Exploring Genre Popularity Across Decades: From Historical Rankings to Predictive Music Recommendations

1st Abhishek Singh
Masters of Data Science
Univ. of Europe for Applied Sciences
14469 Potsdam, Germany.
email address or ORCID

2nd Daniela Oliveira Rocha
Masters of Data Science
Univ. of Europe for Applied Sciences
14469 Potsdam, Germany.
email address or ORCID

3rd Sarawut Boonyarat
Masters of Data Science
Univ. of Europe for Applied Sciences
14469 Potsdam, Germany.
email address or ORCID

4th Mati Nakphon
Masters of Data Science
Univ. of Europe for Applied Sciences
14469 Potsdam, Germany.
email address or ORCID

5th Muthamizhan Alagu
Masters of Data Science
Univ. of Europe for Applied Sciences
14469 Potsdam, Germany.
email address or ORCID

Abstract*—The growth of data in sectors like music streaming has created a huge need to have efficient data processing methodologies that allow actionable insights from large datasets. Organizations use tools like Apache Airflow to automate and scale ETL pipelines, converting unrefined data into meaningful information that supports sophisticated analyses, including genre prediction, user behavior modeling, and tailored recommendations. Modern music analytics frameworks have faced issues of scalability, real-time processing of data, and seamless integration with the heterogenous data sources of CSV files and APIs, which have marred their ability to deliver comprehensive and effective insights. This research fills these gaps by designing an automated Apache Airflow-based data pipeline that smooths out data integration, enhances scalability, and supports real-time generation of personalized music recommendations and trend analyses. The approach is to create a multistage, automated data pipeline using Apache Airflow for analyzing the data from Spotify. It combines different sources of data and uses PostgreSQL and Python to clean and process large datasets for advanced analytics, such as genre trend prediction and personalized recommendations, in order to produce scalable and actionable insights. On the other hand, results are strong and show that such audio features as energy and loudness have a significant correlation with the popularity of the song, and with the popularity of the genre and the artist, that correlate with user preferences. There was a striking seasonal trend during sustainability, e.g., the relevance of Christmas themes to track visibility. Therefore, this prediction framework will exploit textual metadata to perform personalized recommendations to enhance user engagement scalability for music discovery. This research implements an automated data pipeline using Apache Airflow that integrates multiple sources of data, enhances scalability, and provides advanced analytics to achieve real-time comprehension of music trends, the impact of artists, and personalized recommendations.

***Index Terms*—Apache Airflow, Spotify data analytics, predictive recommendation, audio features, workflows automation**

I. Introduction

Due to the exploding amount of data in all industries, for today and tomorrow's decision making, data processing and analysis is key. These datasets contain valuable information about music consumption and user preferences for this purpose, which streaming platforms like Spotify produce every day to understand usage patterns, songs played and streaming activity. But one must use proper data management techniques to extract meaningful information from such data and ETL (Extract, Transform and Load) pipelines are strong enough to work with raw data as a job. A ETL structured pipelines are ready to be used for analysis. Of all the movement, the transformation tools available, Apache Airflow has proven to be a powerful orchestration platform that can automate and streamline these processes. It's very good at managing complex workflows, scalable and real time analysis. For Spotify these capabilities are useful for things like identifying genre trends, predicting popularity and creating personalized recommendations. Such tools can be used to further understand user preferences and behavior and move from raw data to actionable insights.

Although analyzing large music data is really tough, despite the fact that data processing tools have been evolving rapidly. The traditional methods are slow, error prone and not programmable to manage the volume of work of platforms like Spotify. Analyzing a time span of decade to identify, for example, seasonal music trends or looking at the evolution of a genre or the characteristics of a hit song requires tedious and often impractical methods. This study addresses these challenges by developing and deploying large-scale automated data pipelines using Apache Airflow. It enables Spotify to get insights faster and more accurate than before through an

efficient data ingestion, transformation and analysis process.

This paper presents a real world application of workflow automation tools like Apache Airflow for data engineering. It focuses on building a data pipeline for the music data of Spotify to address common issues of scalability and performance in data intensive industries. To build a robust and automated process for data intake, transformation and analysis, Apache Airflow's components such as Directed Acyclic Graphs (DAGs) are used. In addition to Spotify, this work may be useful for data engineers, data scientists and others interested in how automation can improve data management and decision making. In the long run the findings support the importance of scalable data pipeline automation in driving innovation and data driven strategies across sectors.

II. Literature Review

Recent years have witnessed a growing tendency to rely on Apache Airflow within music data analysis for data orchestration, workflow automation, and sophisticated feature extraction methodologies. Automation of pipelines has fundamentally changed the landscape, speeding up tasks relating to data collection, transformation, and analysis by orders of magnitude, with high precision. Such pipelines reduce the need for manual procedures and thus make the handling of large datasets much smoother and far less error-prone. Numerous academic studies have demonstrated that the application of these techniques is able to clarify meaningful and revelatory patterns in music data on manifold dimensions. However, despite their effectiveness, some limitations still persist to this day that keep the scalability and comprehensiveness of current methods at bay. To overcome these challenge, we focus on enhancing the effciency of existing tools. There are four leading methods widely adopted in this area: sentiment analysis, feature engineering, big data technologies, and workflow automation. (Table I)

Sentiment analysis has emerged as an important technique to enhance music recommendation systems, matching songs with the users' emotional states. Carol et al. (2022) developed a real-time music recommendation with sentiment analysis technique. Kanika Kamalhans et al. (2022) came up with a new approach, "Musify," which recommends based on the sentiment of the user's mood. This technique goes beyond traditional features like tempo and genre by delving into the emotional tones within the music itself. In this respect, Sandra Grace et al. (2022) also pointed out the emotional relevance of music recommendations since most of the traditional systems fully disregard the emotional depth of the songs. Panda et al. (2021) highlighted that there is a pressing need for emotionally more relevant features since the current feature set present in Spotify caters to only a partial part of the needs of advanced sentiment-driven systems. It has further been supported by Sharan Duggirala et al. (2020), who integrated Sentiment Analysis with big data technologies and found that this enhances the accuracy of song classification with respect to mood and emotional content, thus making the user's experience more enriching. These studies together point to the increasing role of Sentiment Analysis in facilitating more personalized and emotionally aware recommendations of music.

Feature engineering has been one of the most relevant strategies to improve genre classification and music recommendation systems. The primary previous approaches focused solely on musical features involving tempo, rhythm, and pitch, among other features. Sharan Duggirala et al. (2020) describe the case in which these musical features, in isolation, are not strong enough to afford finer and more accurate predictions. Today, these foundational features are combined with state-of-the-art computational methods that yield more exacting and significant results. Panda et al. (2021) showed that interpretable features of Spotify such as energy and valence could be increased manifold when combined with other emotionally relevant features. Kanika Kamalhans et al. (2022) similarly installed the ways in which pace, genre, and mood-based totally functions combine to make sure better discovery and personalization of track. Sandra Grace et al. (2022) additionally investigated how the inclusions of artist traits and song metadata can introduce new dimensions in enhancing category accuracy. All these form a backdrop in which cautious feature engineering indeed presents an vital factor in the improvement of greater powerful music analytics structures that seize person preferences and musical developments extra aptly.

Due to the ever-growing volume of data in music, scalability and speed become central aspects of MIR systems, entrenched with big data technologies. Sharan Duggirala et al. in 2020 showed how huge data sets could be handled more speedily and efficiently with Apache Spark and provided considerable improvements in time and accuracy for genre classification. Sandra Grace et al. (2022) have pointed out that massive information gear are necessary for processing millions of facts factors, which help in finding valuable developments and insights in song intake styles. Kanika Kamalhans et al. (2022) have recognized the function of big records in managing huge-scale user behavior records, which is critical for turning in personalized mood-primarily based song suggestions. Bas Harenslak et al. (2021) highlighted how data pipelines, orchestrated using tools like Apache Airflow, can handle real-time data transformation and volumes of data that are huge. These improvements show how big data tools contribute to solving complex challenges in music analytics. In fact, all these together provide a backbone for maintaining the ever-growing, dynamic data-sets in the music industry with high accuracy in analysis and efficiency in data processing.

Automation of workflows has become an important way to improve information processing in MIR systems. In the work, Emmanuel Ok and Johnson Eniola. (2024) said that Jenkins can make workflows automatize tasks. Besides, Bas Harenslak et al. (2021) pointed to the importance of using Apache Airflow when orchestration of records pipelines is very

difficult, allowing scaleable and dependably processed across multiple systems. Kanika Kamalhans et al. (2022) highlighted the fact that through automated pipelines, actual real-time analysis of several song capabilities and user behaviors actually can take place in a very environment-friendly fashion. Panda et al. (2021) mentioned that though automation cuts human errors, this speeds up extracting meaningful insights from huge and numerous datasets. Sandra Grace et al. (2022) pressed further the benefits that could be accruable from automation of workflows, especially in making sure multiple information sources are cleanly integrated and enhancing the performance structure of recommendations. These studies have brought to the fore how automation has addressed challenges around the handling of large-scale records inside the track enterprise. This enhances both efficiency and accuracy in the analytics of songs.

III. Our Contribution

A. GAP Analysis

Most of the state-of-the-art music analytics systems are torn between huge chunks of big data created through music streaming. They work traditionally with human labor involvement, taking an unbearable time with loads of errors. Insights of trends in genre, feature correlation, user behaviors can be provided from most of the systems; however, very few guarantee scalability or real-time processing. Most of the frameworks do not fully integrate different data sources, such as CSV files and APIs, thus limiting their depth of analysis. There is a noticeable hole in automating and streamlining tactics that generate and compare tune pointers. Personalized pointers, which think about similarity ratings across genres and artists, also are no longer supported in a scalable or green manner. The proposed research fills these gaps through the development of an automated data pipeline using Apache Airflow for workflow orchestration. The proposed pipeline will integrate CSV files and fetch APIs to provide real-time analysis and actionable insights on trends and recommendations for music.

B. Research Questions

In this study, we try to answer five key questions; each question is very important for the discovery of insights into trends in music and the behavior of listeners from this analysis:

1) RQ1: Analysis of Genre Popularity-What genres have the best average recognition rankings across exceptional many years, including the 1990s, 2000s, and 2010s?.
2) RQ2: Feature-Popularity Correlation-How do features related to tempo (BPM), danceability, and energy relate to the popularity of the track?.
3) RQ3: Artist Popularity-How does the average popularity score of the tracks by popular artists like Taylor Swift or Ed Sheeran differ from the mean of this dataset?.
4) RQ4: Seasonality Cycles-Is the popularity of songs subject to seasonality, including all during holidays like Christmas in December?.
5) RQ5: Predictive Insights for Recommendations-If the dataset can predict users' preferred music based on genre, artist, and functions within the audio itself.

C. Problem Statement

This is because the rapid growth of music streaming services creates an immense amount of big data, which is much too complex to be analysed using traditional methods. In addition, constant changes in genre trends, strongly related audio features, and changing user behaviour lead to complex patterns, making analysis by hand time-consuming and error-prone. This presents a real bottleneck to music professionals and data scientists, whose informed decisions need to be based on actionable insights. These challenges, in turn, raise an increasing need for efficient, scalable, and automated solutions that can process, analyse, and visualise music data in real time. The research discussed herein tries to address this pressing issue by designing an automated data pipeline using Apache Airflow to process Spotify music data. This pipeline tries to address scalability challenges, digging deep into insight: popularity in genre, correlation between features, influence of an artist, seasonality, and predictive recommendations. It closes the gap from raw data to actionable insights and showcases how workflow automation can really make a difference in music analytics.

D. Novelty of this study

This research investigates the trend of music and listener behavior along multiple dimensions, adding novelty in several ways:

- Novelty 1. Improvement of knowledge and understanding of how the music business works.
- Novelty 2. Mapping the changes in genre popularity over decades.
- Novelty 3. The study of the effect of song audio features on user interaction.
- Novelty 4. Shed light on how top artists have contributed to setting music trends.
- Novelty 5. Popularity of the song and establishment of predictive models considering seasonal variations enable personalized recommendations with respect to practical implications of marketing strategy and user experience.

E. Significance of Our Work

It showcases at once an automated multistage ETL pipeline for Spotify record evaluation by means of Apache Airflow for orchestration, PostgreSQL for information storage, and Python for advanced analytics in conjunction with Power BI. This will allow the pipeline to integrate Kaggle datasets with the Spotify Web API and return deep insights into track trends, feature correlations, and listener behavior. It similarly allows the invention of seasonal trends, rising artist popularity, and predictive recommendation models. This work overcomes demanding situations in scalability and efficiency in song records processing even as generating actionable insights. Similarity rankings to permit personalized tune suggestions

TABLE I
LITERATURE REVIEW TABLE SHOWING THE CONTRIBUTIONS OF VARIOUS AUTHORS FOR SPOTIFY ANALYSIS AND DATA PIPELINE TECHNIQUES.

Year	Author and Citation	Paper Title	Dataset Used	Method(s) Used	Contributions	Limitations
2024	Emmanuel Ok, Johnson Eniola. [1]	Streamlining Business Workflows: Leveraging Jenkins for Continuous Integration and Continuous Delivery	Airbnb, Netflix, Spotify	NLP, Jenkins, Continuous Integration, Continuous Delivery, Automation	Demonstrated how Jenkins can streamline workflows by automating repetitive tasks.	Focuses on automation, but does not address potential scalability issues or integration with all deployment environments.
2024	Carol Jeffri et al. [2]	Enhancing Music Discovery: A Real-Time Recommendation System using Sentiment Analysis and Emotional Matching with Spotify Integration	Spotify API	NLP, Sentiment Analysis, Reinforcement Learning, Audio Processing	Developed a real-time music recommendation system integrating emotion detection, sentiment analysis, and Spotify API for personalized recommendations.	Challenges in handling audio complexity, improving reinforcement learning, and refining real-time emotion-to-genre matching.
2022	Kanika Kamalhans et al. [3]	Musify: An application for Aspect Analysis and Visualization of Spotify's Song Data	Spotify Song Dataset	Aspect-based analysis, sentiment analysis, audio feature extraction, data visualization	Visualizes relationships between various song aspects like mood, genre, and energy	Limited to Spotify's dataset, may not generalize well to other platforms or music services
2022	A Sandra Grace et al. [4]	Song and Artist Attributes Analysis For Spotify	Spotify API, Spotify Song Dataset	Data Analysis, Feature Engineering	Analyzed various song and artist attributes to build good recommendation systems.	Limited to data from Spotify; not applicable to other music platforms.
2021	Panda et al. [5]	How Does the Spotify API Compare to the Music Emotion Recognition State-of-the-Art?	Spotify API features were obtained for 704 of our 900-song dataset, annotated in terms of Russell's quadrants	Feature ranking, emotion classification	Evaluates Spotify's audio features for MER	Spotify features underperform compared to state-of-the-art, Limited improvement with combined feature sets.
2021	Bas Harenslak et al. [6]	Data Pipelines with Apache Airflow	Custom Dataset	Apache Airflow, ETL Automation, Data Transformation	Proposed a robust pipeline for automating ETL processes using Apache Airflow.	Scalability could be limited with complex workflows
2020	Sharan Duggiral et al. [7]	A Novel Approach to Music Genre Classification using Natural Language Processing and Spark	Song lyrics dataset with multiple music genres	Hierarchical Attention Networks (HAN) for text classification Apache	Scalable with Apache Spark for large datasets	Limited to lyrics-based features and focused on contempory music genres

across genres and artists were a widespread improvement. The results indeed show the insight-driven approach to decision-making in navigating song discovery and consumer interaction. This scalability of the machine will surely promise to handle increasing datasets and provide valued insights for further applications, such as enhancement of personalized guidelines and predictive analytics. Besides, advanced analytics tools at its core have empowered continuous tracking and iterative enhancements of recommendation algorithms.

IV. METHODOLOGY

Our approach is the multistage fetching of relevant information on Spotify data. This architecture automatically extracts, stores, analyzes, and visualizes the data using leading software packages. Apache Airflow runs the whole data pipeline smoothly and in automated workflow management. The base data is kept in PostgreSQL, promising scalability and

reliability to large-sized data sets. Python, coupled with Power BI, will provide advanced analytics to visualize meaningful trends and patterns in the music data, holistic in approach to make the entire workflow time-efficient and effective (Fig 1).

A. Dataset

The tracks.csv file, which is on the Kaggle website [8], is the one that provides the whole information about each song. The id column is a unique identifier for every track, whereas name is the title of the song. Popularity is a number that indicates how much the track is engaged with streams and users' interactions. The duration_ms column tells how long the song would be in terms of milliseconds, and explicit specifies whether or not there is an explicit content in the track. The artists column lists the performers for the song, and id_artists provides a unique identifier for the artist(s). Release_date gives the public announcement of the song, while danceability quantifies to what extent the track would be good dancing-wise insofar as it is based on the tempo and rhythm of the song. Energy describes the intensity of the track, while key describes the musical key (generally expressed as an integer according to standard music theory). Loudness would be the mean volume of the track over a few sample seconds, measured in decibels, while mode indicates whether the track is in a major (1) or minor (0) key. Speechiness assesses to what extent the song has spoken-word content, and acousticness assesses the probability that the track is acoustic. Instrumentalness predicts whether the track is likely to be instrumental, and liveness measures the presence of an audience. Valence indicates how positive this track is, with higher values indicating a more cheerful sound. Tempo is the speed of a song—meaured in beats per minute (BPM)—and time signature is the measurement of the number of beats per bar, giving an impression of how the rhythmic structure of the song will be (Fig 2).

The API dataset (Fig 3) gives complete audio competencies for individual Spotify tracks [9] [10] [11]. Each tune is assigned a totally unique Spotify ID, which allows for proper evaluation. Important attributes include acousticness, indicating the self-guarantee that a track is acoustic. Danceability assesses how suitable music is for dancing, while strength measures the intensity and hobby level. Instrumentalness estimates the likelihood that a tune has no vocals. Other capabilities encompass liveness, which gauges the risk of a live overall performance, and loudness, which shows the average decibel stage. Speechiness identifies the presence of spoken phrases in music. Additionally, metadata inclusive of evaluation URLs and track URIs gives access to greater certain records about every track.

B. Detailed Methodology

This architecture is split into 7 separate but interdependent DAGs to segment the tasks. DAG 1 is where the first action is taken—cleaning the artists.CSV file from unwanted characters like square brackets and quotes. This makes the data analysable. The cleaned data is stored in PostgreSQL and saved in fresh files, clean_artists.csv, for easy access in later processes. The second task processes the tracks.csv file, standardises the release dates, and merges it with the cleaned artist data to ensure the dataset is consistent. Finally the data is inserted into PostgreSQL using SQLAlchemy's chunking to deal with large data and avoid memory overload and scaling the process. Now DAG 2 builds on top of this—automating the extraction from the Spotify API of a user's recently played tracks. It starts by refreshing the access token for authentication, then fetches the 50 newest tracks and enriches those tracks with audio features. Finally standardises the release dates and upserts the enriched data into PostgreSQL to make replication impossible. Those two ETL pipelines make sure what's stored in raw data is cleaned and structured and ready for more advanced analysis in later stages of the study.

Data Analysis and Wrangling (DAG 3 - 5). DAG 3, "Spotify Analysis Q1," marks the start of the data analysis. This pipeline extracts, validates, and cleans the track data to analyse the popularity of genres over time. After validating the columns, the data is cleaned and stored as validated_tracks.csv. The pipeline then calculates the number of songs released each year and the average genre popularity across decades and stores the results in genre_decade_popularity.csv to answer the research question on genre trends. DAG 4, "Spotify Analysis Q2," does the correlation and regression analysis to see the relationship between the audio features and the song popularity. The records are processed in chunks to keep away from reminiscence overload, and the pipeline calculates the correlation between capabilities like tempo and strength with the popularity rating. The consequences are stored in correlation_matrix.Csv to see how specific capabilities affect reputation. DAG 5: "Spotify Analysis Q3" makes a speciality of artist-level analysis, evaluating the recognition of pinnacle artists with the general Spotify dataset. This pipeline starts off evolved through exploding the artist columns to permit granular evaluation of multiple artists in step with tune and then enriches the dataset with additional artist information like follower matter and genre from the Spotify API, which provides greater context to the evaluation. The pipeline calculates both typical and weighted average recognition to examine the recognition distribution. To calculate average popularity, it genuinely takes the common reputation score of all of the tracks in the dataset 1.

$$OAP = \frac{\sum(popularityofallsongs)}{Totalnumberofsongs} \tag{1}$$

In the case of weighted common choice, the calculation includes the amount of water (or other appropriate weighting) for each track, so that the best-matched tracks have the greatest impact. This method assures that tracks with more listener interaction have a much closer factor in the final popularity rating. The weighting method (2 stages) accounts for this by varying the size of the song's target market, assigning more weight to songs with higher stream levels, and then hiding

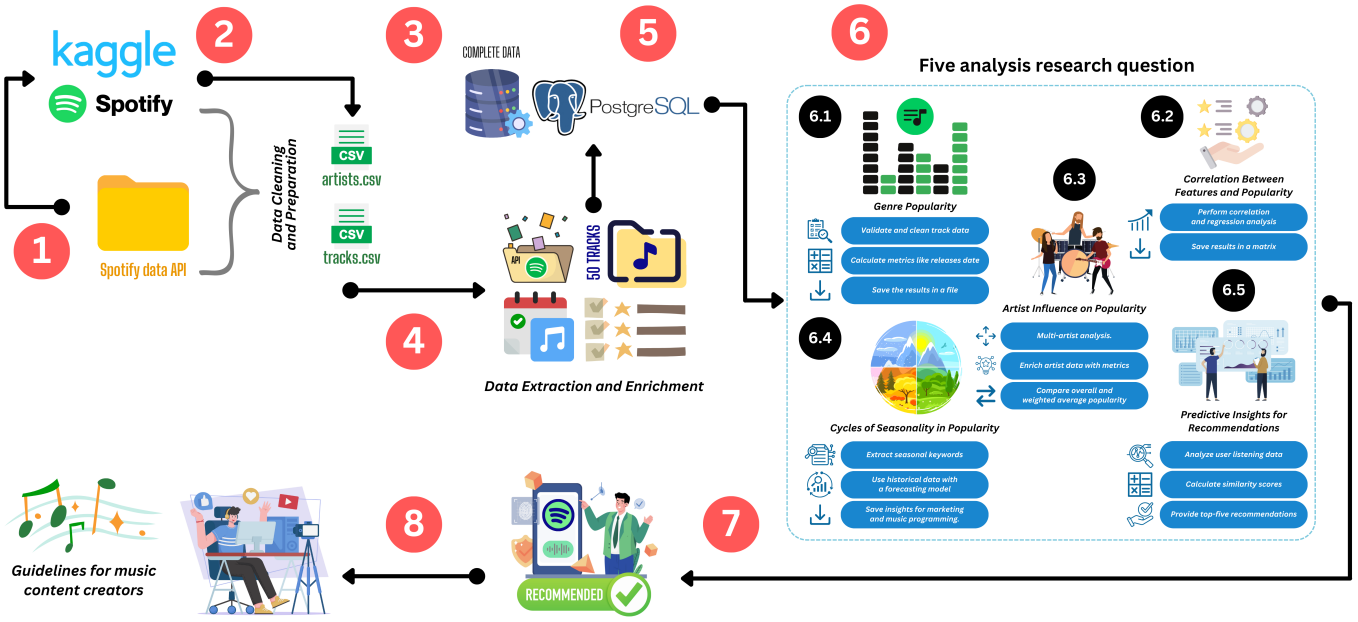

Fig. 1. Data Pipeline.

		Popularity count		Artists name	Artists id	Release date		Energy	loudness			Instrumentalness				Time signature	
35iwgR4jXetI318WEWsa1Q	Carve	6	126903	['Uli']	['45tIt06XoI0Iio4LBEVpls']	22/2/1922	0.645	0.445	0	-13.338	1	0.451	0.674	0.744	0.151	0.127	104.851	3
021ht4sdgPcrDgSk7JTbKY	Cap٧≠tulo 2.16 - Banquero Anarquista	0	98200	['Fernando Pessoa']	['14jtPCOoNZwquk5wd9DxrY']	1/6/1922	0.695	0.263	0	-22.136	1	0.957	0.797	0	0.148	0.655	102.009	1
07A5yehtSnoedViJAZkNnc	Vivo para Quererte - Remasterizado	0	181640	['Ignacio Corsini']	['5LiOoJbxVSAMkBS2fUm3X2']	21/3/1922	0.434	0.177	1	-21.18	1	0.0512	0.994	0.0218	0.212	0.457	130.418	5
08FmqUhxtyLTn6pAh6bk45	El Prisionero - Remasterizado	0	176907	['Ignacio Corsini']	['5LiOoJbxVSAMkBS2fUm3X2']	21/3/1922	0.321	0.0946	7	-27.961	1	0.0504	0.995	0.918	0.104	0.397	169.98	3
08y9GfoqCWfOGsKdwojr5e	Lady of the Evening	0	163080	['Dick Haymes']	['3BiJGZsyX9sJchTqcSA7Su']	1922	0.402	0.158	3	-16.9	0	0.039	0.989	0.13	0.311	0.196	103.22	4

Track id, Track name, Duration (ms), Danceability, key, mode, Speechiness, Acousticness, Liveness, Valence, Tempoo

Fig. 2. Sample track dataset csv file.

			Duration (ms)		ID		Key		Mode	Tempo						
0.00242	https://api.spotify.com/v1/audio-analysis/2takcwOaAZ	0.585	237040	0.842	2takcwOaAZWiXQijPHIx7B	0.0069	9	0.0866	-5.883	0	0.0556	118.21	4	https://api.spotify.com/v1/tracks/2takcwC	audio_features	spotify:track:2takcwOaAZWiX	0.428
0.13421	https://api.spotify.com/v1/audio-analysis/3dfad9LXHo	0.723	199812	0.732	3dfad9LXHoMlPmxJZ7CdFv	0.0123	4	0.0654	-6.245	1	0.0482	122.5	4	https://api.spotify.com/v1/tracks/3dfad9L	audio_features	spotify:track:3dfad9LXHoMlF	0.542
0.00185	https://api.spotify.com/v1/audio-analysis/4abk3mHXy	0.648	244150	0.915	4abk3mHXyIb2hx3iyU5Hy1	0.0049	11	0.0789	-4.731	0	0.0611	127.32	3	https://api.spotify.com/v1/tracks/4abk3m	audio_features	spotify:track:4abk3mHXyIb2	0.664
0.08912	https://api.spotify.com/v1/audio-analysis/5xzY8sm4Aa	0.784	180250	0.694	5xzY8sm4AaRwE6wshJ8GkD	0.0214	6	0.0914	-8.523	1	0.0346	115.89	4	https://api.spotify.com/v1/tracks/5xzY8sm	audio_features	spotify:track:5xzY8sm4AaRwI	0.391
0.06548	https://api.spotify.com/v1/audio-analysis/6klezJ3B54b	0.51	200180	0.845	6klezJ3B54bGPlRsbE45Cz	0.0159	0	0.0831	-5.032	0	0.0577	119.6	4	https://api.spotify.com/v1/tracks/6klezJ3B	audio_features	spotify:track:6klezJ3B54bGPl	0.511

Acousticness, Analysis url, Danceability, Energy, Instrumentalness, Liveness, Loudness, Speechiness, Time signature, Track href, Type, URL, Valence

Fig. 3. Sample API dataset response.

this effect in popularity_comparison.CSV, and allows you to compare the popularity of artists in detail. Learning this data makes it easier for the pipeline to determine whether track charts are dominated by a few well-known artists or whether reputation is evenly distributed throughout the data set. In addition, the findings may recommend that if mainstream artists draw on traditional trends in music consumption and exhibit appropriate dynamics in music selection and listener choice 2.

$$WAP = \frac{\sum(total\_popularityofeachartist)}{\sum(song\_countofeachartist)} \quad (2)$$

Predictive Insights and Personalised Recommendations

(DAG 6 and 7). DAG 6, "Spotify Analysis Q4," attempts to extract seasonal keywords in song titles and predict how trends can become popular. Used libraries include Pandas for data manipulation, SQLAlchemy for database interaction, and Prophet for the forecasting of data. It first extracts the song titles and then vectorizes them to obtain the keywords for each season: "holiday", "Christmas", "summer", "winter". These keyword counts are saved in seasonal_keywords.csv. In the pipeline, to forecast the popularity trends, it uses Prophet, setting yearly seasonality to capture the annual pattern of the trends. After training the model with historical data, a future dataframe is made to predict the trend for the subsequent 12 months—it is an amazingly useful insight that can be deployed in music programming and marketing strategies. DAG 7, "Spotify Analysis Q5," aims at the betterment of personalised musical recommendations by attempting the analysis of textual metadata involving song titles and artists' names. It first queries the user for recently listened-to tracks and extracts pertinent metadata. If recent listening data is empty, then it terminates by logging the appropriate statistics. When there are some recent tracks, it processes the pipeline into a text representation of user preferences for potential use in the calculation of similarity scores for other tracks within the catalogue. It recommends top five tracks that are similar in cosine similarity and writes these recommendations in a CSV format for further analysis using the pipeline to integrate into applications. This really will scale up great with large-sized datasets and help provide user-preferred music recommendations. Here is example Python code below:

- **Analyse Seasonal Keywords:**

```
seasonal_keywords =
keyword_counts[keyword_counts.index.str.contains
("holiday|christmas|summer|winter", case=False)]
seasonal_keywords_df =
seasonal_keywords.reset_index()
seasonal_keywords_df.columns =
['Keyword', 'Count']
```

- **Forecast Popularity Trends:**

```
model = Prophet(yearly_seasonality=True)
model.fit(df_prophet)
future =
model.make_future_dataframe(periods=12, freq='M')
forecast = model.predict(future)
```

- **Computing Textual Similarity:**

```
vectorizer = CountVectorizer
(stop_words='english')
recent_vectors = vectorizer.fit_transform
(recent_tracks['combined_text'])

candidate_texts = chunk['name'].fillna('')
+ " " + chunk['artists'].fillna('')
candidate_vectors =
vectorizer.transform(candidate_texts)

similarity_scores = cosine_similarity
(recent_vectors, candidate_vectors)
chunk['similarity'] =
similarity_scores.max(axis=0)
```

- **Highly recommended selection:**

```
for idx,
recent_song in recent_tracks.iterrows():
   recent_id = recent_song['id']
   song_recs = chunk[chunk['similarity'] >
   0].nlargest(5, 'similarity')

   for _, rec in song_recs.iterrows():
      recommendations.append({
        'based_on_song_id': recent_id,
        'based_on_song_name':
        recent_song['name'],
        'recommended_song_id': rec['id'],
        'recommended_song_name': rec['name'],
        'recommended_artist': rec['artists'],
        'similarity_score': rec['similarity']
        })
```

C. Evaluation Metrics

The success rate of a task (TSR) is one of the very important performance metrics to show how well tasks are being executed within DAGs on a workflow management system like Apache Airflow. It gives the percentage of tasks that were successfully executed out of the total number of tasks executed by equation 3. For each DAG, we counted how many tasks were in the "success" state and divided that by the total amount of tasks for this DAG times 100 to get the percentage. The higher the success rate, the more reliable the execution of tasks, whereas a low rate may point either to the failures in the system or configuration errors or partial failures in the task management. The results from the execution were that all DAGs were at 100% success (Fig 4), and literally every single task in the system had executed without failures. That would indicate a really stable and optimised environment for workflows. The type of analysis above is quite important in system performance monitoring, problem detection in workflows, and assurance of reliable execution of tasks within larger automations.

$$TSR = \frac{Number of Successful Tasks}{Total Number of Tasks} \times 100 \tag{3}$$

```
deproject=> SELECT
    dag_id,
    COUNT(CASE WHEN state = 'success' THEN 1 END) AS successful_tasks,
    COUNT(*) AS total_tasks,
    (COUNT(CASE WHEN state = 'success' THEN 1 END) * 100.0 / COUNT(*)) AS task_success_rate
FROM
    task_instance
WHERE
    dag_id = 'dag1_spotify_etl_pipeline' -- specify your DAG ID here
GROUP BY
    dag_id;
         dag_id          | successful_tasks | total_tasks |  task_success_rate
-------------------------+------------------+-------------+---------------------
 dag1_spotify_etl_pipeline |               15 |          15 | 100.0000000000000000
(1 row)
```

Fig. 4. Task success rate result.

D. Experimental settings

There is some DAG setup, with several DAGs making up the workflows in Apache Airflow. The dag1_spotify_etl_pipeline has a non-schedule interval since the data is static and is Kaggle hosted, so frequent updates are not required. The start date was January 1, 2025. In contrast, the dag2_recently_played_songs_spotify_data_pipeline runs on a

regular basis with daily schedule because it fetches more dynamic data and it started execution on January 1, 2025. Lastly, DAGs spotify_analysis_q1 to spotify_analysis_q5 are also set to none schedule interval because they were for analysis purposes and need not run every day. These configurations will allow a good balance of on-demand and automated processing based on the nature of the data that is being handled.

V. Results

A. Genre Popularity Analysis

Peaches-Justin Bieber feat. Daniel Caesar and Giveon (Fig 5): It's such a fantastic blend of everything coming together to click with just such a broad audience. Genre Popularity Analysis: smooth R and B base with a very tropical and soulful feel that's very easygoing yet emotionally deep at the same time. Justin Bieber's international popularity and dedicated fanbase gave a good kick-start, while the collaboration with Daniel Caesar and Giveon also brought eclectic sets of listeners who enjoy their unique vocal styles. The song speaks to everybody, its message being of love and gratefulness, which helped it reach a wide audience of any demographic. The infectious chorus and groovy production make it a perfect cut for radio, playlists, and social media platforms, especially on TikTok, where it blew up with user-generated content. The balancing of the track between mainstream pop appeal and artistic integrity helped the song top several charts and playlists. Released at a time when feel-good and romantic music was suddenly needed globally, the events may have boosted its popularity. It can, therefore, be said that strong collaboration, flexibility in genre, emotional appeal, and viral marketing through social media catapulted Peaches to being a standout track from the analysis.

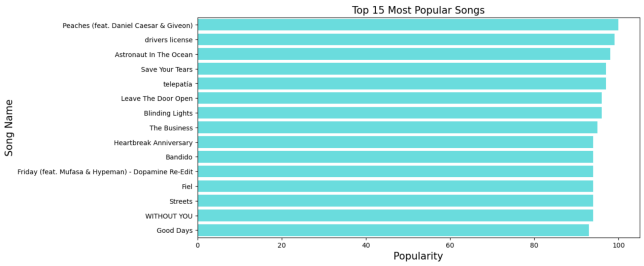

Fig. 5. Landscape of Top Hits.

B. Correlation Between Features and Popularity

We plot the different audio features' correlations in a heatmap, noticing that there might be hidden relationships and intuitively understanding how these features may influence popularity. These links, when visualised, show how things like loudness, energy, and acousticness interact with each other in ways that shape appeal. Among the strongest patterns we were able to discern was a 0.80 correlation between loudness and energy; this means that louder songs tend to carry more energy, a general characteristic maybe of popular genres or songs appealing to a wider crowd. And conversely, we found an important negative correlation of -0.70 between acousticness and energy. That is, energetic tracks are usually less acoustic, and a clear trend is that energetic tracks are normally less likely to feature softer, acoustic arrangements typical of genres such as electronic or rock. (Fig 6).

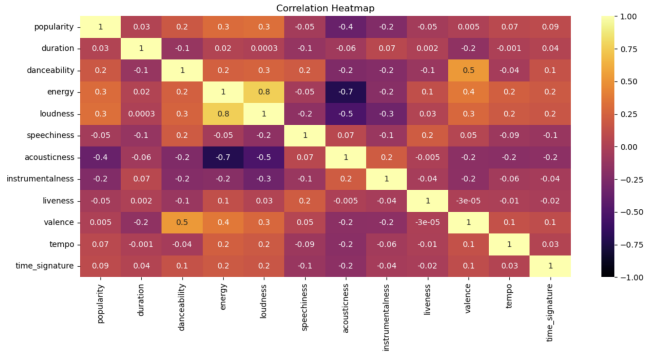

Fig. 6. Energy and Loudness Correlation positive and Energy and Acousticness Correlation negative.

C. Artist Influence on Popularity

Actually, it is interesting to compare the average popularity between the top-rated artists and the rest of the song pool. The ratings from top artists such as Billie Eilish, Bad Bunny, and Harry Styles are commonly from 60 to 77. Indeed, this dwarfs the common average for all tracks: 27.57. This supports the fact that famous artists gain larger followings; they have greater track reputations since more are likely to stream them. Less known artists score closer to the total average. Even superstars such as Justin Bieber might have one or two less-scoring tracks. Therefore, even with superstars, the final score depends entirely on the great music and their ability to capture audience interest. Some of the most consistent best scorers by the stars were Taylor Swift and Ed Sheeran. Through the graph information, it appears that reputation only was not going to make anything fulfilled. Other aspects include style developments, timing, and social media. The common in usual cases is reduction because most of the songs that are not in the mainstream have some level of acknowledgement. Lastly, the facts bring the idea that top artists naturally have an edge, and not every piece of music from them will hit.(Fig 7)

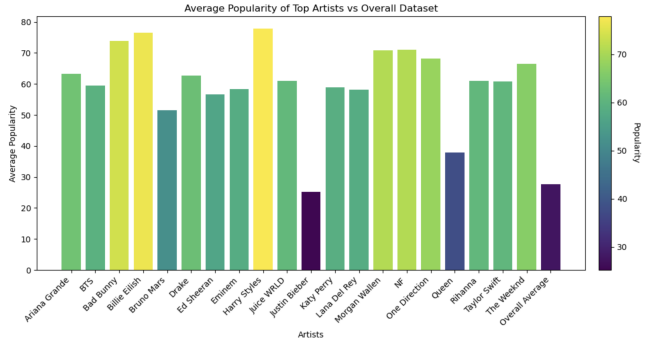

Fig. 7. Artist Influence on Popularity Analysis of the Spotify popularity.

D. Seasonal Cycles of Popularity and Predictive Insights for Recommendations

We created a presentation with the Power BI tool to present the results, and we divided the year into 4 key intervals as we started to explore changing popularity of tracks with the changing seasons: Christmas, Holiday, Summer, and Winter(Fig 8). Seasons come with their vibe with them, therefore affecting the kind of genres people hook onto to songs. And coming to Christmas, we categorized 16 tracks that absolutely had the spirit of Christmas. The most obvious track that proved to be ”What’s Your Name” by Ellinor Springstrike boasted the best reputation with a median reputation rating of fifty-seven. It is possible to trace its success to its wide appeal and strong linkage to the season in that during vacations, it was popular enough to keep it ahead. We researched how specific words within song titles like ”Christmas” and ”Holiday” became connotated with familiarity at a given period during the year. Such seasonal themes also played a tremendous role in making the song more relatable to the spirit of the season. What made this particular music tremendous was that it went on doing well for the rest of Christmas time, showing how songs based on certain situations of the calendar year can have their recognition remain. This was an analysis finding out how well the music, set against the ideal season, can go on to stay in the notice of listeners and also take off at different times of the calendar year. The predictive music recommendation system is going to provide improved listening experiences by suggesting those tracks highly similar to a user’s taste. This is achieved by going through the user’s recently played tracks, finding the textual metadata of these songs—such as song title and artist names—and finding similar songs. As opposed to previous versions of this application, which utilised audio features such as energy, loudness, and tempo, this one uses textual similarity among song metadata. For example, if the song currently playing is ”As the World Caves In,” the system recommends (Fig 9) ”World Of Its Own” by Tingsek, ”World” by Sator, ”Rock the World” by Bubbles, ”Never Once” by Matt Redman, and ”I Am The World” by Takida.

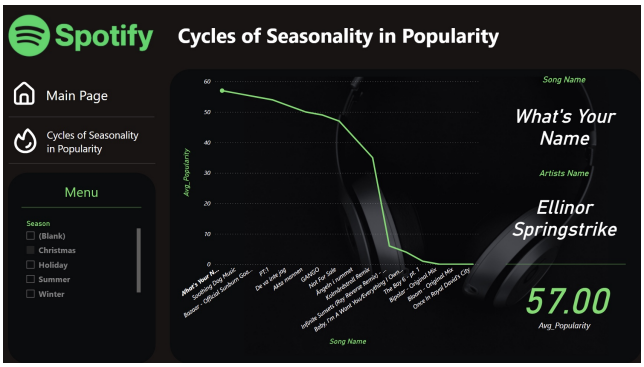

Fig. 8. Seasonal Patterns in Music Popularity on Spotify.

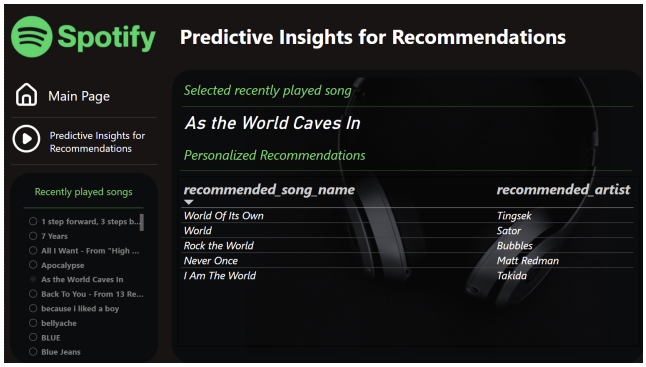

Fig. 9. Recommendation recently played tracks.

VI. Discussion

This computes long-term trends and shifts in genres, based on calculations for the number of songs coming out every year and average popularity across decades. That analysis is really helpful in answering the research question on how the popularity of the genre evolves over time. The structure of the pipeline is sound, but there is an opportunity to increase granularity. For instance, if there is more granular metadata on who collaborated with whom or how long the song is, that might give even more insight into how these variables influence genre popularity. Chart performance from the song ”Peaches” is actually a perfect example of how timing, emotional resonance, and perfect viral marketing elevate a track. The message of love and gratefulness deeply reached its audience through the song, while the rise to the top via platforms like TikTok helped fuel its much-trending virality that becomes influential in the music industry today. The combination of mainstream success and artistic integrity, within the fusion and stylistic collaborations of genres, has made it widely popular. This track is thus one example of where both technical and emotive factors intersect to make a trend for genre.

Besides genre trends, our analysis showed a very strong correlation between features of tracks, such as loudness and energy, with a correlation coefficient of 0.80, hence supporting the supposition that the more energetic the track, the more prominent it is in sound and, therefore, popular. This understanding is useful for artists seeking to make their music more engaging and for streaming services like Spotify in order to fine-tune recommendation algorithms. By integrating textual metadata such as artist names, song titles, and lyrics, our methodology provides a more comprehensive overview of user preferences that allow the creation of personalized playlists which can give good importance to both audio features and thematic elements in music.

This work highlights two new contributions: seasonal trend analysis and metadata-based recommendations. By using text analysis of song titles with Prophet-based seasonal forecasting, seasonal trend analysis will predict music trends and help the platforms be better prepared for demand during the season of the time. Metadata-based recommendations change the

game from traditional audio feature-based methods to textual metadata by the application of cosine similarity on song titles and artist names for scalable, efficient recommendations. However, no quantitative comparisons to existing systems are given, no evaluation metrics are discussed, and there are no discussions of multilingual song title processing or homonym handling. Validation is also limited since examples like "As the World Caves In" rely primarily on superficial word matches. Future work should include A/B testing against traditional methods, precision/recall metrics, handling edge cases like multilingual data or homonyms, and providing examples that demonstrate deeper semantic understanding and insights into user satisfaction.

The advantages brought forth by using Apache Airflow, PostgreSQL, and Python are apparent while comparing the results with other exiting contemporary methods for automating data pipelines. While most traditional approaches suffer with a great level of manual intervention and scalability issues in dealing with huge sets of data, the proposed approach facilitates an efficient and smooth means of music data processing. While this is important, Apache Airflow does allow for scalability-in particular, the big volumes of data generated on streaming platforms. Our methodology further integrates advanced analytics and visualization tools such as Power BI that enrich analytics to help derive actionable insights in real time by all its stakeholders.

This has been an invaluable analysis, but several limitations have to be taken into account. Reliance on Kaggle datasets and the Spotify Web API has some limitations with regard to the reality of actual data access and accuracy. The dataset used in this work here is a representation of the trends that one user followed; therefore, general trends could not be determined. Additionally, textual metadata could not always capture completely the complex user preferences. The modularity in using seven different DAGs within the architecture makes the flexibility and maintenance of the pipeline much better. Each DAG has a certain functionality, be it cleaning the data or doing some sort of research-based analysis; hence, updates and scaling of the system would be easier without affecting the whole pipeline. Integrating Python and Power BI enhances analytics to find deeper insights and more complex relationships. The method would add significant value over conventional approaches by giving stakeholders interactive visualizations and predictive models.

A. Limitations

This study does provide quite a bit of insight into trends within music and predictive recommendations, but there are a few limitations that should be taken into consideration while interpreting the results. The first of the major constraints pertains to the sources of data and the policies surrounding Spotify's API. Spotify has placed certain restrictions on the use of its content, in particular around training machine learning or AI models [12]. Specifically, it is against the policy to consume Spotify's data for training any machine learning algorithms. Due to the aforementioned reason, this work will conduct analysis and modelling on open-sourced datasets collected from various third-party sources instead of collecting all the information available in real time on Spotify itself. Another key limitation is that the dataset on which this analysis is based originates from one user's listening behavior. A single user can never reflect more generalised trends across the population. Similarly, the preferred listening pattern and the pattern variations of one person may be considerably different from the pattern of variations of other users. This calls for a large quantity and diversity in the dataset that could give the correct insight into various users' listening patterns, allowing them to build stronger and generalised truths.

B. Future Directions

Future research with even more diverse data will include the discovery of less acknowledged artists and further genres to level out biases taken in this work. Moreover, trends in social media interactions, reviews given by users, and geotags can depict an overall trend around the globe over what genres of music consumers are enjoying more. Beyond this, Kamalhans et al. (2022) suggested that sentiment analysis may serve to enhance the emotional dimensions in music recommendations and hence offer more scope for the system to suggest tracks that appeal to the current mood or emotional state of a user. Wider coverage and deeper analysis should be able to contribute a lot to the area of music recommendation and trend analysis. The given study fully agrees with general research on the influence of energy, loudness, and tempo audio features on music popularity; however, without including sophisticated techniques like mood detection or sentiment analysis, the recommendations might lose a bit of their emotional wealth. Music is both a technical and an emotional experience, and taking into consideration the emotional shades of recommendations might make them more personal and interesting. That bridges the gap by considering sentiment analysis in the holistic music discovery experience.

VII. Conclusion

With a complete analysis of Spotify from Kaggle and the Spotify API, this study highlighted key findings on music tendencies and reputation dynamics. The steel genres are drastically dominant concerning basic popularity scores, while the genre desire has honestly shifted across many years, with drastically glam punk leading within the 2000s. The analysis also identified important correlations of audio features; for example, loudness was strongly positively correlated with energy, 0.80, and the acousticness is strongly negatively correlated with energy, -0.77. Influencer analysis based on the artists demonstrated that top-rated artists usually sustain higher scores on popularity, but the number of followers does not yet guarantee song success. This allowed the team to realize seasonal analysis showing specific patterns of music popularity during festive periods and the implemented predictive recommendation system showing effective song suggestions based on textual metadata rather than traditional audio features.

These findings, powered by Apache Airflow's automation of data pipelines, provide very useful insights for music industry professionals and data practitioners on how important automated data processing is in understanding patterns and trends of music consumption for informed decision-making.

REFERENCES

[1] E. Ok and J. Eniola, "Streamlining business workflows: Leveraging jenkins for continuous integration and continuous delivery," *Department of Computer Sciences, University College of the Cayman Islands*, 2024, 2nd Jan, 2024.

[2] J. A. Carol Jeffri and A. Tamizhselvi, "Enhancing music discovery: A real-time recommendation system using sentiment analysis and emotional matching with spotify integration," in *2024 8th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, 2024, pp. 1365–1373.

[3] K. Kamalhans, A. Gupta, N. Sharma, and D. K. Sharma, "Musify: An application for aspect analysis and visualization of spotify's song data," in *2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST)*, 2022, pp. 1–5.

[4] A. Sandra Grace, "Song and artist attributes analysis for spotify," in *2022 International Conference on Engineering and Emerging Technologies (ICEET)*, 2022, pp. 1–6.

[5] R. Panda, H. Redinho, C. Gonçalves, R. Malheiro, and R. P. Paiva, "How does the spotify api compare to the music emotion recognition state-of-the-art?" 07 2021.

[6] B. P. Harenslak and J. de Ruiter, *Data Pipelines with Apache Airflow*. Simon and Schuster, 2021.

[7] S. Duggirala and T.-S. Moh, "A novel approach to music genre classification using natural language processing and spark," in *2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, 2020, pp. 1–8.

[8] L. Narnauli, "Spotify dataset on kaggle," 2025, accessed: January 13, 2025. [Online]. Available: https://www.kaggle.com/datasets/lehaknarnauli/spotify-datasets/data

[9] Spotify, "Spotify web api: Recently played tracks," https://api.spotify.com/v1/me/player/recently-played, 2025, accessed: January 13, 2025.

[10] ——, "Spotify web api: Get audio features," 2025, accessed: 2025-01-13. [Online]. Available: https://developer.spotify.com/documentation/web-api/reference/get-audio-features

[11] ——, "Spotify web api: Access token," 2025, accessed: 2025-01-13. [Online]. Available: https://developer.spotify.com/documentation/web-api/concepts/access-token

[12] ——, "Spotify developer terms of service: Section iv - restrictions," 2025, accessed: 2025-01-13. [Online]. Available: https://developer.spotify.com/terms#section-iv-restrictions