notar, incluso teniendo mayor consideración por los géneros menos populares a la hora de hacer la clasificación, el dataset termina siendo desbalanceado y sesgado para el género *rock*. Para evitar este sesgo y que esté más balanceado el dataset, se descartaron las filas de las canciones con género *rock* hasta que coincidieran con el monto total de las canciones con género *metal* (32.641):

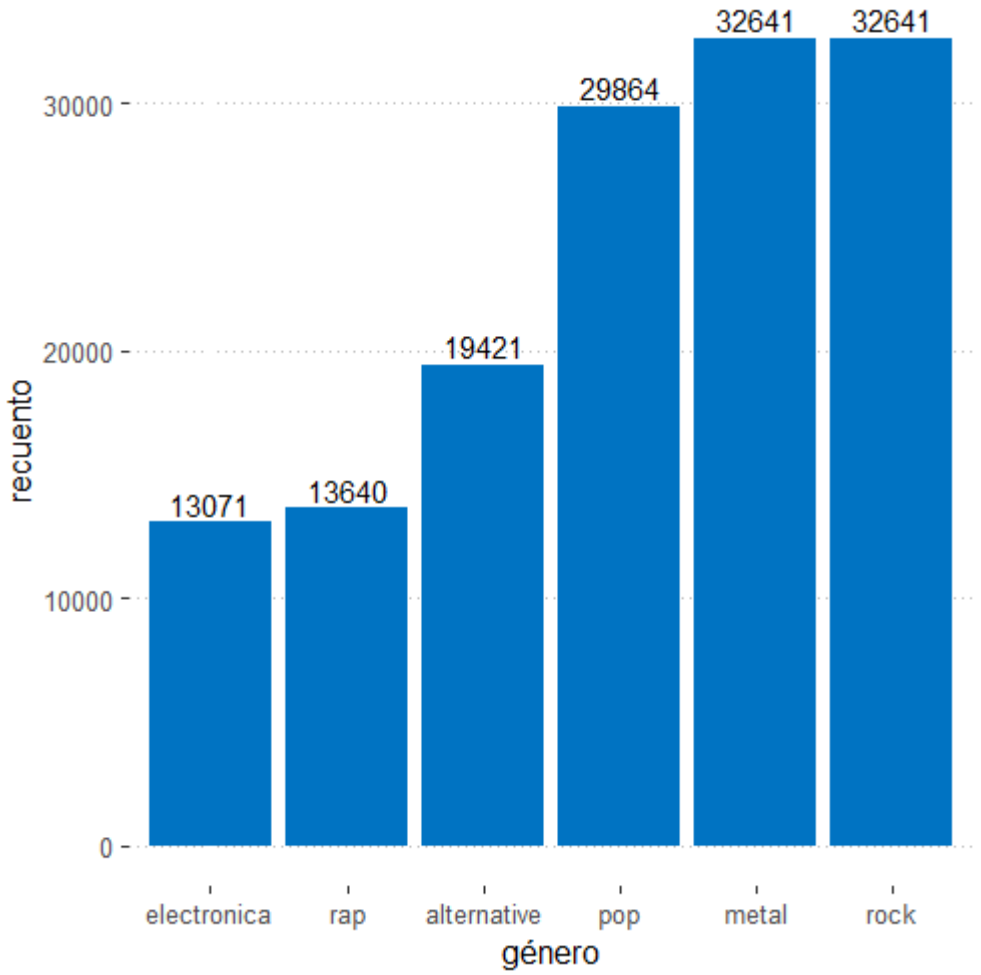

Figura 5. Distribución del dataset en base a género después del balanceo de clases. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

De esta manera, de las 210.926 canciones originales del dataset quedó un subgrupo de 141.278 con clases más balanceadas.

Sorprendentemente, cuando se ven los géneros contra sus respectivos promedios en términos de popularidad, los géneros *rock*, *metal* y *pop* tienen valores muy cercanos y *rap* se posiciona en primer lugar:

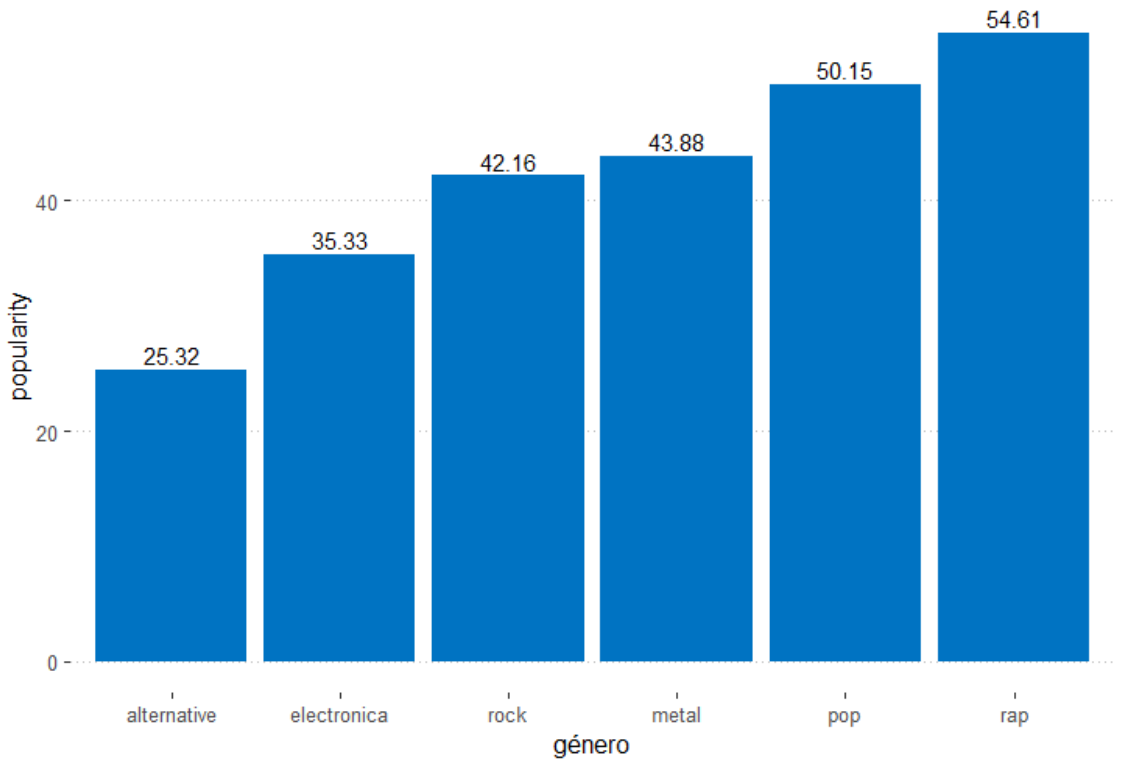

Figura 6. Popularidad promedio de las canciones por género. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Con respecto a las variables técnicas, se graficó un mapa de calor con todas las correlaciones entre dichas variables y *popularity*. La más correlacionadas son las variables *loudness* y *danceability* con una relación positiva y la variable *instrumentalness* con una relación negativa:

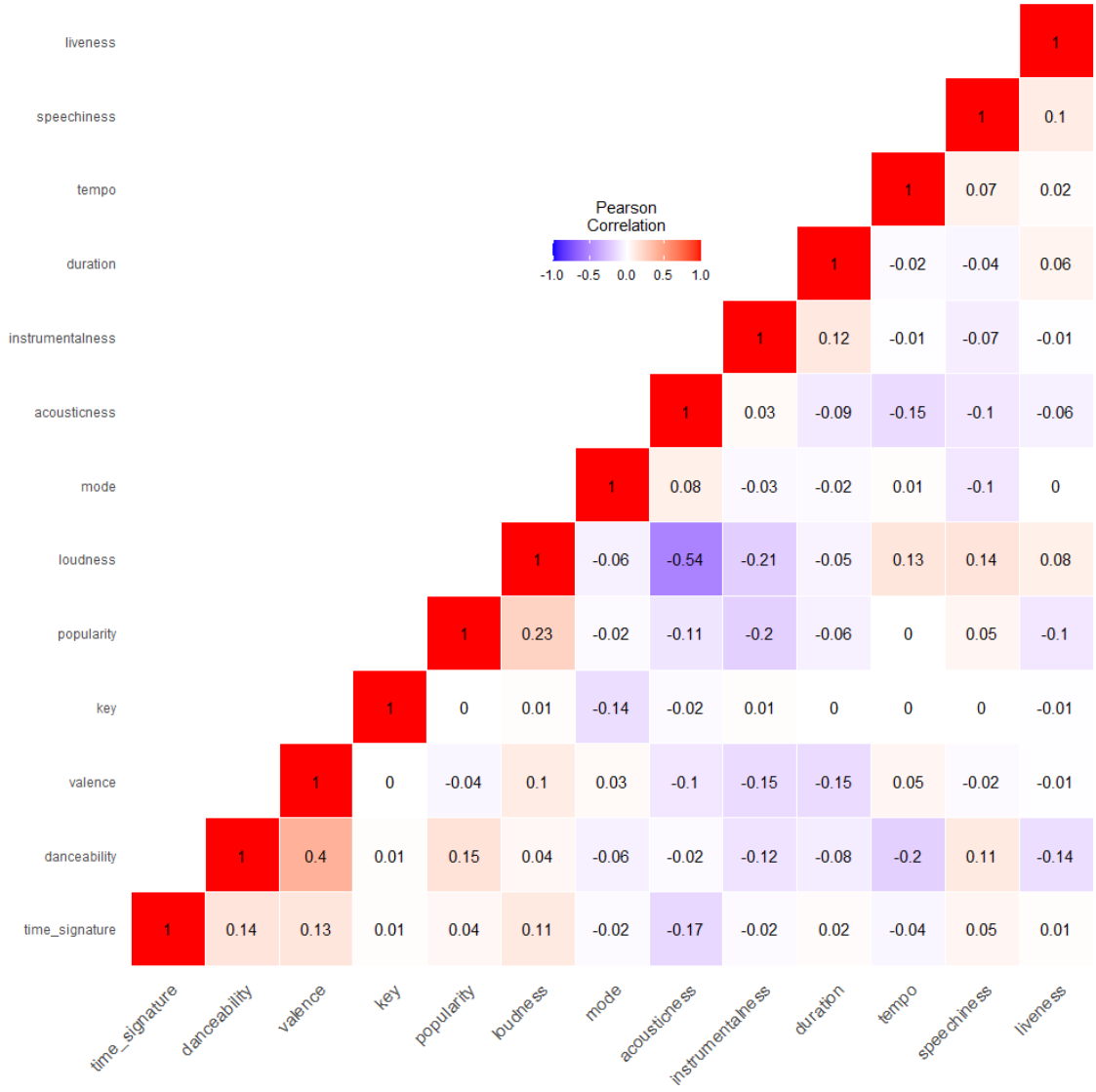

Figura 7. Mapa de calor con las correlaciones de las distintas variables numéricas del dataset. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Esto se puede corroborar para la variable *loudness* al dividir a la población total en cuatro grupos en base a su percentil en dichos valores. Con valores más altos de esa variable, la popularidad promedio de la canción es más alta:

Cuartil	Popularidad Promedio
0-25%	32,99550
25-50%	39,54838
50-75%	45,99066
75-100%	50,73245

Tabla 1. Popularidad promedio de las canciones en base a cuartiles calculados por la variable *loudness.*

También se puede corroborar viendo el promedio de la variable *danceability* por género, como se mostró anteriormente el género *rap* era el que tenía mayor promedio en términos de la variable *popularity* y esto también se cumple para el valor promedio de *danceability*:

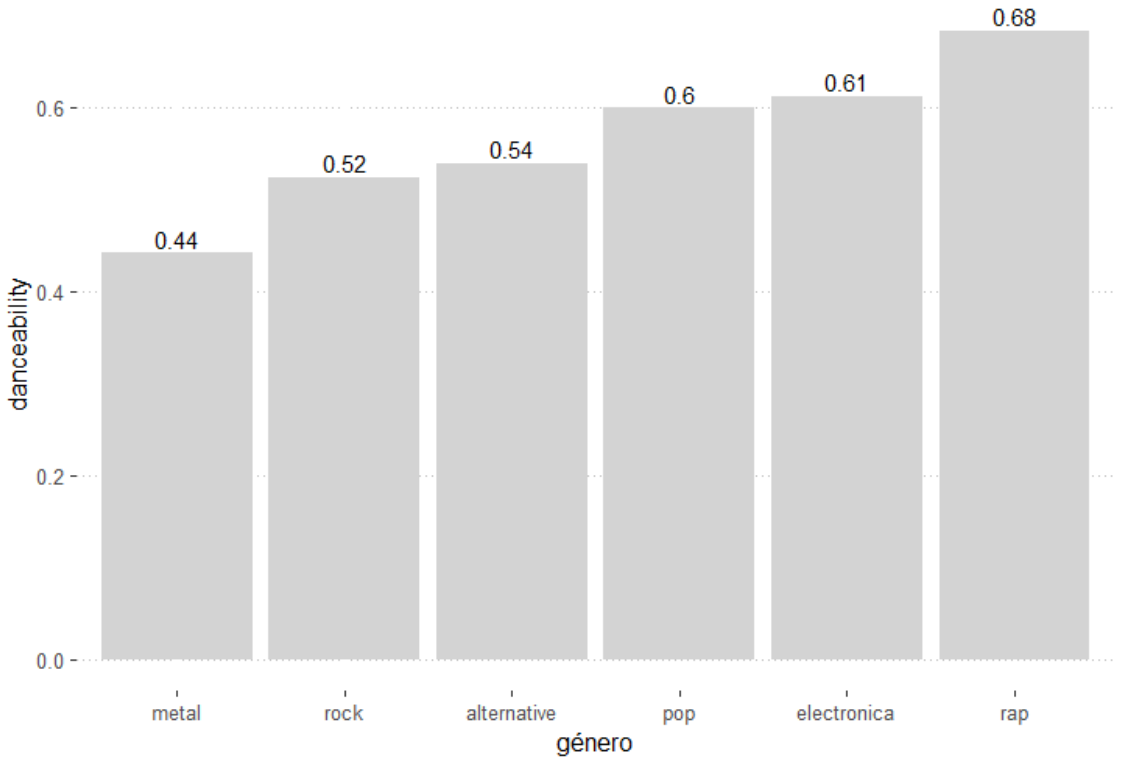

Figura 8. Promedio de la variable *danceability* por género de las canciones. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify.

Lo mismo sucede para la variable *loudness*, donde sorprendentemente los niveles sonoros de las canciones del género *rap* son mayores en promedio que los del género *metal*:

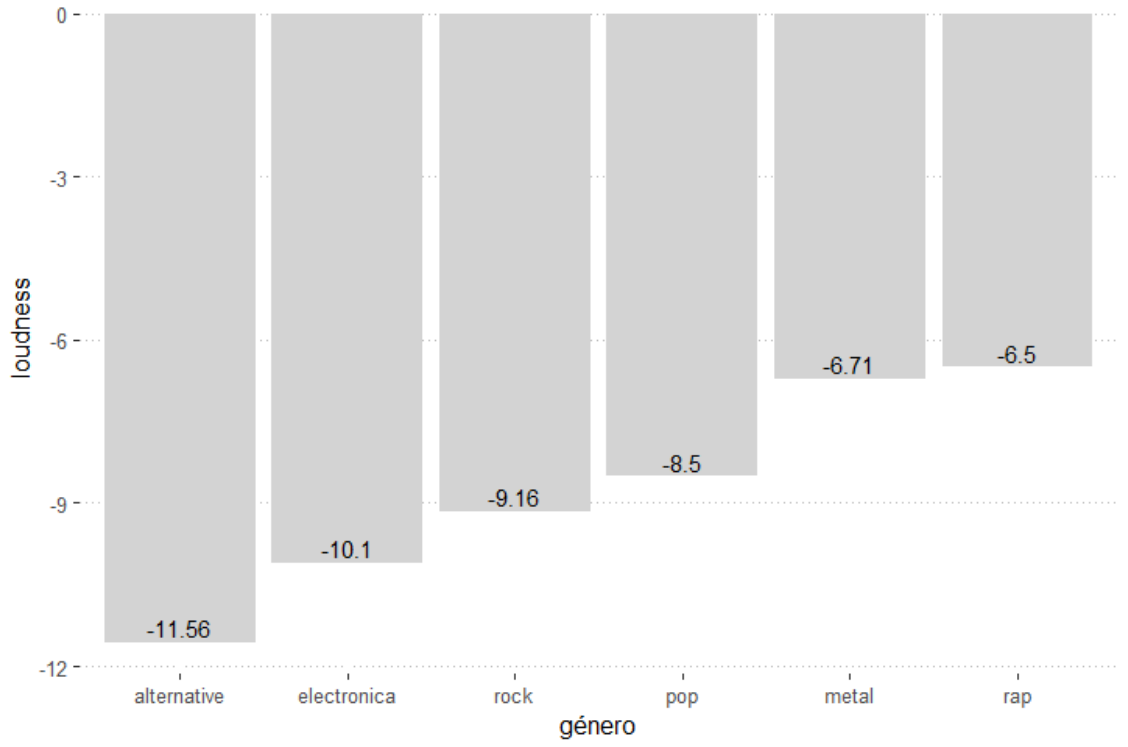

Figura 9. Promedio de la variable loudness por género de las canciones. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

También se puede notar una clara tendencia sobre mayores niveles sonoros en la producción de las canciones a través de los años. El pico se encuentra a mediados de la década de los 2000s, lo cual coincide con la crítica muy prevalente en esa época denominada "Loudness War" sobre el bajo rango dinámico y el gran nivel de ruido de las canciones en la producción [7].

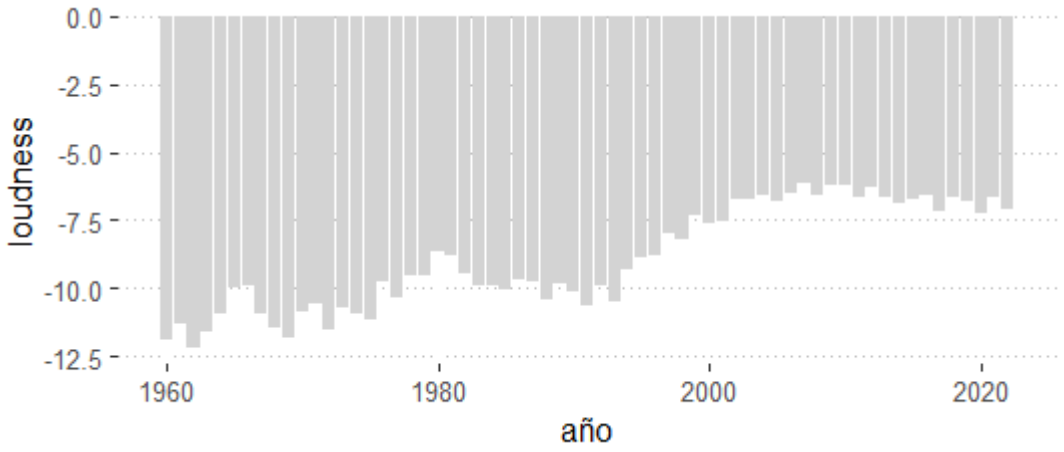

Figura 10. Promedio de la variable *loudness* por año de las canciones. Fuente: Elaboración propia en base a datos proveídos por la API de Spotify.

4. Resultados y Conclusiones

Una vez transformadas las variables y teniendo el dataset completo para efectuar las predicciones, este capítulo se centrará en los resultados proveídos por los modelos probabilísticos mencionados anteriormente, así como también en el análisis de ingreso potencial en base a estos resultados.

4.1. Resultados de Modelos

Primero se utilizaron los modelos de aprendizaje no supervisado mencionados en el capítulo 3 para encontrar cómo se distribuye la varianza en los datos del dataset mediante la utilización de *Principal Component Analysis* y también para encontrar subgrupos homogéneos mediante el cálculo de centroides con *K-Means*.

Para dicho cálculo se utilizaron solo las variables técnicas (*danceability, loudness, acousticness,* etc.) y se utilizó un bucle para saber la cantidad de *clusters* a utilizar, comparando la variación *intra-cluster* para cada una de ellas:

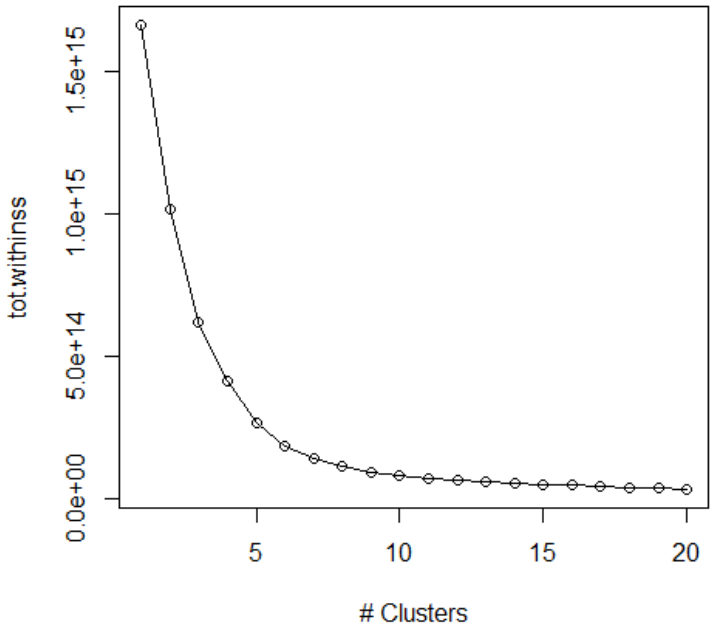

Figura 11. Gráfico de "codo", es decir la variación *intra-cluster* por cantidad de segmentos usados en *k-means*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Si utilizamos el criterio del "codo" [XI] para determinar la cantidad de *clusters* a utilizar, se puede notar en la Figura 11 que el mismo se encuentra cuando se utilizan alrededor de 5 segmentos, por lo cual este será el número por el cual se dividirá el dataset. Sin normalizar las variables, la decisión tomada por el modelo para dividir los segmentos estaba concentrada exclusivamente en la duración de la canción, esto se debe a que dicha variable fluctúa en rangos numéricos muchos más altos que el resto de las variables numéricas. Al normalizar todas las variables y volver a calcular los centroides, esta vez varias de las variables técnicas de audio *(mode, acousticness, speechiness, valence y danceability)* fueron utilizadas para realizar el corte.

Se puede notar la diferencia de las medias entre las variables anteriores (cada una de ellas normalizadas) para cada *cluster*:

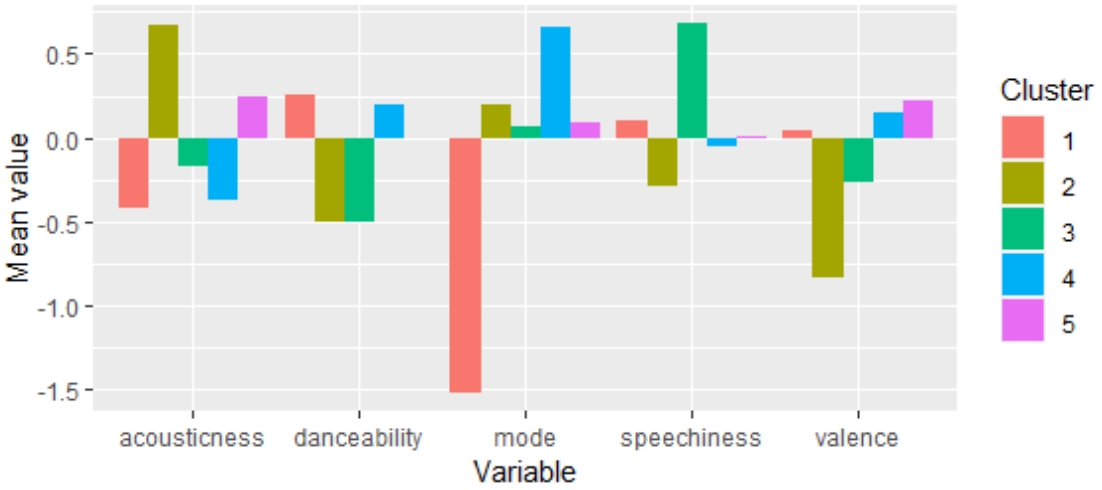

Figura 12. Valores promedios de las variables técnicas mencionadas anteriormente para cada *cluster*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

De esta forma, quedan bien definidas las cinco segmentaciones:

- En el primer *cluster* se encuentran aquellas canciones con bajos niveles acústicos, altos niveles de *danceability* y escalas menores. Canciones de electrónica o *dancepop* por ejemplo tienden a caer en este segmento.
- En el segundo *cluster* caen aquellas canciones con altos niveles acústicos y pocos niveles de *danceability y valence*. En este segmento caerían baladas y canciones folclóricas por ejemplo.
- En el tercer *cluster* se concentran aquellas canciones con bajos niveles acústicos y de *danceability,* pero mantienen en promedio altos niveles de *speechiness.* En este segmento se tenderían a ver canciones de *spoken word* o acapella.
- El cuarto segmento es muy similar al segundo pero las canciones del mismo se producen en promedio con escalas mayores.
- En el quinto *cluster* por su lado se encuentran aquellas canciones con valores cercanos a los promedios para cada una de las variables.

Posteriormente, se utilizó *Principal Component Analysis* para determinar si la varianza de los datos está concentrada en pocas componentes y si conviene reducir la dimensionalidad del dataset para correr los modelos probabilísticos:

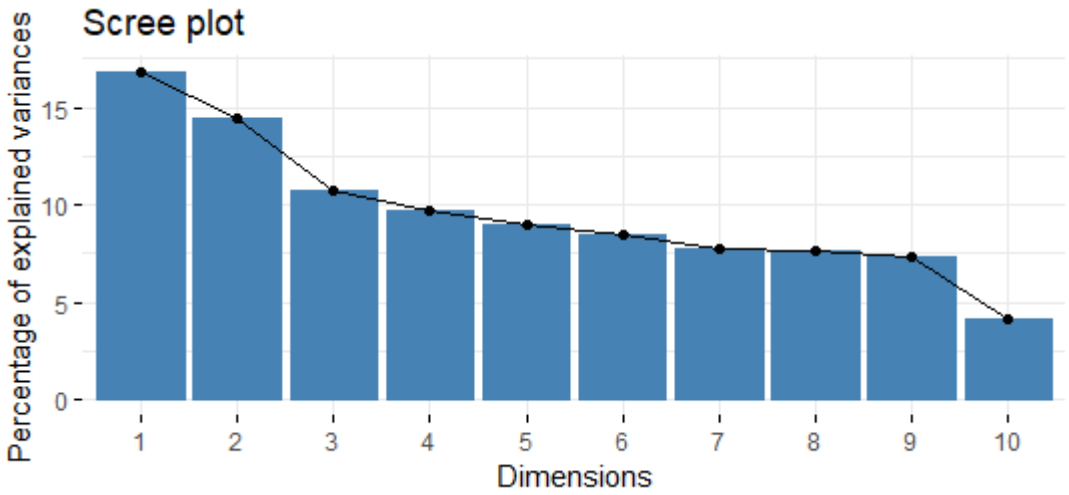

Figura 13. Porcentaje de la varianza explicada del dataset por la cantidad de dimensiones utilizados en *PCA*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Se puede notar por la Figura 13 entonces que no hay una concentración de la varianza en pocas componentes (un 50% de la varianza del dataset es explicada recién por cuatro dimensiones distintas). Esto difiere de los resultados encontrados en el paper [VIII] donde la varianza acumulada del dataset estaba concentrada mayormente en dos componentes principales y las mismas fueron utilizadas en una regresión lineal para la predicción. Se concluye entonces que incluso aunque se estén utilizando datos de la misma API para construir el dataset, los resultados pueden variar de gran manera.

De todas formas, se utilizaron ocho componentes distintas para predecir la popularidad de las canciones mediante una regresión lineal. También se realizó la misma predicción con las variables normalizadas pero sin reducir la dimensionalidad de los datos, los resultados fueron los siguientes en los datos de validación (un 20% aleatorio del dataset):

Modelo	ME	MAE	RMSE
Regresión Lineal con PCA	1,573754	15,25844	18,89279
Regresión Lineal sin PCA	1,411435	14,8411	18,50966

Tabla 2. Métricas de rendimiento entre los modelos de regresión lineal con aplicación de PCA y sin aplicación del mismo.

Como era de esperarse, no hay una variación significativa en los resultados de ambos modelos (incluso las métricas son un poco peores para cuando sí se utiliza PCA). Para el resto de los modelos, entonces, no se reducirá la dimensión de los datos por dos razones principales:

- Como muestra la tabla 2, no hay una mejora en el rendimiento del modelo cuando se utiliza PCA.
- Si se utiliza PCA se estaría perdiendo la interpretabilidad de las variables más significativas las cuales son sumamente importantes para el desarrollo de esta tesis.

En cuanto a la regresión lineal sin reducción de dimensionalidad, se obtuvieron los siguientes resultados al correrla para las variables numéricas:

	Dependent variable:
	popularity
danceability	25.446***
	(0.356)
loudness	1.186***
	(0.014)
speechiness	-18.027***
	(0.699)
acousticness	-2.616***
	(0.192)
instrumentalness	-13.419***
	(0.189)
liveness	-9.941***
	(0.288)
valence	-14.211***
	(0.225)
tempo	0.013***
	(0.002)
duration	-0.00000***
	(0.00000)
time_signature	0.653***
	(0.138)
Constant	49.002***
	(0.637)
Observations	148,739
R^2	0.155
Adjusted R^2	0.155
Residual Std. Error	18.510 (df = 148728)
F Statistic	2,730.219*** (df = 10; 148728)
Note:	*p<0.1; **p<0.05; ***p<0.01

Figura 14. Resultados de la regresión lineal de las variables numéricas sobre la popularidad. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Se puede notar por la Figura 14 que todos los coeficientes obtenidos para las variables numéricas son estadísticamente significativos al 99% y también aclara cómo afecta a la variable dependiente (*popularity*) el cambio en una unidad de cada variable independiente:

- Las variables *danceability, loudness, tempo y time_signature* tienen una correlación positiva con la popularidad de la canción.
- Las variables *speechiness, acousticness, instrumentalness, liveness, valence* y *duration* poseen, por su lado, una correlación negativa con la popularidad.

También se concluye que estas variables explican de manera insuficiente la varianza de la variable *popularity* al tener un R^2 considerablemente bajo (0.155), lo cual coincide con el estudio realizado en el paper [IV]. Estos resultados se mantienen equivalentes al escalar las variables.

Como se mencionó anteriormente, se utilizó una simple división de un 80% aleatorio del dataset como datos de entrenamiento y el 20% restante como datos de validación (todas las métricas de los resultados van a ser sobre este último grupo solamente) que se aplicará a todos los modelos.

Una vez dividido el dataset, se procedió a utilizar los otros modelos, adaptando las variables para cada modelo donde sea necesario y optimizando los hiperparámetros mediante una búsqueda aleatoria que es más robusto a la presencia de hiperparámetros irrelevantes y puede acelerar el proceso de búsqueda [XI]. De esta forma, se predijo la popularidad para cada canción con los siguientes modelos y sus respectivos parámetros:

- **Random Forest:** Por validación cruzada se decidió usar un modelo de *Random Forest* con treinta nodos máximos y ocho variables de muestra aleatoria para cada corte.
- **Redes Neuronales:** Se utilizó un perceptrón multicapa con dos capas intermedias y *learning rate* de 0,1 con las variables escaladas.
- **Support Vector Machine:** Se utilizó un *Kernel* lineal con las variables escaladas y una constante de regularización de valor 4.

Sin embargo, el modelo basado en boosting de árboles llamado *XGBoost* terminó siendo, en comparación con los otros modelos, el de mejor rendimiento en los datos de validación según las métricas de *MAE* y *RMSE* con una significancia estadística del 95%:

Modelo	MAE	RMSE
XGBoost	5,906444 (5,86-5,95)	9,489906 (9,41-9,57)
Random Forest	13,72603 (13,6-13,8)	16,89526 (16,8-17,0)
Redes Neuronales	14,62614 (14,5-14,7)	17,90954 (17,8-18,1)
Support Vector Machine	14,18489 (14,1-14,3)	17,93921 (17,8-18,1)
Regresión Lineal con PCA	15,25844 (15,1-15,4)	18,89279 (18,7-19,0)
Regresión Lineal sin PCA	14,8411 (14,7-15,0)	18.50966 (18,4-18,7)

Tabla 3. Métricas de rendimiento entre los distintos modelos utilizados.

En la tabla 3 se puede notar un resumen de los resultados de todos los distintos modelos mencionados anteriormente y sus respectivas métricas de rendimiento. El modelo *XGBoost*, entonces, es el más potente de todos al predecir la popularidad de la canción en los datos de validación utilizando solamente datos de la API de la aplicación Spotify o derivados de la misma. En este caso, los hiperparámetros óptimos para el modelo y las variables más importantes para la predicción final fueron los siguientes (estas variables van a ser profundizadas en el apartado siguiente):

```
max_depth eta gamma colsample_bytree subsample min_child_weight
        7 0.4   0.7              0.4         1                2
```

Figura 15. Hiperparámetros óptimos para el modelo *XGBoost*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

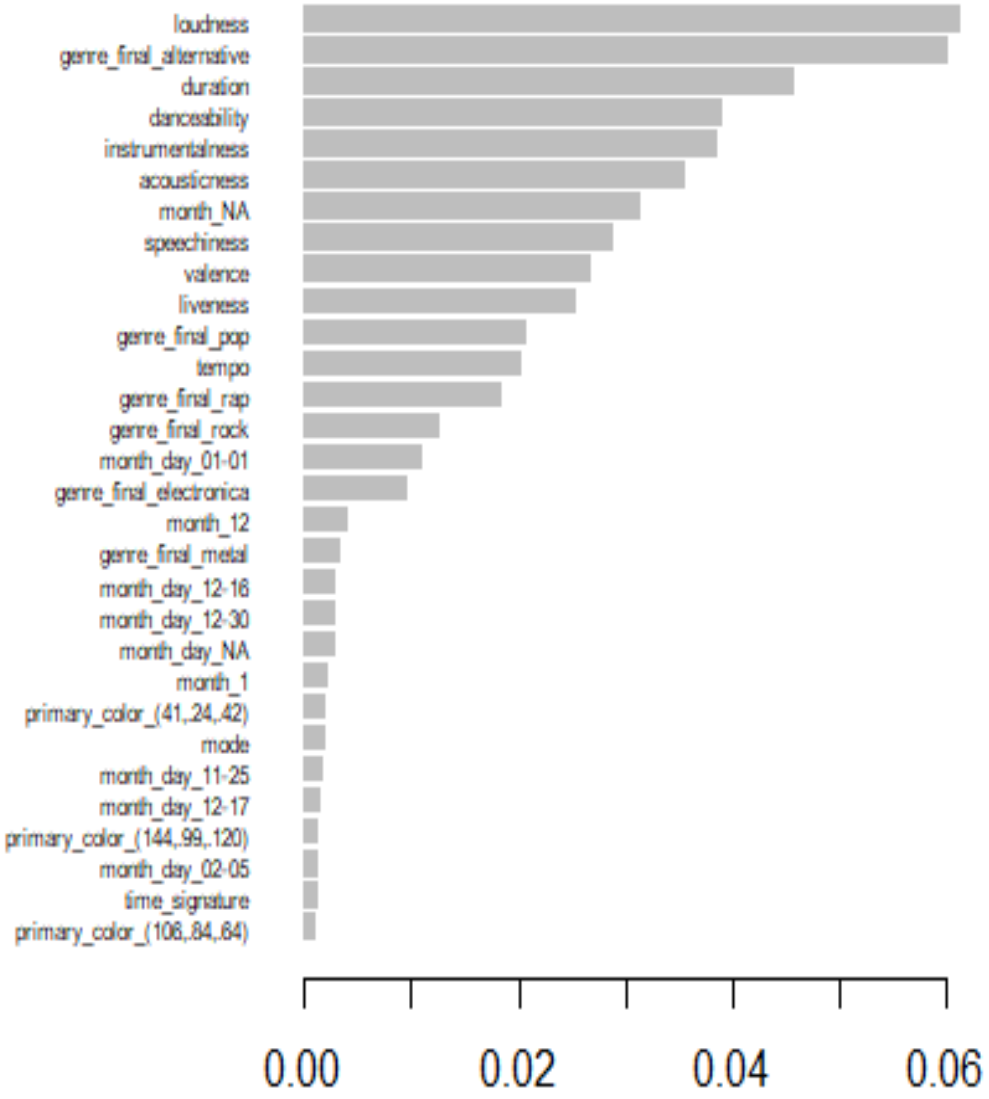

Figura 16. Variables más significativas para la predicción para el modelo *XGBoost*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

4.2. Análisis de Ingreso Potencial

Con la figura 16, entonces, se pueden notar aquellas variables más significativas a la hora de realizar la predicción para el modelo *XGBoost*. De esta manera, también se obtienen cuáles acciones tomar para aumentar la probabilidad de tener una mayor cantidad de reproducciones y, por consiguiente, un mayor ingreso.

Al no tener disponible la cantidad de reproducciones exactas de las canciones (solo la variable *popularity* se encuentra disponible en la API), se determinará de manera aproximada a cuántas reproducciones equivale el cambio marginal de la variable *popularity* en una unidad. Para ello, se tomó el promedio de la cantidad de reproducciones en la aplicación de Spotify de diez canciones distintas para los valores de *popularity* entre 45 y 55. Cabe destacar que se tuvieron en cuenta para el promedio solo aquellas canciones cuyo año de lanzamiento fue en el 2022 ya que el factor tiempo puede sesgar los valores al tener la variable una ponderación mayor por las reproducciones más recientes (una canción que se encuentra disponible desde hace muchos años en la aplicación puede tener una gran cantidad de reproducciones pero que pocas de ellas sean recientes por lo cual puede llevar a que se sobreestime el promedio si se incluye en el cálculo):

Popularidad	Promedio Reproducciones
45	7.820.844
46	8.463.552
47	13.275.549
48	14.558.677
49	15.072.144
50	16.713.902
51	18.671.911
52	21.891.731
53	24.734.892
54	25.898.103
55	27.483.300

Tabla 4. Promedio de reproducciones en *Spotify* para las canciones por popularidad.

Si se calcula la diferencia entre la cantidad de reproducciones para cada grupo, el promedio de todas estas se encuentra en aproximadamente dos millones de reproducciones. Dicha diferencia fue calculada para estos valores de la variable *popularity* porque se encuentran en los valores medios de la variable y son más representativos para el análisis. Estos mismos números varían en gran medida para canciones con popularidad en valores más extremos, por ejemplo, aquellas canciones con popularidad 99 o 100 superan en promedio los miles de millones de reproducciones y para aquellas con popularidad 0 o 1 ni siquiera hay información disponible de la cantidad de reproducciones en la aplicación.

Para simplificar el análisis entonces, se asumirá que con el logro de aumentar la popularidad de la canción en una unidad se traducirá en un incremento en promedio de dos millones de reproducciones. *Spotify* paga una tarifa fija de 0,0033 dólares por reproducción como se aclaró en el capítulo 2, por lo cual se traduciría a un incremento de $6.600 dólares en el ingreso promedio. Con esto, se analizará cómo cambia la popularidad promedio de la canción en base a cambios en las variables más determinantes para la predicción realizada por el modelo *XGBoost.*

En el capítulo anterior ya se analizó cómo cambia el promedio de la popularidad de la canción en base a sus valores (cuartiles) de la variable *loudness,* la cual fue la más significativa para la predicción según el modelo (esto coincide con la figura 7 donde se puede notar que dicha variable es la que tiene mayor correlación con la popularidad de la canción):

Rango Valor *Loudness*	Popularidad Promedio
(-39,103, -11,32)	32,99550
(-11,32, -8,125)	39,54838
(-8,125, -5,673)	45,99066
(-5,673,0)	50,73245

Tabla 5. Promedio de popularidad para las canciones según los cuartiles de la variable *loudness.*

La tendencia de cómo sube la popularidad de la canción a mayores niveles sonoros se puede ver claramente, aquellas canciones con decibeles mayores a -5 dbFS tienen una popularidad promedio de alrededor de 18 puntos mayor que aquellas en los rangos más bajos. Siguiendo el criterio anterior, esto significaría un aumento de $118.800 dólares de ingresos en promedio al producir la canción con mayores niveles sonoros (en comparación con los valores en los rangos más bajos).

Lo mismo se analizó para cada una de las variables técnicas de audio, dividiendo sus valores en cuartiles y calculando la popularidad promedio para cada grupo:

Rango Valor	Popularidad Promedio *Danceability*	Popularidad Promedio *Instrumentalness*	Popularidad Promedio *Acousticness*	Popularidad Promedio *Speechiness*	Popularidad Promedio *Valence*	Popularidad Promedio *Liveness*
(0-0,25)	37,70802	46,40087	43,21881	43,46550	42,22516	43,28031
(0,25-0,50)	42,15065	45,17045	44,74239	41,43968	43,46462	43,33245
(0,50-0,75)	43,55385	42,05796	43,79027	41,52408	42,59578	42,21259
(0,75-1)	45,68853	35,43472	37,30680	42,62414	40,79482	40,22663

Tabla 6. Promedio de popularidad para las canciones según los cuartiles de las distintas variables técnicas de audio.

Las variables de la duración de la canción y el mes de lanzamiento también fueron de las más significativas para la predicción y se calcularon los mismos valores para ellas:

Rango Valor *Duration*	Popularidad Promedio
(14.080, 178.307)	37,81551
(178.316, 218,013)	44,76950
(218.014, 263.947)	46,04592
(263.948,4.571.800)	40,43679

Tabla 7. Promedio de popularidad para las canciones según los cuartiles de la variable *duration*.

Mes de Lanzamiento	Popularidad Promedio
Enero	39,82317
Febrero	43,28920
Marzo	44,73152
Abril	43,98835
Mayo	46,23962
Junio	46,30135
Julio	44,70163
Agosto	46,81708
Septiembre	45,88857
Octubre	47,50532
Noviembre	46,43127
Diciembre	39,97594

Tabla 8. Promedio de popularidad para las canciones según el mes de lanzamiento.

En el resultado de la Figura 16 también se puede observar que tres categorías del color primario de la tapa del álbum de la canción tuvieron impacto en la predicción del modelo. Sin embargo, se analizaron estas categorías y en los tres casos se correspondían a valores atípicos dentro del dataset. Por ejemplo, la variable *primary_color(41,24,42)* se encuentra en el gráfico debido a que existen diferentes canciones en el dataset correspondiente al álbum *Midnights* de *Taylor Swift* y todas tienen una popularidad muy alta.

Con las tablas anteriores entonces se obtiene el rango o valor óptimo de las variables más significativas para aumentar la cantidad de reproducciones y obtener un mayor ingreso potencial si se parte del rango menos óptimo:

Variable	Rango o Valor Óptimo	Ingreso Potencial (USD)
Loudness	(-5,673,0)	$118.800
Danceability	(0,75-1)	$52.800
Instrumentalness	(0-0,25)	$72.600
Acousticness	(0,25-0,50)	$49.038
Speechiness	(0-0,25)	$13.200
Valence	(0,25-0,50)	$17.622
Liveness	(0,25-0,50)	$15.180
Duration	(218.014, 263.947)	$54.318
Mes	Octubre	$63.954

Tabla 9. Rango o valor óptimo para cada variable significativa con su respectivo incremento potencial en ingresos.

De esta manera, al producir una nueva canción y subirla a la aplicación de *Spotify* las variables técnicas de producción (y la duración y el mes de lanzamiento de la canción) deberían idealmente caer en esos rangos para maximizar la probabilidad del aumento del ingreso. La popularidad de una canción no está solamente explicada por estas variables sin embargo, por lo cual para poder medir el impacto verdadero que poseen las mismas se podría realizar un test A/B con las canciones en las aplicaciones. Si bien no hay una función disponible hoy en día dentro de la aplicación de *Spotify* para publicar una canción con ciertas variables para un determinado grupo de usuarios y la misma con diferentes atributos para otros usuarios (A/B testing tradicional), se podría publicar dos veces la misma canción con el mismo título pero con distintos valores en sus variables técnicas de audio y comparar la cantidad de reproducciones entre ambas después de un tiempo determinado.

5. Cierre y Mirada al Futuro

Desde mediados de la década de los 2000, el formato físico para escuchar música ha disminuido lentamente en favor de alternativas más cómodas para el usuario final como son las aplicaciones de streaming. Si bien fue un cambio positivo para el consumidor, esto ha sido un problema para los artistas o empresas discográficas que fueron afectados en términos de ingresos teniendo en cuenta la poca paga por reproducción de las aplicaciones de streaming. Para dimensionar esto, solo un 1% de todos los artistas en Spotify explican el 80% de todas las reproducciones de la aplicación [8]. Esto derivó en que el soporte económico de muchos artistas y bandas sea la venta de mercadería, los shows en vivo y las regalías por usar sus canciones en otros medios como películas o videojuegos [9].

Hoy en día ya existen papers y estudios utilizando la información proveída por las aplicaciones de streaming de música para distintos propósitos. Sin embargo, la mayoría de ellos están enfocados en un solo aspecto ya sea utilizando los datos para un problema de aprendizaje no supervisado (estudios relacionados a clasificación de playlists y visualización de los datos por ejemplo) o para un problema de aprendizaje supervisado (el paper *SpotiPred* por ejemplo que fue referido anteriormente), raramente combinando ambos enfoques. En mucho de ellos también el análisis está hecho de forma básica desarrollando los modelos, pero no explicando de forma sustancial las conclusiones (como en el caso de *Prediction of an Artist's Success on Spotify* que también fue mencionado anteriormente) y en ninguno de ellos se puede encontrar recomendaciones o consejos en términos de negocio.

Bajo este contexto, los artistas o las bandas casi tienen que descartar el ingreso proveniente de las reproducciones en las plataformas de streaming y enfocarse en otras incursiones como las que se mencionaron anteriormente. Sin embargo, el excesivo foco en los tours como fuente principal de ingreso para las bandas o artistas está trayendo problemas para la industria. Cada vez es más común la cancelación de giras por problemas de salud mental o cansancio debido a la sobre exigencia de los artistas para ser redituables [10]. También significó una desventaja muy grande estos dos últimos

años por la cuarentena donde este ingreso principal se vio pausado y muchos músicos tuvieron que dejar la industria por esta consecuencia [11].

Con lo introducido anteriormente, el objetivo principal de esta tesis era, entonces, desarrollar una metodología que permita optimizar alguna variable de monetización por parte del usuario en las plataformas digitales. Se tomó como caso concreto la optimización de variables técnicas de producción musical hecho a base de un estudio con datos proveídos de una plataforma de *streaming*, lo cual puede ser un accionable para aumentar los ingresos en la plataforma para un usuario final como puede llegar a ser un artista o una empresa discográfica. Este análisis está hecho a partir de datos públicos proveídos por la API de la plataforma *Spotify* pero sería válido para cualquier otra plataforma de streaming musical como *Apple Music* o *Tidal*.

El criterio de éxito era, entonces, que el modelo tenga buen desempeño utilizando solo las variables accionables para poder predecir de manera correcta la popularidad y determinar las variables más importantes y así poder determinar de manera correcta el potencial aumento de ingreso para los artistas o bandas (en términos aproximados ya que no se va a estar prediciendo exactamente la cantidad de reproducciones).

Se pudo ver que utilizando mayormente las variables disponibles en la API el modelo de *XGBoost* tuvo un buen rendimiento, por lo cual se puede concluir que se pueden tomar acciones para aumentar la probabilidad de recibir un ingreso mayor de una forma eficiente en las aplicaciones de streaming musicales solamente utilizando variables técnicas de producción musical. Claro está que existen muchísimos factores que explican la popularidad de la canción más allá de las variables técnicas de la producción, más que nada la popularidad ya existente del artista a la hora de producir la misma u otras variables relacionadas al marketing y la publicidad. Por esto mismo, este estudio no se centra en el rendimiento de los modelos en sí ni en la optimización de los mismos agregando variables, sino que se basa en saber si cambiando estas variables que conllevan un costo más bajo al no estar relacionadas con la publicidad de forma directa, se puede lograr un ingreso mayor en la plataforma.

De esta manera, se determinó cuáles de las variables fueron las más determinantes a la hora de predecir la popularidad de la canción y, en base a estos resultados, se indicó el

rango óptimo para cada una de ellas para aumentar la cantidad de reproducciones y, en consecuencia, incrementar también el ingreso. Para el caso de las canciones con una popularidad promedio, por ejemplo, este incremento de ingreso al cambiar el valor de una de las variables técnicas de producción podía llegar a valores por arriba de 100 mil dólares.

Cabe destacar que no es la intención de este trabajo incentivar a la homogeneización de la producción en la industria musical y llegar a una eventual sobresaturación en el mercado. Si bien, probablemente, el ingreso mayor que ostenta el artista o la compañía al optimizar las variables de producción musical no sea suficiente para revertir por completo la situación descripta anteriormente y que puedan tener a la paga por reproducciones en streaming como principal ingreso, esta tesis busca poder mitigar esto lo mayormente posible mostrando que se puede lograr aumentar la ganancia solamente modificando las variables técnicas de producción y no tener que depender solamente de gastos publicitarios.

Vale la pena mencionar también que este estudio está hecho con datos de una ventana de tiempo específica y que la industria musical, así como las tendencias en las preferencias del público, se encuentra en un dinamismo constante. Por esto mismo, los resultados presentados en esa tesis podrían cambiar de gran manera con el paso del tiempo y si se utiliza esta misma metodología se recomienda ir actualizando los datos de entrenamiento para captar estas nuevas tendencias o cambios en la industria (como nuevos géneros o técnicas de producción musical por ejemplo).

En futuras investigaciones o planes comerciales se podría replicar la metodología de este estudio con un enfoque distinto en términos de lo que ofrece la plataforma, por ejemplo, si se desea empezar un podcast se podría adaptar el dataset para obtener datos relevantes de podcasts (esto está disponible en la misma API de Spotify) y aplicar modelos de aprendizaje automático para determinar las variables accionables más significativas. A su vez, este estudio se podría adaptar con datos provenientes de APIs de otras plataformas, ya sea el mismo KPI a optimizar o no, e incluso se podría expandir para utilizar datos no estructurados y tomar el audio completo de las canciones como entrada para los modelos por ejemplo.

Referencias

Papers Académicos y Libros

[I] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning with Applications in R* (pp. 245-246).

[II] Krismayer, T., Schedl, M., Knees, P., et al. (2019). *Predicting user demographics from music listening information. Multimedia Tools and Applications, 78, 2897-2920.*

[III] Georgieva, E., Suta, M., & Burton, N. (2013). *Hitpredict: Predicting Hit Songs Using Spotify Data. Stanford Computer Science 229: Machine Learning.*

[IV] Middlebrook, K., & Shaik, K. (2019). *SONG HIT PREDICTION: PREDICTING BILLBOARD HITS USING SPOTIFY DATA.*

[V] Nijkamp, R. (2018). *Prediction of product success: explaining song popularity by audio features from Spotify data.*

[VI] Chen, Y., Dixit, A., Sanyal, S., et al. (Year). *Show Me What You Got: Song Popularity Prediction Using FMA Dataset.*

[VII] Shete, A., Mohanani, N., & Gondal, S. (2021). *Prediction of an Artist's Success on Spotify.*

[VIII] Reiman, M., & Ornell, P. (Year). *Predicting Hit Songs with Machine Learning.*

[IX] Gulmatico, J. S., Susa, J. A. B., Malbog, M. A. F., Acoba, A., Nipas, M. D., & Mindoro, J. N. (2022). *SpotiPred: A Machine Learning Approach Prediction of Spotify Music Popularity by Audio Features.*

[X] Jetzinger, F., Heumer, F., & Schedl, M. (2017). *Towards Predicting the Popularity of Music Artists.*

[XI] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning with Applications in R (p. 388).*

[XII] Bergstra, J., & Bengio, Y. (2012). *Random Search for Hyper-Parameter Optimization.*

Artículos Periodísticos y Páginas

[1] Financial Times. (2023). *The rise of the platform economy.*

[2] Search Engine Journal. (2022). *TikTok vs. YouTube: Which Is Better For You?*

[3] Medium. (2021). *This Is What Music Streaming Services Pay Artists.*

[4] Spotify Developers. Spotify API. URL: https://developer.spotify.com/documentation/general/guides/

[5] Spotipy Library Documentation. URL: https://spotipy.readthedocs.io/en/2.22.1/

[6] Colorthief Library Documentation. (n.d.). URL: https://github.com/fengsp/color-thief-py

[7] Granville-Fall, N. (2016). *The Mastering Loudness War: Can The Effects of Hyper-Compression and Increasing Loudness in Commercially Released and Broadcast Music Be Reduced?*

[8] The Guardian. (2021). *Odds Are Against You, The Problem With The Music Streaming Boom.*

[9] The Guardian. (2018). *How Do Musicians Make Money?*

[10] The Guardian. (2022). *This Should Not Be Normalised: Why Musicians Are Cancelling Tours to Protect Their Mental Health.*

[11] The Guardian. (2020). *Third of British musicians may quit industry amid pandemic.*

[12] spotipy-random Library Documentation. URL: https://pypi.org/project/spotipy-random/

[13] Forever Library Documentation. URL: https://pypi.org/project/forever/

Apéndice A. Scraping de Datos

La API es muy accesible para acceder a datos ya conocidos como playlists o consultar directamente por una canción con su *track_id* respectivo. Sin embargo, a la hora de tratar de acceder información sobre canciones de forma aleatoria, las funciones disponibles en la librería principal no son recomendables.

Primero, se intentó utilizar la función *sp_search()* con filtro como *track* en la categoría. Esta función permite extraer información de hasta 50 canciones por vez. Sin embargo, la misma no distingue las canciones aleatoriamente, sino que trae información sobre aquellas que están en el top más escuchado globalmente. Como la página web de Spotify solo muestra 50 canciones a la vez en la pestaña principal, el límite del tamaño del lote para extraer la información es de 50 pero esto puede ser evitado modificando el valor del *offset* de la función.

Para no sesgar los datos y permitir tener canciones en el dataset con popularidad baja se tuvo que recurrir a la librería *spotipy_random* [12]. Esta misma permite buscar aleatoriamente un objeto dentro de la API (álbum, canción, artista, etc.) y con diversos filtros como género, año y un filtro booleano llamado *hipster_tag* que permite traer específicamente canciones con popularidad baja. Esta función también permite bajar la información en lotes con el filtro de *limit.*

Si bien la librería *spotipy_random* es muy útil para generar un dataset bien balanceado al tener un filtro disponible para extraer canciones con popularidad baja, esta misma es propensa a perder su conexión con la API (más allá del límite diario de peticiones que tiene la API misma) y los errores de la misma en consecuencia son comunes al no estar trayendo información:

```
  File "c:\Users\aahedo\Downloads\from spotipy import Spotify.py", line 47, in <module>
    random_pop_song_json: str = get_random(spotify=sp, type="track", limit= 1, tag_hipster = random.cho
ice(random_boolean), genre = genero, year=random.randrange(1960,2023,1))
  File "C:\Users\aahedo\AppData\Local\Programs\Python\Python310\lib\site-packages\spotipy_random\spotip
y_random.py", line 97, in get_random
    element: dict = result[random_type_result_key]["items"][0]
IndexError: list index out of range
```

Figura 17. Error en el código por *timeout* de la conexión a la API de *Spotify*. Fuente: Consola del programa Visual Studio en base a código propio.

Para eludir esto se desarrolló otro código que corre directamente desde la terminal con la librería *forever* [13] en Python. Esta misma permite correr el código indefinidamente y reiniciarlo en caso de que se produzca un error:

```
C:\Users\aahedo>python -m forever.run -t 3 -i 3 python -u -m "SpotifyAPI.py"
```

Figura 18. Código para correr indefinidamente el script para recolectar una canción aleatoriamente de la API de *Spotify*. Fuente: Elaboración propia.

Con todo lo anterior, se presenta el código completo y comentado que se utilizó para la extracción de los datos de la API de *Spotify* y la posterior construcción del dataset:

```
import spotipy
from spotipy.oauth2 import SpotifyClientCredentials
import sys
import time

if sys.version_info < (3, 0):
    from urllib2 import urlopen
else:
    from urllib.request import urlopen

import io
import random
import pandas as pd
from colorthief import ColorThief
from spotipy_random import get_random

#Insertar cid y secret en base a los valores proveídos por
la app en la API de Spotify
cid = 'yourcid'
secret = 'yoursecret'

#Conexión a la API y a la librería Spotipy con las
credenciales correspondientes
client_credentials_manager=
SpotifyClientCredentials(client_id=cid,
client_secret=secret)
sp = spotipy.Spotify(client_credentials_manager =
client_credentials_manager)
```

```
#Loop para extraer información pertinente a la canción y
agregarla a la lista
for i in range(0,1000000):
    #Lista de géneros para la búsqueda que se va a
determinar de forma aleatoria con la función get_random()
    random_genre =
    ["pop","rock","metal","punk","electronic","alternative
    ","dance","folk","indie"]
    #Lista booleana para el tag_hipster de la función
get_random()
    random_boolean = [True, False]
    #Se definen las listas para almacenar la información y
posteriormente agregarla al dataset
    artist_name = []
    track_name = []
    popularity = []
    track_id = []
    album_name = []
    danceability = []
    key = []
    loudness = []
    mode = []
    speechiness = []
    acousticness = []
    instrumentalness = []
    liveness = []
    valence = []
    tempo = []
    duration = []
    time_signature = []
    genre = []
    release_date = []
    primary_color = []
    color_palette = []
    #Se extrae la información de la canción aleatoria con
la función get_random()
    random_pop_song_json: str = get_random(spotify=sp,
    type="track", limit= 1, tag_hipster =
    random.choice(random_boolean), genre =
    random.choice(random_genre),
    year=random.randrange(1960,2023,1))
    #Una vez extraída la información se agrega a la lista
que corresponda
    artist_name.append(random_pop_song_json['artists'][0][
    'name'])
    album_name.append(random_pop_song_json['album']['name'
    ])
    track_name.append(random_pop_song_json['name'])
    track_id.append(random_pop_song_json['id'])
    popularity.append(random_pop_song_json['popularity'])
```

```
            features: str =
         sp.audio_features(random_pop_song_json['uri'])
        genres: str =
        sp.artist(random_pop_song_json['artists'][0]['external
        _urls']['spotify'])
        album: str =
        sp.album(random_pop_song_json["album"]["external_urls"
        ]["spotify"])
        release_date.append(album["release_date"])
        genre.append(genres['genres'])
        danceability.append(features[0]['danceability'])
        key.append(features[0]['key'])
        loudness.append(features[0]['loudness'])
        mode.append(features[0]['mode'])
        speechiness.append(features[0]['speechiness'])
        acousticness.append(features[0]['acousticness'])
         instrumentalness.append(features[0]['instrumentalness'
         ])
        liveness.append(features[0]['liveness'])
        valence.append(features[0]['valence'])
        tempo.append(features[0]['tempo'])
        duration.append(features[0]['duration_ms'])
        time_signature.append(features[0]['time_signature'])
        #Se extrae la URL de la tapa del álbum de la canción y
    se identifica el color primario así como la paleta entera
    de colores del mismo
        fd =
        urlopen(random_pop_song_json["album"]["images"][0]["ur
        l"])
        f = io.BytesIO(fd.read())
        color_thief = ColorThief(f)
        primary_color.append(color_thief.get_color(quality=1))
         color_palette.append(color_thief.get_palette(quality=1
         ))
        #Se juntan las distintas listas en un solo dataframe
    que es exportado como CSV posteriormente
        track_dataframe = pd.DataFrame({'artist_name' :
        artist_name, 'album_name' : album_name, 'track_id' :
        track_id, 'track_name' : track_name, 'popularity' :
        popularity, 'danceability' : danceability, 'key' :
        key,'loudness' : loudness,'mode' : mode,'speechiness'
        : speechiness,'acousticness' :
        acousticness,'instrumentalness' :
        instrumentalness,'liveness' : liveness,'valence' :
        valence,'tempo' : tempo,'duration' :
        duration,'time_signature' : time_signature,'genre' :
        genre,'release_date' : release_date,'primary_color' :
        primary_color,'color_palette' : color_palette})
        track_dataframe.to_csv('Dataset_Spotify.csv',mode="a",
        header=False,index=False)
```

Apéndice B. Misceláneo Datos y Modelos

Con respecto a lo mencionado en la Figura 10 sobre la tendencia de mayores niveles sonoros en la producción a través de los años (también conocido como "Loudness War"), es interesante notar que lo mismo se mantiene para todos los géneros y no solo para aquellos considerados como "pesados" o "ruidosos" como pueden llegar a ser el *rock* o el *metal:*

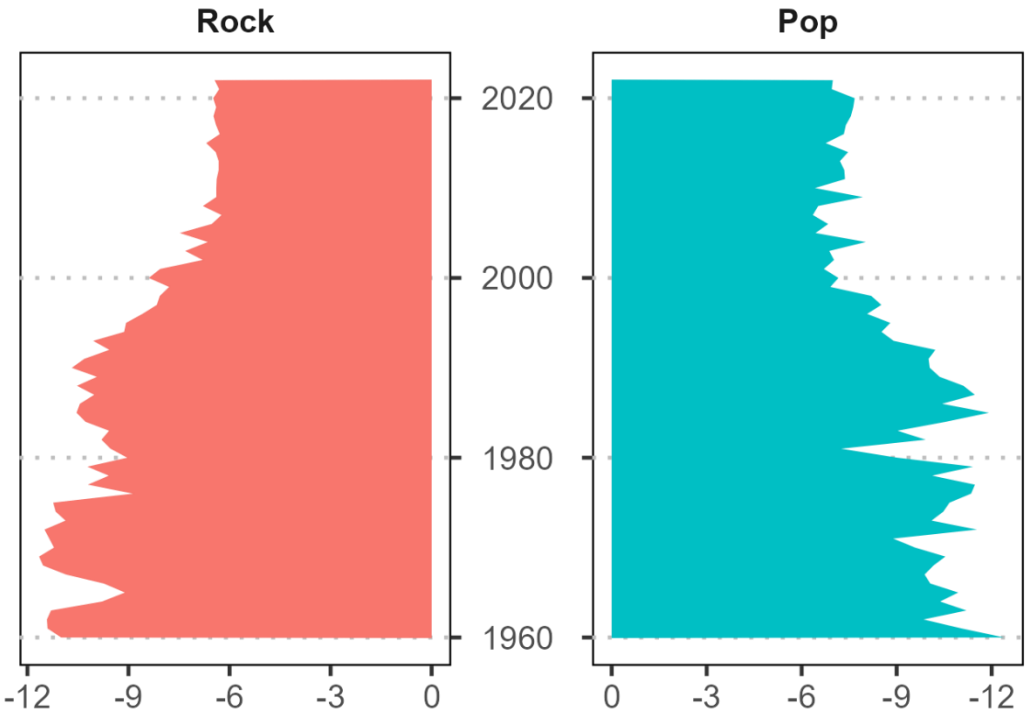

Figura 19. Variación de la variable *loudness* por año divido por género musical. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Como complemento a la Tabla 1, se graficó una muestra de mil canciones aleatorias para cada cuartil en base a la variable *loudness* y se puede notar que la mayor concentración de canciones más populares se encuentra en los cuartiles más altos:

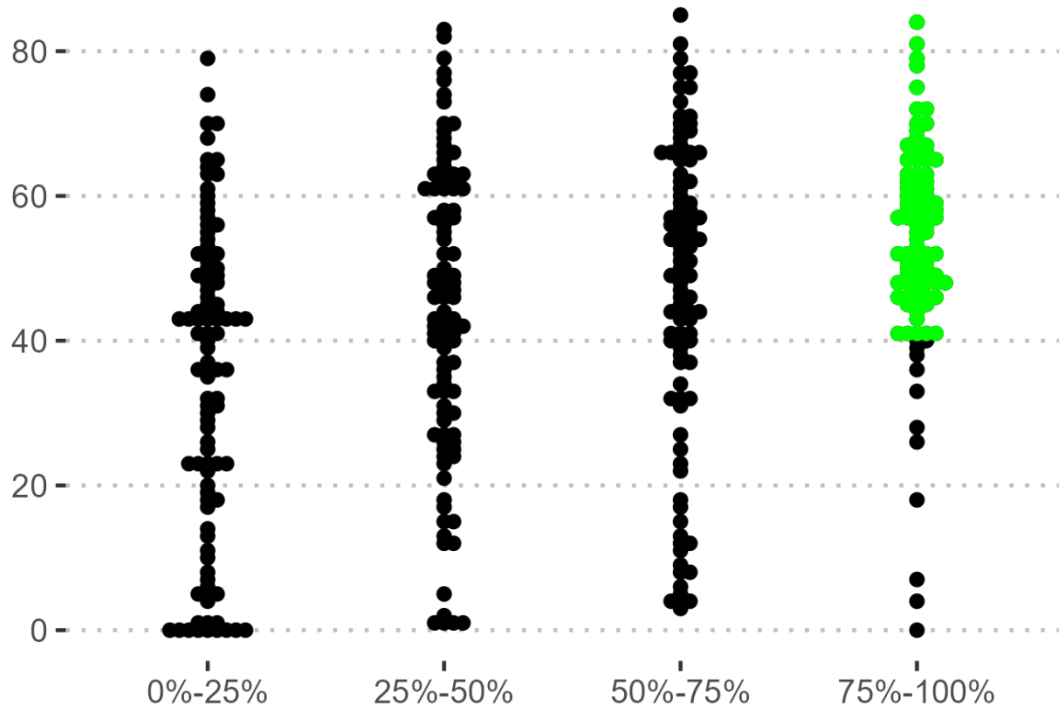

Figura 20. Distribución de la popularidad de las canciones por cuartiles en base a la variable *loudness*. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Complementando al análisis de *k-means* presentado en el capítulo 4, la decisión de cuáles variables utilizar por parte del modelo para determinar la división de los distintos segmentos se puede visualizar con árboles de decisión:

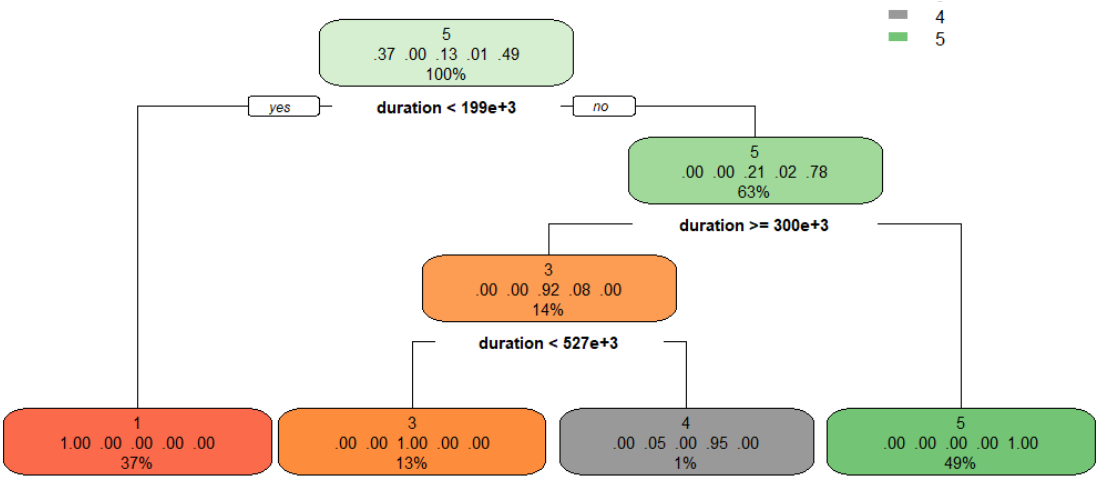

Figura 21. Decisión del modelo para determinar el segmento para cada canción, visualizado con árboles de decisión. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

De esta forma, se puede notar con mayor facilidad el cambio de la utilización de las variables para los cortes en los segmentos al estandarizar las variables:

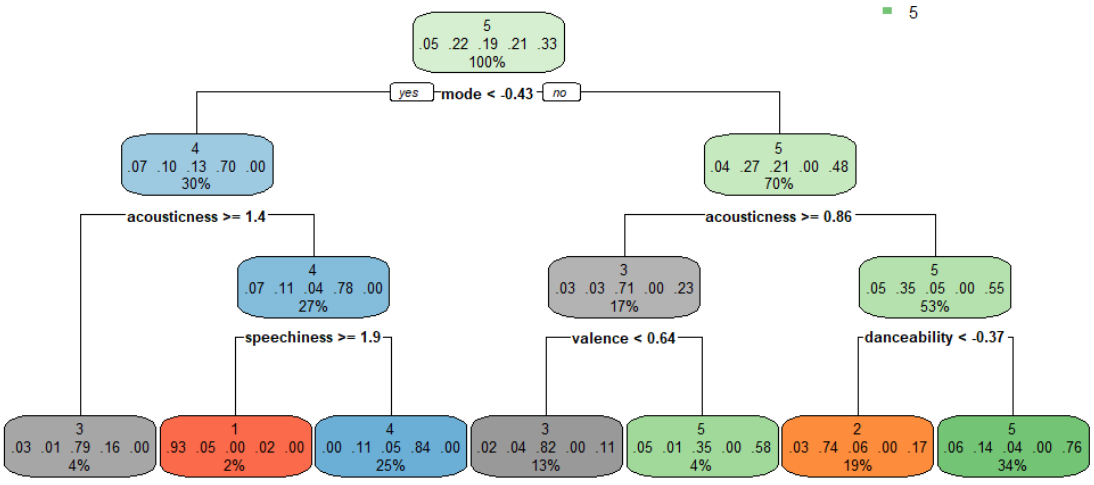

Figura 22. Decisión del modelo para determinar el segmento para cada canción y con las variables normalizadas, visualizado con árboles de decisión. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.

Con respecto a la optimización de hiperparámetros que se realizó para la aplicación del modelo de Maquinas de Vector Soporte, se graficó la variación del rendimiento del modelo al cambiar los hiperparámetros *gamma* y *cost*:

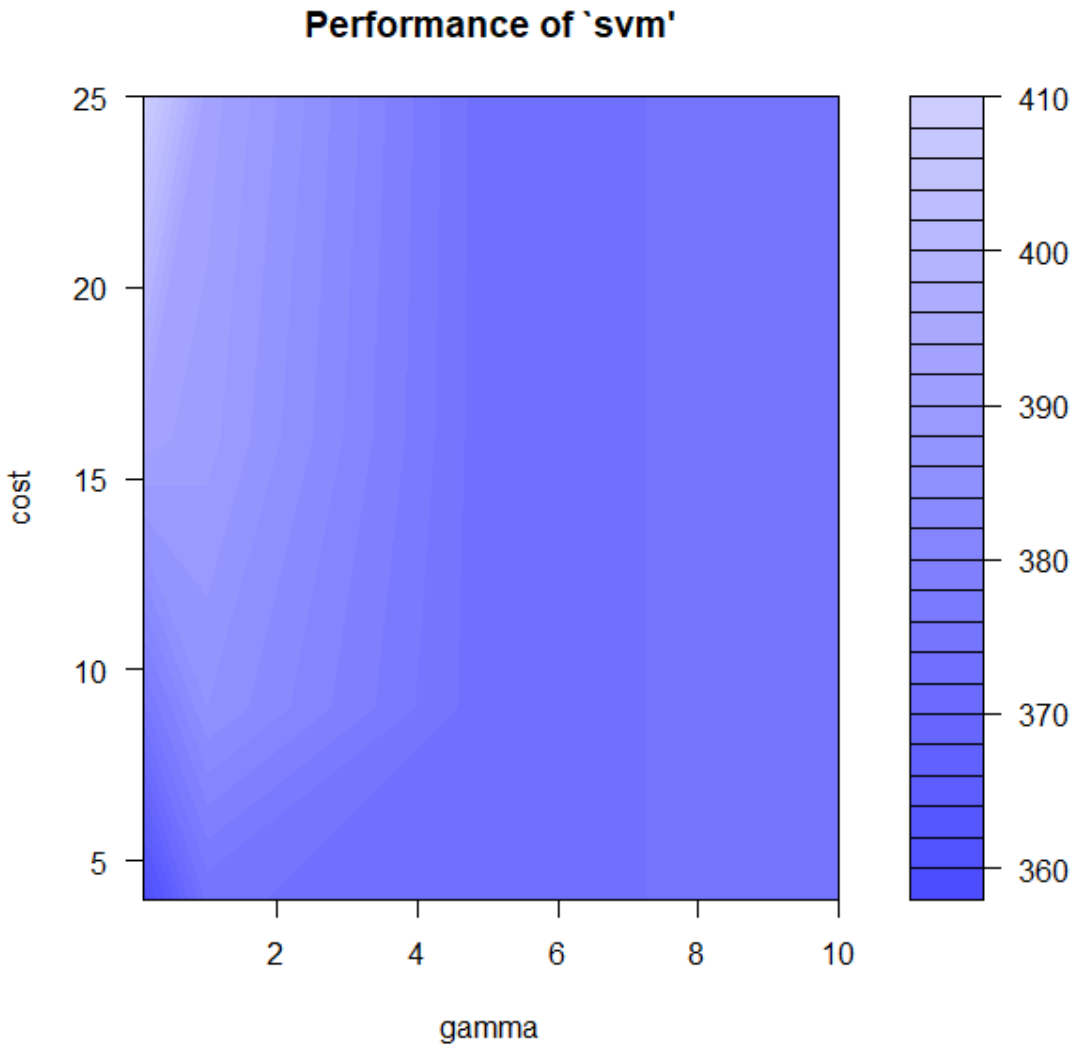

Figura 23. Variación del rendimiento en el modelo de máquina de vector soporte al cambiar los hiperparámetros *gamma* y *cost* del mismo. Fuente: Elaboración propia en base a datos proveídos por la API de *Spotify*.