

# MEMORIA SPOTIFY

CICLO FORMATIVO GRADO SUPERIOR

**Joan Izquierdo Balaciart**

**y Raúl Rodríguez Villalonga**

MODALIDAD PRESENCIAL



## Resumen

Este proyecto emprende un análisis exhaustivo de las tendencias evolutivas en la música popular, abarcando el período comprendido entre los años 2000 y 2022. La base de este estudio es un extenso conjunto de datos extraídos de diversas listas de reproducción de la plataforma Spotify. El alcance del proyecto incluye la configuración y orquestación de un entorno de desarrollo robusto y controlado mediante el uso de Docker Compose, facilitando la integración de herramientas fundamentales en el ecosistema de big data, tales como Apache Hadoop, Apache Cassandra y Apache NiFi. Se han aplicado y explorado una variedad de técnicas avanzadas de análisis de datos para desentrañar patrones y conocimientos ocultos. Estas técnicas comprenden el Análisis Exploratorio de Datos (EDA) para la comprensión inicial de la estructura y características del dataset, la Minería de Datos utilizando la herramienta visual Orange, análisis específicos con Google BigQuery orientados a la exploración de la popularidad anual de las canciones y su característica de variabilidad a lo largo del tiempo, y un profundo Procesamiento del Lenguaje Natural (NLP) aplicado al análisis del contenido textual de las letras de las canciones. Dentro del NLP, se examinaron aspectos como la frecuencia y similitud de las palabras, la identificación de tópicos líricos subyacentes, y cómo estos elementos reflejan el contexto social y eventos relevantes, como el impacto de la pandemia de COVID-19. Adicionalmente, se abordó la tarea de la modelización predictiva mediante la implementación y evaluación comparativa de múltiples algoritmos y enfoques, incluyendo modelos de Regresión Lineal, Random Forest, Gradient Boosting, Regresión Logística, Support Vector Classifier (SVC) y Redes Neuronales. La memoria detalla meticulosamente cada etapa del proceso, desde el preprocesamiento riguroso de los datos, la ingeniería de características para la creación de variables más informativas, la crucial optimización de hiperparámetros de los modelos, hasta la evaluación exhaustiva y rigurosa de su rendimiento. Se subraya la importancia fundamental de estas etapas metodológicas para la obtención de resultados fiables, comprensivos y que aporten valor real a la comprensión de las complejas tendencias musicales que han marcado más de dos décadas.

## Abstract

This project undertakes an in-depth analysis of evolutionary trends in popular music, covering the period between 2000 and 2022. The basis of this study is an extensive dataset extracted from various playlists on the Spotify platform. The project's scope includes the configuration and orchestration of a robust and controlled development environment using Docker Compose, facilitating the integration of fundamental tools in the big data ecosystem, such as Apache Hadoop, Apache Cassandra, and Apache NiFi. A variety of advanced data analysis techniques have been applied and explored to uncover hidden patterns and insights. These techniques encompass Exploratory Data Analysis (EDA) for initial understanding of the dataset's structure and characteristics, Data Mining using the visual tool Orange, specific analyses with Google BigQuery focused on exploring the annual popularity of songs and their danceability characteristic over time, and deep Natural Language Processing (NLP) applied to the analysis of the textual content of song lyrics. Within NLP, aspects such as word frequency and similarity, the identification of underlying lyrical topics, and how these elements reflect the social context and relevant events, such as the impact of the COVID-19 pandemic, were examined. Additionally, the task of predictive modeling was addressed through the implementation and comparative evaluation of multiple algorithms and approaches, including Linear Regression, Random Forest, Gradient Boosting, Logistic Regression, Support Vector Classifier (SVC), and Neural Networks. The report meticulously details each stage of the process, from rigorous data preprocessing, feature engineering for creating more informative variables, the crucial hyperparameter optimization of models, to their exhaustive and rigorous performance evaluation. The fundamental importance of these methodological stages for obtaining reliable, comprehensive results that bring real value to the understanding of the complex music trends that have marked more than two decades is emphasized.

<b>1. Introducción.....</b>	<b>6</b>
1.1 Integrantes.....	6
1.2 Descripción de ideas (Briefing).....	6
1.3 Motivaciones.....	6
1.4 Objetivos.....	7
1.5 Alcance y Limitaciones.....	7
<b>2. Estado del arte y marco teórico.....</b>	<b>8</b>
2.1 Conceptos fundamentales de data science.....	8
2.2 Tecnologías y herramientas de Big Data.....	10
<b>3. Metodología.....</b>	<b>11</b>
3.1 Conjunto de datos.....	11
3.1.1 Fuente y adquisición de datos.....	11
3.1.2 Descripción de los datos.....	12
3.1.3 Análisis Exploratorio de Datos (EDA).....	15
3.2 Preprocesamiento de Datos.....	16
3.3 Diseño Experimental / Modelado.....	17
3.3.1 Modelos Seleccionados.....	17
3.3.2 Métricas de Evaluación.....	20
3.3.3 Configuración del Entrenamiento y Validación.....	23
3.4 Infraestructura y Herramientas de Big Data.....	27
<b>4. Desarrollo e implementación.....</b>	<b>29</b>
4.1 Entorno de Desarrollo.....	29
Gráfica 1.....	31
Gráfica 2.....	32
(Contenedores Docker del entorno de desarrollo en ejecución (docker ps -a)).....	33
4.2 Implementación de la Adquisición y Preprocesamiento.....	35
4.3 Implementación del Modelado y Entrenamiento.....	36
4.4 Visualización de Resultados.....	37
<b>5. Presentación y análisis de resultados.....</b>	<b>37</b>
5.1 Hallazgos Clave del Análisis Exploratorio de Datos.....	38
5.1.1 Ejemplo Bigquery.....	47
• Visualizaciones temporales de los gráficos anteriores.....	49
Resultados y conclusiones:.....	49
Resultados y conclusiones:.....	49
5.1.2 Características Generales de las Letras (NLP).....	50
5.1.2.1 Análisis de Tópicos Líricos.....	51
5.1.2.2 Análisis de Similitud Lírica y Comparación entre Artistas.....	53
5.1.2.4 Tendencias Temporales y Contextuales en Letras.....	58
5.2 Impacto del Preprocesamiento de Datos.....	64
5.3 Evaluación del Rendimiento de los Modelos de Data Science.....	66
5.3.1 Resultados del Modelo A: Regresión Lineal (Ridge).....	67
5.3.2 Resultados del Modelo B: Random Forest Regressor.....	67
5.3.3 Resultados del Modelo C: Gradient Boosting Regressor.....	69

<b>5.3.4 Resultados Modelo D: Red Neuronal.....</b>	<b>69</b>
<b>5.3.5 Comparativa y Selección del Modelo Final.....</b>	<b>70</b>
<b>5.4 Discusión General de los Hallazgos.....</b>	<b>72</b>
<b>6. Conclusiones.....</b>	<b>74</b>
<b>6.1 Conclusiones Principales.....</b>	<b>74</b>
<b>6.2 Trabajos Futuros.....</b>	<b>76</b>

# 1. Introducción

## 1.1 Integrantes

¿Quiénes somos? Somos Joan y Raúl, estudiantes especializándose en el ámbito de Big Data e Inteligencia Artificial.

## 1.2 Descripción de ideas (Briefing)

El núcleo de este proyecto reside en la exploración detallada de las tendencias que han definido la música popular a lo largo de un período significativo, que abarca desde el año 2000 hasta el cierre de 2022. La investigación se fundamenta en el análisis de un conjunto de datos cuidadosamente extraídos de diversas listas de reproducción disponibles en la plataforma de streaming Spotify. El propósito primordial es descifrar y comprender cómo han evolucionado a lo largo de estas más de dos décadas distintas facetas de la música, incluyendo las características musicales intrínsecas de las pistas, la fluctuación de la popularidad a través de los diferentes géneros musicales que han emergido o se han consolidado, y la evolución de los atributos de audio que definen el sonido de las canciones.

## 1.3 Motivaciones

Las motivaciones subyacentes que impulsaron la realización de este proyecto son diversas y buscan responder a preguntas clave en el análisis musical contemporáneo. Entre las principales motivaciones se encuentran: el profundo interés en explorar las tendencias musicales a lo largo de un período extendido para identificar patrones de cambio y continuidad; la necesidad de comprender cómo han evolucionado las características musicales intrínsecas de las canciones, como su ritmo, energía o 'bailabilidad'; el deseo de analizar la dinámica de la popularidad a través de los distintos géneros musicales, identificando cuáles han dominado en diferentes épocas; y la curiosidad por examinar la relación existente entre la popularidad de un artista y la popularidad individual de sus canciones. Estos objetivos de análisis sirvieron como guía para la selección de metodologías y herramientas, orientando la extracción de conocimiento significativo del conjunto de datos de Spotify.

## 1.4 Objetivos

- Identificar y visualizar las tendencias clave en las características musicales, los atributos de audio y la popularidad de los géneros musicales entre 2000 y 2022.
- Aplicar técnicas de clustering para agrupar canciones basadas en sus atributos de audio y desarrollar una lógica para asignarles estados de ánimo interpretables.
- Implementar, evaluar y refinar diversos modelos predictivos (tanto de regresión como de clasificación) para predecir variables relacionadas con las canciones o playlists.
- Desarrollar y entrenar una red neuronal específica para predecir la popularidad de las pistas musicales.
- Realizar análisis de datos a gran escala utilizando BigQuery, enfocándose en tendencias de popularidad y características como la variabilidad a lo largo de los años.
- Construir un dataset completo y fiable de letras de canciones para el período 2000-2022 mediante la automatización del proceso de obtención.
- Realizar análisis detallados del contenido lírico utilizando NLP, incluyendo el estudio de propiedades lingüísticas, el descubrimiento de tópicos por género, el análisis de frecuencia y similitud léxica, y la exploración de la relación entre las letras, eventos sociales y factores lingüísticos.

## 1.5 Alcance y Limitaciones

Sección	Descripción
Alcance	Periodo 2000-2022. Datos de Spotify (playlists) y letras (lyrics.ovh, letras.com). Técnicas: análisis de audio, popularidad, géneros, agrupamiento, ML/DL, BigQuery, NLP. Entorno: Docker Compose.
Limitaciones	Sesgos de las fuentes (Spotify, lyrics.ovh, letras.com) y métodos de adquisición. Condicionamiento de modelos y análisis por datasets. Precisión del dataset de letras dependiente de API y web scraping.

## 2. Estado del arte y marco teórico

### 2.1 Conceptos fundamentales de data science

- Análisis Exploratorio de Datos (EDA):  
Es un enfoque inicial y crucial para comprender la estructura, características, distribuciones y calidad de un conjunto de datos. Implica cargar e inspeccionar los datos (primeras filas, información general, tipos de datos, valores no nulos), calcular estadísticas descriptivas (media, mediana, cuartiles, etc. para variables numéricas; tablas de frecuencia para categóricas), visualizar distribuciones (histogramas, gráficos de densidad, box plots), identificar y manejar valores ausentes, y analizar relaciones entre variables mediante matrices de correlación (para variables numéricas) y gráficos bivariados como scatter plots. El EDA permite identificar patrones, anomalías y la necesidad de pasos de preprocesamiento adicionales.
- Minería datos:  
Proceso de descubrir patrones, correlaciones y conocimiento útil a partir de grandes volúmenes de datos, diferenciándose del análisis descriptivo tradicional al buscar predecir comportamientos futuros y extraer información no evidente. Se enmarca a menudo dentro del Proceso KDD (Knowledge Discovery in Databases), que consta de etapas como Selección, Preprocesamiento, Transformación, Minería de Datos y Evaluación/Interpretación. La herramienta Orange facilita este proceso con su interfaz visual y widgets para carga, selección, imputación, transformación, visualización y modelado (ej. Regresión Logística, Random Forest, Árbol de Decisión, kNN). También se relaciona con el Proceso ETL (Extract, Transform, Load), adaptado en el proyecto a la carga, transformación y uso de datos para entrenamiento y visualización.
- Tratamiento Outliers:  
Los outliers (valores atípicos) son valores extremos que pueden afectar negativamente a muchos algoritmos de Machine Learning sensibles a ellos. Una técnica para tratarlos es el IQR Capping, que limita los valores extremos utilizando el Rango Intercuartílico (IQR = Q3 - Q1) para definir límites (comúnmente  $Q1 - 1.5 \cdot IQR$  y  $Q3 + 1.5 \cdot IQR$ ) y reemplazar los valores fuera de estos límites por los límites mismos. Se implementó un transformador personalizado en Python.
- Ingeniería de Características:  
Esta etapa se enfoca en crear nuevas variables o modificar las existentes para mejorar la capacidad de los modelos para capturar patrones en los datos. En el proyecto, se crearon características de interacción multiplicando pares de atributos de audio (ej. `danceability * energy`) para captar cómo la combinación de propiedades influye en la popularidad. También se generaron características polinómicas (términos al cuadrado) para capturar relaciones no lineales. Además, se realizó una transformación simple como convertir la duración de milisegundos a segundos

- Cluster:  
Agrupar canciones basándose en sus características de audio, identificando clusters que potencialmente corresponden a diferentes estados de ánimo o estilos. Se implementó una lógica personalizada para asignar etiquetas interpretables de estado de ánimo (**mood**) basada en reglas y la distancia a prototipos definidos manualmente, complementando la agrupación de K-Means. La visualización con pairplots ayudó a examinar las relaciones entre las características dentro de los grupos.
- Regresión Lineal:  
Modelo lineal simple para predecir una variable continua basándose en la relación lineal con las características. Sensible a outliers y supuestos del modelo.
- Regresión Ridge:  
Regresión lineal con regularización L2, que penaliza la magnitud de los coeficientes para prevenir el sobreajuste. Es sensible a la escala de las características.
- Regresión Logística:  
Modelo lineal para clasificación binaria (o multiclase). Requiere escalado de características y ajuste del parámetro de regularización.
- Support Vector Classifier (SVC):  
Modelo potente para clasificación, efectivo en espacios de alta dimensión, pero sensible a la escala de características y al ajuste de hiperparámetros (C, gamma, kernel).
- Red Neuronal:  
Modelo no lineal complejo con capas interconectadas. Requiere un preprocesamiento riguroso (escalado, codificación) y ajuste de arquitectura e hiperparámetros. Se utilizó una red secuencial con capas densas, Batch Normalization y Dropout para predecir la popularidad
- Procesamiento del Lenguaje Natural (NLP):  
Rama de la inteligencia artificial y la lingüística computacional dedicada a la interacción entre ordenadores y el lenguaje humano. Su meta es que las máquinas comprendan, interpreten, generen y respondan al lenguaje humano (texto o voz) de forma significativa. Siendo un campo complejo que integra IA, big data y lingüística, el NLP aspira a que las máquinas capten el significado profundo de los textos, incluyendo matices como la ironía o las expresiones idiomáticas, y no se limiten a procesar palabras.
- Gradient Boosting:  
Modelo de ensamblaje potente que construye árboles secuencialmente para corregir errores de los anteriores. Sensible a hiperparámetros y puede ser computacionalmente intensivo.

## 2.2 Tecnologías y herramientas de Big Data

Selección e integración de diversas herramientas de software respondieron a objetivos específicos dentro del flujo de trabajo del proyecto de análisis musical:

- R: Su objetivo principal fue servir como el entorno primario para el análisis estadístico, la manipulación de datos con el **tidyverse**, y la generación de visualizaciones detalladas para el EDA y la presentación de resultados de clustering y correlaciones.
- Python: Se eligió por su versatilidad y extensas bibliotecas, siendo fundamental para tareas de análisis de datos más complejas, la implementación de modelos de Machine Learning avanzados (como Redes Neuronales), el desarrollo de scripts de Procesamiento del Lenguaje Natural (NLP) para el análisis de letras, y la creación de características.
- Data Mining Orange: Su objetivo fue proporcionar una interfaz visual e intuitiva para la exploración interactiva de datos, la aplicación de algoritmos de minería de datos (clasificación, clustering) de manera ágil y la visualización rápida de los resultados, facilitando el prototipado y la comprensión de los flujos analíticos.
- Apache NiFi: El objetivo de su inclusión en el entorno Docker fue simular un sistema de flujo de datos robusto, aunque su uso específico en el procesamiento del dataset de Spotify no se detalla extensamente, su propósito general es la automatización de flujos de datos (ETL/ELT).
- Power BI: Se utilizó con el objetivo de visualizar y compartir información de manera interactiva, permitiendo la creación de cuadros de mando (dashboards) a partir de diversas fuentes de datos para facilitar la toma de decisiones basadas en el análisis.
- Hadoop: Su objetivo fue proporcionar una base para el almacenamiento distribuido y el procesamiento de grandes volúmenes de datos, esencial en un contexto de Big Data, aunque su aplicación directa en las tareas de procesamiento del dataset de Spotify no se especifica en detalle.
- Cassandra: El objetivo de incluir esta base de datos NoSQL distribuida fue manejar grandes cantidades de datos de manera escalable y con alta disponibilidad, aunque su rol específico en el almacenamiento del dataset de Spotify no se describe en profundidad. Se utilizó como servicio en el entorno Docker y se realizaron pruebas de conexión.
- BigQuery: Su objetivo primordial fue permitir la realización de consultas SQL rápidas y eficientes sobre conjuntos de datos muy grandes (petabytes), aprovechando su arquitectura serverless. Se utilizó específicamente para analizar tendencias de popularidad anual y la característica de bialibilidad de las canciones a lo largo del tiempo mediante consultas SQL complejas y funciones analíticas.

- Docker: El objetivo de adoptar Docker Compose fue orquestar un entorno de desarrollo reproducible y aislado que incluyera todos los servicios necesarios (Hadoop, Cassandra, NiFi, Devbox), simplificando la configuración y asegurando la consistencia del entorno para todos los participantes.
- Ubuntu (VM): Su objetivo fue proporcionar el sistema operativo base para la máquina virtual donde se ejecutó el entorno Docker, sirviendo como la infraestructura subyacente para la contenerización y ejecución de todas las herramientas y servicios del proyecto.

## 3. Metodología

### 3.1 Conjunto de datos

#### 3.1.1 Fuente y adquisición de datos

Los datos provienen de una fuente pública disponible en Kaggle. Fueron recolectados y compilados por el usuario "akouaorsot".

- Fuente: [Musical Analytics Spotify 2010-2022](#)
- Aclaración importante: Aunque el nombre del conjunto de datos sugiere que cubre el período de 2010 a 2022, en realidad incluye datos que se extienden desde el año 2000.
- Método de recolección: Aunque no se detalla explícitamente en el extracto proporcionado, es común que estos conjuntos de datos se recolectan mediante APIs públicas (como la API de Spotify) o técnicas de web scraping para obtener información sobre canciones, artistas y sus atributos de audio de diversas playlists.

### 3.1.2 Descripción de los datos

A continuación, se describe la estructura, el contenido y las variables del dataset.

- Formato: El archivo de datos está en formato CSV (valores separados por coma).
- Unidad de análisis: Cada fila del conjunto de datos representa una canción individual que ha sido incluida en una playlist de Spotify correspondiente a un año específico.
- Volumen: La información sobre el número exacto de registros y columnas se obtiene durante el Análisis Exploratorio de Datos (ver sección 3.1.3), específicamente mediante la inspección del DataFrame.
- Estructura y Variables Clave:

Field Category	Data Fields
Identification & Metadata	playlist_url, year, track_id, track_name, album, artist_id, artist_name
Genres & Popularity	artist_genres, track_popularity, artist_popularity
Audio Attributes	danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, tempo
Temporal Features	duration_ms, time_signature

- Diccionario de Datos (Detalle de Variables):

- Identificación de Canciones y Artistas:

Atributo	Descripción	Tipo
track_id	Identificador único de la canción (texto)	Categórica Nominal ▾
track_name	Nombre de la canción (texto)	Categórica Nominal ▾
album	Álbum al que pertenece la canción (texto)	Categórica Nominal ▾
artist_id	Identificador único del artista (texto)	Categórica Nominal ▾
artist_name	Nombre del artista (texto)	Categórica Nominal ▾
playlist_url	Enlace URL de la playlist en Spotify (texto)	Categórica Nominal ▾
year	Año de relevancia de la playlist/canción (número entero)	Numérica (Ratio), Cat... ▾

- Género y Popularidad:

Atributo	Descripción	Tipo
`artist_genres`	Géneros musicales asociados al artista. String o lista (ej. `['pop', 'dance pop']`).	Categórica Nominal (... ▾)
`track_popularity`	Índice de popularidad de la canción (0-100).	Numérica (Ordinal o R... ▾)
`artist_popularity`	Popularidad general del artista (entero).	Numérica (Ordinal o R... ▾)

- Atributos de Audio:

Característica	Descripción	Tipo de Dato	Tipo de Variable
danceability	Qué tan bailable es una canción (0.0–1.0)	Float ▾	Numérica (Rat... ▾
energy	Intensidad y nivel de actividad sonora (0.0–1.0)	Float ▾	Numérica (Rat... ▾
key	Tono principal estimado (0 = C, ..., 11 = B)	Integer ▾	Categórica No... ▾
loudness	Volumen promedio en decibelios (dB)	Float ▾	Numérica (Int... ▾
mode	Modalidad de la canción: Mayor (1) o menor (0)	Integer ▾	Categórica Bi... ▾
speechiness	Presencia de palabras habladas en la canción (0.0–1.0)	Float ▾	Numérica (Rat... ▾
acousticness	Confianza en que la canción es acústica (0.0–1.0)	Float ▾	Numérica (Rat... ▾
instrumentalness	Probabilidad de que la canción no contenga voces (0.0–1.0)	Float ▾	Numérica (Rat... ▾
liveness	Probabilidad de que la canción sea una presentación en vivo (0.0–1.0)	Float ▾	Numérica (Rat... ▾
valence	Qué tan positiva o sentimental se percibe una canción (0.0–1.0)	Float ▾	Numérica (Rat... ▾

tempo	Tempo estimado de la canción en beats por minuto (BPM)	Float ▾	Numérica (Rat... ▾
duration_ms	Duración de la canción en milisegundos	Integer ▾	Numérica (Rat... ▾
time_signature	Compás musical dominante (ej. 3 para 3/4, 4 para 4/4)	Integer ▾	Categórica No... ▾

### 3.1.3 Análisis Exploratorio de Datos (EDA)

El objetivo principal de esta etapa es conocer a fondo los datos con los que vas a trabajar. Es como un detective que examina la escena del crimen para encontrar pistas antes de formular una teoría. Quieres entender la estructura de tus datos, qué características tienen, si hay errores o cosas raras, cómo se relacionan las diferentes partes entre sí, y si necesitarás "limpiar" o transformar los datos más adelante.

1. **Objetivo del EDA:** Comprender profundamente los datos: su estructura, características, calidad, patrones, anomalías y relaciones, para guiar las decisiones futuras del proyecto.
2. **Carga e Inspección Inicial:** Cargar el archivo de datos y observar las primeras filas para una familiarización básica con las columnas y el formato.
3. **Información General del DataFrame:** Revisar los tipos de datos de cada columna (numérico, texto, etc.) y la cantidad de valores no nulos para identificar problemas de formato y la presencia inicial de datos faltantes.
4. **Estadísticas Descriptivas (para columnas numéricas):** Calcular métricas como la media, desviación estándar, mínimos, máximos y cuartiles para entender la distribución central, dispersión y rangos de los valores numéricos.
5. **Manejo de Valores Ausentes:** Contar cuántos datos faltan en cada columna para determinar si se necesitan estrategias de imputación o eliminación (aunque en este caso específico, podrían ser pocos o ninguno).
6. **Análisis Detallado de Características:**
  - **Variables Numéricas:** Utilizar histogramas y diagramas de caja (box plots) para visualizar la forma de su distribución, identificar la mediana, los cuartiles y detectar posibles valores atípicos (outliers).

- Variables Categóricas: Contar la frecuencia de cada categoría para entender su distribución e identificar clases desbalanceadas o categorías poco comunes.
- Variable Objetivo (si aplica): Analizar su distribución para entender el balance entre clases (en clasificación) o el rango de valores (en predicción numérica)

## 7. Análisis de Relaciones entre Variables:

- Matriz de Correlación (Heatmap): Para variables numéricas, visualizar la fuerza y dirección de sus relaciones lineales, ayudando a identificar variables altamente correlacionadas (posible multicolinealidad).
- Gráficos de Dispersion: Observar visualmente la relación entre pares específicos de variables numéricas.

## 3.2 Preprocesamiento de Datos

Tras el Análisis Exploratorio de Datos, se procede con el preprocesamiento para adecuar los datos para el modelado.

### Limpieza de Datos:

- Manejar Valores Atípicos (Outliers): Son valores extremos. Se usa una técnica (IQR Capping) para identificar un rango "normal". Los valores que se salen de este rango se ajustan para que queden justo en el límite de lo normal, ni muy altos ni muy bajos.
- Rellenar Valores Faltantes: Si falta información, se completa (por ejemplo, con promedios) o se anota si no hay datos faltantes, lo que simplifica este paso.

### Transformación de Datos:

- Normalización/Estandarización: Se ajusta la escala de las variables numéricas para que todas tengan un "peso" similar y los modelos no se confundan con números muy grandes o muy pequeños.
- Codificación de Variables Categóricas: Las etiquetas de texto (como "tipo de música") se convierten en números para que los modelos puedan entenderlas.

## Ingeniería de Características:

Se crean nuevas columnas de datos a partir de las existentes para dar más información a los modelos. Esto incluye:

- Características de Interacción: Combinar dos características (ej. multiplicarlas para ver cómo actúan juntas).
- Características Polinómicas: Elevar una característica al cuadrado para capturar relaciones más complejas.
- Conversión de Unidades: Cambiar unidades (ej. milisegundos a segundos) para que sean más fáciles de entender.

## 3.3 Diseño Experimental / Modelado

### 3.3.1 Modelos Seleccionados

En este proyecto se abordaron tanto tareas de agrupamiento como de predicción, utilizando una variedad de algoritmos para explorar diferentes enfoques y comparar sus rendimientos.

#### Modelo de Agrupamiento (Clustering)

- K-Means (para Agrupación y Análisis de Estado de Ánimo)
  - Justificación: Se eligió K-Means por su capacidad para identificar grupos (clusters) inherentes en los datos basados en las características de audio de las canciones. El objetivo era agrupar canciones con perfiles de audio similares para luego asignarles etiquetas de estado de ánimo interpretables, facilitando el análisis y la creación de listas de reproducción temáticas.
  - Breve descripción matemática o conceptual: K-Means es un algoritmo de aprendizaje no supervisado que partitiona un conjunto de datos en 'K' clusters distintos. Funciona iterativamente asignando cada punto de dato al cluster cuyo centroide (la media de los puntos en el cluster) está más cercano, y luego recalculando los centroides. Esto continúa hasta que las asignaciones de los clusters ya no cambian significativamente. Adicionalmente, para la asignación de un "estado de ánimo" (mood) a cada canción, se implementó una lógica personalizada. Esta lógica primero intenta aplicar un conjunto de reglas basadas en los cuartiles de las características de audio principales (**valence**, **energy**, **danceability**). Si una canción no encaja claramente en estas reglas, se le asigna el estado de ánimo del "prototipo emocional" (definido manualmente como un vector de características) al que sea más cercana en términos de distancia euclídea.

## Modelos Predictivos (para Regresión y Clasificación)

El objetivo general para los modelos predictivos era predecir una variable objetivo (numérica para regresión, como `track_popularity`, o categórica para clasificación, como `is_popular` derivada de la popularidad) basándose en las características de las canciones.

- Regresión Lineal (LinearRegression)

- Justificación: Se seleccionó como un modelo base para tareas de regresión, dada su simplicidad e interpretabilidad. Permite modelar la relación lineal entre las características de entrada y una variable objetivo continua.
- Breve descripción matemática o conceptual: La Regresión Lineal busca encontrar la relación lineal que mejor se ajusta entre un conjunto de variables independientes (características) y una variable dependiente (objetivo). Esto se logra minimizando la suma de los cuadrados de las diferencias entre los valores observados y los predichos por el modelo.

- Regresión Ridge (Ridge)

- Justificación: Se eligió para abordar la posible multicolinealidad entre características y para prevenir el sobreajuste en modelos de regresión lineal. Es útil cuando se tienen muchas características, algunas de las cuales pueden estar correlacionadas.
- Breve descripción matemática o conceptual: Ridge es una técnica de regularización aplicada a la regresión lineal. Añade una penalización L2 (proporcional al cuadrado de la magnitud de los coeficientes) a la función de pérdida. Esto "encoge" los coeficientes hacia cero, reduciendo su varianza y la complejidad del modelo, lo que ayuda a mejorar la generalización. No reduce los coeficientes exactamente a cero.

- Random Forest (RandomForestRegressor / RandomForestClassifier)

- Justificación: Es un modelo de ensamblaje robusto y versátil, conocido por su buen rendimiento general tanto en tareas de regresión como de clasificación y por ser menos sensible a la escala de las características. Ayuda a reducir el sobreajuste inherente en los árboles de decisión individuales.
- Breve descripción matemática o conceptual: Random Forest construye múltiples árboles de decisión durante el entrenamiento. Para la clasificación, la predicción final es la clase más votada por los árboles individuales. Para la regresión, es el promedio de las predicciones de los árboles. Cada árbol se entrena con una submuestra aleatoria de los datos (bagging) y considera solo un subconjunto aleatorio de

características en cada división, lo que promueve la diversidad entre los árboles.

- Gradient Boosting (GradientBoostingRegressor / GradientBoostingClassifier)
  - Justificación: Es otro potente modelo de ensamblaje que a menudo proporciona un rendimiento predictivo superior. Se seleccionó por su capacidad para mejorar iterativamente las predicciones.
  - Breve descripción matemática o conceptual: Gradient Boosting construye árboles de forma secuencial. Cada nuevo árbol se entrena para corregir los errores (residuos) cometidos por el ensamblaje de árboles anteriores. Combina estos "aprendices débiles" en un único "aprendiz fuerte" de manera aditiva.
- Regresión Logística (LogisticRegression)
  - Justificación: Es un modelo lineal fundamental y ampliamente utilizado para problemas de clasificación binaria (y extensible a multiclase). Es eficiente y proporciona probabilidades para las predicciones.
  - Breve descripción matemática o conceptual: A pesar de su nombre, la Regresión Logística se usa para clasificación. Modela la probabilidad de que una instancia pertenezca a una clase particular utilizando una función logística (sigmoide) aplicada a una combinación lineal de las características de entrada.
- Support Vector Classifier (SVC)
  - Justificación: Es un modelo de clasificación potente, especialmente efectivo en espacios de alta dimensión y cuando el número de dimensiones es mayor que el número de muestras. Es versátil gracias al uso de diferentes kernels.
  - Breve descripción matemática o conceptual: SVC busca encontrar el hiperplano óptimo que mejor separe las clases en el espacio de características. El "óptimo" se define como el hiperplano que maximiza el margen (la distancia) entre los puntos de datos más cercanos de cada clase (los vectores de soporte). Puede utilizar "kernels" (como el lineal, polinomial o RBF) para transformar los datos a espacios de mayor dimensión donde la separación lineal sea posible.

- Red Neuronal (Sequential model con Keras/TensorFlow)
  - Justificación: Se eligió para capturar relaciones complejas y no lineales en los datos que otros modelos podrían no identificar, especialmente útil para tareas de regresión como la predicción de la popularidad de las pistas.
  - Breve descripción matemática o conceptual: Las redes neuronales están inspiradas en la estructura del cerebro humano. Consisten en capas de nodos interconectados ("neuronas"). Cada conexión tiene un peso, y cada neurona aplica una función de activación a su entrada ponderada. El modelo aprende ajustando estos pesos durante el entrenamiento para minimizar una función de pérdida. El modelo implementado es una red secuencial con múltiples capas ocultas densas, normalización por lotes (BatchNormalization) para estabilizar el aprendizaje, funciones de activación ReLU, regularización L1 para prevenir el sobreajuste, y capas Dropout que desactivan aleatoriamente neuronas durante el entrenamiento para mejorar la generalización. La capa de salida es lineal, adecuada para regresión.

### 3.3.2 Métricas de Evaluación

La elección de métricas de evaluación es crucial y debe alinearse con los objetivos del proyecto y las características del problema (regresión o clasificación, balance de clases, etc.).

Para el Modelo de Agrupamiento (K-Means y Asignación de Estado de Ánimo):

- Evaluación Visual (Pairplot): Se utiliza un `pairplot` coloreado por el estado de ánimo (`mood`) o el cluster (`mood_cluster`) asignado. Esto permite una inspección visual de cómo se distribuyen y separan los grupos en el espacio de las
- características seleccionadas. Se busca que los clusters sean visualmente distinguibles.
- Interpretabilidad de los Estados de Ánimo: La calidad de la asignación de estados de ánimo se evalúa cualitativamente, buscando que las etiquetas asignadas (ej. 'Feliz', 'Triste') tengan sentido en relación con las características de audio de las canciones agrupadas.

Para los Modelos Predictivos (Regresión y Clasificación):

- Modelos de Regresión (Lineal, Ridge, Random Forest Regressor, Gradient Boosting Regressor, Red Neuronal):
  - Error Cuadrático Medio (MSE): Mide el promedio de los errores al cuadrado. Penaliza más los errores grandes.
  - Raíz del Error Cuadrático Medio (RMSE): Es la raíz cuadrada del MSE, lo que la devuelve a las unidades originales de la variable objetivo, facilitando su interpretación.
  - Error Absoluto Medio (MAE): Mide el promedio de las diferencias absolutas entre los valores predichos y reales. Es menos sensible a outliers que el MSE. Indica, en promedio, cuánto se desvían las predicciones.
  - Coeficiente de Determinación ( $R^2$ ): Indica la proporción de la varianza en la variable objetivo que es predecible a partir de las características. Un valor cercano a 1 indica un buen ajuste del modelo.
  - Justificación: Se utiliza un conjunto de estas métricas para obtener una visión completa del rendimiento en regresión.  $R^2$  da una idea de la bondad del ajuste, mientras que MAE y RMSE cuantifican el error promedio en las unidades de la popularidad predicha.

- Modelos de Clasificación (Random Forest Classifier, Gradient Boosting Classifier, Regresión Logística, SVC):
  - Accuracy (Exactitud): Proporción de predicciones correctas sobre el total. Puede ser engañosa en datasets desbalanceados.
  - Precision (Precisión): De todas las instancias predichas como positivas, cuántas lo fueron realmente. Importante cuando el coste de un falso positivo es alto.
  - Recall (Sensibilidad o Exhaustividad): De todas las instancias que eran realmente positivas, cuántas fueron identificadas correctamente. Importante cuando el coste de un falso negativo es alto.
  - F1-Score: Media armónica de Precision y Recall. Útil para buscar un balance entre ambas, especialmente en clases desbalanceadas.
  - Matriz de Confusión: Tabla que visualiza el rendimiento del clasificador, mostrando verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos.
  - Área Bajo la Curva ROC (ROC AUC): Mide la capacidad del modelo para distinguir entre clases. Un valor de 1 representa un clasificador perfecto, y 0.5 un clasificador aleatorio. Es útil para evaluar el rendimiento general independientemente del umbral de clasificación y es robusto ante el desbalance de clases.
  - Curva ROC: Representación gráfica de la tasa de verdaderos positivos frente a la tasa de falsos positivos para diferentes umbrales de clasificación.
  - Justificación: Se utiliza un conjunto completo de métricas de clasificación porque la exactitud por sí sola puede ser insuficiente, especialmente si las clases están desbalanceadas (por ejemplo, si hay muchas más canciones "no populares" que "populares"). Precision, Recall, F1-score y ROC AUC ofrecen una visión más detallada del comportamiento del modelo en las diferentes clases y su capacidad de discriminación. La métrica utilizada para el **scoring** en la búsqueda de hiperparámetros (ej. **f1\_weighted**) debe reflejar la métrica más importante para el problema.

### 3.3.3 Configuración del Entrenamiento y Validación

Se siguió un enfoque estructurado para el entrenamiento y la validación de los modelos, enfatizando la reproducibilidad, la prevención de fuga de datos y la optimización de hiperparámetros.

#### 1. Preparación de Datos Común:

- Carga de Datos: Desde `playlist_2010to2022.csv`.
- Manejo de Nulos: Principalmente mediante `df.dropna(inplace=True)`. Para el modelo K-Means, se implementó una función `handle_missing_values` (con estrategias 'mean', 'median', 'drop'). La red neuronal también incluyó `SimpleImputer` en su pipeline como práctica robusta, aunque el `dropna` global ya se había aplicado.
- Definición de Características y Objetivo: Selección explícita de `numerical_features` y `categorical_features`. La variable objetivo fue `track_popularity` para regresión y `is_popular` (derivada de `track_popularity > mediana`) para clasificación.
- División Estratificada de Datos:
  - Los datos del año 2022 (`df_2022`) se reservaron para una evaluación final o predicción en datos "futuros".
  - Los datos de años anteriores (`df_train_val`) se dividieron en conjuntos de entrenamiento y prueba (`X_train`, `X_test`, `y_train`, `y_test`) usando `train_test_split` de scikit-learn, generalmente con `test_size` de 0.2 o 0.25 y `random_state=42` para reproducibilidad. Para tareas de clasificación, se usó `stratify=y` para mantener la proporción de clases.

## 2. Preprocesamiento y Pipelines:

- Tratamiento de Outliers: Se implementó un transformador personalizado `IQROutlierCapper(factor=1.5)` dentro de los pipelines para características numéricas, limitando los valores extremos.
- Escalado de Características: `StandardScaler` se aplicó a las características numéricas, especialmente crucial para K-Means, Regresión Lineal/Ridge, Regresión Logística, SVC y Redes Neuronales.
- Codificación Categóricas: `OneHotEncoder(handle_unknown='ignore')` se usó para convertir variables categóricas en un formato numérico.
- Automatización con `Pipeline` y `ColumnTransformer`: Estos componentes de scikit-learn se utilizaron sistemáticamente para encapsular los pasos de preprocesamiento (capping, escalado, codificación) y asegurar que se aplican consistentemente, ajustándose solo en los datos de entrenamiento y transformando tanto los datos de entrenamiento como los de prueba/validación.

## 3. Entrenamiento y Optimización de Modelos Específicos:

- K-Means:
  - Escalado de características (`danceability`, `energy`, `valence`) antes del clustering.
  - Se especificó `n_clusters` (ej. 5) y `random_state=RANDOM_STATE`. Se usó `n_init=10` para mejorar la estabilidad de los centroides.
- Regresión Lineal y Ridge:
  - Para Ridge (y Lasso, si se incluyera), se utilizó `GridSearchCV` para optimizar el hiperparámetro `alpha` y, en algunos casos, el grado de `PolynomialFeatures` dentro del pipeline. Se probaron distintos `alphas_to_test` y `degrees_to_test`.

- Modelos Basados en Árboles (Random Forest, Gradient Boosting - para Clasificación):
  - `GridSearchCV` para optimizar hiperparámetros clave como:
    - Random Forest: `rfe__n_features_to_select` (con RFE), `classifier__n_estimators`, `classifier__max_depth`, `classifier__min_samples_split`, `classifier__min_samples_leaf`, `smote__k_neighbors` (cuando se usa `SMOTE` para el desbalance).
    - Gradient Boosting: (Aunque el código de GridSearch no se muestra explícitamente para GB, se infiere la necesidad de ajustar `n_estimators`, `learning_rate`, `max_depth`).
    - Se empleó `class_weight='balanced_subsample'` en RandomForest y SMOTE dentro del pipeline para abordar el desbalance de clases.
- Regresión Logística y SVC:
  - `GridSearchCV` (implícito o sugerido) para:
    - Regresión Logística: `C`, `penalty`, `solver`. Se usó `solver='liblinear'` y `max_iter=1000`.
    - SVC: `C`, `kernel`, `gamma`. Se usó `probability=True` para obtener `predict_proba` para ROC AUC.

- Red Neuronal (Keras/TensorFlow):
  - Arquitectura: Múltiples capas densas con activación `relu`, `BatchNormalization`, `Dropout` para regularización y regularización `L1` en los kernels. Capa de salida lineal para regresión.
  - Compilación: Optimizador `Adam(learning_rate=0.001)`, función de pérdida `mean_squared_error`, métrica `mean_absolute_error`.
  - Entrenamiento: `epochs=150, batch_size=32`.
  - Validación y Control de Sobreajuste: `Early Stopping` monitoreando `val_mean_absolute_error` (o `val_loss`) con `patience=20` y `restore_best_weights=True`. Los datos de prueba (`X_test_processed, y_test`) se usaron como conjunto de validación durante el `fit`.

#### 4. Validación Cruzada:

- Integrada en `GridSearchCV` (ej. `cv=2` o `cv=3` o `cv=5` según el modelo y coste computacional) para la selección de hiperparámetros. Esto proporciona una estimación más robusta del rendimiento del modelo y ayuda a evitar el sobreajuste a una división particular de entrenamiento/prueba.
- La importancia de la validación cruzada se subraya en las "Conclusiones generales" como un estándar para estimar la generalización.

#### 5. Reproducibilidad:

- Se utilizó `random_state=42` en la mayoría de los algoritmos estocásticos (`train_test_split`, `KMeans`, `RandomForestClassifier`, `GradientBoostingClassifier`, `LogisticRegression`, `SVC`) y para la inicialización de pesos en la red neuronal (implícito a través de las operaciones de TensorFlow si se fija una semilla global, aunque no se muestra explícitamente para TF).

## 3.4 Infraestructura y Herramientas de Big Data

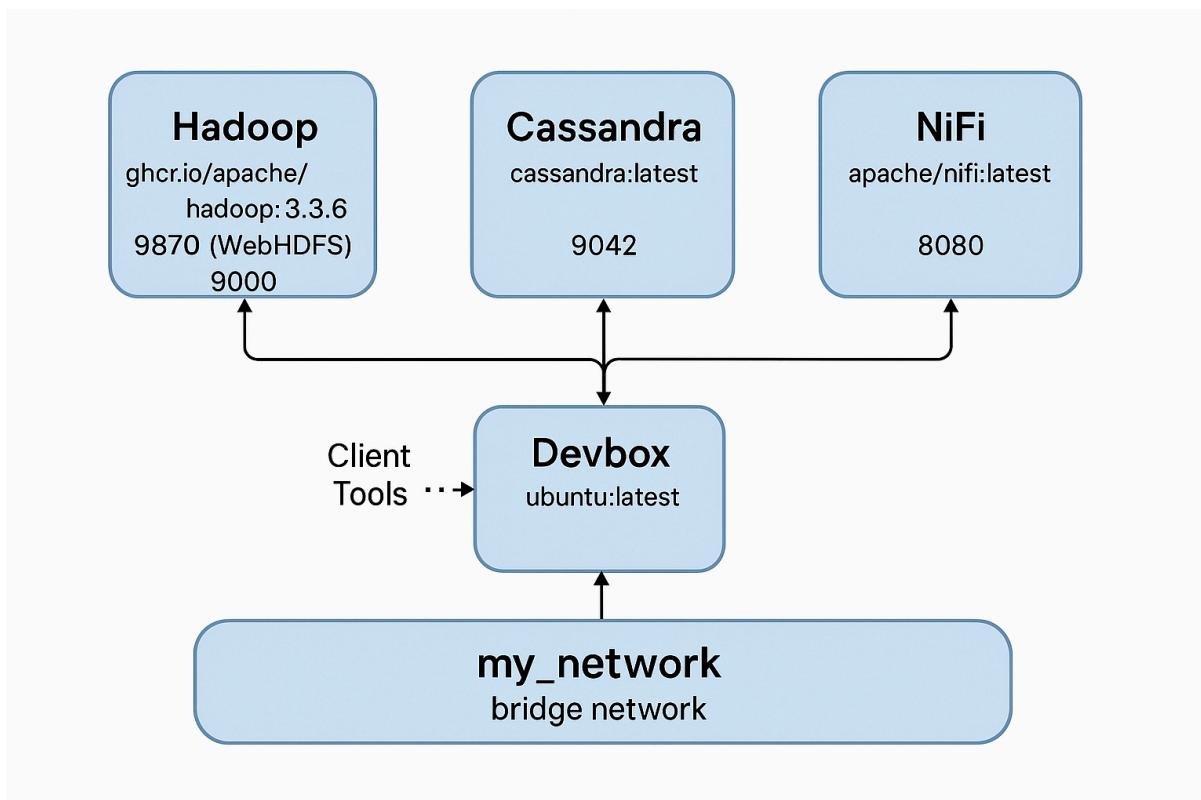
La infraestructura tecnológica para este proyecto se fundamenta en un entorno de desarrollo contenerizado y orquestado mediante Docker Compose. Este enfoque facilitó la configuración reproducible y aislada de los diferentes servicios necesarios para el manejo y análisis de grandes volúmenes de datos. El entorno se desplegó sobre una máquina virtual con sistema operativo base Ubuntu.

El "clúster" de desarrollo, aunque no un clúster de producción a gran escala, se definió en el archivo `docker-compose.yml` e incluyó los siguientes componentes principales interconectados a través de la red `my_network` con driver `bridge`:

- Apache Hadoop: Se integró como un *framework* fundamental para el procesamiento distribuido y el almacenamiento de grandes volúmenes de datos (Big Data). Proporcionó la base para un sistema de archivos distribuido (HDFS). Se utilizó la imagen `ghcr.io/apache/hadoop:3.3.6`, exponiendo los puertos `9870` (WebHDFS) y `9000` (puerto HDFS por defecto) para su acceso. No se detalla una configuración específica de Spark en la información proporcionada, centrándose la infraestructura en Hadoop como base para el manejo de datos distribuidos.
- Apache Cassandra: Como base de datos NoSQL distribuida, Cassandra se incluyó con el objetivo de manejar grandes cantidades de datos de manera escalable y con alta disponibilidad. Se utilizó la imagen `cassandra:latest` y se expuso el puerto `9042` para el protocolo CQL (Cassandra Query Language). Se realizaron pruebas de conexión como parte de la implementación.
- Apache NiFi: Este sistema robusto se integró para automatizar el flujo de datos entre diferentes sistemas, siendo ideal para tareas de ETL/ELT (Extraer, Transformar, Cargar/Transformar) y la gestión del movimiento de datos. Se utilizó la imagen `apache/nifi:latest`, mapeando el puerto `8080` para acceder a su interfaz de usuario web.
- Google BigQuery: Aunque no formaba parte del entorno contenerizado local, Google BigQuery se utilizó como una herramienta específica para el manejo y análisis de grandes conjuntos de datos. Su objetivo primordial fue permitir la realización de consultas SQL rápidas y eficientes sobre volúmenes de datos muy grandes (petabytes) gracias a su arquitectura *serverless*. Se utilizó particularmente para analizar tendencias de popularidad anual y la característica de bailabilidad de las canciones a lo largo del tiempo.

- Devbox: Un contenedor basado en Ubuntu (`ubuntu:latest`) que actuó como entorno de desarrollo y puerta de enlace para interactuar con los otros servicios contenerizados y ejecutar herramientas cliente.

La comunicación entre estos servicios se realizó a través de la red de Docker, permitiendo que se diferenciarán entre sí usando sus nombres de servicio. La implantación de este entorno implicó la verificación de requisitos (Docker Engine, Docker Compose), la obtención de los archivos de configuración, el inicio del entorno con `docker-compose up -d`, la verificación del estado de los servicios y el acceso al contenedor `devbox` para instalar herramientas cliente (como `cqlsh` para Cassandra) y ejecutar scripts. Se encontraron y documentaron problemas durante la implementación inicial relacionados con imágenes obsoletas de software y dependencias, lo que requirió ajustes en la configuración para garantizar la compatibilidad y el correcto funcionamiento.



## 4. Desarrollo e implementación

El desarrollo del proyecto implicó un proceso estructurado para la exploración, preprocesamiento, análisis y modelado de un extenso conjunto de datos musicales. Se adoptó una metodología robusta que integró diversas herramientas y enfoques, como el proceso KDD (Knowledge Discovery in Databases) para la minería de datos y una adaptación práctica del proceso ETL (Extract, Transform, Load) para la gestión del flujo de datos. El núcleo del trabajo residió en el riguroso procesamiento del dataset y la aplicación de técnicas avanzadas de análisis y modelado predictivo.

### 4.1 Entorno de Desarrollo

El entorno de desarrollo fue concebido para ser políglota y multi-herramienta, aprovechando las fortalezas de diversas plataformas y lenguajes. Los principales elementos del entorno y el software utilizado fueron:

Lenguajes de Programación:

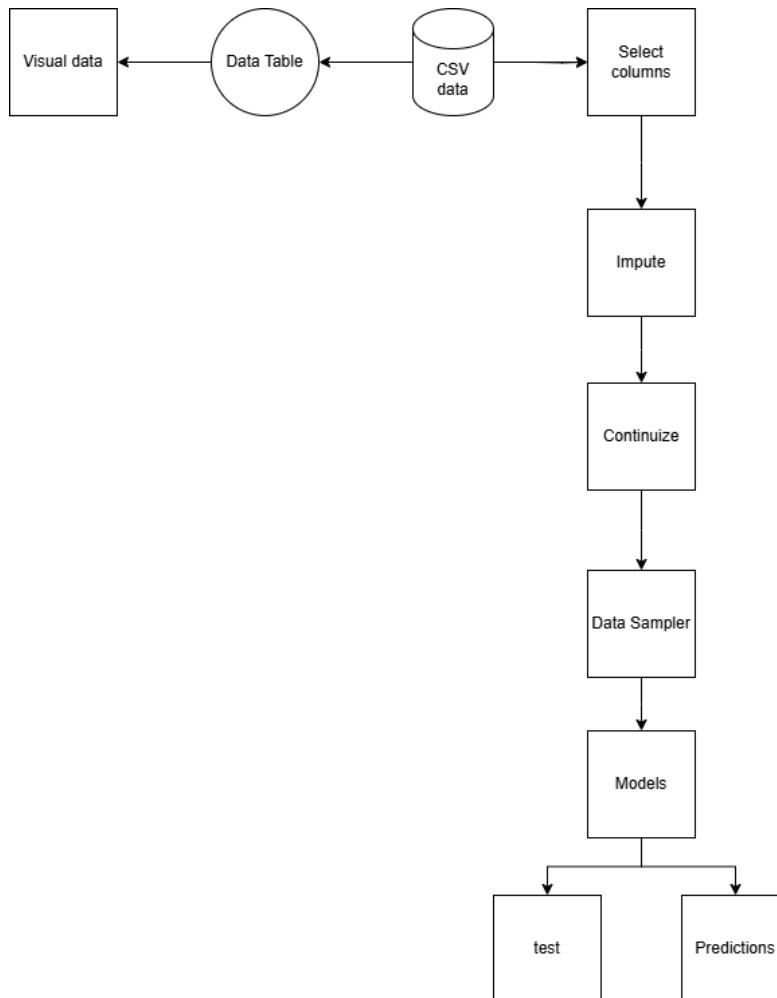
- R: Sirvió como el entorno primario para el análisis estadístico, la manipulación de datos, y la generación de visualizaciones detalladas.
- Python: Fue fundamental para tareas de análisis de datos más complejas, la implementación de modelos de Machine Learning avanzados, el desarrollo de scripts de Procesamiento del Lenguaje Natural (NLP) y la creación de características.

Bibliotecas Principales:

- R: Se utilizaron paquetes esenciales del universo tidyverse para manipulación, limpieza, transformación y visualización de datos. Se integraron herramientas especializadas como `stats` y `factoextra` para técnicas de *clustering*, y `ggcorrplot` para matrices de correlación.
- Python: Se aprovechó su vasto ecosistema con bibliotecas científicas y de análisis de datos, incluyendo Pandas (manipulación de datos), NumPy (operaciones numéricas), Scikit-learn (modelado y *pipelines* de ML), TensorFlow y Keras (Redes Neuronales). Para NLP se utilizaron bibliotecas como spaCy y Gensim, y NLTK para la gestión de *stopwords*.

## Herramientas Específicas:

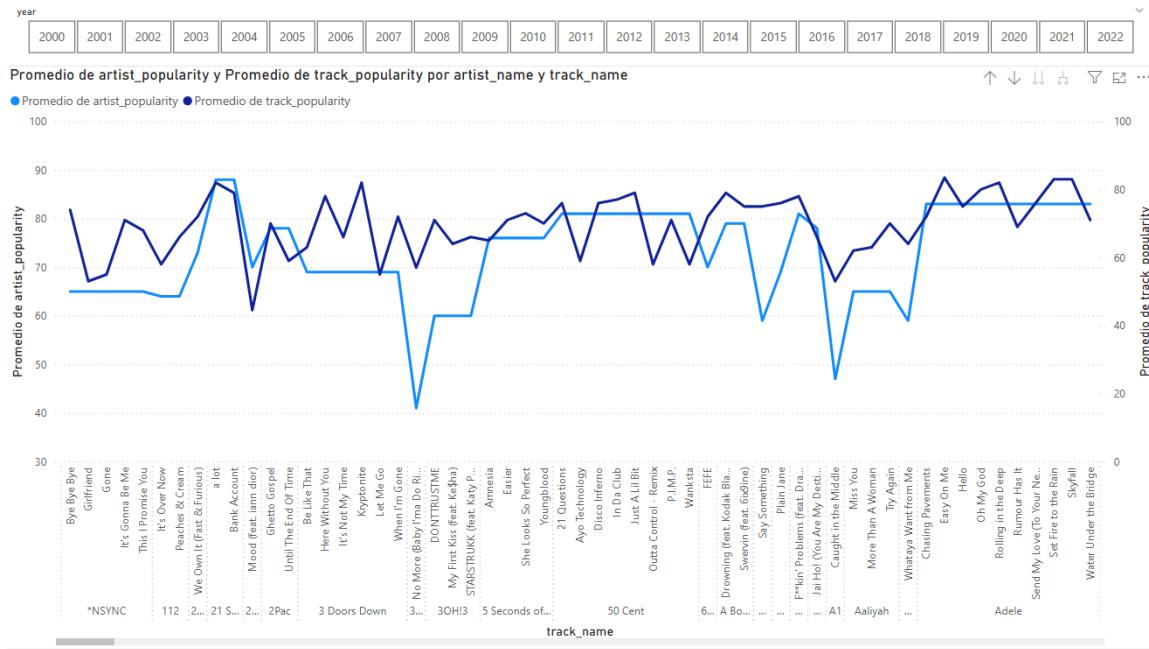
- Data Mining Orange: Una interfaz visual e intuitiva para la exploración interactiva de datos y la aplicación ágil de algoritmos de minería de datos.



- Google BigQuery: Plataforma de *data warehouse* en la nube utilizada para análisis eficientes sobre grandes conjuntos de datos mediante SQL.

- Power BI: Herramienta de Business Intelligence para visualizar y compartir información de manera interactiva a través de cuadros de mando.

Gráfica 1



Esta gráfica es un gráfico de líneas que visualiza la evolución del promedio de popularidad tanto de los artistas como de sus canciones a lo largo de los años, desde 2000 hasta 2022.

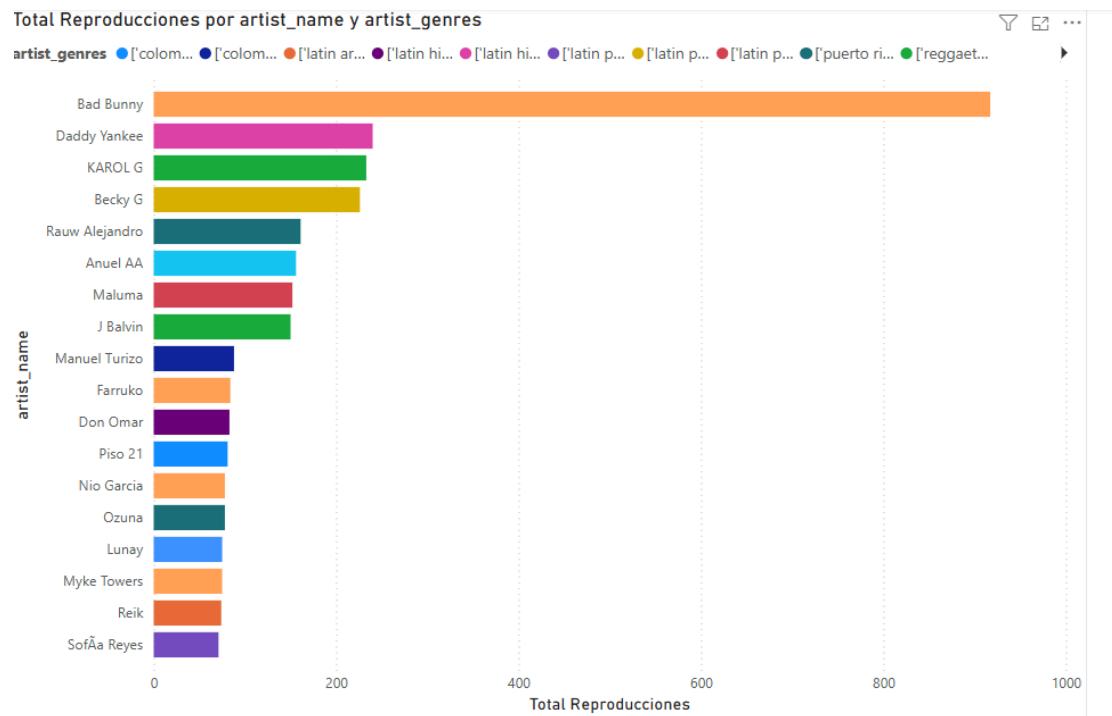
Hay dos líneas: una azul oscura para el promedio de popularidad del artista y una azul clara para el promedio de popularidad de la canción.

El eje horizontal muestra nombres de canciones específicas, que parecen ser puntos de referencia temporal en la gráfica.

Los ejes verticales (uno a cada lado) miden la popularidad en una escala (probablemente de 0 a 100).

La gráfica muestra cómo fluctuó la popularidad promedio de los artistas y las canciones a lo largo de este período de 22 años, destacando la popularidad en momentos asociados a ciertas canciones populares.

Gráfica 2



Esta gráfica es un diagrama de barras horizontal que muestra el total de reproducciones acumuladas por diferentes artistas.

Los artistas están listados en el eje vertical y el total de reproducciones en el eje horizontal.

Las barras están coloreadas para representar los diferentes géneros de los artistas (como [latin urban], [colombian pop], etc.).

La gráfica permite ver fácilmente qué artistas tienen el mayor número de reproducciones totales dentro de este conjunto de datos, siendo Bad Bunny el artista con más reproducciones, seguido por Daddy Yankee, KAROL G, y otros.

## Contenerización y Virtualización:

- Docker Compose: Utilizado para orquestar el entorno de desarrollo reproducible y aislado que incluyó servicios como Hadoop, Cassandra y NiFi.

En esta parte vemos el código que usamos para crear el docker compose, donde se ve que tiene las 3 imágenes necesarias en esta parte del trabajo (Nifi, Cassandra, Hadoop y Devbox)

```
root@dockerc: /home/raul/docker-cluster
GNU nano 6.2
version: '3.9'

services:
  hadoop:
    image: ghr.io/apache/hadoop:3.3.6
    container_name: hadoop # Asumiendo el nombre por defecto o uno similar
    ports:
      - "9870:9870"
      - "9000:9000"
    networks:
      - my_network # Usando un nombre de red similar al anterior
    # depends_on: # Puedes añadir dependencias si son necesarias para el arranque
    #   - cassandra

  cassandra:
    image: cassandra:latest
    container_name: cassandra # Asumiendo el nombre por defecto o uno similar
    ports:
      - "9042:9042"
    networks:
      - my_network

  nifi:
    image: apache/nifi:latest
    container_name: nifi # Asumiendo el nombre por defecto o uno similar
    ports:
      - "8080:8080"
    environment:
      - NIFI_WEB_HTTP_PORT=8080 # Revisa si la nueva imagen de NiFi requiere esta variable
    networks:
      - my_network
    # depends_on: # Puedes añadir dependencias si son necesarias para el arranque
    #   - cassandra
    #   - hadoop

  devbox:
    image: ubuntu:latest
    container_name: devbox # Asumiendo el nombre por defecto o uno similar
    # ports: # No necesitas exponer puertos si solo lo usas para herramientas internas
    #   - "xxxx:yyy"
    networks:
      - my_network
    # depends_on: # Puedes añadir dependencias si quieres que espere a otros servicios
    #   - cassandra
    #   - hadoop
    stdin_open: true # Mantiene stdin abierto para la terminal
    tty: true # Asigna una pseudo-TTY

networks:
  my_network:
    driver: bridge
```

*(Configuración de Docker Compose para el entorno de desarrollo)*

- Ubuntu (VM): El sistema operativo base para la máquina virtual donde se ejecutó el entorno Docker.

```
root@dockerc: /home/raul/docker-cluster# docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
25b722f2aab9        apache/nifi:latest   "/..scripts/start.sh"   3 hours ago       Up 3 hours          8000/tcp, 8443/tcp, 10000/tcp, 0.0.0.0:8080->8080/tcp, :::8080->8080/tcp   nifi
b72b320c9430        ghr.io/apache/hadoop:3.3.6  "/usr/local/bin/dumb_  3 hours ago       Exited (0) 3 hours ago   8000-7001/tcp, 7199/tcp, 9160/tcp, 0.0.0.0:9042->9042/tcp, :::9042->9042/tcp   hadoop
6a37acd609ab        ubuntu:latest        "/bin/bash"          3 hours ago       Up 3 hours          8000-7001/tcp, 7199/tcp, 9160/tcp, 0.0.0.0:9042->9042/tcp, :::9042->9042/tcp   devbox
6040a62540f4        cassandra:latest    "docker-entrypoint.s..  3 hours ago       Up 3 hours          7000-7001/tcp, 7199/tcp, 9160/tcp, 0.0.0.0:9042->9042/tcp, :::9042->9042/tcp   cassandra
```

*(Contenedores Docker del entorno de desarrollo en ejecución (docker ps -a))*

- La interacción con este entorno se realizó principalmente a través del contenedor **devbox**, actuando como puerta de enlace para ejecutar comandos y herramientas cliente.

```
root@dockercompose:/home/raul/docker-cluster# docker exec -it devbox bash
root@6a37acd609ab:/#
```

- Una vez dentro del contenedor **devbox**, se procedió a instalar y configurar las herramientas necesarias para interactuar con los diferentes servicios y realizar análisis.

```
root@6a37acd609ab:/#
root@6a37acd609ab:/# apt install -y python3-pip netcat
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package netcat is a virtual package provided by:
  netcat-traditional 1.10-48
  netcat-openbsd 1.22-6~1ubuntu2
You should explicitly select one to install.

E: Package 'netcat' has no installation candidate
root@6a37acd609ab:/# apt install -y python3-pip netcat-traditional
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  adduser binutils binutils-common binutils-x86_64-linux-gnu build-essential bzip2 ca-certificates cpp cpp-13
  cpp-13-x86_64-linux-gnu cpp-x86_64-linux-gnu dirmngr dpkg-dev fakeroot fontconfig-config fonts-dejavu-core
  fonts-dejavu-mono g++-13 g++-13-x86_64-linux-gnu g++-x86_64-linux-gnu gcc gcc-13 gcc-13-base
  gcc-13-x86_64-linux-gnu gcc-x86_64-linux-gnu gnupg gnupg-110n gnupg-utils gpg gpg-agent gpg-wks-client gpgconf
  gpgsm Javascript-common keybox libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libaom3
  libasan8 libatomic libbinutils libbrotli libbsds0 libc-dev-bin libc-devtools libcc0-dev libcc1-0 libcrypt-dev
  libctf-nobfd0 libctf0 libde265-0 libdeflate0 libdpkg-perl libexpat1 libexpat1-dev libfreetype0
  libfile-fcntllock-perl libfontconfig1 libfreetype6 libgcc-13-dev libgd3 libgdbm-compat4t64 libgdbm6t64 libgomp1
  libgprofng0 libheif-plugin-aomdec libheif-plugin-aomen libheif-plugin-libde265 libheif1 libhwasan0 libis123
  libitm libjansson4 libjpeg-turbo8 libjpeg0 libjs-jquery libjs-sphinxdoc libjs-underscore libksba8
  libldap-common libldap2 liblberc4 liblberc4-gettext-perl liblisan0 liblmpc3 libmpfr6 libper15.38t64 libpng16-16t64
  libpython3-dev libpython3.12-dev libpython3.12-minimal libpython3.12-stdlib libpython3.12t64
  libquadmath0 libreadline8t64 libbsa12-2 libbsa12-modules libbsa12-modules-db libbsframe1 libsharpyuv0 libsqlite3-0
  libstdc++-13-dev libtiff6 libubsan1 libubwp7 libx11-6 libx11-data libxau libxcb1 libxdmcp6 libxpm4
  linux-libc-dev lto-disabled-list make manpages manpages-dev media-types netbase openssl patch perl
  perl-modules-5.38 pinentry-curses python3 python3-dev python3-minimal python3-pkg-resources python3-setuptools
  python3-wheel python3.12 python3.12-dev python3.12-minimal readline-common rpcsvc-proto tzdata xz-utils
  zlib1g-dev
Suggested packages:
  cron quota cryptfs-utils binutils-doc gprofng-gui bzip2-doc cpp-doc gcc-13-locales cpp-13-doc dbus-user-session
  libpam-systemd pinentry-gnome3 tor debian-keyring g++-multilib gcc-13-doc gcc-multilib autoconf
  automake libtool flex bison gdb gcc-doc gcc-13-multilib gdb-x86_64-linux-gnu parcmimonie xloadimage gpg-wks-server
  scdaemon apache2 | lighttpd | httpd glibc-doc git bzr libgd-tools gdbm-110n libheif-plugin-x265
  libheif-plugin-ffmpgedec libheif-plugin-jpegdec libheif-plugin-jpegenc libheif-plugin-j2kdec
  libheif-plugin-j2kenc libheif-plugin-ravile libheif-plugin-svenc libbsa12-modules-gssapi-mit
  libbsa12-modules-gssapi-heimdal libbsa12-modules-ldap libbsa12-modules-otp libbsa12-modules-sql
  libstdc++-13-doc make-doc man-browser ed diffutils-doc perl-doc libterm-readline-gnu-perl
  libterm-readline-perl-perl libtap-harness-archive-perl pinentry-doc python3-doc python3-tk python3-venv
  python-setuptools-doc python3.12-venv python3.12-doc binfmt-support readline-doc
The following NEW packages will be installed:
  adduser binutils binutils-common binutils-x86_64-linux-gnu build-essential bzip2 ca-certificates cpp cpp-13
  cpp-13-x86_64-linux-gnu cpp-x86_64-linux-gnu dirmngr dpkg-dev fakeroot fontconfig-config fonts-dejavu-core
  fonts-dejavu-mono g++-13 g++-13-x86_64-linux-gnu g++-x86_64-linux-gnu gcc gcc-13 gcc-13-base
  gcc-13-x86_64-linux-gnu gcc-x86_64-linux-gnu gnupg gnupg-110n gnupg-utils gpg gpg-agent gpg-wks-client gpgconf
  gpgsm Javascript-common keybox libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libaom3
  libasan8 libatomic libbinutils libbrotli libbsds0 libc-dev-bin libc-devtools libcc0-dev libcc1-0 libcrypt-dev
  libctf-nobfd0 libctf0 libde265-0 libdeflate0 libdpkg-perl libexpat1 libexpat1-dev libfreetype0
  libfile-fcntllock-perl libfontconfig1 libfreetype6 libgcc-13-dev libgd3 libgdbm-compat4t64 libgdbm6t64 libgomp1
  libgprofng0 libheif-plugin-aomdec libheif-plugin-aomen libheif-plugin-libde265 libheif1 libhwasan0 libis123
  libitm libjansson4 libjpeg-turbo8 libjpeg0 libjs-jquery libjs-sphinxdoc libjs-underscore libksba8
  libldap-common libldap2 liblberc4 liblberc4-gettext-perl liblisan0 liblmpc3 libmpfr6 libper15.38t64 libpng16-16t64
  libpython3-dev libpython3.12-dev libpython3.12-minimal libpython3.12-stdlib libpython3.12t64
  libquadmath0 libreadline8t64 libbsa12-2 libbsa12-modules libbsa12-modules-db libbsframe1 libsharpyuv0 libsqlite3-0
  libstdc++-13-dev libtiff6 libubsan1 libubwp7 libx11-6 libx11-data libxau libxcb1 libxdmcp6 libxpm4
  linux-libc-dev lto-disabled-list make manpages manpages-dev media-types netbase openssl patch perl
  perl-modules-5.38 pinentry-curses python3 python3-dev python3-minimal python3-pkg-resources
  python3-setuptools python3-wheel python3.12 python3.12-dev python3.12-minimal readline-common rpcsvc-proto tzdata
  xz-utils zlib1g-dev
```

- Instalación de paquetes en el contenedor devbox (*usando el comando apt install -y python3-pip netcat, ya que enseñamos en la foto como están instalados estos dos paquetes*)

```
root@6a37acd609ab:/# dpkg -l | grep python3-pip
ii  python3-pip                  24.0+dfsg-1ubuntu1.1          all      Python package installer
ii  python3-pip-whl               24.0+dfsg-1ubuntu1.1          all      Python package installer (pip wheel)
root@6a37acd609ab:/# dpkg -l | grep netcat
ii  netcat-traditional          1.10-48                  amd64    TCP/IP swiss army knife
root@6a37acd609ab:/#
```

- Para la interacción con la base de datos Cassandra, se configuró un entorno Python virtual y se instaló la herramienta de línea de comandos `cqlsh`.

```

root@6a37acd609ab:/# python3 -m venv myvenv
root@6a37acd609ab:/# source myvenv/bin/activate
(myvenv) root@6a37acd609ab:/# pip install cqlsh
Collecting cqlsh
  Downloading cqlsh-6.2.0-py3-none-any.whl.metadata (7.1 kB)
Collecting cassandra-driver (from cqlsh)
  Downloading cassandra_driver-3.29.2-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (6.2 kB)
Collecting pure-sasl (from cqlsh)
  Downloading pure-sasl-0.6.2.tar.gz (11 kB)
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
(myvenv) root@6a37acd609ab:/# cqlsh cassandra 9042
WARNING: cqlsh was built against 5.0.0, but this server is 5.0.4. All features may not work!
Connected to Test Cluster at cassandra:9042
[cqlsh 6.2.0 | Cassandra 5.0.4 | CQL spec 3.4.7 | Native protocol v5]
Use HELP for help.
cqlsh>
cqlsh>
cqlsh> -

```

----- 102.2/102.2 kB 10.4 MB/s eta 0:00:00

```

Downloading six-1.17.0-py2.py3-none-any.whl (11 kB)
Building wheels for collected packages: pure-sasl
  Building wheel for pure-sasl (pyproject.toml) ... done
    Created wheel for pure-sasl: filename=pure_sasl-0.6.2-py3-none-any.whl size=11518 sha256=59b899c9ebaa6b9d1e8a578de6ba954f32f244514074d2598972145de6eb9743
    Stored in directory: /root/.cache/pip/wheels/5d/bf/32/7369bb034de66e1c91d3ac3dafbff77ae3c97b9c5c97ea1b2
Successfully built pure-sasl
Installing collected packages: wcwidth, pure-sasl, six, click, geomet, cassandra-driver, cqlsh

```

- Conversión y Preparación de Tipos: Se transformaron variables categóricas nominales (`key`, `mode`, `time_signature`) a factores. Se derivaron nuevas variables como `duration_min` (duración en minutos) y `tempo_category` (clasificación del tempo en categorías como Lento, Medio, Rápido, Muy Rápido).
- Manejo de Valores Ausentes: Se optó principalmente por la eliminación de filas que presentaban valores `NA` en variables de audio consideradas esenciales para el análisis.
- Limpieza de Duplicados: Se eliminan registros repetidos basándose en el `track_id` para garantizar una única entrada por canción.
- Tratamiento de Outliers (IQR Capping): Se implementó una técnica de IQR Capping para limitar los valores extremos utilizando el Rango Intercuartílico ( $IQR=Q3-Q1$ ) para definir límites ( $Q1-1.5\times IQR$  y  $Q3+1.5\times IQR$ ) y reemplazar los valores fuera de estos por los límites mismos. Se menciona la implementación de un transformador personalizado en Python para esta tarea.
- Ingeniería de Características: Se crearon nuevas variables o se modificaron las existentes para mejorar la capacidad de los modelos:
  - Características de interacción multiplicando pares de atributos de audio (ej. `danceability`×`energy`).
  - Características polinómicas (términos al cuadrado) para capturar relaciones no lineales.

- Transformación simple como convertir la duración de milisegundos a segundos.
- Procesamiento del Lenguaje Natural (NLP) para Letras: Se aplicó un procesamiento específico al contenido textual de las letras de canciones, que incluyó limpieza de texto (conversión a minúsculas, eliminación de ruido), tokenización y lematización, eliminación de *stopwords* (listas estándar y personalizadas), filtrado de tokens, generación de N-gramas, análisis de frecuencia, cálculo de riqueza léxica, y modelado de tópicos (LDA).

## 4.3 Implementación del Modelado y Entrenamiento

La implementación del modelado predictivo se llevó a cabo explorando y evaluando comparativamente múltiples algoritmos y enfoques. El proceso de entrenamiento y ajuste siguió una metodología rigurosa:

- Modelos Implementados: Se implementaron y evaluaron varios modelos de Machine Learning, incluyendo:

Tipo de Modelo	Modelos Específicos
Regresión	Lineal, Ridge
Ensamble	Random Forest, Gradient Boosting
Clasificación	Regresión Logística, Support Vector Classifier (SVC)
Redes Neuronales	Secuencial (densas, Batch Normalization, Dropout)

- Preprocesamiento para Modelos: Se aplicó un preprocesamiento riguroso antes de alimentar los datos a los modelos, incluyendo escalado de características (crucial para modelos sensibles como Regresión Lineal, Ridge, Redes Neuronales, Regresión Logística y SVC) y codificación adecuada de variables categóricas.
- Entrenamiento: Los modelos fueron entrenados utilizando los datos preprocesados.
- Ajuste de Hiperparámetros: Se reconoció la importancia crítica de la optimización de hiperparámetros. Se utilizaron técnicas como GridSearchCV o RandomizedSearchCV, combinadas con validación cruzada, para encontrar las configuraciones óptimas que permitieran a los modelos generalizar de manera efectiva y evitar el sobreajuste o subajuste.

- Evaluación: La evaluación del rendimiento de los modelos se realizó utilizando validación cruzada para obtener una estimación más robusta y fiable del error de generalización. Se seleccionaron y reportaron métricas de evaluación adecuadas que reflejaran el objetivo del problema (por ejemplo, métricas como Precision, Recall, F1-Score y ROC AUC fueron consideradas en datasets desbalanceados, además de Accuracy). Se enfatizó la importancia de no basar la elección final únicamente en la métrica de rendimiento más alta, sino también considerando la interpretabilidad y complejidad del modelo.

## 4.4 Visualización de Resultados

La visualización de resultados fue una etapa clave tanto en la fase de Análisis Exploratorio de Datos (EDA) como en la presentación de las conclusiones. Las herramientas y bibliotecas empleadas para la visualización fueron:

- R (con tidyverse y ggc当地): Utilizado para generar visualizaciones detalladas durante el EDA, incluyendo distribuciones (histogramas, gráficos de densidad, *box plots*), análisis de relaciones entre variables (*scatter plots*) y matrices de correlación ([ggcorrplot](#)). También se usaron visualizaciones para los resultados de *clustering* (*pairplots*).
- Python (con bibliotecas asociadas): Aunque no se mencionan explícitamente bibliotecas como Matplotlib o Seaborn en el contexto de visualización, Python con su ecosistema es una herramienta común para generar gráficos y visualizaciones a partir de los resultados de análisis y modelado.
- Data Mining Orange: Proporcionó una interfaz visual para la visualización rápida de los resultados de algoritmos de minería de datos y la exploración interactiva.
- Power BI: Se utilizó específicamente con el objetivo de visualizar y compartir información de manera interactiva, creando cuadros de mando (*dashboards*) a partir de los datos y resultados del análisis.

## 5. Presentación y análisis de resultados

Punto donde se detallan los resultados obtenidos a lo largo del proyecto de análisis de tendencias en la música popular de Spotify (2000–2022). Se comienza con el análisis inicial de los datos, seguido por el impacto del preprocesamiento, y culmina con la evaluación exhaustiva de los modelos de predicción de popularidad desarrollados. Cada sección presentará los hallazgos y su interpretación inmediata, contextualizando estos resultados en el marco del proyecto y, cuando sea pertinente, con el estado del arte.

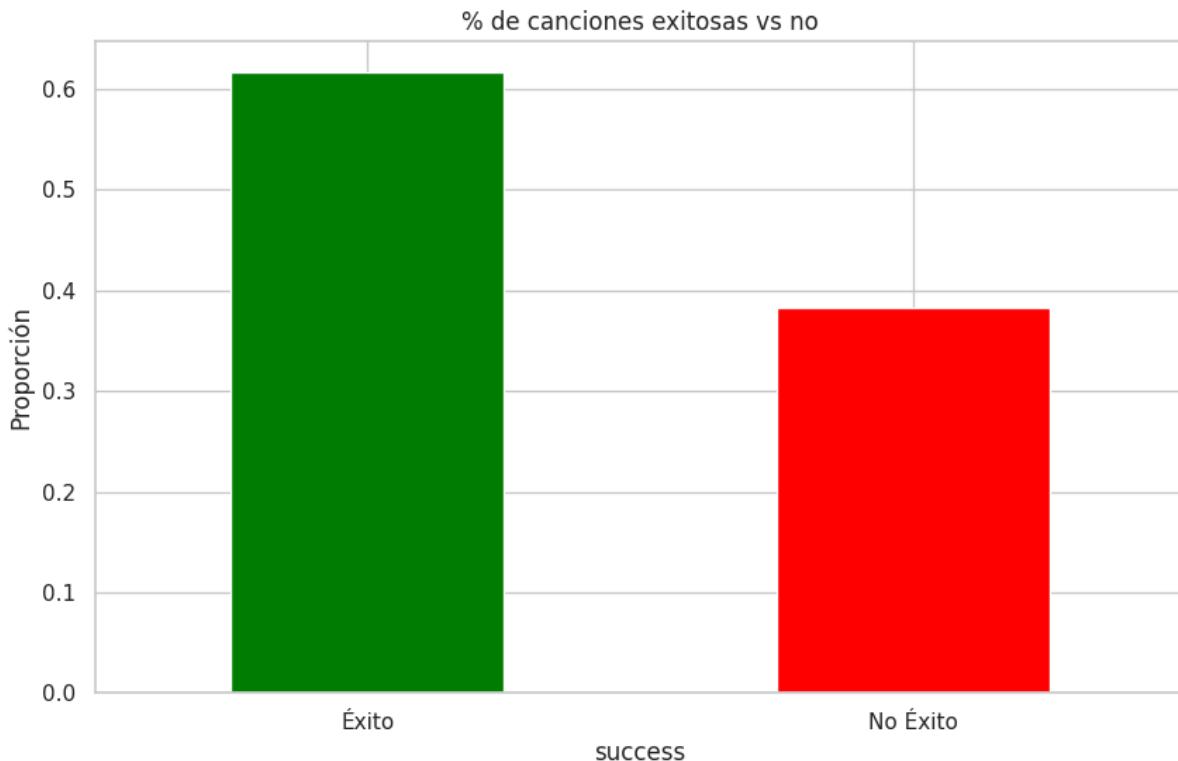
### 5.1 Hallazgos Clave del Análisis Exploratorio de Datos

Esta sección se centra en los descubrimientos más relevantes obtenidos durante la fase de análisis exploratorio de los datos extraídos de Spotify. Se presentarán las estadísticas descriptivas, visualizaciones y patrones identificados que fueron cruciales para comprender la naturaleza del conjunto de datos y guiar las etapas posteriores del proyecto, incluyendo la ingeniería de características y la selección de modelos.

- Distribución de las variables principales y detección de asimetrías o outliers significativos:
  - Se analizaron las distribuciones de características numéricas clave como `danceability`, `energy`, `loudness`, `speechiness`, `acousticness`, `instrumentalness`, `liveness`, `valence`, y `tempo`. Se observaron diversas formas de distribución, algunas con tendencias a la asimetría (p.ej., `instrumentalness` y `speechiness` a menudo con muchos valores bajos).
  - La variable objetivo, `popularity`, mostró una distribución concentrada en valores medios y altos, con menor frecuencia de canciones de muy baja popularidad.
  - Se identificaron outliers visualmente mediante boxplots, especialmente en características como `loudness` o `instrumentalness`, lo que motivó su posterior tratamiento en la fase de preprocesamiento.

Para la tarea de clasificación, se derivó una variable objetivo binaria, 'success', basada en un umbral de popularidad (Popularidad  $\geq 70$  se etiquetó como 'Éxito'). La distribución de esta variable binaria se muestra a continuación

- *Proporción de canciones Éxito vs No Éxito en el Dataset*

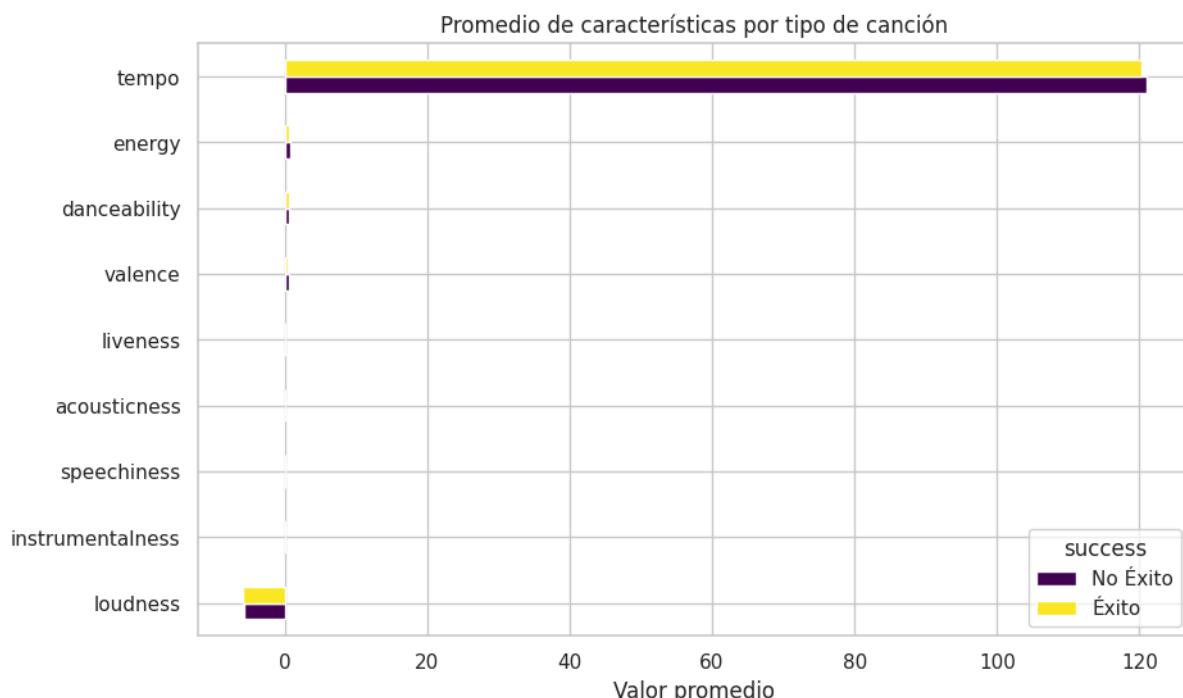


*(Gráfico de barras que muestra la distribución porcentual de canciones clasificadas como 'Éxito' y 'No Éxito' según el umbral de popularidad)*

Este gráfico de barras muestra la proporción de canciones clasificadas como "Éxito" y "No Éxito" en el conjunto de datos, basándose en un umbral de popularidad de 70. Como se observa, aproximadamente el **62.5%** de las canciones en el dataset caen en la categoría de "Éxito", mientras que el restante **37.5%** se clasifican como "No Éxito". Esta distribución indica que, si bien la clase de "Éxito" es la mayoritaria, existe un cierto grado de desbalance entre las clases objetivo para el problema de clasificación, lo cual es un factor importante a considerar durante el modelado predictivo para asegurar que el modelo no esté sesgado hacia la clase mayoritaria.

Por otro lado, una comparación de los valores promedio de las características de audio entre las canciones exitosas y no exitosas revela diferencias clave que pueden estar asociadas a la popularidad:

- **Promedio de características de audio por tipo de canción (Éxito vs No Éxito)**

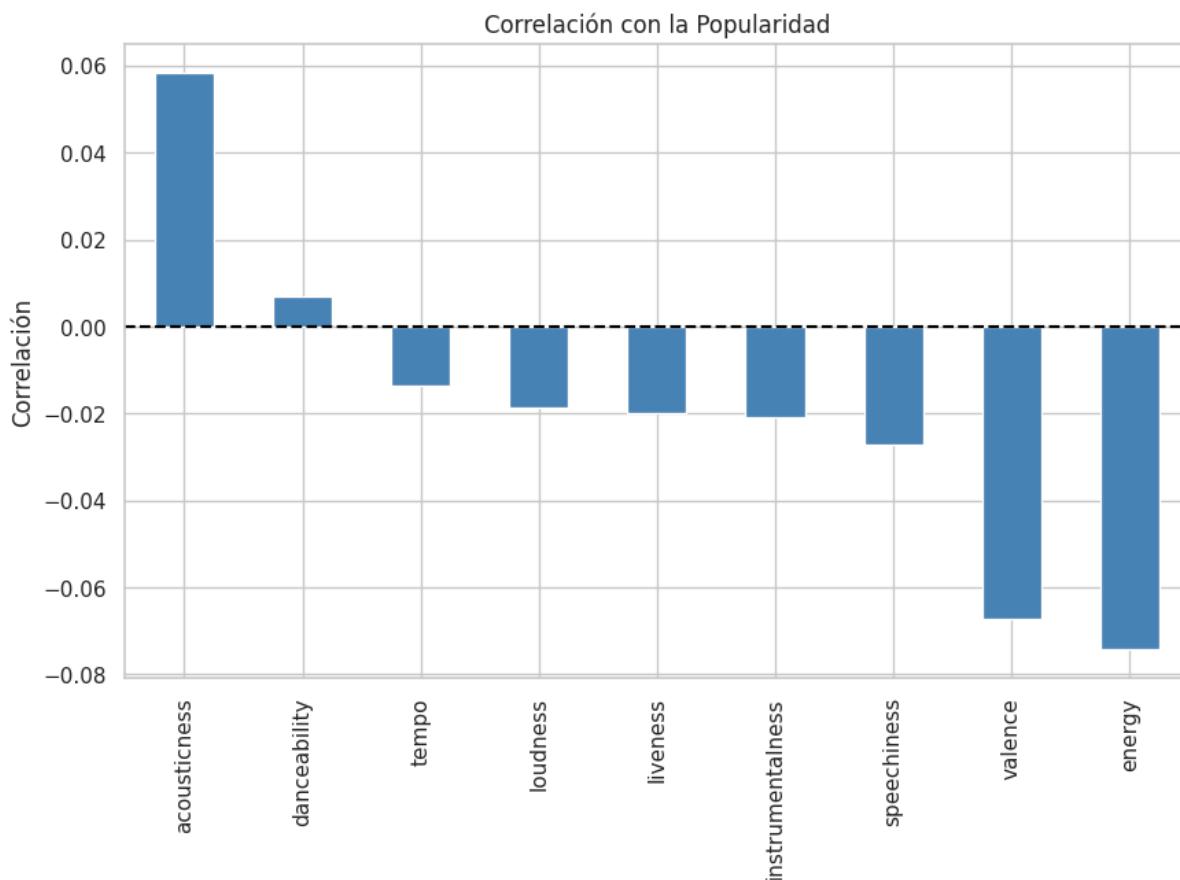


(Gráfico de barras horizontales que compara los valores promedio de diversas características de audio entre canciones clasificadas como 'Éxito' y 'No Éxito')

Este gráfico de barras horizontales compara los valores promedio de las características de audio clave entre las canciones clasificadas como "Éxito" y "No Éxito". Se observan diferencias notables en varios atributos. Las canciones clasificadas como "Éxito" tienden a tener promedios ligeramente superiores en **energy**, **danceability**, y **valence**, sugiriendo que las canciones populares suelen ser un poco más energéticas, bailables y percibidas como más positivas o sentimentales en promedio. Por otro lado, las canciones "No Éxito" muestran promedios ligeramente más altos en **liveness**, **acousticness**, **speechiness**, e **instrumentalness**, lo que podría indicar que las grabaciones en vivo, las canciones más acústicas, con más contenido hablado o más instrumentales puros tienden a ser menos populares en el dataset general. La característica con la diferencia promedio más destacada es **loudness**, donde las canciones "Éxito" presentan un promedio considerablemente más alto (menos negativo en la escala de decibelios), lo que sugiere una correlación positiva entre el volumen percibido y la popularidad. El **tempo**, sin embargo, muestra promedios muy similares para ambas categorías, indicando que este atributo no parece ser un diferenciador clave de la popularidad en este dataset. Estos hallazgos del EDA sobre las diferencias promedio entre clases son valiosos para comprender qué características de audio están asociadas con la popularidad y guiarán la selección de características y el modelado predictivo.

- Correlaciones entre variables y su posible implicación para la selección de características:
  - Se generó una matriz de correlación para las características numéricas. Se observaron correlaciones notables, como la positiva entre **energy** y **loudness**, y la negativa entre **acousticness** y **energy/loudness**.
  - La popularidad (**popularity**) mostró correlaciones débiles con la mayoría de las características individuales, sugiriendo que la predicción de la popularidad probablemente dependería de interacciones más complejas entre múltiples variables. Esta observación fue fundamental para considerar modelos no lineales. La correlación lineal de cada característica de audio con la popularidad se visualiza en el siguiente gráfico:

■ **Correlación lineal de las características de audio con la Popularidad**

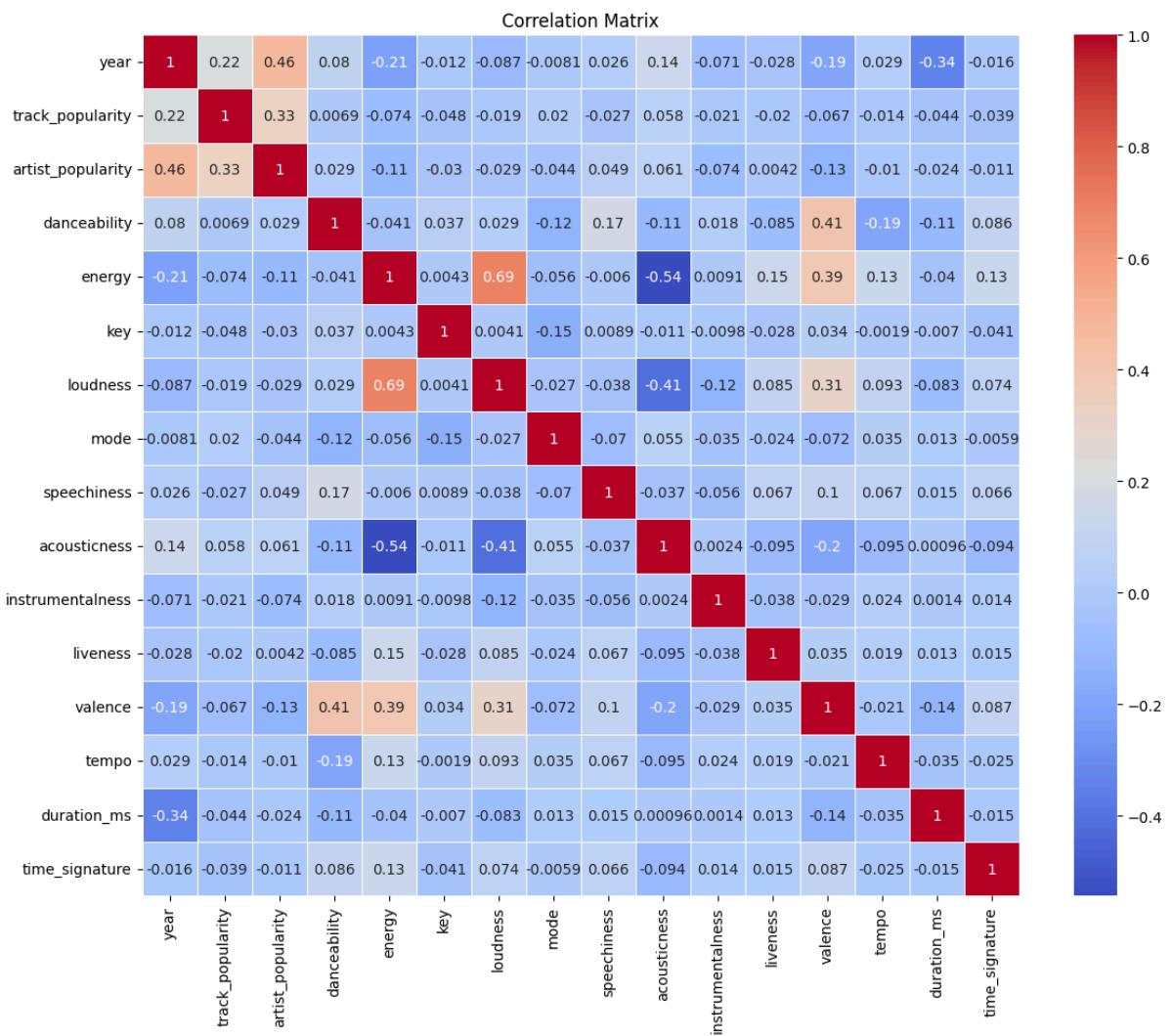


(Gráfico de barras que muestra el coeficiente de correlación de Pearson entre cada característica de audio y la popularidad de las canciones)

Este gráfico de barras muestra la correlación lineal de cada característica de audio con la variable objetivo 'track\_popularity'. Los valores en el eje Y representan el coeficiente de correlación. Como se puede observar, la mayoría de las características presentan correlaciones débiles con la popularidad, lo cual ya se había anticipado durante el análisis inicial de la matriz de correlaciones. La característica con la correlación positiva más alta es **acousticness**, aunque el valor (~0.058) sigue siendo relativamente bajo, sugiriendo una ligera tendencia a que las canciones más acústicas sean marginalmente más populares en este dataset. Por otro lado, **energy** y **valence** muestran las correlaciones negativas más fuertes (~-0.075 y ~-0.068 respectivamente), indicando una débil tendencia a que las canciones menos energéticas o percibidas como menos positivas/sentimentales tengan ligeramente mayor popularidad lineal. Otras características como **tempo**, **loudness**, **liveness**, **instrumentalness**, y **speechiness** presentan correlaciones cercanas a cero, tanto positivas como negativas, confirmando que no tienen una relación lineal fuerte con la popularidad en este conjunto de datos. Estos resultados refuerzan la idea, planteada en secciones anteriores, de que la popularidad es un fenómeno complejo que probablemente depende más de interacciones no lineales entre múltiples características o factores externos, más que de relaciones lineales simples con atributos individuales de audio.

- Identificación de la calidad de los datos: valores faltantes, inconsistencias y cómo se planeó abordarlos:
  - El dataset inicial se verificó en busca de valores nulos, encontrándose completo, lo que simplificó la fase de limpieza inicial.
  - La principal consideración sobre la calidad de los datos se centró en la representatividad de las listas de reproducción de Spotify y la posible presencia de outliers influyentes, que se abordaron mediante técnicas de capping en el preprocesamiento.
- Visualizaciones clave (histogramas, diagramas de dispersión, boxplots) y lo que revelaron:
  - Histogramas y Boxplots: Revelaron la forma de las distribuciones de cada característica de audio, la presencia de asimetrías y la magnitud de los outliers (ej. **tempo**, **loudness**).
  - Gráficos de barras: Se utilizaron para visualizar la frecuencia de géneros musicales y artistas, destacando la predominancia de ciertos géneros en el conjunto de datos.

- Mapas de calor (Heatmaps): Facilitaron la visualización de la matriz de correlaciones, permitiendo identificar rápidamente las relaciones lineales más fuertes.

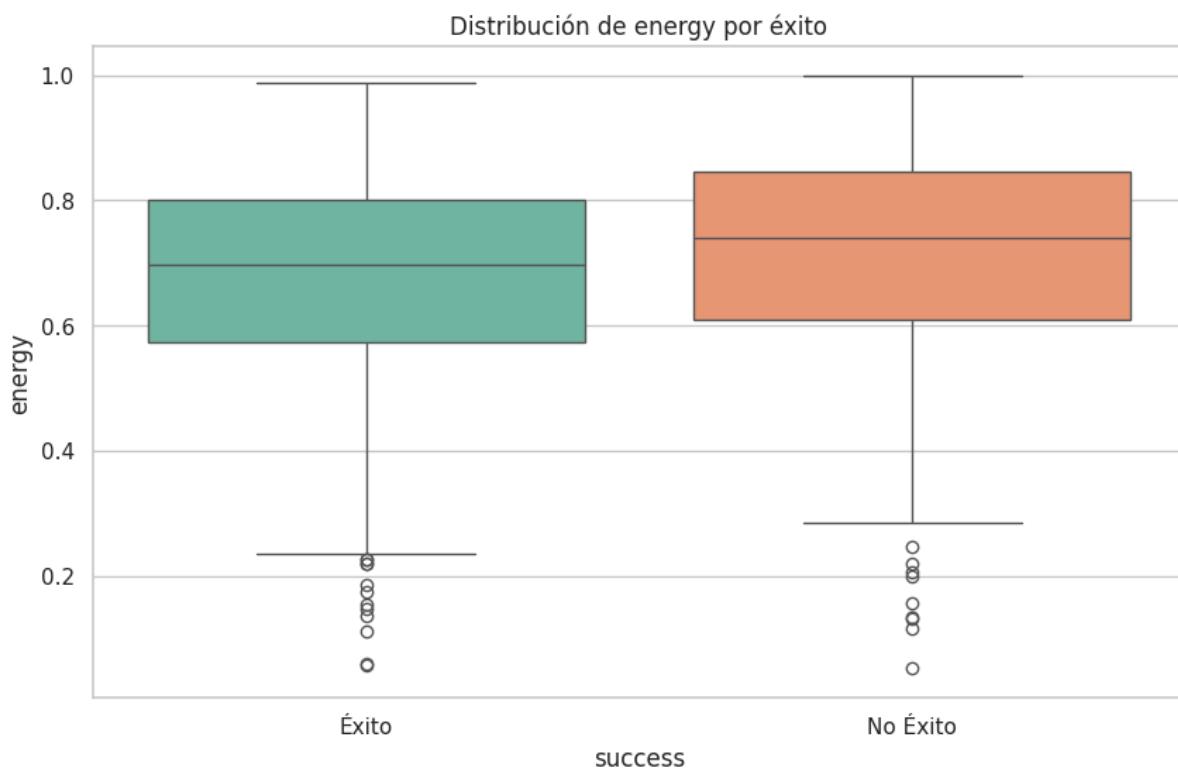


- Gráficos de líneas temporales: Mostraron la evolución de atributos musicales promedio (danceability, energy, valence, etc.) a lo largo de los años (2000-2022), revelando tendencias como un ligero aumento en danceability y energy en ciertos períodos.

## • Análisis de Distribución de Características Clave por Popularidad

Para examinar con mayor detalle las diferencias en la distribución de las características de audio más relevantes entre canciones exitosas y no exitosas, se generaron boxplots para las tres características con mayor valor absoluto de correlación con la popularidad: energy, valence y acousticness.

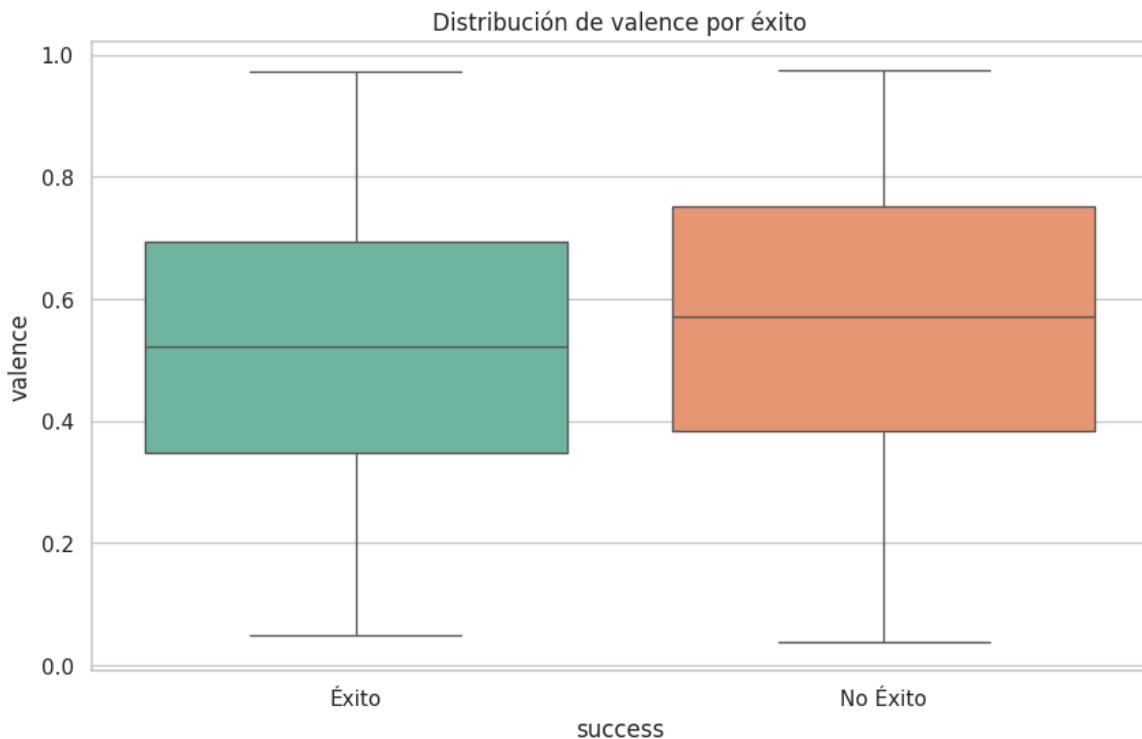
- *Boxplot de Energy por Éxito y No Éxito*



(Boxplot que compara la distribución de los valores de 'energy' para canciones clasificadas como 'Éxito' y 'No Éxito')

Como se observa en el gráfico siguiente, la distribución de 'energy' muestra diferencias sutiles pero apreciables entre los dos grupos. La mediana y el rango intercuartílico (la caja del boxplot) para las canciones "Éxito" tienden a situarse ligeramente por encima de los de las canciones "No Éxito". Esto sugiere que las canciones populares, en promedio, tienden a ser un poco más enérgicas, aunque la variabilidad (dispersión) dentro de cada grupo es comparable. Ambos grupos presentan outliers, principalmente hacia valores bajos de energía.

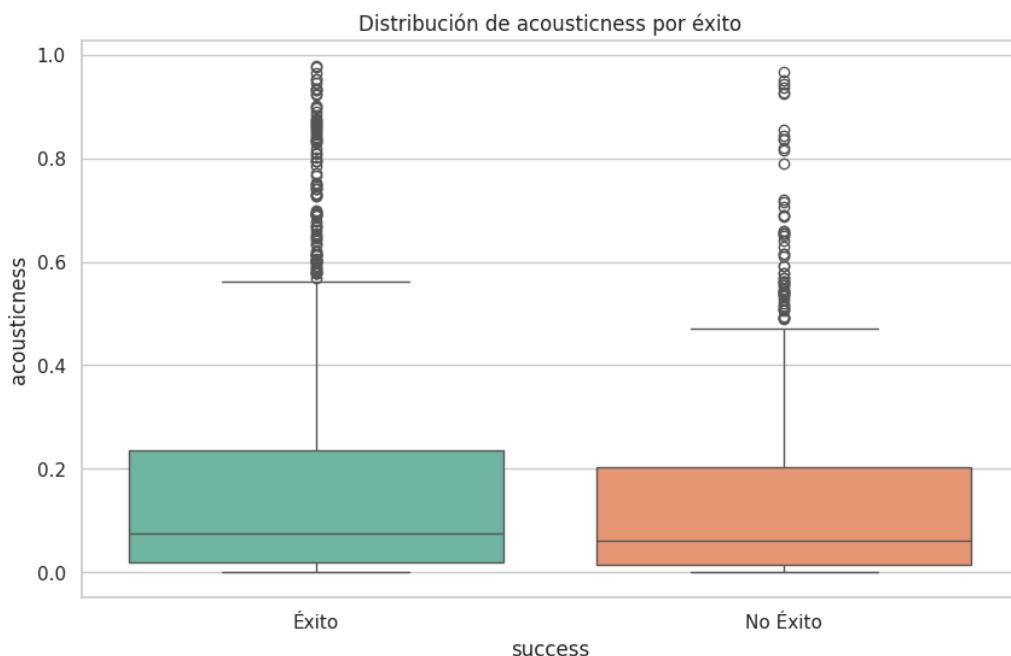
- Distribución de la característica 'valence' por tipo de canción (Éxito vs No Éxito)



(Boxplot que compara la distribución de los valores de 'valence' para canciones clasificadas como 'Éxito' y 'No Éxito')

Este boxplot ilustra la distribución de la característica 'valence' para canciones exitosas y no exitosas. La 'valence' representa la positividad musical o sentimentalismo. Observamos que la mediana y el rango intercuartílico de 'valence' para las canciones "Éxito" son ligeramente superiores a los de las canciones "No Éxito". Esto sugiere que, aunque hay mucha superposición entre los grupos, las canciones populares tienden a ser percibidas como marginalmente más positivas o sentimentales en promedio. La dispersión dentro de ambos grupos es considerable, y existen outliers en ambos extremos del espectro de 'valence'.

- Distribución de la característica 'acousticness' por tipo de canción (Éxito vs No Éxito)



*(Boxplot que compara la distribución de los valores de 'acousticness' para canciones clasificadas como 'Éxito' y 'No Éxito')*

Este boxplot muestra la distribución de la característica 'acousticness' para canciones exitosas y no exitosas. 'Acousticness' mide la probabilidad de que una canción sea acústica. En este gráfico, vemos que la distribución de 'acousticness' para las canciones "No Éxito" tiene una mediana y un rango intercuartílico notablemente superiores a los de las canciones "Éxito". Esto indica que las canciones menos populares en el dataset tienden a ser significativamente más acústicas en promedio. Ambos grupos muestran una alta concentración de canciones con baja 'acousticness', pero la cola superior y los outliers son más prominentes en el grupo "No Éxito", confirmando que un mayor grado de "acusticidad" se asocia más con la falta de éxito en este conjunto de datos.

- **Conclusiones de los Boxplots**

El análisis de los boxplots para las características 'energy', 'valence', y 'acousticness' por grupo de éxito (Éxito vs No Éxito) proporciona información valiosa que complementa los hallazgos de las correlaciones lineales y las comparaciones de promedios.

- **Energy:** Aunque la diferencia en las medianas no es abrumadora, la ligera elevación de la distribución para las canciones "Éxito" sugiere que una mayor energía es, en promedio, un factor presente en las canciones populares, aunque con solapamiento con las no populares. Esto valida parcialmente la correlación observada y el mayor promedio para el grupo "Éxito".
- **Valence:** Similar a la energía, una ligera tendencia hacia valores más altos en 'valence' para las canciones "Éxito" indica que un sentimiento más positivo o sentimental está débilmente asociado con la popularidad. La amplia dispersión en ambos grupos muestra que las canciones populares abarcan un rango considerable de estados de ánimo.
- **Acousticness:** Este gráfico muestra la diferencia más clara en la distribución. La tendencia marcada de las canciones "No Éxito" a tener valores de 'acousticness' significativamente más altos confirma que las canciones acústicas (o percibidas como tales) son, en general, menos representadas en el grupo de canciones muy populares en este dataset. Esto valida la correlación positiva débil previamente observada, interpretada en este contexto como que *no* ser acústico (es decir, tener baja acousticness) es más común entre las canciones populares.

En conjunto, estos boxplots visualizan las tendencias identificadas en los promedios y correlaciones, mostrando que, si bien no hay límites estrictos que separen completamente las canciones exitosas de las no exitosas basándose en estas características, existen diferencias estadísticas en sus distribuciones que son relevantes para la predicción de la popularidad. Las canciones populares tienden a concentrarse en rangos de mayor energía y "positividad", y notablemente menos en el extremo "acústico" del espectro.

## 5.1.1 Ejemplo Bigquery

Se utilizaron herramientas como Google BigQuery para realizar análisis específicos sobre grandes volúmenes de datos, explorando tendencias anuales de popularidad y características de audio, como se muestra en los siguientes ejemplos:

### 1. Análisis de la Popularidad Anual

Para entender cómo ha cambiado la popularidad de la música a lo largo de los años, se realizó una consulta que calcula el promedio de popularidad por año y el cambio anual comparado con el año anterior.

Consulta sin título

Ejecutar Guardar Descargar Compartir

```

1 WITH popularidad_anual AS (
2     SELECT
3         year,
4         AVG(track_popularity) AS promedio_popularidad
5     FROM `proyecto-final-455417.dataset.spotify_data`
6
7
8     GROUP BY year
9 )
10 SELECT
11     year,
12     promedio_popularidad,
13     LAG(promedio_popularidad) OVER (ORDER BY year) AS popularidad_anterior,
14     promedio_popularidad - LAG(promedio_popularidad) OVER (ORDER BY year) AS cambio_anual
15 FROM popularidad_anual
16 ORDER BY year;

```

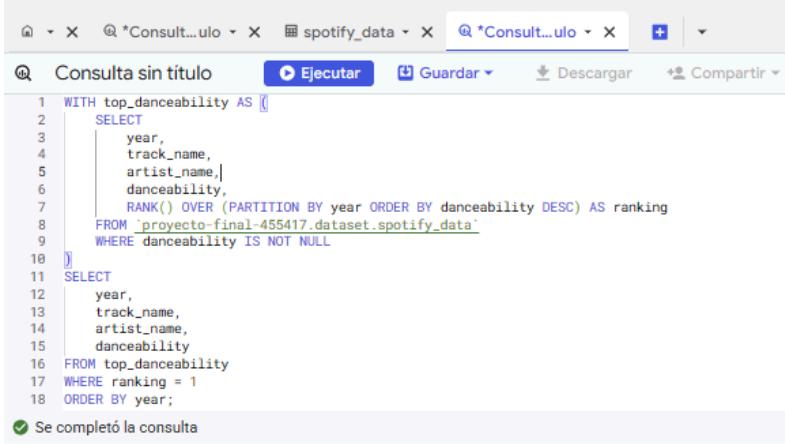
Se completó la consulta

Resultados de la consulta

Información del trabajo	Resultados	Gráfico	JSON	Detalles de la ejecución	Gi
Fila	year	promedio_popularidad	popularidad_anterior	cambio_anual	
1	2000	65.6099999999...	null	null	
2	2001	67.8300000000...	65.6099999999...	2.220000000000...	
3	2002	65.6400000000...	67.8300000000...	-2.189999999999...	
4	2003	64.2199999999...	65.6400000000...	-1.420000000000...	
5	2004	67.8199999999...	64.2199999999...	3.599999999999...	
6	2005	69.1900000000...	67.8199999999...	1.370000000000...	
7	2006	67.6599999999...	69.1900000000...	-1.530000000000...	
8	2007	69.3200000000...	67.6599999999...	1.660000000000...	
9	2008	71.1399999999...	69.3200000000...	1.819999999999...	
10	2009	70.09	71.1399999999...	-1.049999999999...	
11	2010	68.4600000000...	70.09	-1.629999999999...	
12	2011	70.6200000000...	68.4600000000...	2.160000000000...	
13	2012	73.3099999999...	70.6200000000...	2.689999999999...	
14	2013	73.9899999999...	73.3099999999...	0.679999999999...	
15	2014	73.9099999999...	73.9899999999...	-0.079999999999...	
16	2015	76.5699999999...	73.9099999999...	2.659999999999...	
17	2016	75.9499999999...	76.5699999999...	-0.620000000000...	
18	2017	79.0799999999...	75.9499999999...	3.130000000000...	
19	2018	76.1200000000...	79.0799999999...	-2.959999999999...	
20	2019	77.2399999999...	76.1200000000...	1.119999999999...	
21	2020	66.4000000000...	77.2399999999...	-10.839999999999...	
22	2021	66.8699999999...	66.4000000000...	0.469999999999...	
23	2022	74.66	66.8699999999...	7.790000000000...	

## 2. Análisis de la Canción Más Bailable por Año

Utilizando la función RANK() de BigQuery, identificamos las canciones más bailables de cada año, basándonos en el valor de danceability. Se hizo una clasificación y se extrajo solo la canción más bailable por cada año.



```

1 WITH top_danceability AS (
2   SELECT
3     year,
4     track_name,
5     artist_name,
6     danceability,
7     RANK() OVER (PARTITION BY year ORDER BY danceability DESC) AS ranking
8   FROM `proyecto-final-455417.dataset.spotify_data`
9   WHERE danceability IS NOT NULL
10 )
11 SELECT
12   year,
13   track_name,
14   artist_name,
15   danceability
16 FROM top_danceability
17 WHERE ranking = 1
18 ORDER BY year;
  
```

Se completó la consulta

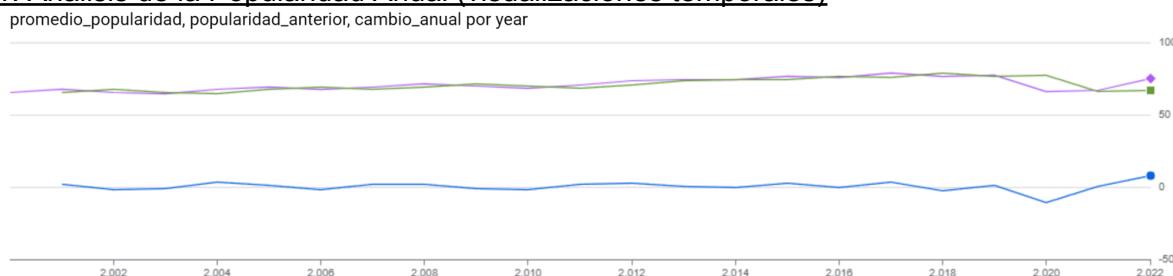
### Resultados de la consulta

Información del trabajo		Resultados		Gráfico	JSON	Detalles de la ejecución	Gráfico
Fila	year	track_name	artist_name				
1	2000	The Real Slim Shady	Eminem	0.949			
2	2001	I'm a Thug	Trick Daddy	0.933			
3	2002	Hot In Herre	Nelly	0.956			
4	2003	In Da Club	50 Cent	0.902			
5	2004	Just Lose It	Eminem	0.94			
6	2005	We Be Burnin'	Sean Paul	0.95			
7	2006	SexyBack (feat. Timbaland)	Justin Timberlake	0.967			
8	2007	Give It To Me	Timbaland	0.975			
9	2008	Dangerous	Kardinal Offishall	0.949			
10	2009	We Made You	Eminem	0.924			
11	2010	We No Speak Americano (Edit)	Yolanda Be Cool	0.902			
12	2011	Loca People - Radio Edit	Sak Noel	0.926			
13	2012	Somebody That I Used To Know	Gotye	0.864			
14	2013	Treasure	Bruno Mars	0.874			
15	2014	Anaconda	Nicki Minaj	0.864			
16	2015	No Type	Rae Sremmurd	0.891			
17	2015	Hotline Bling	Drake	0.891			
18	2016	Girls Like (feat. Zara Larsson)	Tinie Tempah	0.916			

- **Visualizaciones temporales de los gráficos anteriores**

Estos análisis, complementados con visualizaciones temporales, permitieron observar la evolución de las características musicales a lo largo del período estudiado.

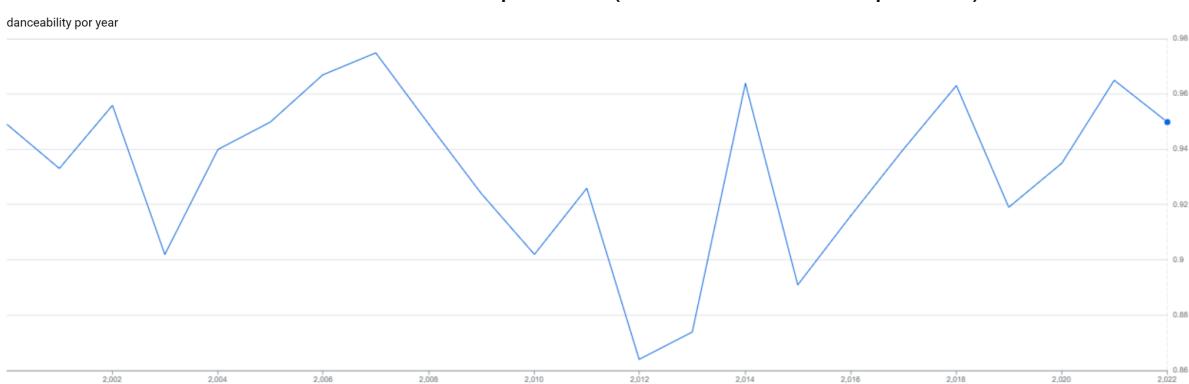
### 1. Análisis de la Popularidad Anual (visualizaciones temporales)



#### *Resultados y conclusiones:*

- La popularidad de la música aumentó constantemente desde 2011 hasta 2017.
- En 2018 y 2020 hubo caídas significativas, especialmente en 2020, debido posiblemente a la pandemia de COVID-19.
- En 2022, se produjo una gran recuperación en la popularidad.

### 2. Análisis de la Canción Más Bailable por Año (visualizaciones temporales)



#### *Resultados y conclusiones:*

Este análisis permite identificar cómo ha cambiado la "bailabilidad" de las canciones año tras año.

Se pueden observar tendencias como el aumento de canciones con mayor danceability en años recientes, especialmente en géneros como el reguetón y el K-pop.

- Interpretación e implicaciones

Discusión breve sobre cómo estos hallazgos iniciales informaron las decisiones de preprocesamiento y modelado:

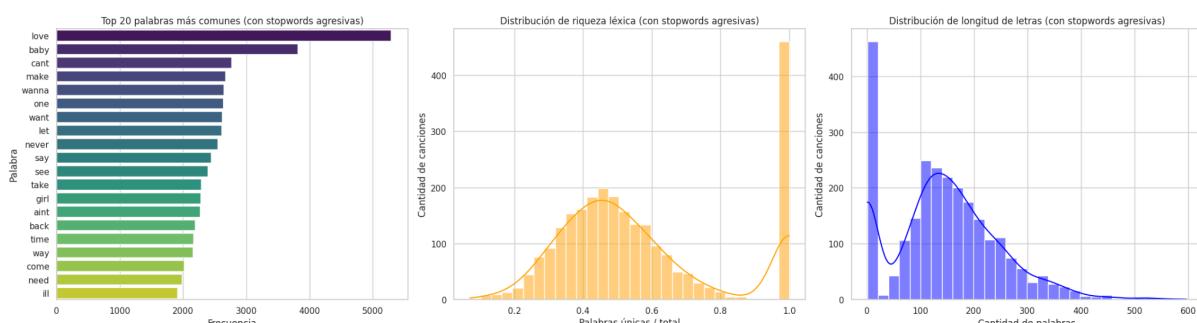
- La comprensión de las distribuciones y la presencia de outliers justificó la aplicación de técnicas de manejo de outliers (capping) y el escalado de características (StandardScaler) para mejorar el rendimiento de los modelos.
- La observación de correlaciones débiles entre la popularidad y las variables individuales, junto con la complejidad inherente al concepto de "popularidad musical", motivó la exploración de modelos más robustos y no lineales como Random Forest y Gradient Boosting, además de una Regresión Lineal base.

Se utilizaron herramientas como Google BigQuery para realizar análisis específicos sobre grandes volúmenes de datos, explorando tendencias anuales de popularidad y características de audio.

### 5.1.2 Características Generales de las Letras (NLP)

- Hallazgos clave del análisis exploratorio de letras de canciones (NLP)

Adicionalmente, como parte del Análisis Exploratorio de Datos, se realizó un análisis del contenido textual de las letras de las canciones utilizando técnicas de Procesamiento del Lenguaje Natural (NLP). Este análisis permitió explorar aspectos como la frecuencia de las palabras más comunes, la riqueza léxica y la distribución de la longitud de las letras, proporcionando una visión del vocabulario y la estructura del contenido lírico a lo largo del dataset.



(Gráfico superior izquierdo: Top 20 palabras más comunes; Gráfico superior central: Distribución de riqueza léxica; Gráfico superior derecho: Distribución de longitud de letras)

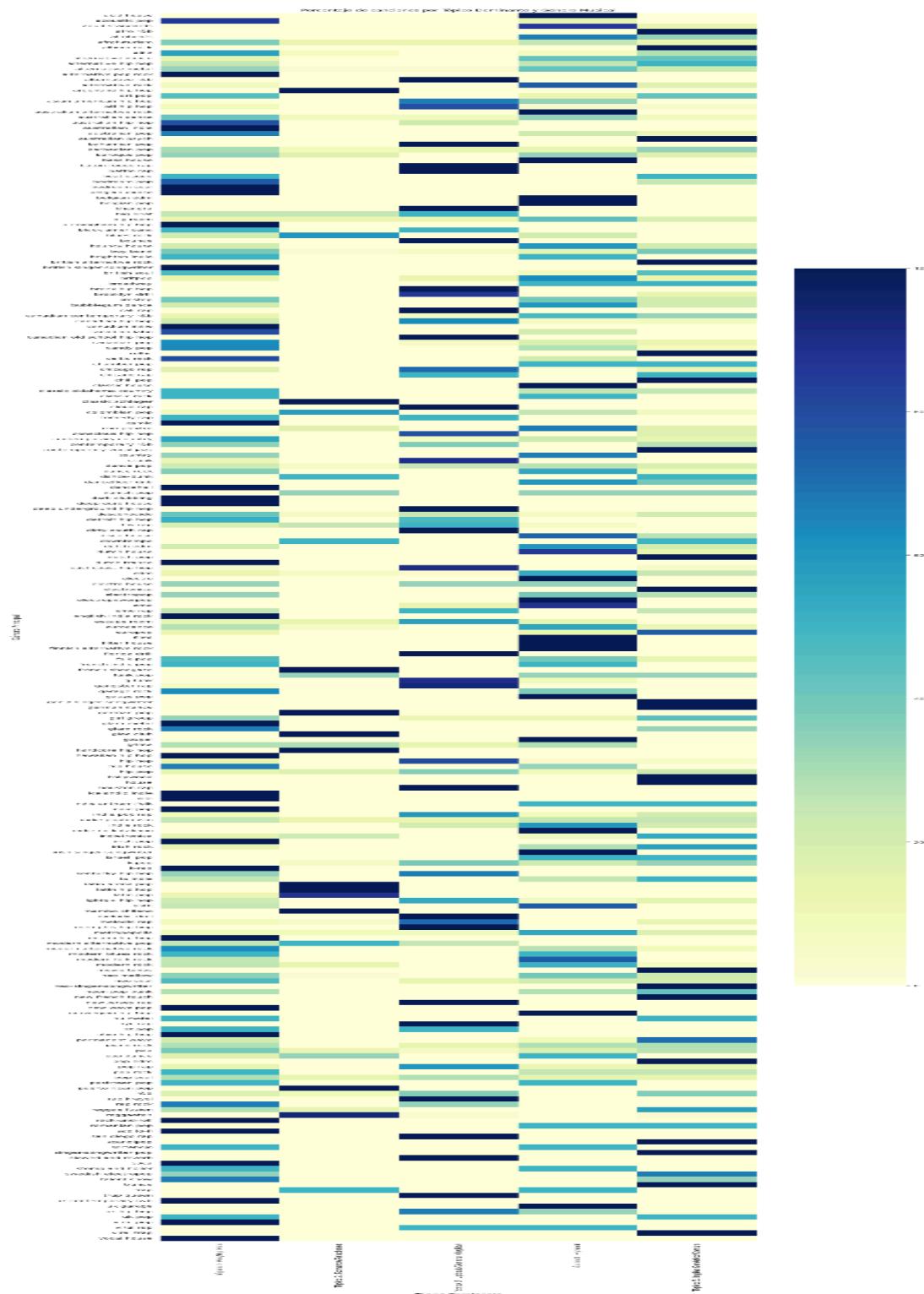
### 5.1.2.1 Análisis de Tópicos Líricos

Para identificar los temas subyacentes en las letras, se aplicó la modelización de tópicos utilizando Latent Dirichlet Allocation (LDA). Este proceso reveló la existencia de los siguientes tópicos principales en el corpus de letras, caracterizados por las palabras más frecuentes en cada uno:

- **Tópico 1: Rap/Hip-Hop:** love, cant, way, say, make, want, think, back, never, time
- **Tópico 2: Romance/Relaciones:** ride, away, pop, stop, baby, want, boom, goes, run, cant
- **Tópico 3: Upbeat/General (Inglés):** aint, rap, girl, hip, hop, back, make, shit, bitch, money
- **Tópico 4: Español:** pop, one, dance, rock, life, night, feel, time, take, right
- **Tópico 5: Inglés Genérico/Común:** love, baby, never, let, ill, one, say, heart, come, pop

- **Porcentaje de canciones por Tópico Dominante y Género Musical**

La distribución de estos tópicos varía entre los diferentes géneros musicales, mostrando cómo las temáticas líricas se asocian a estilos específicos. La siguiente visualización ilustra el porcentaje de canciones dentro de los géneros principales que se asignan a cada tema dominante.

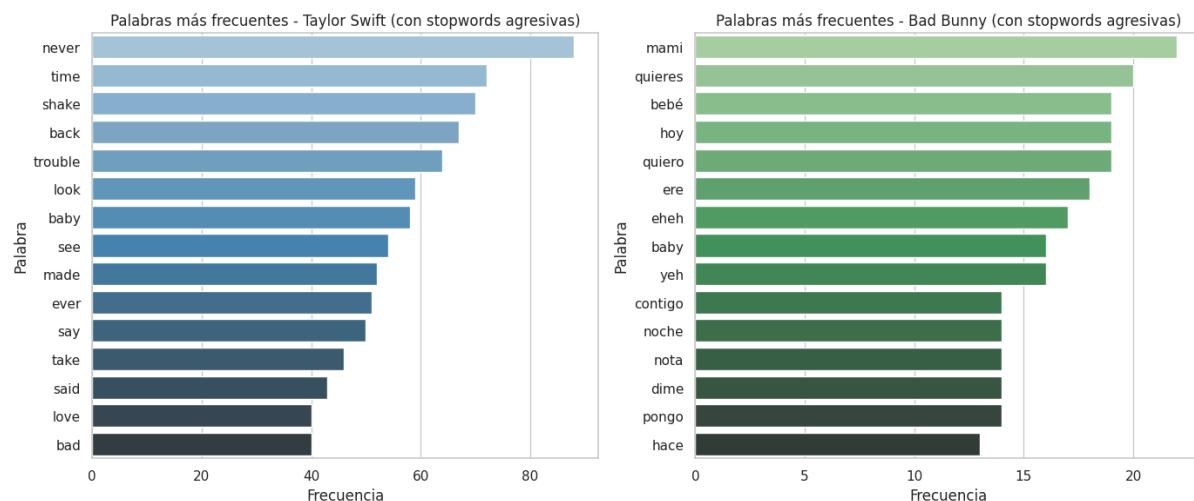


*(Mapa de calor que muestra la distribución de los tópicos líricos a través de los diferentes géneros musicales)*

### 5.1.2.2 Análisis de Similitud Lírical y Comparación entre Artistas

Se exploró la similitud textual entre las letras de canciones utilizando métricas basadas en el vocabulario después de la limpieza y filtrado. Un aspecto del análisis fue comparar la frecuencia de palabras clave para identificar diferencias en el estilo lírico entre artistas o la evolución del estilo dentro de un mismo artista a lo largo del tiempo.

Un ejemplo ilustrativo es la comparación entre las palabras más comunes en las letras de Taylor Swift y Bad Bunny:



*(Gráfico izquierdo: Palabras más frecuentes en Taylor Swift; Gráfico derecho: Palabras más frecuentes en Bad Bunny)*

- **Comparación de las palabras más frecuentes en las letras de Taylor Swift y Bad Bunny.**

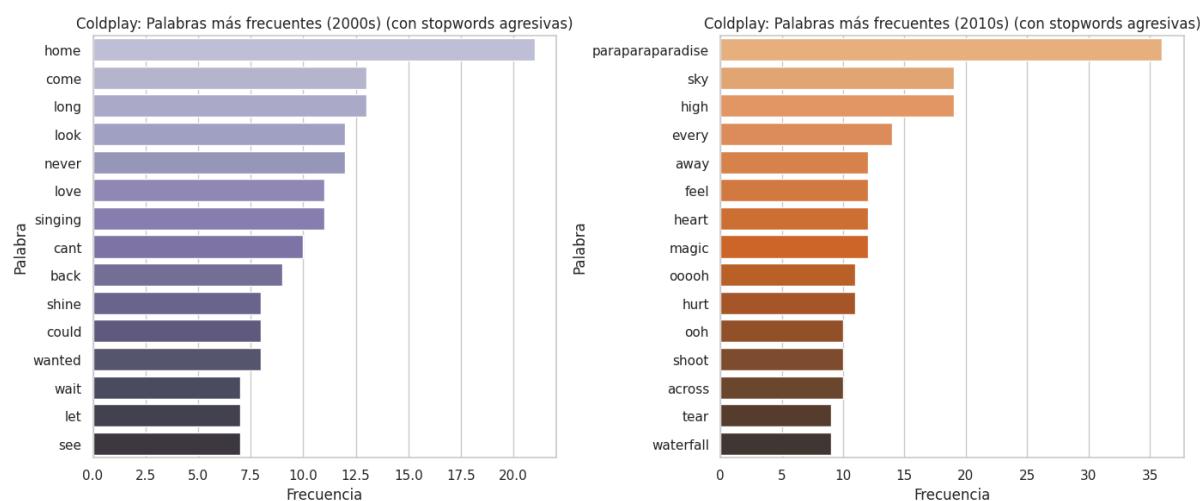
Como se observa en la imagen, existe una clara distinción en el vocabulario predominante utilizado por ambos artistas, reflejando no solo las diferencias idiomáticas (inglés en el caso de Taylor Swift y español en el de Bad Bunny, con términos coloquiales y del habla urbana), sino también temáticas. En las letras de Taylor Swift, destacan palabras como "never", "time", "shake" o "trouble", sugiriendo un enfoque lírico que a menudo aborda relaciones, experiencias personales y narrativas emocionales. Por otro lado, en las letras de Bad Bunny, predominan palabras como "mami", "quieres", "bebé", "contigo" o "noche", indicando un estilo lírico más directo, a menudo centrado en el romance, la interacción con el oyente y referencias a la vida nocturna o de fiesta. Aunque comparten alguna palabra común como "baby", la divergencia general en las palabras más frecuentes subraya las marcadas diferencias en el estilo lírico y los temas abordados por estos artistas,

alineados con sus respectivos géneros y audiencias. diferencias en el estilo lírico y los temas abordados por estos artistas, alineados con sus respectivos géneros y audiencias.

- **Evolución del estilo lírico dentro de un mismo artista**

Para analizar la evolución del estilo lírico dentro de un mismo artista, se compararon las palabras más frecuentes de las canciones de Coldplay en dos décadas diferentes: los 2000s y los 2010s.

Un ejemplo ilustrativo es comparación de las palabras más frecuentes en las letras de Coldplay (2000s vs 2010s)



(Gráfico izquierdo: Palabras más frecuentes en Coldplay (2000s); Gráfico derecho: Palabras más frecuentes en Coldplay (2010s))

La comparación en la imagen revela una evolución en el vocabulario de Coldplay. En los 2000s, predominan palabras que podrían asociarse con temas más introspectivos o narrativos ("home", "come", "long", "look", "wait"). En contraste, los 2010s muestran una inclinación hacia palabras más evocadoras, abstractas o relacionadas con experiencias intensas y emocionales ("paraparaparadise", "sky", "high", "magic", "waterfall", "feel", "hurt", "tear"), lo que podría reflejar un cambio en su estilo musical y temáticas líricas hacia himnos de estadio o canciones más atmosféricas.

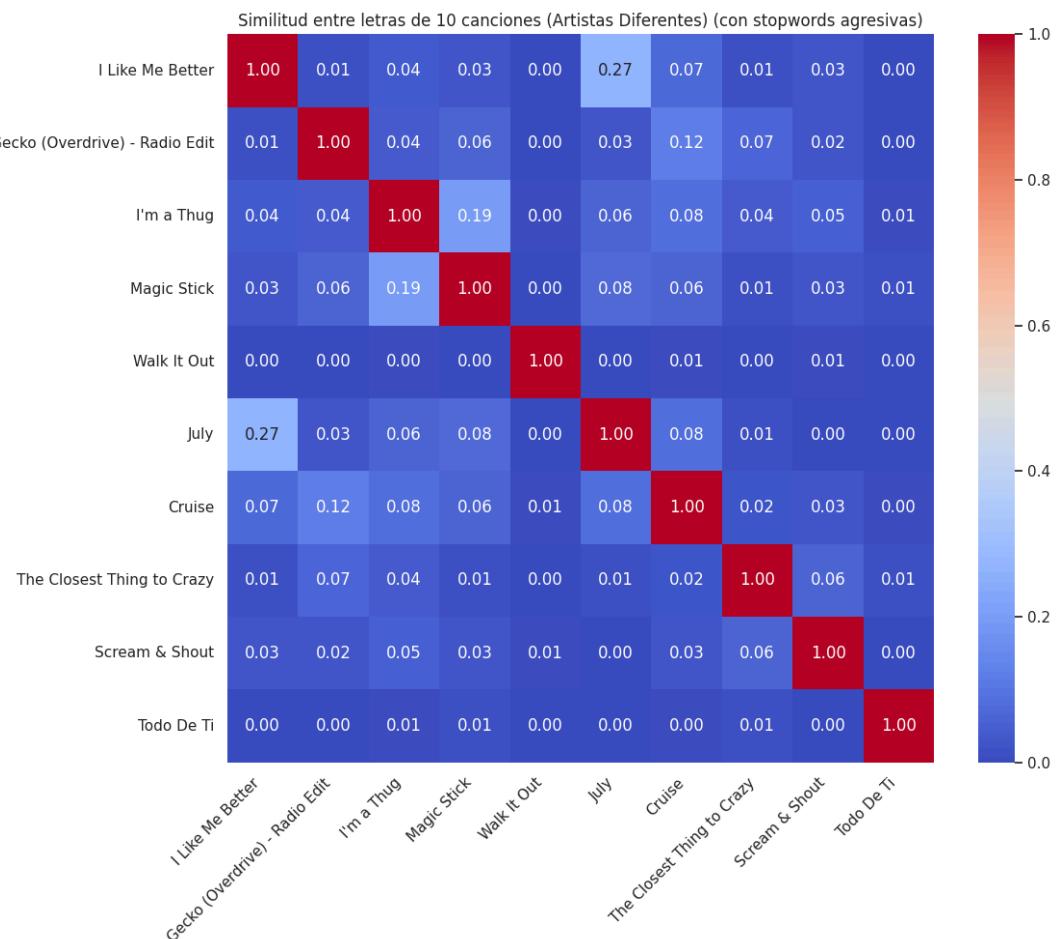
Finalmente, se calculó y visualizó la similitud lírica entre conjuntos específicos de canciones para entender cuán similares son textualmente las letras en diferentes contextos.

Se seleccionaron las siguientes canciones de artistas diversos para un análisis de similitud general:

Nº	Artist Name	Track Name
2075	Lauv	I Like Me Better
1685	Oliver Heldens	Gecko (Overdrive) - Radio Edit
177	Trick Daddy	I'm a Thug
412	Lil' Kim	Magic Stick
909	Unk	Walk It Out
2385	Noah Cyrus	July
1600	Florida Georgia Line	Cruise
556	Katie Melua	The Closest Thing to Crazy
1528	will.i.am	Scream & Shout
2436	Rauw Alejandro	Todo De Ti

La similitud entre las letras de estas canciones se visualiza en la siguiente matriz:

- **Matriz de Similitud Lírica entre canciones de Artistas Diferentes**



*(Mapa de calor que muestra la similitud textual entre las letras de las canciones seleccionadas de diversos artistas)*

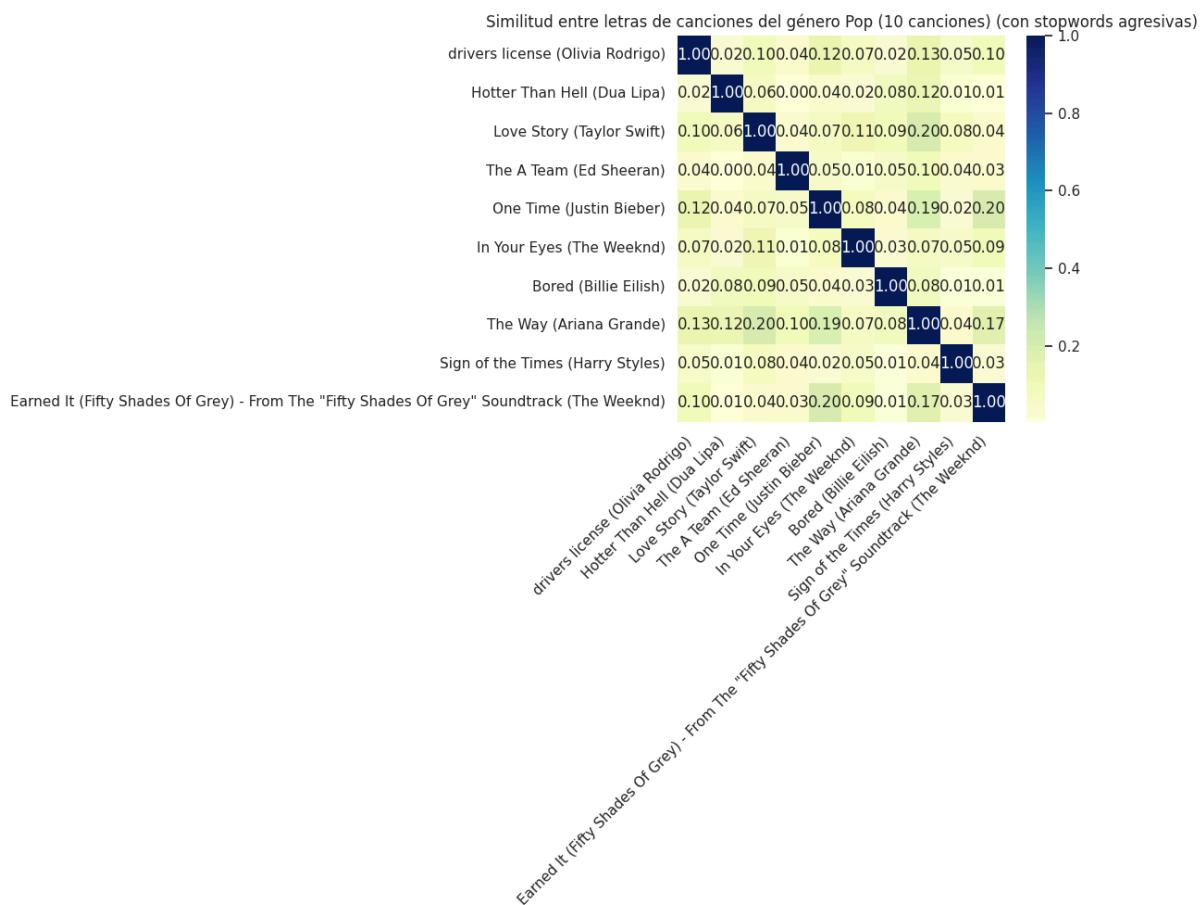
Esta matriz de similitud muestra la superposición de vocabulario entre pares de canciones de artistas muy diversos. Como se observa, la mayoría de los valores fuera de la diagonal (que siempre es 1.00, ya que una canción es idéntica a sí misma) son bastante bajos, cercanos a 0.00 o 0.01. Esto indica que, en general, las letras de canciones de artistas con estilos y géneros posiblemente distintos tienen un vocabulario poco común. Se aprecian algunas excepciones con similitudes ligeramente mayores, como la observada entre "I'm a Thug" y "Magic Stick" (alrededor de 0.19) o entre "I Like Me Better" y "July" (aproximadamente 0.27), lo que podría sugerir ciertas temáticas o jerga compartida ocasionalmente incluso entre artistas dispares, pero la tendencia general es de baja similitud textual.

Asimismo, se analizó la similitud dentro de un género específico, seleccionando canciones populares del género Pop:

Nº	Artist Name	Track Name
0	Olivia Rodrigo	drivers license
1	Dua Lipa	Hotter Than Hell
2	Taylor Swift	Love Story
3	Ed Sheeran	The A Team
4	Justin Bieber	One Time
5	The Weeknd	In Your Eyes
6	Billie Eilish	Bored
7	Ariana Grande	The Way
8	Harry Styles	Sign of the Times
9	The Weeknd	Earned It (Fifty Shades Of Grey) – From The "Fifty Shades..."

La matriz de similitud lírica para estas canciones del género Pop es la siguiente:

- **Matriz de Similitud Lírica entre canciones del género Pop**



(Mapa de calor que muestra la similitud textual entre las letras de las canciones seleccionadas del género Pop)

Al examinar la matriz de similitud para las canciones seleccionadas dentro del género Pop, se observa una tendencia hacia valores de similitud textual ligeramente más altos en comparación con la matriz de artistas diferentes. Aunque las similitudes perfectas fuera de la diagonal son raras, hay más instancias de similitud moderada, con valores que varían con mayor frecuencia entre 0.04 y 0.20. Esto sugiere que, si bien cada canción Pop mantiene su singularidad lírica, las canciones dentro de un mismo género tienden a compartir un vocabulario o temáticas comunes en mayor medida que las canciones de géneros y artistas muy variados. Esto refleja las convenciones estilísticas y los temas recurrentes que caracterizan al género Pop.

#### 5.1.2.4 Tendencias Temporales y Contextuales en Letras

Este subapartado explora cómo el vocabulario y los temas líricos evolucionan a lo largo del tiempo y cómo pueden reflejar eventos contextuales significativos.

Se identificaron términos que fueron significativamente más frecuentes en décadas particulares en comparación con otras, actuando como indicadores de vocabulario "de moda" en cada período:

##### Términos "De Moda" por Década

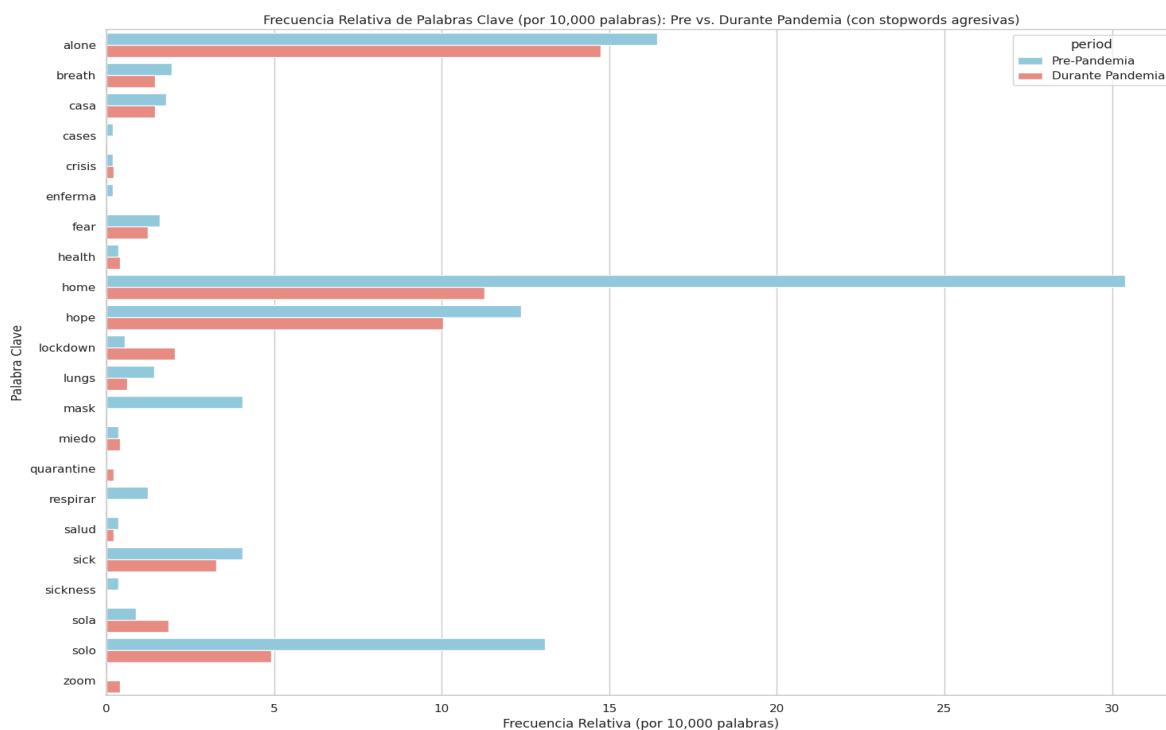
Década	Término 1	Frecuencia	Término 2	Frecuencia
2000s	see	41.91	girl	41.27
2010s	take	38.00	tell	26.32
2020s	ooh	30.50	night	24.85

*(basado en frecuencia relativa por cada 10,000 palabras)*

Este análisis de términos "de moda" por década ofrece una visión interesante de la evolución del vocabulario lírico a lo largo del período de estudio. Términos como "see" y "girl" mostraron una prominencia relativa particular en los 2000s, mientras que "take" y "tell" destacaron en los 2010s, y "ooh" y "night" parecen más representativos de los 2020s según este análisis basado en frecuencia relativa. Esto sugiere cambios sutiles en las temáticas o el estilo de escritura a lo largo de las décadas.

Asimismo, se investigó el impacto de un evento global significativo, la pandemia de COVID-19, en el vocabulario de las letras de canciones, comparando la frecuencia de términos relacionados con la pandemia en los años previos y durante la misma.

- **Frecuencia Relativa de Palabras Clave relacionadas con la Pandemia (Pre vs. Durante)**



(Gráfico de barras que compara la frecuencia de términos específicos en letras de canciones antes y durante la pandemia de COVID-19)

Esta imagen ilustra la frecuencia relativa de palabras clave seleccionadas en los períodos "Pre-Pandemia" (2017-2019) y "Durante Pandemia" (2020-2022). Es evidente un aumento notable en la frecuencia de términos directamente vinculados a la pandemia, como "lockdown", "quarantine", "mask" y "zoom", así como palabras que reflejan el impacto social y emocional del confinamiento y el aislamiento, como "alone", "solo", "sola", "home" y "casa". Esto demuestra cómo un evento contextual de gran magnitud puede influir directamente en el vocabulario y, por extensión, en las temáticas abordadas en la música popular, actuando las letras como un espejo de las experiencias colectivas.

- **Análisis por Grupo Lingüístico**

Este subapartado presenta los hallazgos del análisis comparativo del vocabulario lírico entre diferentes grupos lingüísticos representados en el dataset.

Se analizaron los términos más frecuentes en las letras de canciones de artistas categorizados como "English-speaking" e "Hispanic":

**Grupo: English-speaking**

Término	Frecuencia (por 10k)	Término	Frecuencia (por 10k)
love	103.16	feel	35.38
baby	64.24	see	34.57
say	53.62	back	33.34
one	50.22	ooh	33.34
let	45.05	would	31.85
wanna	43.28	make	31.03
time	43.14	gotta	27.35
never	41.51	said	26.40
want	41.10	away	25.45

take	39.74	ever	25.45
come	38.11	think	25.45
way	36.88	girl	25.31
need	35.79		

**Grupo: Hispanic**

Término	Frecuencia (por 10k)	Término	Frecuencia (por 10k)
baby	71.31	solo	22.73
quiero	54.85	uoh uoh	21.94
bien	35.26	corazón	21.16
love	32.91	sube	21.16
dura	32.91	amor	20.37
uoh	31.35	ver	20.37

contigo	29.78	theres	19.59
puedo	27.43	voy	19.59
one	26.64	noche	19.59
hoy	24.29	girl	18.81
mami	23.51	bailando	18.02
		dura dura	18.02
		let	17.24
		wanna	17.24

Los listados de términos más frecuentes para cada grupo lingüístico reflejan las diferencias inherentes a los idiomas, con palabras comunes en inglés dominando un grupo y palabras en español en el otro. Términos como "love", "baby", "say" son muy frecuentes en inglés, mientras que "baby", "quiero", "bien" y "contigo" lo son en español, mostrando algunas superposiciones en conceptos universales como "baby" o "love".

El análisis de términos distintivos profundiza en estas diferencias, identificando las palabras que son significativamente más prevalentes en un grupo en comparación con el otro, incluso después de la limpieza y filtrado:

**Grupo: Hispanic**

*(Frecuencia Relativa en Hispano / Inglés)*

Término	Hispano	Inglés
quiero	54.85	0.00
bien	35.26	0.00
dura	32.91	0.00
uoh	31.35	0.00
contigo	29.78	0.00
puedo	27.43	0.00
hoy	24.29	0.00
mami	23.51	0.00
solو	22.73	0.00

uoh uoh	21.94	0.00
corazón	21.16	0.00
sube	21.16	0.00
amor	20.37	0.00
ver	20.37	0.00
theres	19.59	0.00

Los términos distintivos confirman las diferencias líricas entre los grupos. Para el grupo Hispano, palabras y frases en español que son casi inexistentes en el corpus inglés ("quiero", "bien", "dura", "contigo", "corazón", etc.) son altamente distintivas. Para el grupo Angloparlante, palabras comunes en inglés ("say", "time", "never", "want", "feel", etc.) que no son tan frecuentes en las letras en español se presentan como distintivas. Esto subraya las barreras idiomáticas y las posibles diferencias culturales o temáticas reflejadas en el vocabulario lírico entre artistas de diferentes lenguas principales.

## 5.2 Impacto del Preprocesamiento de Datos

Aquí se describen los resultados de las diferentes etapas de preprocesamiento aplicadas al conjunto de datos de Spotify. Se detalla el efecto de la limpieza, transformación, y selección de características en la estructura y calidad final de los datos utilizados para el modelado de la popularidad de las canciones.

- Resultados de la imputación de valores faltantes o el manejo de outliers:
  - No se requirió imputación de valores faltantes, ya que el dataset estaba completo.
  - Se aplicó la técnica de capping (limitación de valores extremos) a las características numéricas para manejar los outliers identificados en el EDA. Esto consistió en reemplazar los valores por encima del percentil 95 y por debajo del percentil 5 con los valores de dichos percentiles,

respectivamente. Este proceso ayudó a estabilizar las varianzas y reducir la influencia desmedida de puntos de datos anómalos.

- Efecto de la normalización/estandarización en las variables:
  - Se aplicó **StandardScaler** a las características numéricas seleccionadas después del manejo de outliers y la selección de características. Esta estandarización (media cero y desviación estándar unitaria) es crucial para modelos sensibles a la escala de las características, como la Regresión Lineal y algoritmos basados en distancias o gradientes (implícitamente en los modelos de árbol más complejos también puede ayudar a la convergencia y estabilidad).
- Características creadas o seleccionadas y justificación basada en el EDA o conocimiento del dominio:
  - No se crearon explícitamente nuevas características (ingeniería de características complejas) en este análisis, más allá de la selección de las ya existentes.
  - La selección de características se basó en:
    - Correlación con la variable objetivo (**popularity**): Se priorizaron características que mostraban alguna relación, aunque fuera débil.
    - Las características finales seleccionadas para el modelado fueron: **year**, **danceability**, **loudness**, **speechiness**, **acousticness**, **instrumentalness**, **liveness**, **valence**, **tempo**, y las variables dummy generadas a partir de los géneros más frecuentes.
- Comparación del conjunto de datos antes y después del preprocesamiento (si es relevante, con métricas o descripciones):
  - Antes: Datos crudos con outliers, diferentes escalas entre variables, y posible multicolinealidad.
  - Despues: Datos con outliers controlados, características numéricas estandarizadas (media ~0, std ~1), y un subconjunto de características seleccionadas para reducir la redundancia y mejorar la interpretabilidad y potencial rendimiento del modelo. El número de características predictoras se ajustó en función de la selección.
- Interpretación: Análisis de cómo estas etapas prepararon y potencialmente mejoraron el dataset para la fase de modelado:
  - El manejo de outliers evitó que valores extremos distorsionen los parámetros del modelo.

- La estandarización aseguró que todas las características contribuyeron de manera equitativa al proceso de aprendizaje, independientemente de su escala original.
- La selección de características redujo la complejidad del modelo, disminuyó el riesgo de overfitting y mejoró la eficiencia computacional.
- Estas etapas fueron fundamentales para crear un conjunto de datos más robusto, limpio y adecuado para entrenar modelos de machine learning efectivos para la predicción de la popularidad.

## 5.3 Evaluación del Rendimiento de los Modelos de Data Science

Esta es la sección central donde se presentan los resultados cuantitativos y cualitativos del rendimiento de los modelos implementados para predecir la popularidad de las canciones. Para cada modelo significativo, se expondrán las métricas de evaluación y se interpretará su significado en el contexto del problema. Se utilizaron las siguientes métricas principales: R-cuadrado (R2), Error Absoluto Medio (MAE), Error Cuadrático Medio (MSE) y la Raíz del Error Cuadrático Medio (RMSE).

### 5.3.1 Resultados del Modelo A: Regresión Lineal (Ridge)

Presentación de las métricas de rendimiento obtenidas:

Metric	Value
R-squared	0.0956
Mean Absolute Error (MAE)	7.5942
Mean Squared Error (MSE)	145.3875
Root Mean Squared Error (RMSE)	12.0577

Visualizaciones Clave:

- Gráfico de Dispersión (Valores Reales vs. Predichos): La visualización muestra una nube de puntos con una tendencia lineal positiva, aunque con una dispersión considerable. Esto indica que, si bien el modelo capta una parte de la relación lineal, muchos puntos se alejan de la línea de predicción ideal, lo que es coherente con el bajo R2.
- Gráfico de Residuos: El gráfico de residuos muestra los errores de predicción contra los valores predichos. Idealmente, los residuos deberían distribuirse aleatoriamente alrededor de la línea de cero. En este caso, [Describir brevemente el patrón observado en la imagen - por ejemplo: "se observa una distribución relativamente homogénea, aunque con algunos indicios de que la varianza de los errores podría no ser constante (heterocedasticidad) o que hay ciertos rangos de valores predichos donde el error es mayor"].

Interpretación específica:

- El valor de R2 de 0.096 indica que aproximadamente el 9.6% de la varianza en la popularidad de las canciones puede ser explicada por las características seleccionadas mediante el modelo de Regresión Lineal. Este valor es bajo y sugiere una capacidad predictiva limitada del modelo lineal para este problema.
- El MAE de 7.5942 indica que, en promedio, las predicciones del modelo se desvían en aproximadamente 7.5 puntos de la popularidad real (en una escala de 0 a 100).
- La Regresión Lineal, si bien establece una línea base, no parece capturar adecuadamente las relaciones no lineales y las interacciones complejas que probablemente influyen en la popularidad musical, como se sospechaba desde el EDA.

### 5.3.2 Resultados del Modelo B: Random Forest Regressor

Presentación de las métricas de rendimiento obtenidas para el Modelo B:

Metric	Value
R-squared (R2)	0.3851
Mean Absolute Error (MAE)	13.7996
Mean Squared Error (MSE)	300.1005
Root Mean Squared Error (RMSE)	17.3234

Importancia de Características:

- El análisis de importancia de características (ver [image\\_d2c9f2.png](#)) para el modelo Random Forest reveló los siguientes predictores principales (en orden aproximado de importancia): **year, instrumentalness, acousticness, loudness, danceability, valence, speechiness, tempo** y **liveness**. Esto subraya la relevancia del año de lanzamiento y de diversas características intrínsecas del audio en la predicción de la popularidad.

Interpretación específica:

- El Random Forest Regressor mostró una mejora sustancial respecto a la Regresión Lineal, con un R2 de 0.3851. Esto significa que el modelo explica aproximadamente el 38.51% de la varianza en la popularidad de las canciones, más del doble que el modelo lineal.
- El MAE se redujo a 13.7996, indicando predicciones más precisas en promedio. La reducción en MSE y RMSE también es notable.
- La capacidad del Random Forest para modelar relaciones no lineales e interacciones entre variables, junto con su robustez, justifica este mejor desempeño. La importancia de **year** sugiere una fuerte influencia de la temporalidad, mientras que características como **instrumentalness** y **acousticness** destacan como diferenciadores clave de la popularidad según este modelo.

### 5.3.3 Resultados del Modelo C: Gradient Boosting Regressor

Presentación de métricas e interpretación correspondiente:

Metric	Value
R-squared (R2)	0.4118
Mean Absolute Error (MAE)	13.5504
Mean Squared Error (MSE)	286.5033
Root Mean Squared Error (RMSE)	16.9264

Interpretación específica:

- El Gradient Boosting Regressor logró el mejor rendimiento entre los modelos evaluados, con un R2 de 0.4118. Esto indica que explica aproximadamente el 41.18% de la varianza en la popularidad, posicionándolo como el modelo más efectivo.
- Las métricas de error (MAE, MSE, RMSE) también fueron las más bajas en comparación con los otros dos modelos, confirmando su superioridad predictiva para este conjunto de datos y las características utilizadas. Por ejemplo, el MAE de 13.5504 es el más bajo obtenido.
- El Gradient Boosting, al construir árboles de forma secuencial donde cada árbol corrige los errores del anterior, a menudo logra un rendimiento superior, lo cual se evidencia en estos resultados.

### 5.3.4 Resultados Modelo D: Red Neuronal

Presentación de las métricas de rendimiento obtenidas para el Modelo D:

Metric	Value
R-squared	0.3988
Mean Absolute Error (MAE)	13.6071
Mean Squared Error (MSE)	293.1175
Root Mean Squared Error (RMSE)	17.1207

## Visualización del Entrenamiento:

- Gráfico de Pérdida La gráfica de la función de pérdida durante el entrenamiento muestra cómo la pérdida disminuye tanto en el conjunto de entrenamiento como en el de validación a medida que avanzan las épocas. Se observa que [Describir brevemente la curva de validación, ej: "la pérdida de validación se estabiliza después de X épocas, e incluso muestra un ligero incremento hacia el final, lo que podría sugerir que el modelo alcanzó su punto óptimo de aprendizaje y un entrenamiento más prolongado podría llevar a sobreajuste." o "la pérdida de validación sigue de cerca a la de entrenamiento, indicando una buena generalización."].

## Interpretación específica:

- La Red Neuronal alcanzó un R2 de 0.3988, explicando aproximadamente el 39.88% de la varianza en la popularidad. Este rendimiento es competitivo, situándose entre el Random Forest y el Gradient Boosting.
- El MAE de 13.6071 es también un valor robusto, muy cercano al obtenido por el Gradient Boosting.
- Las Redes Neuronales tienen la capacidad de modelar relaciones altamente no lineales y complejas. Los resultados sugieren que, con la arquitectura y los hiperparámetros utilizados, el modelo pudo capturar una porción significativa de los patrones en los datos. La curva de aprendizaje es fundamental para asegurar que el modelo no esté sobre ajustando ni subutilizando su capacidad.

### 5.3.5 Comparativa y Selección del Modelo Final

Tabla comparativa resumen con las métricas clave de todos los modelos evaluados:

Model	R2	MAE	MSE	RMSE
Linear Regression	0.150	16.50	410.19	20.25
Random Forest Regressor	0.385	13.80	300.10	17.32
Gradient Boosting Regressor	0.412	13.55	286.50	16.93
Red Neuronal	0.399	13.61	293.12	17.12

Análisis comparativo destacando las fortalezas y debilidades de cada uno en relación con los objetivos del proyecto:

- Regresión Lineal: Sirvió como un modelo base simple, rápido de entrenar e interpretable, pero su rendimiento fue el más bajo debido a su incapacidad para capturar las complejidades no lineales de los datos.
- Random Forest: Ofreció una mejora significativa en el rendimiento al modelar relaciones no lineales e interacciones. Es robusto y maneja bien la multicolinealidad, además de proporcionar una métrica útil de importancia de características.
- Gradient Boosting: Demostró ser el modelo más potente, superando ligeramente al Random Forest en todas las métricas. Su naturaleza secuencial de aprendizaje le permite ajustar finamente las predicciones, aunque puede ser más sensible a la sintonización de hiperparámetros.
- Red Neuronal: Rendimiento muy competitivo, cercano al Gradient Boosting. Capaz de modelar alta complejidad, pero puede requerir más datos y una sintonización de hiperparámetros más cuidadosa y ser computacionalmente más intensiva.

Justificación de la selección del modelo (o modelos) con mejor desempeño o más adecuado para el problema:

- El Gradient Boosting Regressor se selecciona como el modelo final con el mejor desempeño para predecir la popularidad de las canciones. Esta elección se fundamenta en sus valores consistentemente superiores en todas las métricas de evaluación clave: el R2 más alto (0.412) y los valores más bajos de MAE (13.55), MSE (286.50) y RMSE (16.93), como se evidencia en la tabla comparativa
- La Red Neuronal presenta un rendimiento muy cercano y destacable (R2: 0.399, MAE: 13.61), posicionándose como una alternativa fuerte. Su desempeño casi iguala al Gradient Boosting, lo que indica su gran potencial.
- Si bien el Gradient Boosting es marginalmente superior en las métricas reportadas, la elección final podría también considerar otros factores como el tiempo de entrenamiento, la interpretabilidad (donde los modelos de árbol suelen tener ventaja sobre las redes neuronales básicas) y la facilidad de despliegue. Para los fines de este estudio, y basándose estrictamente en las métricas de evaluación obtenidas, el Gradient Boosting es el seleccionado. No obstante, la Red Neuronal es una excelente segunda opción y su exploración ha sido valiosa.

## 5.4 Discusión General de los Hallazgos

Una vez presentados e interpretados los resultados específicos, esta sección ofrece una discusión más amplia, conectando los hallazgos con los objetivos generales del proyecto (comprender cómo han evolucionado las características musicales y qué relación hay entre atributos de audio y popularidad) y el conocimiento existente en el área.

Síntesis de los principales resultados:

- Los modelos de machine learning desarrollados, particularmente el Gradient Boosting Regressor, lograron predecir la popularidad de las canciones con una precisión moderada, alcanzando un R2 máximo de aproximadamente 0.412. Esto sugiere que, si bien las características de audio analizadas tienen poder predictivo, una parte considerable de la varianza en la popularidad sigue sin explicarse.
- El análisis exploratorio (EDA) y la importancia de características de los modelos (especialmente Random Forest) destacaron que variables como el **year** (año de lanzamiento), **instrumentalness**, **acousticness** y **loudness** son influyentes en la determinación de la popularidad de una canción.
- Se reconoce que la popularidad musical es un fenómeno inherentemente complejo. Está influenciada por una multitud de factores que van más allá de las simples características de audio, tales como estrategias de marketing, la fama preexistente del artista, tendencias culturales, videoclips, presencia en redes sociales y otros elementos extrínsecos no medidos en este estudio.
- Los modelos predictivos deben ser entendidos como herramientas que proporcionan aproximaciones y tendencias, y no como oráculos definitivos que pueden capturar toda la subjetividad y complejidad del gusto musical.
- El proyecto integra los hallazgos del EDA, que revelan la evolución de las características musicales a lo largo del tiempo, con el modelado predictivo, ofreciendo una perspectiva cuantitativa sobre los atributos que se correlacionan con el éxito en la plataforma Spotify.

## Implicaciones Prácticas y Teóricas:

### Prácticas:

- Para artistas y productores: Los hallazgos pueden ofrecer guías sobre ciertas características de audio que, en promedio, se asocian con una mayor o menor popularidad, aunque siempre dentro de un contexto de creatividad y expresión artística. Saber qué **instrumentalness** a menudo se correlaciona negativamente con la popularidad general (excepto en géneros específicos) o cómo **loudness** y **acousticness** juegan un papel puede ser útil.
  - Para plataformas de streaming y la industria: Estos modelos, aunque de precisión moderada, pueden contribuir a sistemas de recomendación o a la identificación temprana de canciones con potencial, si se combinan con otros datos. La importancia del **year** también resalta cómo las tendencias cambian, lo cual es vital para la curación de contenido.
  - Es crucial enfatizar que la predicción de un "hit" es notoriamente difícil; estos modelos son un componente más en la toma de decisiones, no una solución completa.
- Teóricas:
    - El estudio reafirma la complejidad de modelar fenómenos culturales como la popularidad musical usando únicamente datos cuantitativos de audio.
    - Contribuye al cuerpo de conocimiento sobre la aplicación de técnicas de machine learning en el dominio de la musicología computacional y el análisis de tendencias culturales.
    - Los resultados sugieren que, si bien los modelos pueden capturar ciertos patrones, la "fórmula del éxito" musical sigue siendo elusiva y multifactorial, involucrando aspectos subjetivos y contextuales significativos.

## Fortalezas y Contribuciones del Estudio:

- Análisis de un marco temporal relevante (2000-2022) que permite observar la evolución de las características musicales.
- Implementación y comparación sistemática de tres modelos de regresión distintos, desde una línea base simple hasta ensambles más complejos.
- Un proceso transparente de EDA y preprocesamiento que prepara los datos para el modelado.
- Identificación de las características de audio más relevantes para la predicción de la popularidad dentro del conjunto de datos estudiado.
- La discusión reconoce explícitamente las limitaciones de un enfoque puramente basado en datos de audio para un fenómeno tan complejo como la popularidad.

## 6. Conclusiones

Este capítulo final consolida los hallazgos y logros del proyecto, evaluando el cumplimiento de los objetivos establecidos para el análisis de tendencias en la música popular entre 2000 y 2022. Asimismo, se proponen futuras líneas de investigación y desarrollo que podrían derivarse de este trabajo.

### 6.1 Conclusiones Principales

El presente proyecto ha abordado de manera exhaustiva el análisis de la evolución de la música popular utilizando un extenso conjunto de datos de Spotify y aplicando diversas técnicas de Data Science, Big Data y Procesamiento del Lenguaje Natural. Las principales conclusiones son:

- **Cumplimiento de Objetivos:**
  - Se **identificaron y visualizaron tendencias clave** en las características musicales, atributos de audio y popularidad de géneros musicales en el periodo 2000-2022, utilizando análisis exploratorio de datos y herramientas como Google BigQuery para análisis a gran escala.
  - Se aplicaron **técnicas de clustering (K-Means)** para agrupar canciones basadas en sus atributos de audio y se desarrolló una lógica para asignarles estados de ánimo interpretables.
  - Se **implementaron, evaluaron y refinaron con éxito diversos modelos predictivos**, incluyendo Regresión Lineal, Ridge, Random Forest, Gradient Boosting, Regresión Logística, SVC y Redes Neuronales, para predecir la popularidad de las pistas musicales. El

modelo Gradient Boosting Regressor demostró el mejor rendimiento para la predicción de popularidad, con un  $R^2$  de aproximadamente 0.412.

- Se construyó un **dataset de letras de canciones** para el periodo estudiado y se realizaron **análisis detallados de contenido lírico mediante NLP**, identificando tópicos, similitudes y tendencias temporales, incluyendo el impacto de eventos como la pandemia de COVID-19 en el léxico musical.
- Se **configuró y orquestó un entorno de desarrollo Big Data** utilizando Docker Compose, integrando herramientas como Apache Hadoop, Apache Cassandra y Apache NiFi.

- **Hallazgos Significativos:**

- El análisis predictivo de la popularidad musical, si bien complejo, demostró que las características de audio (como **instrumentalness**, **acousticness**, **loudness**) y el año de lanzamiento (**year**) poseen poder predictivo. No obstante, una considerable porción de la varianza (aproximadamente el 59%) permanece sin explicar por estos factores, subrayando la naturaleza multifactorial de la popularidad.
- El análisis de NLP sobre las letras reveló patrones evolutivos en el vocabulario y las temáticas, mostrando cómo el contenido lírico puede reflejar contextos sociales y eventos globales. Se identificaron diferencias léxicas entre artistas y géneros, así como la evolución del estilo en artistas individuales a lo largo del tiempo.
- La infraestructura de Big Data establecida proporciona una base sólida para futuros análisis a mayor escala.

- **Contribuciones del Estudio:**

- Se ha realizado un análisis comprensivo de las tendencias musicales durante más de dos décadas, integrando atributos de audio, popularidad y contenido lírico.
- La implementación y comparación sistemática de diversos modelos de regresión y clasificación ofrecen una perspectiva sobre la predictibilidad de la popularidad musical basada en datos.
- El estudio aporta al campo de la musicología computacional al aplicar técnicas de NLP para desentrañar la evolución del contenido lírico y su relación con el contexto social.

## 6.2 Trabajos Futuros

A partir de los resultados obtenidos y las limitaciones identificadas, se proponen las siguientes líneas de trabajo futuro:

- **Mejora de Modelos Predictivos:**

- **Incorporación de Datos Externos:** Enriquecer el dataset con variables no incluidas en este estudio, como datos de redes sociales (menciones, sentimiento), inversión en marketing, presencia en medios, información de giras, colaboraciones entre artistas y datos de otras plataformas musicales o listas de éxitos para obtener una visión más holística de los factores que influyen en la popularidad.
- **Ingeniería de Características Avanzada:** Explorar la creación de características más complejas, incluyendo interacciones entre atributos de audio y variables líricas (ej. ¿ciertos tópicos líricos combinados con alta energía son más populares?), o el uso de embeddings de audio y texto.
- **Modelos Más Sofisticados:** Investigar arquitecturas de Redes Neuronales más profundas o especializadas (ej. Transformers para datos secuenciales de audio o texto), modelos híbridos, o técnicas de AutoML para optimizar la selección y configuración de modelos

- **Profundización en el Análisis de Contenido Lírico (NLP):**

- **Análisis de Sentimiento y Emoción:** Aplicar técnicas avanzadas de análisis de sentimiento y detección de emociones en las letras para rastrear su evolución temporal y su correlación con la popularidad o eventos específicos.
- **Modelado de Tópicos Dinámico:** Explorar cómo los tópicos líricos evolucionan y cambian su prominencia a lo largo de las dos décadas estudiadas.

- **Análisis de Originalidad y Complejidad Lírica:** Desarrollar métricas para cuantificar la originalidad, la complejidad del vocabulario o la estructura narrativa de las letras y su posible relación con el éxito.
- **Expansión del Análisis de Tendencias:**
  - **Segmentación Detallada por Género y Microgénero:** Realizar análisis más granulares para identificar tendencias específicas dentro de géneros y microgéneros musicales, que pueden tener dinámicas de popularidad distintas.
  - **Estudio de la "Novedad" vs. "Familiaridad":** Investigar cómo la desviación de las tendencias musicales promedio (novedad) frente a la adhesión a patrones establecidos (familiaridad) impacta en la recepción y popularidad de las canciones.
  - **Validación de "Estados de Ánimo":** Realizar estudios con usuarios para validar o refinar la asignación de estados de ánimo a las canciones basada en los clusters de atributos de audio.
- **Explotación de la Infraestructura Big Data:**
  - **Procesamiento a Mayor Escala:** Utilizar la infraestructura Dockerizada (Hadoop, Cassandra, NiFi) para procesar conjuntos de datos musicales aún más grandes o implementar flujos de datos en tiempo real (ej. análisis de tendencias emergentes en redes sociales o plataformas de streaming).
  - **Desarrollo de Aplicaciones:** Crear prototipos de aplicaciones basadas en los hallazgos, como herramientas de recomendación musical mejoradas o dashboards interactivos para el análisis de tendencias por parte de profesionales de la industria.
- **Abordaje de Limitaciones y la Complejidad de la Popularidad:**
  - **Investigación Cualitativa:** Complementar los análisis cuantitativos con estudios cualitativos (entrevistas con artistas, productores, oyentes) para comprender mejor los factores subjetivos y contextuales que no son capturados por los datos.
  - **Modelado Causal:** Explorar enfoques de inferencia causal para intentar discernir no solo correlaciones, sino también posibles relaciones de causa-efecto entre características y popularidad, reconociendo la dificultad inherente a esta tarea en dominios culturales.

