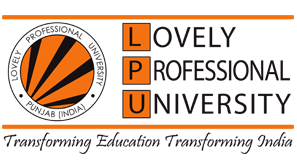
****

**CSE 353 EDA PROJECT REPORT**

**ON**

**DATA SET: Top 10000 songs on Spotify 1960-Now**

**Submitted by:**

**Komal pasumarthy**

**Reg No: 12106222**

**Section: K21UG**

**Roll Number: RK21UGB38**

**Domain Knowledge**

In today's fast-paced and interconnected world, music has cemented its position as a universal and indispensable medium of human expression. It transcends linguistic and cultural boundaries, touching the hearts and souls of people across the globe. From soothing melodies that offer solace in times of distress to pulsating beats that ignite euphoria on dance floors, music is an art form that resonates deeply with us all.

At the forefront of this musical revolution stands Spotify, a true pioneer in the realm of music streaming. In an age where access to virtually any song, artist, or album is just a few clicks away, Spotify has emerged as a dominant force, redefining how we consume and engage with music. Its remarkable prowess lies not only in its extensive catalogue but also in its ability to understand, adapt, and cater to its users' unique tastes and preferences.

What sets Spotify apart is its unparalleled recommendation system, a technological marvel that seamlessly curates and tailors music experiences for each individual listener. Drawing upon a vast trove of data and employing cutting-edge machine learning algorithms, Spotify has perfected the art of suggesting songs, playlists, and artists based on an intricate web of factors. These factors encompass everything from language preferences to artist affinities, moods, occasions, and even intricate listening patterns. It's this ability to understand and predict what a listener desires at any given moment that has left music enthusiasts utterly captivated.

Yet, in the world of technology and data-driven innovation, the pursuit of perfection is an ongoing journey. Spotify, like any visionary entity, recognizes that there is always room for enhancement, evolution, and refinement. The quest to elevate the user experience, to immerse listeners in a world of music that resonates with them on a profound level, continues unabated.

This brings us to the heart of our project—a comprehensive Exploratory Data Analysis (EDA) of Spotify's Top 10,000 Songs. Spanning from the musical landscape of 1960 to the present day, this dataset encapsulates a rich tapestry of sonic experiences, offering a window into the evolution of musical preferences and trends over the decades.

Through this EDA, we embark on a journey to unearth hidden gems of insight and understanding within the vast universe of Spotify's musical data. Our goal is twofold: first, to gain a deeper understanding of the dynamic interplay between songs, artists, and their listeners; and second, to harness this newfound knowledge to further elevate user satisfaction and enrich the overall music listening experience on the platform.

As we delve into the dataset, we will explore the multifaceted world of music, examining attributes such as tempo, danceability, energy and many more. These dimensions, often analyzed by Spotify's recommendation engine, shape the musical landscape and influence the songs that resonate most with individual listeners. Understanding these attributes is essential in deciphering the intricate art of song recommendation.

**Data Understanding**

Track URI: A unique identifier for each track on Spotify, often used for linking to specific songs. Data Type: String (URI).

Track Name: The name or title of the song. Data Type: String.

Artist URI(s): Unique identifiers for the artist(s) associated with the track, typically used for linking to artist profiles. Data Type: String (URI).

Artist Name(s): The name(s) of the artist(s) who performed the track. Data Type: String.

Album URI: A unique identifier for the album that contains the track, often used for linking to album details. Data Type: String (URI).

Album Name: The title of the album where the track is featured. Data Type: String.

Album Artist URI(s): Unique identifiers for the artist(s) who contributed to the album. Data Type: String (URI).

Album Artist Name(s): The name(s) of the artist(s) responsible for the album. Data Type: String.

Album Release Date: The date when the album containing the track was released. Data Type: Date or String (depends on the format).

Album Image URL: A URL pointing to an image representing the album cover. Data Type: String (URL).

Disc Number: Indicates which disc of a multi-disc album the track belongs to. Data Type: Integer.

Track Number: The position of the track within its album. Data Type: Integer.

Track Duration (ms): The duration of the track in milliseconds. Data Type: Integer.

Track Preview URL: A URL that allows users to preview a short segment of the track. Data Type: String (URL).

Explicit: Indicates whether the track contains explicit content (e.g., explicit lyrics). Data Type: Boolean (or Integer with binary values).

Popularity: A measure of the track's popularity on Spotify, typically represented as a numerical value. Data Type: Integer or Float.

ISRC: International Standard Recording Code, a unique identifier for sound recordings. Data Type: String.

Added By: The user or source that added the track to the dataset. Data Type: String or User ID.

Added At: The date and time when the track was added to the dataset. Data Type: Date and Time.

Artist Genres: Genres associated with the artist(s) of the track. Data Type: String or List of Strings.

Danceability: A measure of how suitable the track is for dancing, typically represented as a numerical value. Data Type: Float.

Energy: A measure of the track's energy level, often represented as a numerical value. Data Type: Float.

Key: The key in which the track is composed, represented as a musical key (e.g., C, D, G). Data Type: String.

Loudness: The overall loudness of the track, typically represented as a numerical value. Data Type: Float.

Mode: Indicates whether the track is in a major or minor key. Data Type: Integer (0 for minor, 1 for major).

Speechiness: A measure of how much speech (spoken words) is present in the track. Data Type: Float.

Acousticness: A measure of the track's acoustic qualities (acoustic vs. electronic), typically represented as a numerical value. Data Type: Float.

Instrumentalness: A measure of whether the track is instrumental (no vocals), typically represented as a numerical value. Data Type: Float.

Liveness: A measure of the presence of a live audience in the track, typically represented as a numerical value. Data Type: Float.

Valence: A measure of the track's positivity or happiness, often represented as a numerical value. Data Type: Float.

Tempo: The tempo or beats per minute (BPM) of the track, typically represented as a numerical value. Data Type: Float.

Time Signature: The time signature of the track (e.g., 4/4), indicating the number of beats in each bar. Data Type: String.

Album Genres: Genres associated with the album containing the track. Data Type: String or List of Strings.

Label: The record label that released the track or album. Data Type: String.

Copyrights: Information regarding copyright and ownership of the track. Data Type: String.

**Key Questions/objectives**

In this EDA project on the Spotify Top 10,000 Songs dataset, I have outlined some key questions and strategies to guide our exploration and gain valuable insights:

1. Most popular Genre across decades?
2. How do popularity scores vary among the songs?
3. Trends in Albums release over the years(Finding in which year most of the albums are released?)
4. What is the distribution of loudness levels in the songs?
5. Is there a noticeable pattern in the distribution of tempos (BPM)?
6. What is the distribution of time signatures among the songs?
7. Relation between Frequency and popularity, can we deduce a pattern?
8. How does acousticness vary across different genres?
9. What are the most common artist genres in the dataset?
10. Who are the top artists in terms of the number of tracks they have in the dataset?
11. Which artists have the highest popularity scores for their songs?
12. Average Song duration over years, which year have highest average song duration?
13. What are the most popular albums based on user engagement (e.g., plays, likes)?
14. How does the distribution of album release dates look for specific artists?
15. Are there any noticeable patterns in the distribution of album genres?
16. Can we visualize how album popularity scores have changed over time?
17. What are the most common album artists in the dataset?
18. How does the distribution of album durations look?
19. Is popularity dependent on spechiness?
20. Most frequent artists?

**Libraries Used and their description**

**Pandas**: Essential for data manipulation and analysis. You can load, clean, filter, and transform your dataset using pandas.

**NumPy**: Provides support for mathematical operations and array manipulation, often used in conjunction with pandas.

**Matplotlib**: A powerful library for creating static, animated, and interactive visualizations in Python.

**Seaborn**: Built on top of matplotlib, seaborn provides a higher-level interface for creating attractive and informative statistical graphics.

**SciPy**: Offers a wide range of statistical tests and functions for scientific and technical computing.

**Scikit-learn**: If you decide to include machine learning in your analysis, scikit-learn is a popular library for various machine learning tasks.

**How libraries are used**

**Pandas:**

Description: pandas is a versatile data manipulation library that allows us to load, clean, filter, and transform our dataset with ease. It's the foundation for data analysis in Python.

Usage: we can use pandas to load our music dataset from various file formats, clean and preprocess data, handle missing values, perform data aggregations, and create subsets of our data for specific analyses.

**NumPy:**

Description: NumPy is a fundamental library for numerical operations in Python. It provides support for mathematical operations and array manipulation.

Usage: NumPy complements pandas by enabling mathematical operations on our data. It's particularly useful for tasks like calculating statistical measures, working with arrays of data, and performing element-wise operations.

**Matplotlib**:

Description: matplotlib is a powerful library for creating static and interactive visualizations in Python. It offers a wide range of plotting functions.

Usage: we can use Matplotlib to create various plots and charts to visualize trends, distributions, and relationships in our music dataset. It's essential for creating informative data visualizations.

**Seaborn:**

Description: seaborn is built on top of matplotlib and provides a high-level interface for creating attractive and informative statistical graphics.

Usage: seaborn simplifies the creation of complex statistical plots such as heatmaps, pair plots, and violin plots. It enhances the aesthetics of your visualizations and makes it easier to explore data relationships.

**SciPy:**

Description: SciPy is a library that extends NumPy and provides a wide range of statistical tests and functions for scientific and technical computing.

Usage: we can use SciPy for statistical analysis and hypothesis testing in our music dataset project. It offers functions for t-tests, ANOVA, correlation tests, and more.

**Statsmodels:**

Description: Statsmodels is focused on statistical analysis and provides classes and functions for estimating various statistical models.

Usage: stats models is useful when we need to perform time series analysis, regression analysis, or hypothesis testing in our music dataset project. It's particularly helpful for in-depth statistical investigations.

**scikit-learn:**

Description: sci-kit-learn is a comprehensive machine-learning library that offers a wide range of machine-learning algorithms and tools.

Usage: If we decide to incorporate predictive modelling or clustering into our analysis, sci-kit-learn provides ready-to-use algorithms for tasks such as classification, regression, and clustering.

TensorFlow or PyTorch:

Description: TensorFlow and PyTorch are deep learning frameworks if we plan to explore deep learning models for music analysis, such as genre classification or sentiment analysis.

Usage: These frameworks enable us to build and train neural networks for more complex and deep analysis of your music dataset.

Beautiful Soup and Selenium:

Description: Beautiful Soup and Selenium are libraries for web scraping, which can be helpful if we need to acquire additional data from online sources.

Usage: we can use these libraries to automate the process of collecting data from websites or online APIs to complement your music dataset.

**Steps of EDA performed till now**

**Step 1: Data Loading and Initial Inspection:**

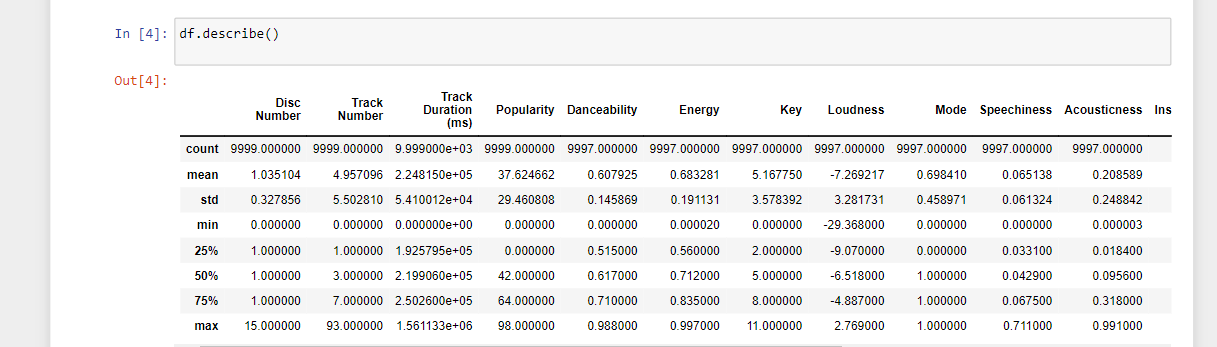
So, I began by loading the music dataset. With the help of Python's pandas library, I imported the dataset like this:

**import pandas as pd**

**df=pd.read\_csv(r"D:\LPU Academic Files\Untitled Folder\top\_10000\_1960-now.csv")**

After that, I wanted to get a quick glimpse of what the data contained, so I checked the first few rows by using df. head(). This was a helpful way to see the column names and have a peek at the actual data.

Next, I wanted to know more about the data types of each column and check if there were any missing values. Using df.info(), I could see the data types and identify any gaps in the dataset. It's crucial to understand the data types as they help in interpreting the meaning of each column. Additionally, spotting missing values is essential for data quality.

To get some basic statistics for the numerical columns, I employed df.describe(). This gave me statistics such as the mean, standard deviation, minimum, maximum, and quartiles. These stats provided an initial insight into the numerical attributes of the data.

**Step 2: Data Cleaning and Preprocessing:**

Having understood the dataset better, the next step was to clean and preprocess the data to make it suitable for analysis

Dealing with missing data was a priority. I had to decide on a strategy, and in this case, I chose to impute missing values. Using df.fillna(), I could fill in those gaps with appropriate values, like the mean, median, or mode of the respective column. Additionally, I removed rows with missing data using df.dropna(), ensuring a cleaner dataset.

Duplicate rows can introduce redundancy and skew the analysis, so I checked for duplicates and removed them using df.drop\_duplicates().

If there were any date columns, I converted them to the datetime format with pd.to\_datetime(). This conversion made it easier to work with time-based analysis.

A screenshot of a computer

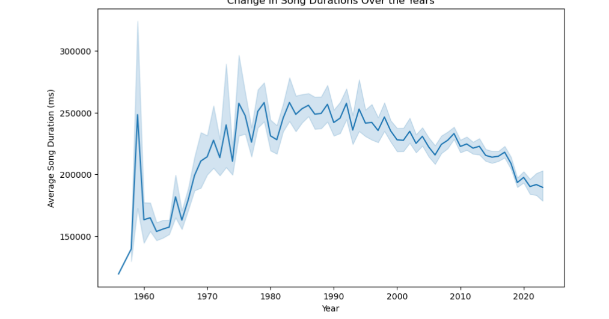
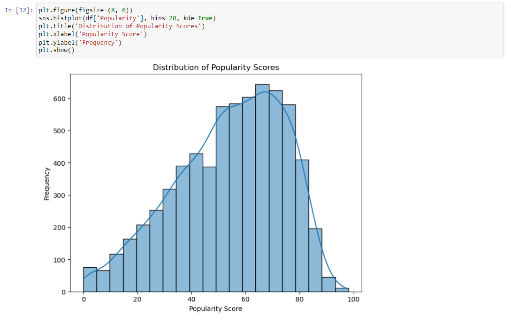
Description automatically generated

**Step 3: Data Visualization - General Exploration:**

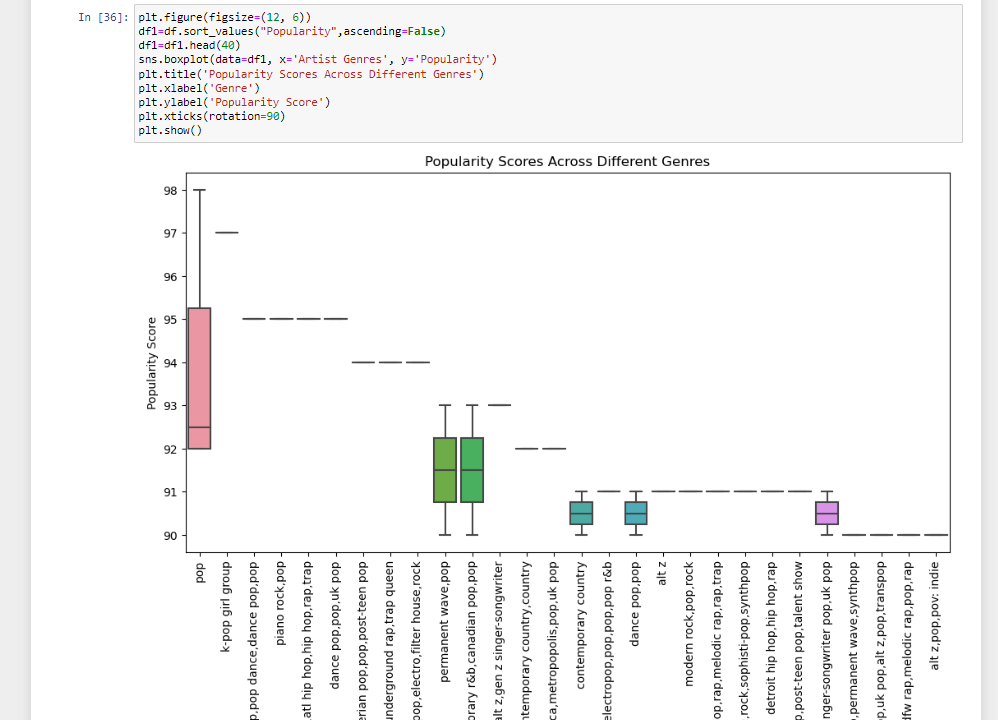
To better understand the distribution and characteristics of the data, I turned to data visualization. Visualizations are powerful tools for uncovering patterns and insights.

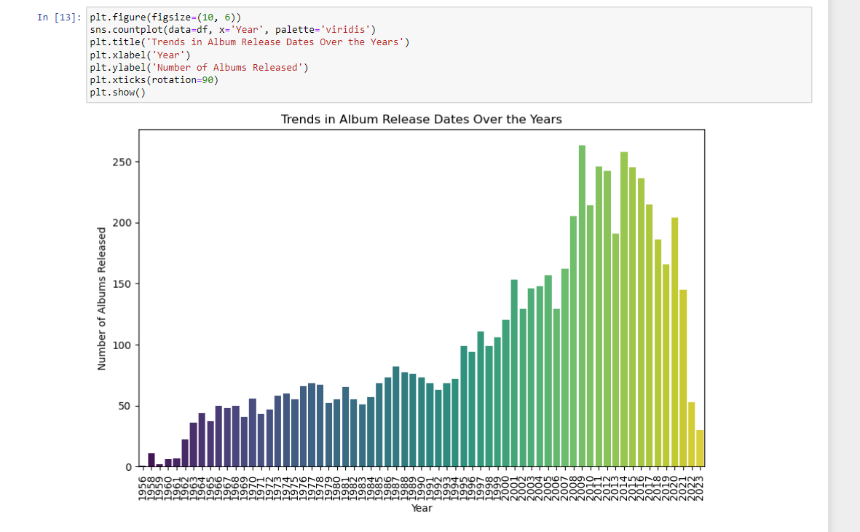
With libraries like matplotlib and seaborn at my disposal, I created various visualizations. For instance, I crafted histograms, scatter plots, and line charts to visually represent attributes like song durations and popularity scores.

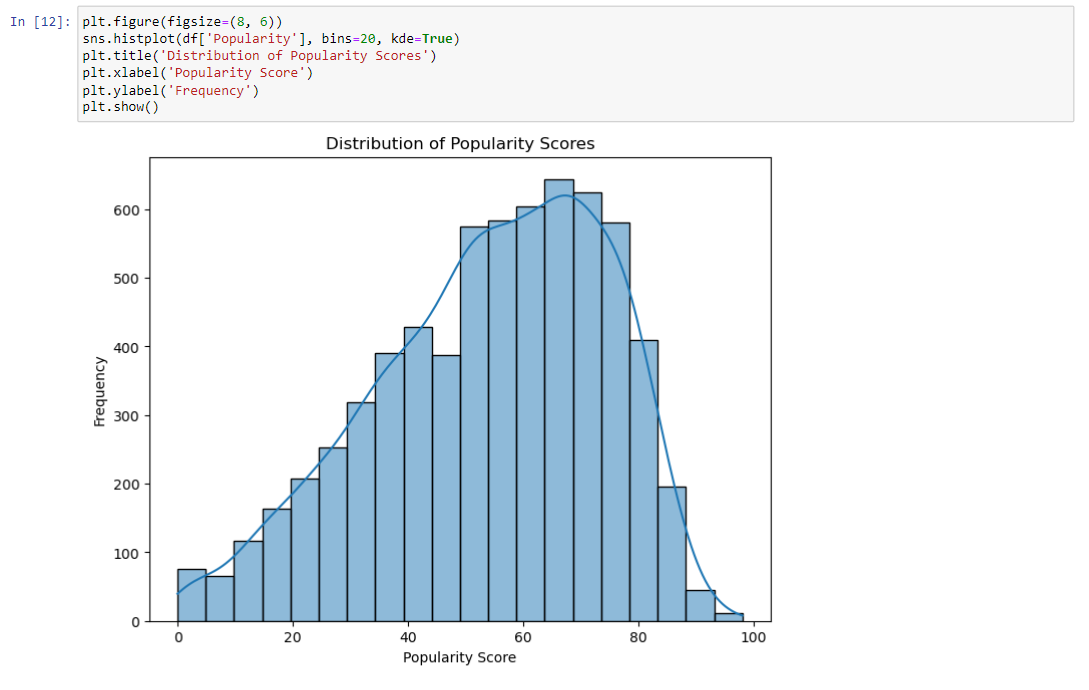
A screen shot of a graph

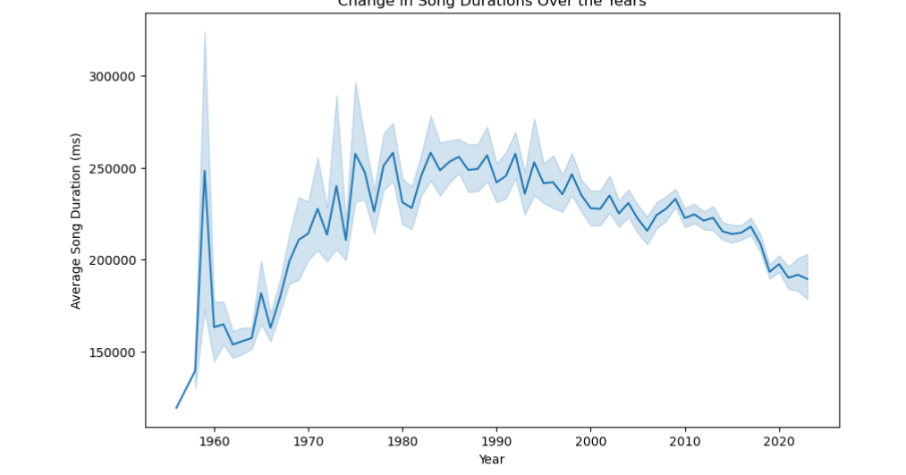
Description automatically generatedThese visualizations allowed me to explore trends, such as those in album release dates, and gain insights into categorical variables, like explicit content and time signatures. It was a valuable step in drawing initial inferences and identifying areas for deeper analysis.

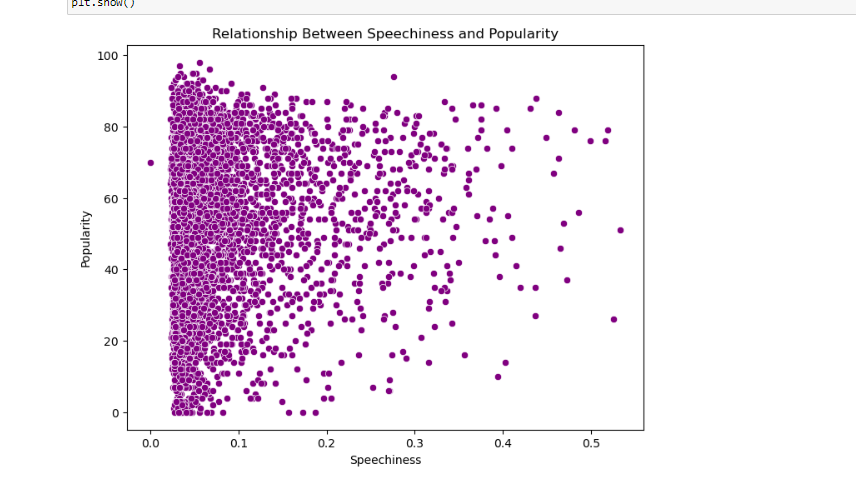
**Answers for my Questions:**



* Most popular Genre across decades?
  + Ans: Pop
  + From 1960 the most popular songs belong to pop genere which has been understood after plotting a boxplot for top genres! With a score varying from 92-98
* Trends in Albums release over the years(Finding in which year most of the albums are released)
  + Ans : from the countplot we can deduce that more than 250 albums are released in 2010



* Relation between Frequency and popularity, can we deduce a pattern? Ans:We can clearly deduce from the histplot that higher frequency songs have higher popularity score!
* Average Song duration over years, which year have highest average song duration?
  + Ans: we can clearly deduce that songs during 1960 have higher average duration and we can also get to know that there was an increase in song duration after 1960
* Most frequent artists?
  + Ans: Ed Sheeran, David Guetta, Madonna, beatles, taylor swift
* Frequency distribution of the songs according to their loudness
  + Ans: many songs have a frequency between 1000 and 1600 with loudness between 0 to 15dB
* Does Song Duration affect it’s popularity?
  + Ans: No it doesnot affect it’s popularity most of the songs have 20\*10^6 ms duration and have various popularity
* A chart with different colored squares

  Description automatically generatedTop 10 most common artist and their top genere
* Is popularity dependent on spechiness?
  + Ans: NO

A graph of different colored lines

Description automatically generated with medium confidence

* Distribution of Acoustic Ness across various eras
* Top artists by Number of tracks
  + Ans: Madona, Ed Sheeran, Elvis presley
* Top artists by Popularity scores
  + Ans: Fifty Fifty, David Guetta, bebe Rexha
* Top albums based on user engagement
  + Ans:Greatest Hits, Celebration, The definitive collection
* Top artists based on number of albums
  + Ans:Ed Sheeran(50+), Elvis(45+), Madonna(40+)
* A graph with blue lines

  Description automatically generatedTrends in popularity scores over year, which year has the highest?

Ans: After 2020 and before 1960 popularity score of the songs was comparatively higher than remaining period