National College of Ireland

Student Name:	Calista Clifford Gonsalves , Sushmita Ghanshyam Gupta , Gulbahar Erol
Student ID:	22186077 , 22219455 , 23136235
Programme:	Master of Science in Data Analytics **Year:** 2023 / 2024
Module:	Database & Analytics Programming
Lecturer:	Anu Sahni
Submission Due Date:	17.12.2023
Project Title:	Spotify Spectrum : Delving Deeper
Word Count:	3344

We hereby certify that the information contained in this is information pertaining to research I conducted for this project. All information other than our own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

Signature: Calista Gonsalves Sushmita Gupta Gulbahar Erol

Date: 17/12/2023

Office Use Only	
Signature:	
Date:	17/12/2023
Penalty Applied (if applicable):	

Spotify Spectrum : Delving Deeper

Calista Gonsalves,
Department of Data Analytics
National College of Ireland

Gulbahar Erol
Department of Data Analytics
National College of Ireland

Sushmita Gupta
Department of Data Analytics
National College of Ireland

Abstract—**This project explores music distribution on Spotify, utilizing datasets from the Spotify API on albums, artists, and tracks.. The project involves importing raw data into MongoDB. Subsequent stages include data extraction, cleaning, and transformation for PostgreSQL. Analytically, visualizations are created to reveal relationships among albums, artists and tracks, considering various factors. The report outlines key decisions, and lessons demonstrating proficiency in handling spotify datasets and extracting meaningful insights through a blend of NoSQL and relational databases and tries to comprehend the impact of popularity, followers, genres and other important factors on Spotify's tracks, artists, and albums datasets exploring the correlations between them based on a variety of factors.**

Keywords : Spotify, plotly, MongoDB, API

I. Introduction

Spotify is a popular music streaming service that allows users to access a vast library of songs, albums, and playlists from various genres. Users can explore music based on their preferences, discover new artists, and create personalized playlists. The central motivation behind selecting this topic was to investigate the potential correlations among the three datasets—albums, tracks, and artists. The objective is to discern whether the popularity of an artist is influenced by the number of followers or popular tracks or albums, and if an album gains recognition is it due to the popularity of its associated artists or because of the inclusion of numerous popular tracks. The focus is on understanding how the popularity of artists, tracks, and albums interrelates and influences one another. Which music genres are trending, and how have these genres evolved over time. The distribution of artists in albums and the categorization of album types. Comprehending the tracks, artists, and albums currently trending. Given the plethora of music applications available, with Spotify standing out as the most popular, the aim is to unravel how these different elements intertwine within the platform's dynamic musical ecosystem.

The main purpose is to see the popular distribution of music on Spotify by albums, artist, track and evaluate it against different factors. The dataset presented unique challenges, from complex relational structures to the need for extensive data cleansing. the goal was to gain hands-on experience in loading, managing and analysing data using NOSQL system like Mongodb and relational database systems like PostgreSQL.

The project started by fetching data from the Spotify API and storing it in MongoDB, a NoSQL database. We then processed and converted the data for a relational database. With visualizations, we established many different relationships in terms of albums, artist and piece of music. We gathered insights from varied viewpoints by considering important factors. Our research question is how all these three datasets correlate with each other.

II. Related Works

In this paper [1], The authors propose a comprehensive approach to assist developers in this integration process. The paper also includes a user study to evaluate the tool's effectiveness in real-world integration scenarios. This paper [2], demonstrates the effectiveness of MongoDB by evaluating it on 307 open-source projects, achieving a precision of 78%. The study contributes significantly to understanding database access patterns in MongoDB applications. In this survey, [3] the author explores the use of MongoDB, a NoSQL database, in handling large, unstructured or semi-structured datasets. The paper [4] focuses on the practical application of open-source tools for data processing and visualization. It emphasizes the use of Python, Jupyter Notebook, Matplotlib, NumPy, and other libraries for effective data analysis and visualization. This approach is particularly valuable for projects involving significant data analysis and visualization components. In this research paper [5] the author analyses user patterns on Spotify. It uncovers daily session and playback trends, showing morning and evening peaks that differ between desktop and mobile users. The study also explores device preference across different genres. It also investigates the correlation between playback time and stopping time between sessions, offering insights into user engagement and behaviours using Spotify. This research is crucial for enhancing system design and user experience in music streaming platforms. The paper [6] focuses on predicting song popularity by analysing various track features using Spotify data. It understands the different patterns and trends using different algorithms for music popularity. Demonstrates that certain features, particularly those related to tracks, play a crucial role in determining a song's popularity. The study in [7] analyses song and artist attributes using Spotify data, emphasizing data visualization techniques like histograms and scatter plots for assessing acoustic characteristics in playlists. The research explores the potential of machine learning and deep learning to enhance music data analysis.

III. Methodology

A. Data Description

All 3 datasets are collected in real time from Spotify API. Listed below are the description of the data.

1) Dataset 1 – Tracks data include all the necessary information about the tracks like popularity, the artists involved in it, and which album is it located in, the total

tracks, and other general information. It contains 20 columns and 1106 rows.

2) Dataset 2 – Album provides detailed information about albums like the name, artist involved, popularity, type of an album, release date and so on. It contains 18 columns and 1050 rows.
3) Dataset 3 – Artist dataset provides basic information about an artist like followers, popularity, genres. In total there are 1111 rows and 6 columns.

All these datasets were stored independently in the database. Every table has a common column that can be linked with other table. Like tracks data consists of artistid and albumid. Album data consists of albumid, artistid. Artist data contains artistid. Hence all the table can be linked to each other.

B. Detailed Description of Data Processing Algorithms

1. API Extraction

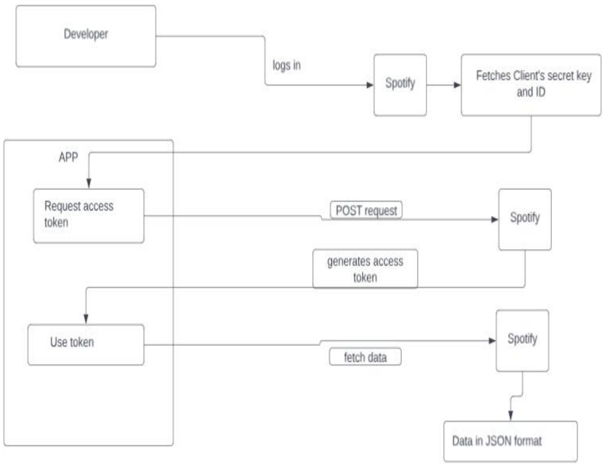

Fig 1: API extraction overview

The 3 datasets tracks, albums and artists are extracted from spotify developer API. Fig 1 gives an high level overview of how the three datasets are retrieved from API. Let’s see the breakdown of each step:

1. Developer Registration on Spotify:

A developer registers on the Spotify Developer Dashboard.

2. App Registration and Credentials:

The developer sets up a new application on the Spotify Developer Dashboard, receiving a unique Client ID and Client Secret that uniquely identify the application.

3. Authentication Request (Token Retrieval):

The Python code/ app uses the obtained Client ID and Client Secret key to make a POST request to Spotify.

4. Token Authentication by Spotify:

Spotify's verifies the client credentials. If the credentials are valid, Spotify generates an Access Token.

5. Access Token Usage in API Requests:

The Python code uses the obtained Access Token to make subsequent API requests to the Spotify API.

This token helps in accessing certain Spotify resources.

6. Fetching Data from Spotify API:

The code utilizes the access token to make requests to the Spotify API. These requests are designed to retrieve information from Spotify. The fetched data, is in JSON format.

2. Process Flow

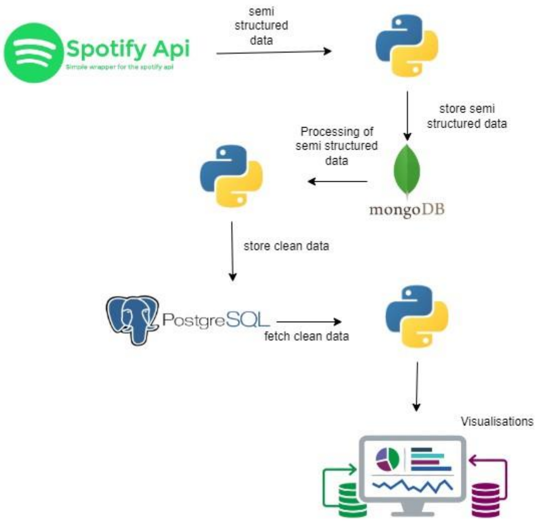
Spotify Api
Simple wrapper for the spotify api
semi structured data
store semi structured data
Processing of semi structured data
mongoDB
store clean data
PostgreSQL
fetch clean data
Visualisations

Fig 2: Process Flow

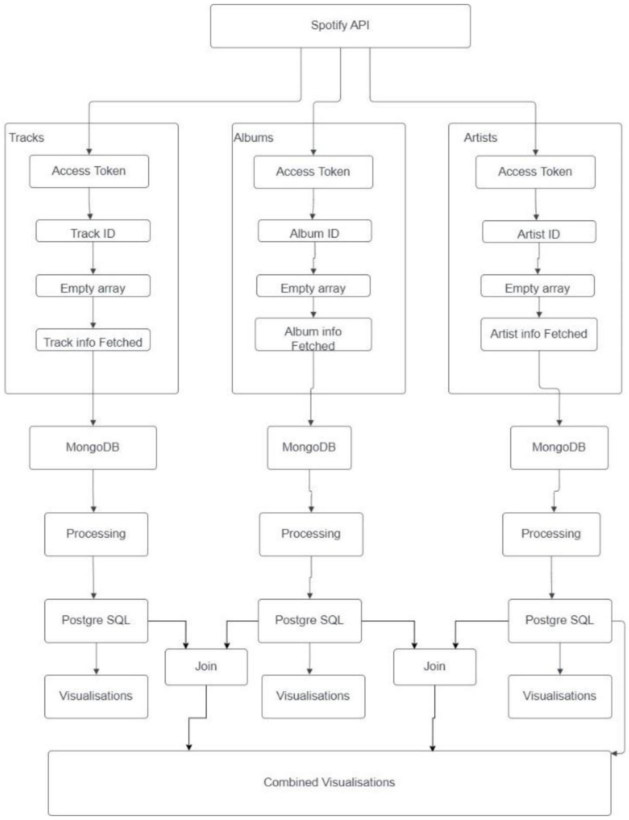

Fig 3: Detailed overview of data

Fig 3 outlines a process beginning with data extraction, followed by storage in a NoSQL database (MongoDB). Next, data undergoes processing for structured formatting before being stored in an SQL database (PostgreSQL). Finally, the stored data is retrieved for diverse visualizations.

1. <u>API extraction of data</u>: As discussed previously, the datasets (albums, tracks, artists) are fetched from spotify API. The API makes use of access tokens to authenticate the request sent from the app/code via Client ID and client secret key. Once the access token is received from the spotify API we can use that access token to make authenticated requests to the Spotify API and fetch information about tracks, albums, and artists.

- To gather track information, the process starts by obtaining trackIDs through a search URL, storing successful fetches in an array. Subsequently, these trackIDs are used to collect detailed track information via the Spotify track URL.
- To fetch album details, the process starts by obtaining albumIDs via search URL. Successful retrievals are stored in an array. Using these albumIDs, detailed information for each album is then gathered by making requests to the Spotify album URL..
- To fetch details about artists, the process starts by obtaining the artistID through a dedicated search URL. Successful artistID retrievals are stored in an empty array. Subsequently, these IDs in the array are used to gather comprehensive artist information via the Spotify artist URL.

2. <u>No SQL database (Mongo DB):</u> All the 3 datasets extracted from API were in JSON format. Post extraction they were stored in Mongodb.
3. <u>Processing of the Data</u>: Data in MongoDB is semi-structured with key-value pairs in documents. To fit into a SQL database, careful processing is needed, transforming MongoDB's semi-structured data into a structured format aligned with the relational model. Various processing steps were involved:

<u>Albums:</u>

- All the columns were fetched from mongodb.
- Null values were present in ‘external_ids_upc’ and handled by replacing it with ‘Not Present’.
- Out of 1300 rows, 387 duplicates rows were identified as duplicates, and the initial occurrence of each duplicated value was removed.
- The artist list in JSON had a nested structure, prompting the creation of new columns like "artist1," "artist2," etc. Empty artist columns were filled with 'Not Present'.
- Unnecessary columns like ‘externalurls’,’href’, tracks as we have a detailed table for track were deleted.
- From the copyrights nested list, just the copyright_type was fetched and remaining were dropped, similarly for externalids nested list only externalid_upc was fetched.
- Post completion of all the above steps, the dataset only consisted of 1050 rows and 18 columns.

<u>Artists:</u>

- Only necessary/important columns were fetched from mongodb.
- This dataset consisted of no null values.
- Out of 1300 rows, there were 266 duplicate values and the first occurrence of the duplicate value was removed.
- After preprocessing, the dataset is now 1111 rows by 6 columns.

<u>Tracks:</u>

- All the columns were fetched from mongodb.
- 'available_markets', 'href','preview_url' columns were dropped as they are not very informative.
- Because the "artist" field is a nested array with three artists, six new columns like artistid1, artistname1, etc., were created during data processing.
- Even the album field being a nested array, only the id, type, name, totaltracks field was extracted and remaining were dropped.
- The new artists columns that were created were checked if any null value is present, if so they were replaced with ‘Not Present’.

- Out of 1300 rows, there were 333 duplicate values and the first occurrence of the duplicate value was removed.
- This dataset consisted no null values.
- Post completion of all the above steps, the dataset only consisted of 1106 rows and 20 columns.

4. PostgreSQL: All the datasets mentioned above are stored in SQL in tabular format in 3 different tables.
5. Visualisations: The data is extracted from PostgreSQL and queried by performing joins and stored in dataframe in order to perform visualisation using plotly.

C. *Technologies used*

1. MongoDB: The reason for choosing Mongodb is that MongoDB is great for storing JSON data because it doesn't require a fixed structure, letting you save data that can change or have different fields. MongoDB supports nested structures, which is much needed as this dataset consists of a lot of nesting.
2. PostgreSQL: PostgreSQL is a robust, open-source relational database. It offers a rich set of data types, including arrays, even JSON. This flexibility allows developers to handle diverse data formats.

3. Python: This was adopted as the programming language owing to its user-friendly nature and widespread applicability in the contemporary market. Even libraries, like pymongo, psycopg2, datetime were used for integration with mongodb and Postgre, and plotly, pandas were used for visualisation.

IV. Result and Evaluation

Datasets are individually visualized and merged through SQL joins using queries, enabling combined visualizations to reveal interdependencies and the impact of one dataset on another.

A. **Visualizations with respect to genres**

1. In Figure 4, the popular genres are displayed, and all 10 genres exhibit a high level of popularity.

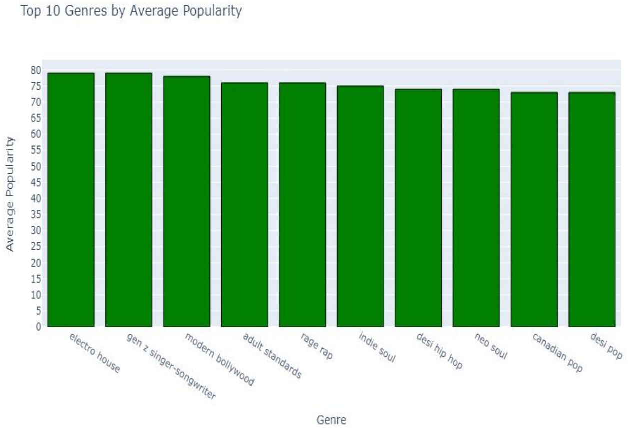

Fig 4: Popular genres

2. The provided figure tells us the duration of a song for a particular genre. Among all the genres, alternative metal exhibits the longest duration, while ambient low-fi has the shortest duration.

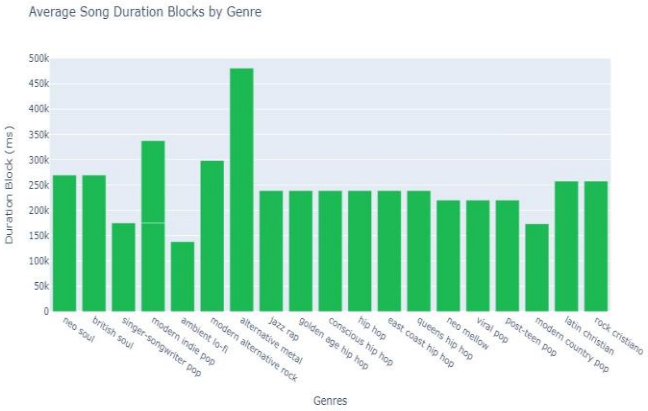

Fig 5: Genre Song Durations

3. Figure 6 shows how genres develop over time, and we can see how over the years the block gets bigger which shows addition of genres

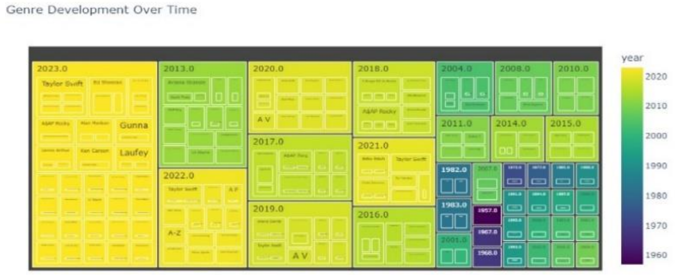

Fig 6: Genre Evolution Over Time

4. Figure 7 gives us an overview of how genre is affected by popularity and followers of an artist with the help of a 3d plot. Each circle in the plot represents a genre, and we can see the genres that are yellow are more popular than those that are blue

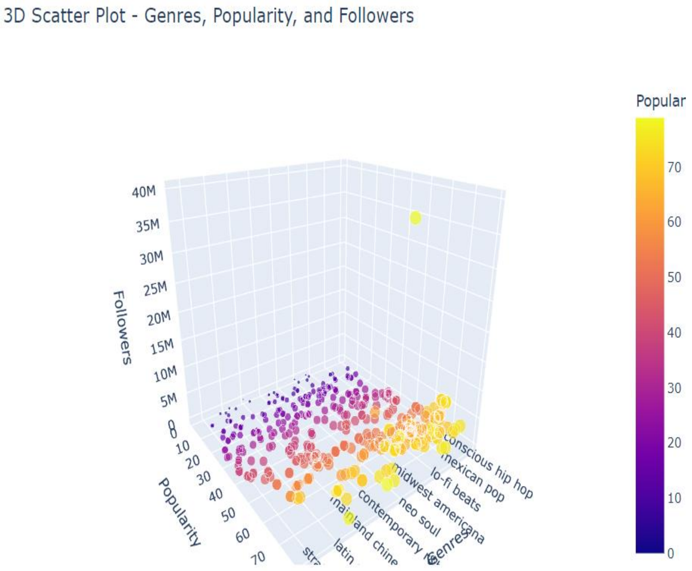

Fig 7: 3D Scatter Plot

5. In Figure 8, a zigzag plot illustrates the top genre counts for an artist, with "stomp and holler" achieving the highest count among all genres in the dataset.

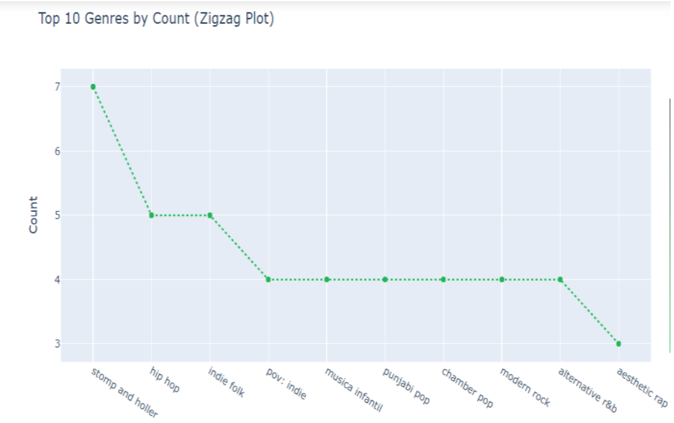

Fig 8: Genre counts

Genre Insights: Analysis of Figure 4 reveals that electrohouse is a popular genre, yet it doesn't have the maximum count as shown in Figure 8. It's notable that genres with higher popularity tend to have lower counts in the dataset as none of the popular genres have high count.

B. **Visualizations with respect to albums**

1. Figure 9 shows the pie chart of album types in the dataset, including singles, compilations, and albums, is illustrated. The visualization indicates that a majority are of type albums.

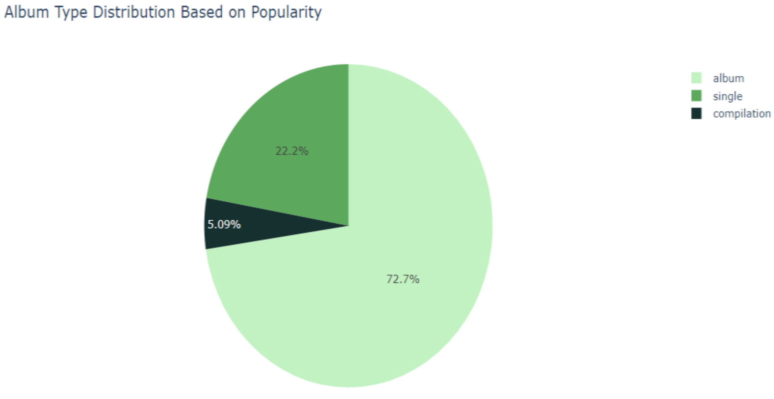

Fig 9: Album type distribution

2. Figure 10 gives an overview if an album consists of 1 or 2 artists.

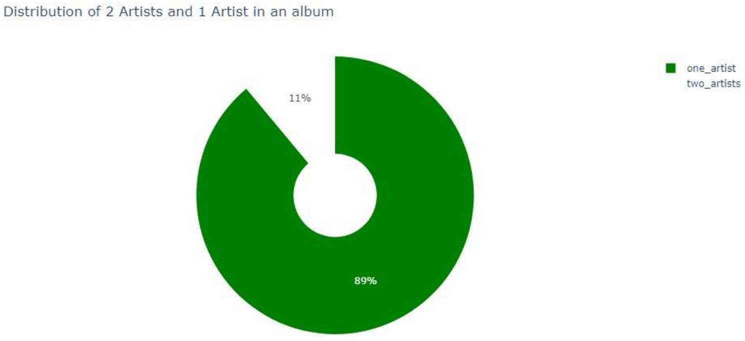

Fig 10: Artists in album donut chart

3. Figure 11 presents a line chart depicting the total number of tracks for an album over the years, commencing from 2010

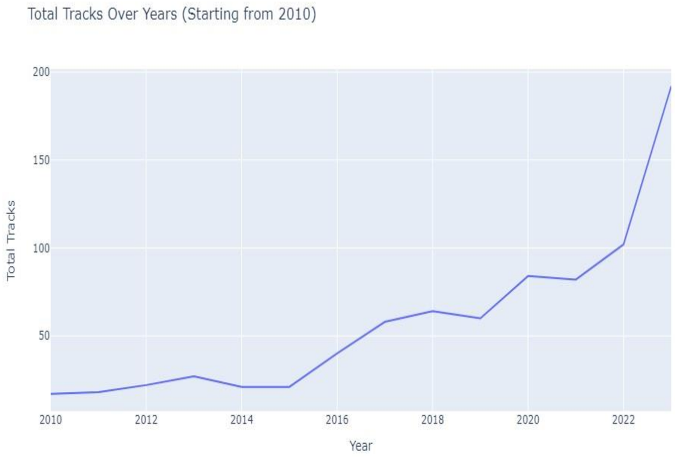

Fig 11: Donut chart of artists in album

4. Figure 12 shows a 3d plot indicating that most of the popular albums(circle in yellow) have high number of tracks.

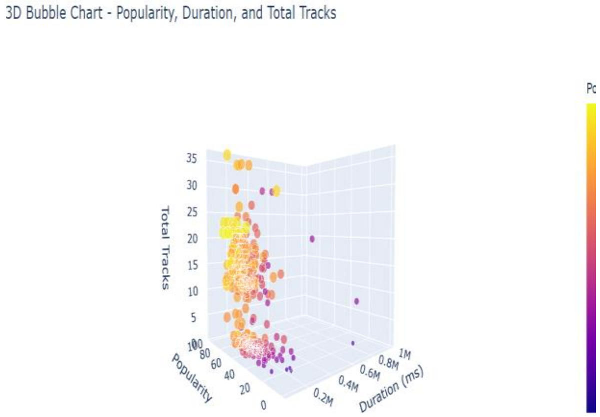

Fig 12: 3D album plot

5. Figure 13 shows the top 10 popular albums.

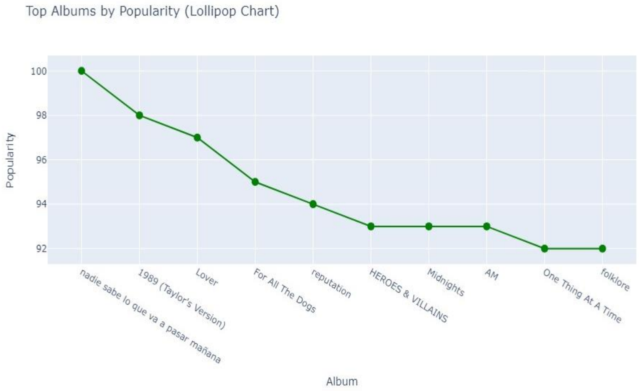

Fig 13: Popular albums

6. In Figure 14, the visualization displays the artists associated with popular albums. Notably, Taylor Swift stands out with a count of 5 popular albums.

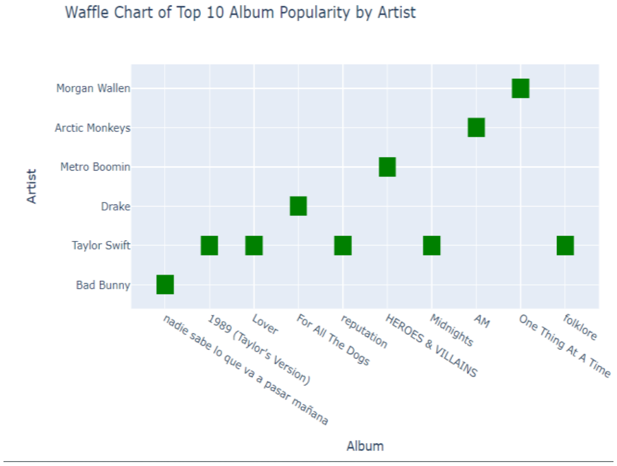

Fig 14. Artists of popular albums

7. Figure 15 shows a correlation between artist and album popularity and shows slight correlation between artist and album popularity

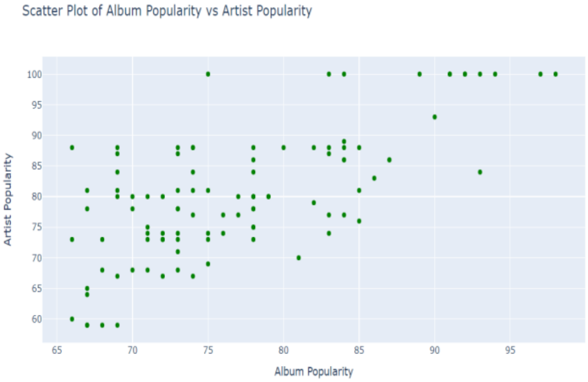

Fig 15: Artist-Album Popularity Relationship

Key Takeaways: The top-rated albums, as seen in Figure 14, prominently feature 5 albums by Taylor Swift, who is also identified as one of the popular artists (Fig 17) Additionally, the correlation graph (Fig 15) indicates a connection, suggesting that the popularity of some albums may be influenced by the popularity of the associated artist, and vice versa.

C. **Visualizations with respect to artists**

1. Figure 16 shows a funnel chart for top5 artists based on followers

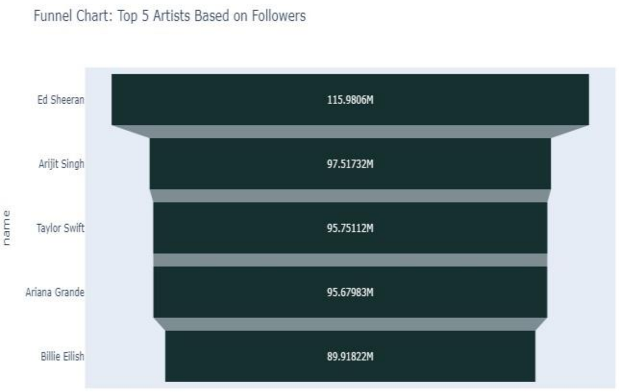

Fig 16: Top artists by followers

2. Figure 17 shows top5 artists based on popularity

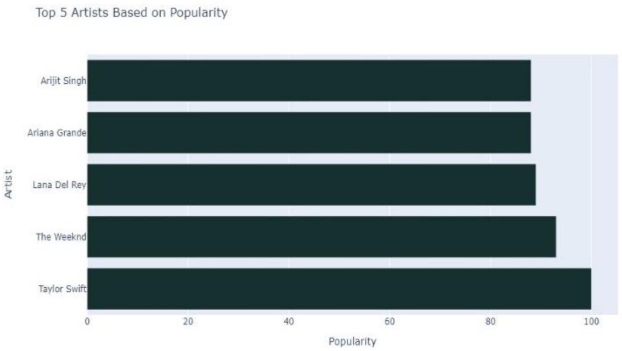

Fig 17: Top artists by popularity

3. Figure 18 shows a scatter plot between artist's followers and popularity and it can be seen that as the popularity increases the followers increase too.

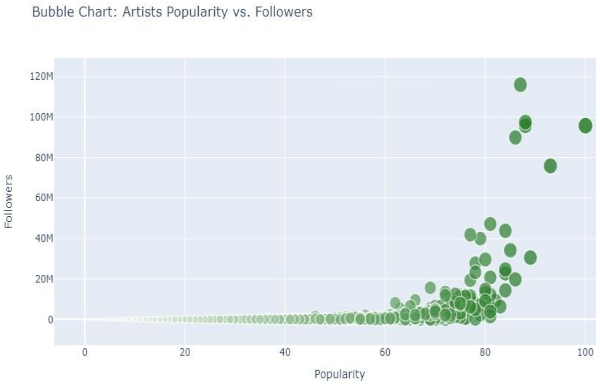

Fig 18: Popularity vs followers

4. Figure 19 shows the list of artists having maximum popularity and followers

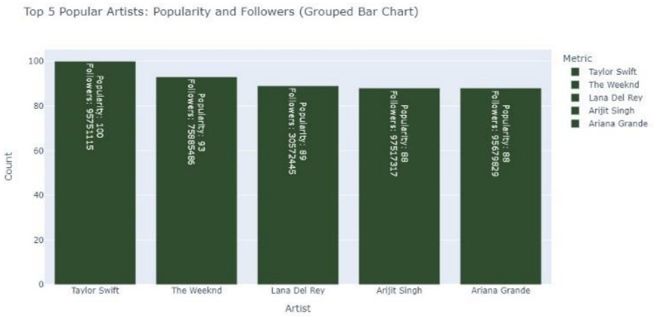

Fig 19: Popularity vs followers

Key Takeaways: From figure 18 it can be understood that popularity affects the followers and vice-versa, more the followers more the popularity. Fig 19 shows that Taylor Swift is the most followed and most popular artist. It even shows Taylor Swift, Arjit Singh, Ariana Grande have maximum followers(fig 16) and maximum popularity (fig 17).

D. **Visualizations with respect to tracks**

1. Figure 20 shows the distribution of explicit and non-explicit tracks. Black color shows explicit distribution. Maximum tracks are non-explicit.

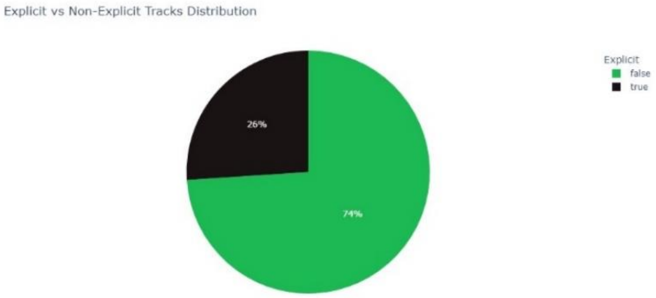

Fig 20: Explicit vs Non Explicit Tracks

2. Figure 21 shows a sunburst chart of popular tracks and their artists. The inner circle represents the artists name while the outer circle are the name of tracks

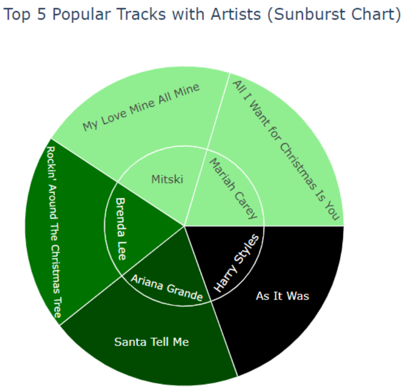

Fig 21: Explicit vs Non Explicit Tracks

3. Figure 22 shows the tracks popularity for popular albums, it can be seen that 1989 album has maximum popular tracks

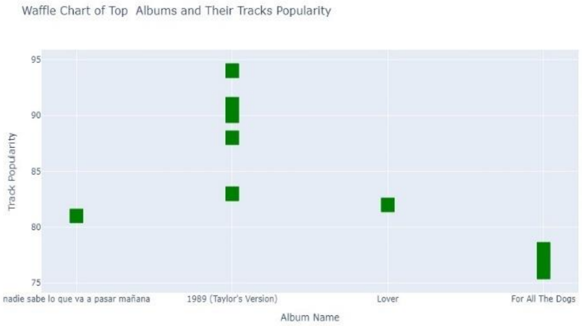

Fig 22: Top albums and their tracks

4. Figure 23 shows popular artists and their tracks. It can be seen that although Ariana Grande has maximum tracks, Taylor Swift's tracks are more popular.

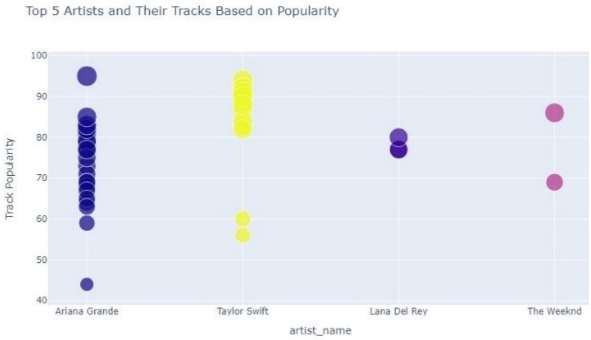

Fig 23: Top artists and their tracks

5. Figure 24 shows the albums having maximum tracks and even displays the album's popularity

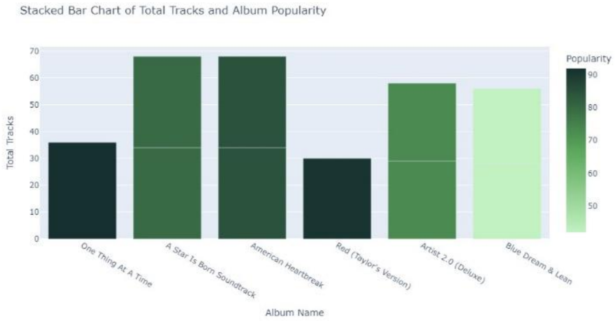

Fig 24: Max Tracks Albums

6. Figure 25 shows relation between album popularity and track popularity, it can be seen that there is slight correlation between album popularity and track popularity.

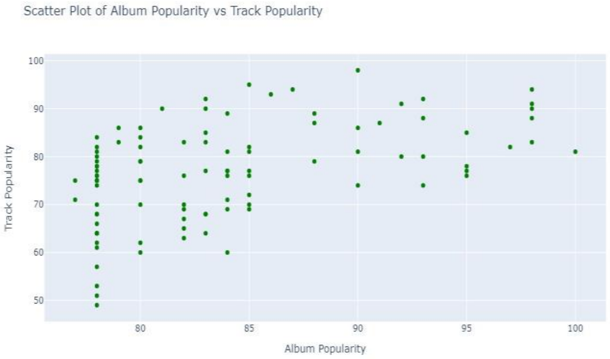

Fig 25: Album-Track Popularity Correlation.

Key Takeaways: Popular albums feature highly popular tracks(fig 22), highlighting a subtle correlation between album and track popularity(fig 25)

V. Conclusion and Future Work

This project explored Spotify's music distribution dynamics, analysing relationships between albums, tracks, and artists. In this project, we conducted data extraction, cleaning, transformation, and visualization on Spotify data.. The results and evaluation section presented insightful visualizations into tracks, albums, and artists. Below figures (fig 26 and fig 27) show correlation between the popularity of artists, tracks and albums. It is notable that the popularity of album affects artist and tracks and vice-versa. The scatter plot shows that if artist popularity (yellow circles) is high the track and artist popularity also increases. Hence we can conclude that this project helps in showing correlations between the three datasets, album, tracks and artist.

Utilizing machine learning, we can predict music trends, classify genres for better track recommendations, and conduct comparative analyses with other streaming platforms. Algorithms can identify emerging trends or external factors affecting popularity. A recommendation system considering genres and mood analysis can be developed, and integrating Spotify with social media enhances the overall user experience.

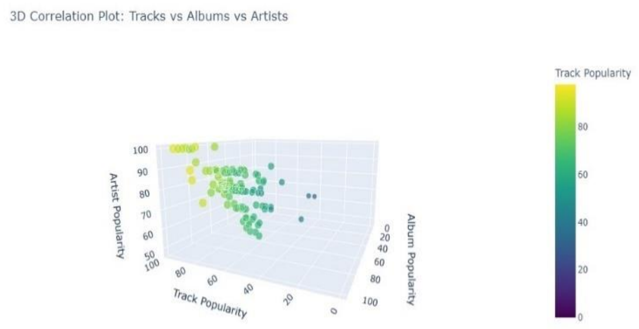

Fig 26: 3d plot of popularity

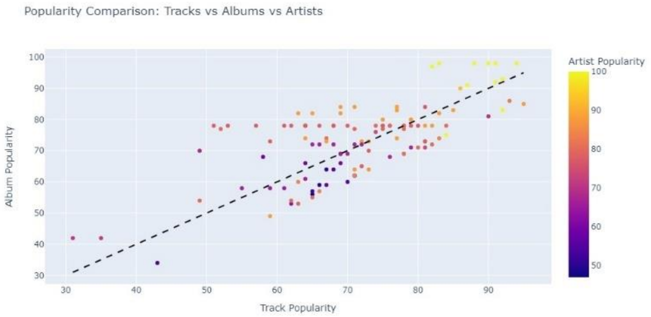

Fig 27: Scatter plot of popularity

VI. REFERENCES

[1] Q. Shen, S. Wu, Y. Zou and B. Xie, "Comprehensive Integration of API Usage Patterns," 2021 IEEE/ACM 29th International Conference on Program Comprehension (ICPC), Madrid, Spain, 2021, pp. 83-93, doi: 10.1109/ICPC52881.2021.00017.

[2] B. Cherry, P. Benats, M. Gobert, L. Meurice, C. Nagy and A. Cleve, "Static Analysis of Database Accesses in MongoDB Applications," 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), Honolulu, HI, USA, 2022, pp. 930-934, doi: 10.1109/SANER53432.2022.00111.

[3] Y. Gu, S. Shen, J. Wang and J. -U. Kim, "Application of NoSQL database MongoDB," 2015 IEEE International Conference on Consumer Electronics - Taiwan, Taipei, Taiwan, 2015, pp. 158-159, doi: 10.1109/ICCE-TW.2015.7216831.

[4] S. G. Babić and K. Cetina, "Processing and Visualization of Collected Data Based on Open-Source Tools and Principles," 2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO), Opatija, Croatia, 2020, pp. 1736-1739, doi: 10.23919/MIPRO48935.2020.9245215.

[5] B. Zhang et al., "Understanding user behavior in Spotify," 2013 Proceedings IEEE INFOCOM, Turin, Italy, 2013, pp. 220-224, doi: 10.1109/INFCOM.2013.6566767.

[6] L. Vardo, J. Jerkić and E. Žunić, "Predicting Song Success: Understanding Track Features and Predicting Popularity Using Spotify Data," 2023 22nd International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, Bosnia and Herzegovina, 2023, pp. 1-6, doi: 10.1109/INFOTEH57020.2023.10094172.

[7] N. Lin, P. -C. Tsai, Y. -A. Chen and H. H. Chen, "Music recommendation based on artist novelty and similarity," 2014 IEEE 16th International Workshop on Multimedia Signal Processing (MMSP), Jakarta, Indonesia, 2014, pp. 1-6, doi: 10.1109/