Web Mining Final Project

**EchoVerse: Decoding the Sentiments and Success of Songs**

# INTRODUCTION

In today's digital era, music has become an integral part of our lives. With millions of songs available online, we can perform exploratory analysis and predictions to build powerful web applications using web mining tools for music enthusiasts. The goal is to utilize machine learning algorithms to generate useful insights and build a platform for both casual listeners and industry professionals.

# OBJECTIVE

The goal of this Web Mining project is to create a hub for music and song-related functionalities. We have performed web scraping to collect metadata for 4k songs (approx) and utilized machine learning algorithms for song summary generation, and categorization of songs based on mood and other attributes. Moreover, we created a content-based recommendation system and song success predictor for new and upcoming songs. Additionally, we created visualizations to provide meaningful insights into various aspects of song analysis. To summarize, The objectives of the project are as follows:

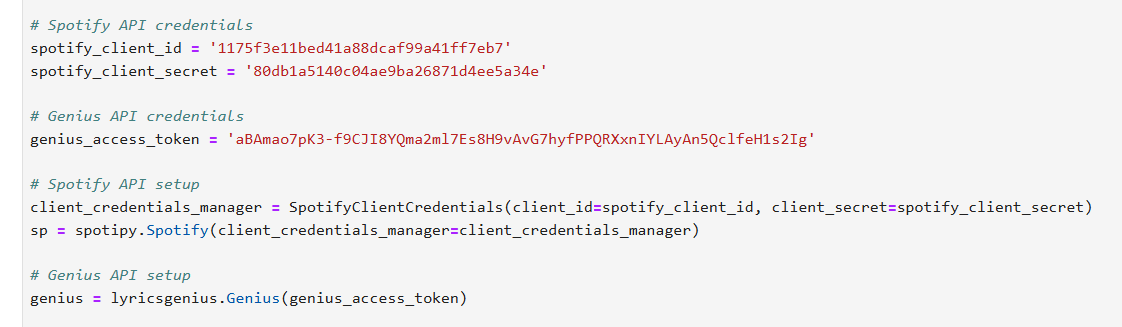
1. Summary Generation of lyrics of the songs
2. Sentiment Analysis and Categorization of songs into various moods
3. Content-based (Item-Item) Recommendation System
4. Song Success Prediction to predict if a new song would be successful based on the song characteristics through a trained model.

# DATA COLLECTION

**Information About Data:** We have collected the following features: **Title, Artist, Album, Release Date, Available Markets, Genre, ISRC, Lyrics, Tempo, Key, Mode, Danceability, Energy, Instrumentalness, Acousticness, Speechiness, Valence, Loudness, Track Popularity, Duration (ms),** and **Formatted Duration**.

* **API Integration** 
  + Used Spotipy for Spotify Web API and LyricsGenius for Genius API
  + Authenticated using appropriate credentials for each API
* **Data Retrieval**
  + Queried Spotify for track information, audio features, and popularity metrics
  + Fetched lyrics from Genius based on track and artist names
* **Error Handling and Rate Limiting** 
  + Implemented try-except blocks for API request failures
  + Added delays between calls to respect API rate limits
* **Data Structuring and Validation** 
  + Compiled retrieved data into a structured DataFrame
  + Performed basic checks for data integrity and handled missing values
* **Storage** 
  + Saved the compiled and validated data to an Excel file for further analysis

Implementation snippets for data collection



*Implementation of Spotify & Genius API’s*



*Features scraped for our dataset*

## **SENTIMENT ANALYSIS AND TREND ANALYSIS IN MUSIC LYRICS**

### Methods and Key Findings

* **Sentiment Analysis**
  + The main goal of this study was to explore and quantify the attitudes towards different aspects of life as expressed in music lyrics at different time periods and across different music genres.
* **Sentiment Classification Approach**
  + The sentiment analysis was done with the DistilBERT model that was fine-tuned on the SST-2 dataset for binary classification of text as having a positive or negative sentiment.
  + The summarized lyrical data was tokenized and fed into the model and it produced sentiment labels and their scores. This approach gave accurate results while at the same time being light on the computer's resources.
* **Sentiment Distribution Insights**
  + Sentiment distribution by genre showed some patterns in the data. The two genres that had the highest positive scores were Pop and Jazz, which is in line with the fact that these genres have been associated with positive and uplifting messages. Large-Scale Genre Classification
  + This is because the mapping of specific genres into broader categories allowed for a general view of the sentiment trends.

For instance:

**Pop**: It was observed that the highest percentage of positive songs was in the Pop genre.

**Hip-Hop**: It was also quite balanced with about 40% of the songs having negative sentiments which captured aspects such as the issues with the society and the personal struggles of the artists.

### Temporal Sentiment Trends

**Sentiments Over Decades**

* Based on the longitudinal analysis, it is possible to observe some changes in the patterns of sentiment expression in music.
* 1980s and 1990s: These decades were marked by the increasing use of positive sentiments which may be associated with the cultural optimism of the decade, technological progress, and increased interconnectedness of the world.
* Post-2010: There was a noticeable decrease in positive sentiment especially in the 'Hip-Hop' and 'Electronic' categories. This shift may be in line with the social and political conditions of the contemporary world and its issues regarding the economy and world affairs.

### Genre-Specific Temporal Trends

* Every genre had its own variations of temporal trends of sentiment:
* "Pop" was positive all the time and with only occasional variations.
* "Hip-Hop" showed an alternating pattern with highs of negative sentiment at certain time steps especially during the socio-political context of the early 2000s and the late 2010s.

### Popularity Analysis and Sentiment Correlation

**Sentiment and Popularity**

* The relationship between sentiment and track popularity was analyzed using the Track\_Popularity metric. Insights revealed that tracks with positive sentiments generally scored higher in popularity, a trend particularly evident in "Pop" and "Jazz." However, exceptions were observed in emotionally intense genres like "Rock" and "Hip-Hop." For instance:
* In "Rock," negative sentiments resonated deeply with audiences, contributing to the success of tracks reflecting themes of rebellion and emotional authenticity.
* Similarly, "Hip-Hop" tracks with negative sentiments gained significant popularity, underscoring the genre's role in amplifying voices and expressing raw emotions.

**Sentiment and Genre Trends**

* Boxplots depicting track popularity by sentiment for various genres showed distinct patterns:
* "Pop" and "Electronic" genres favored positivity, with positive tracks consistently achieving higher popularity scores.
* In "Rock" and "Hip-Hop," the median popularity of negatively sentimental tracks was comparable to, or sometimes exceeded, their positive counterparts, highlighting the complex relationship between sentiment and audience reception.

## **VIBECARE – A MUSIC THERAPY APPLICATION**

The Music Therapy Application leverages the emotional and therapeutic potential of music to provide users with curated playlists tailored to specific emotional states and therapeutic needs. Music has been scientifically proven to influence emotions, alleviate stress, and boost motivation. This application taps into these qualities by analyzing song lyrics, categorizing them based on sentiment, and organizing them into mood-specific playlists.

### **Key Features**

* **Sentiment Analysis**: The lyrics of songs are analyzed to determine their emotional polarity (positive or negative).
* **Mood-Based Categorization**: Songs are categorized into predefined moods such as "Anxiety Relief", "Motivation", and "Relaxation", among others, based on their sentiment score.
* **Dynamic Playlist Generation**: The application groups songs by category and generates dynamic playlists, which can be saved and used for specific therapeutic purposes.
* **Visualization**: Data visualizations provide insights into the distribution of song categories and average sentiment scores across different categories, helping users to understand the data better.

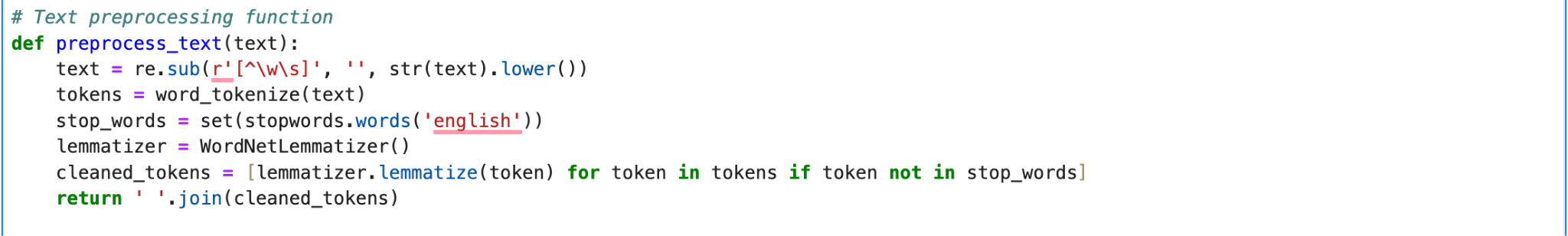
### **Applications**

* **Stress relief**: Through relaxing or calming music.
* **Emotional support**: By listening to music that alleviates anxiety or promotes positive emotions.
* **Motivation and empowerment**: By curating playlists that inspire and encourage action.
* **Personalized music experiences**: Tailored playlists based on emotional preferences.

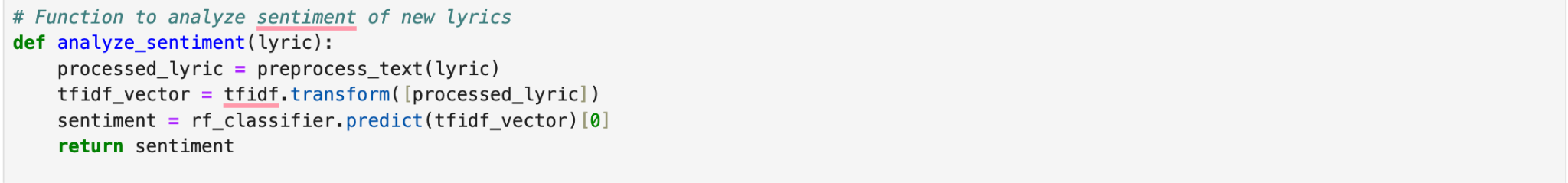
### **Implementation**

This code uses Random Forest Classification and TF-IDF for sentiment-based classification of song lyrics into predefined categories.

1. **Text Preprocessing**



1. **Feature Extraction**  
   
2. **Analysing Sentiment**

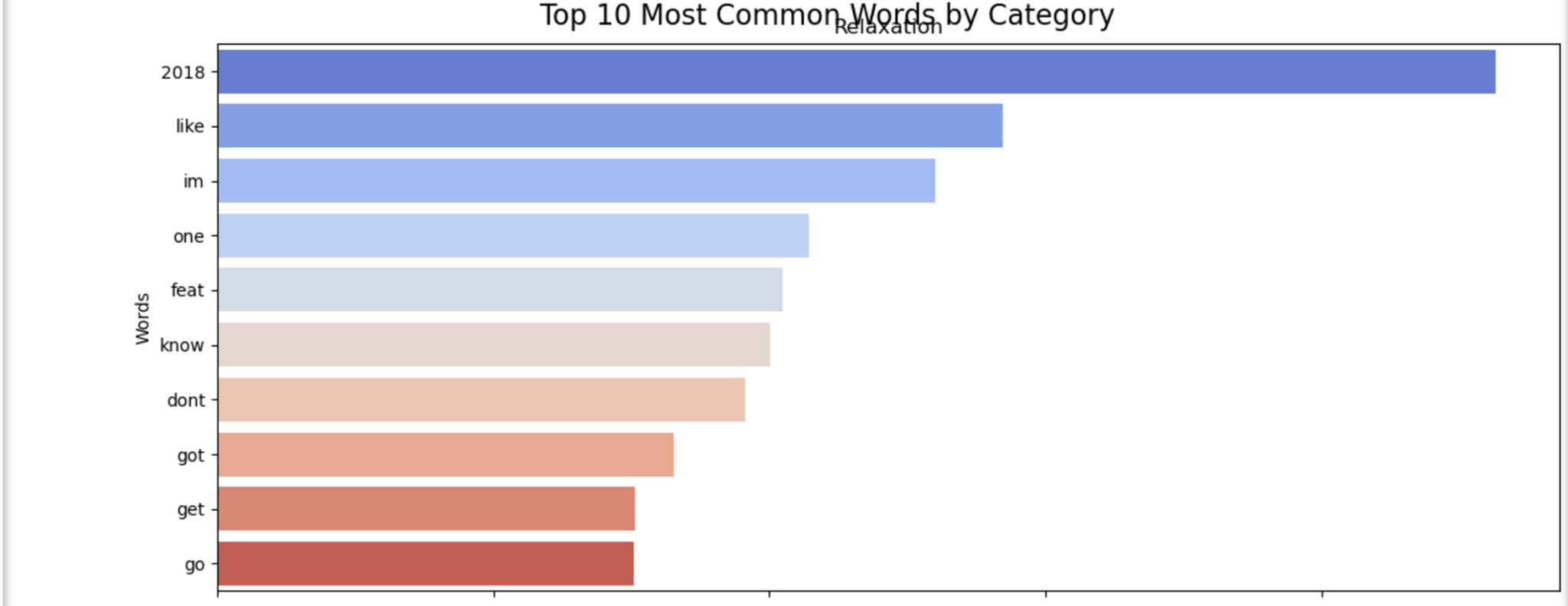


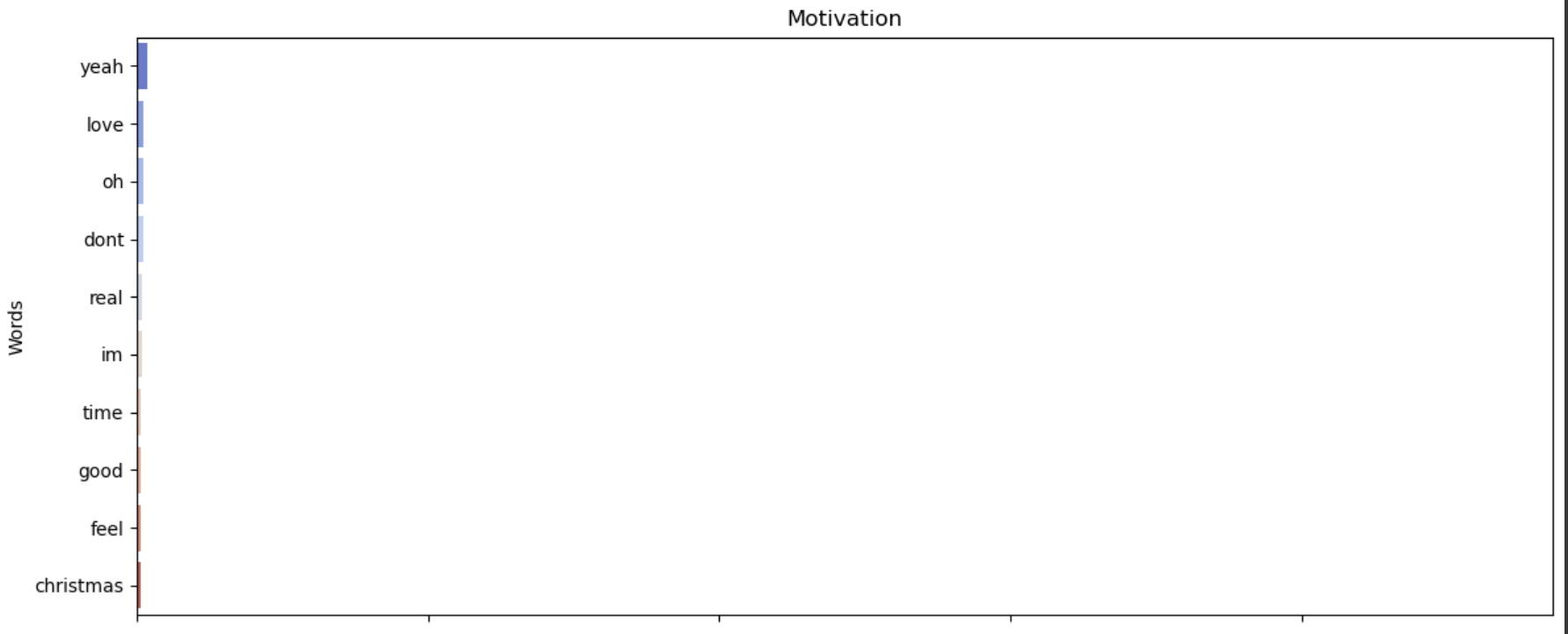
1. **Playlist creation**

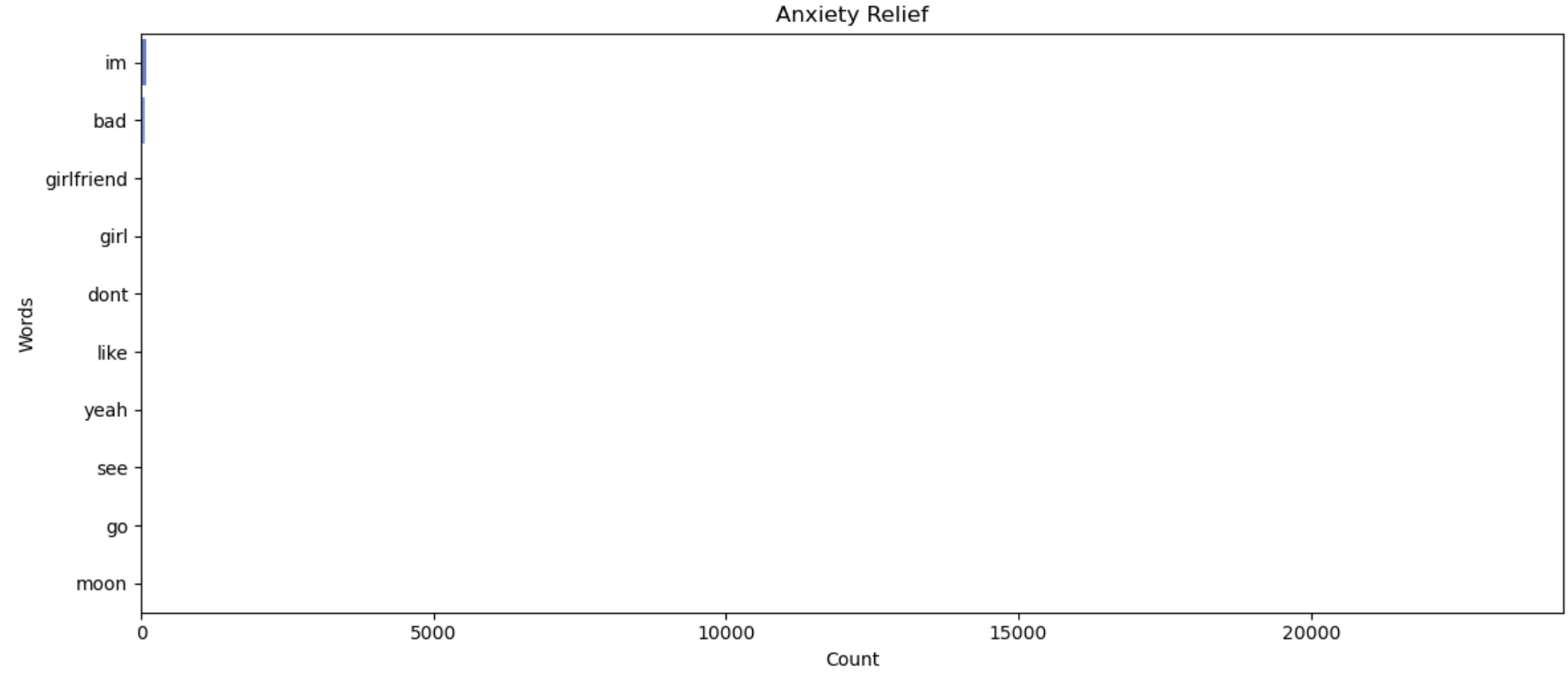


1. **Visualizations**









## **MOODIFY - MOOD PLAYLIST GENERATOR**

This Mood Playlist Generator is an advanced system that analyzes song lyrics, audio features, and sentiment to categorize music into different emotional states and create personalized mood-based playlists. By leveraging machine learning and natural language processing techniques, it offers a sophisticated approach to curating music that aligns with specific moods or emotional needs.

### **Key Features**

1. Sentiment Analysis: Uses NLTK's SentimentIntensityAnalyzer to evaluate the emotional tone of lyrics.
2. Topic Modeling: Applies Latent Dirichlet Allocation (LDA) to identify underlying themes in song lyrics.
3. Feature Fusion: Combines audio characteristics (Valence, Energy, Danceability), sentiment scores, and topic distributions for comprehensive song profiling.
4. Mood Clustering: Employs K-means clustering to group songs into distinct emotional categories.
5. Automated Playlist Generation: Creates mood-specific playlists based on the clustered emotional categories.
6. Data Visualization: Includes functions to visualize mood distributions across the music library.

### **Applications**

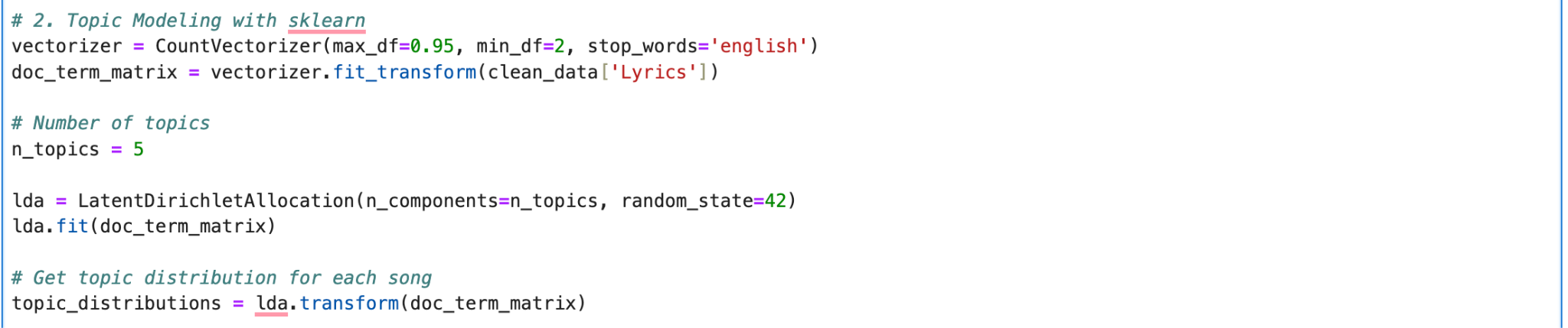
1. Personalized Music Experience: Offers tailored playlists that match users' current emotional states or desired moods.
2. Music Streaming Platforms: Enhances recommendation systems with mood-based song suggestions.
3. Emotional Wellness: Supports mental health by providing music aligned with therapeutic goals or emotional needs.
4. Content Creation: Assists in selecting background music that matches specific moods for videos, podcasts, or other media.
5. Event Planning: Helps in curating playlists for various occasions based on the desired emotional atmosphere.
6. Research: Provides a tool for studying the relationship between music, lyrics, and emotional impact.

### **Implementation**

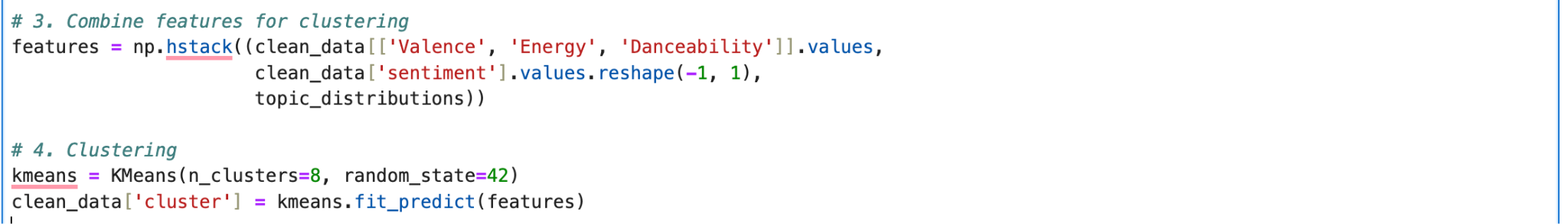
1. Sentiment Analysis



1. Topic Modelling



1. Feature Combination & Clustering



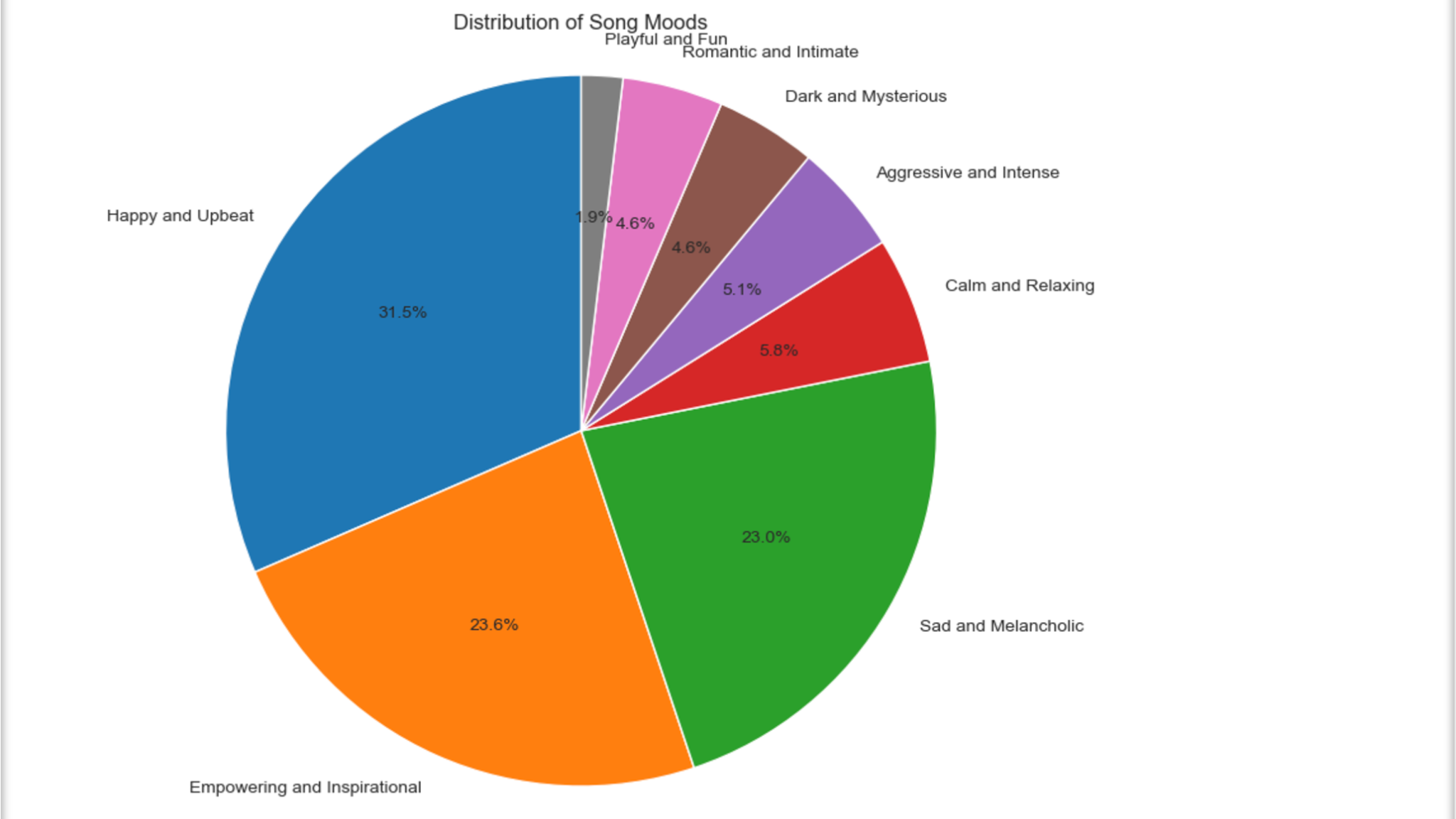
1. Mood labelling & Playlist Generation



1. Visualizations







## **CONTENT-BASED RECOMMENDATION SYSTEM**

* Content-based filtering uses item features to recommend other items similar to what the user likes. Here, we will use attributes and features about the song metadata to generate top recommendations for a song provided by the user.
* Content-based Item-Item filtering is an information retrieval method that uses item features to select and return items relevant to a user’s query. To get similar song recommendations, we have utilized machine learning library scikit-learn to generate relevant similarity scores.
* The recommendation system models utilized in this section include:
  + Cosine similarity
  + Sigmoid\_kernal

### **Application**

* Song Streaming companies: Can use recommendation systems to create personalized playlists for their listeners.
* Social Media: these platforms can leverage the song recommendation to incorporate music in reels and videos.
* Music Therapy: Recommendation systems can be used to create customized playlist for healing and relaxation purposes.

### **Implementation**

### **Data Cleaning**

* Removed empty values: Empty cells and values were identified and dropped from the dataset.

df= df.dropna()

* Eliminated duplicate entries: Duplicate rows were identified and dropped from the dataset

df=df.drop\_duplicates()

### Feature Transformation

* Standardization of Numerical Features: Data Scaling method, where numerical data is centered around mean with a unit standard deviation.
* Converted categorical features to numerical: The categorical Features are converted to numerical features using one-hot-encoding.
* Text features to vector form: used Tf-idf vectorizer to convert a collection of text data to a matrix of TF-IDF features(vectors).

## 

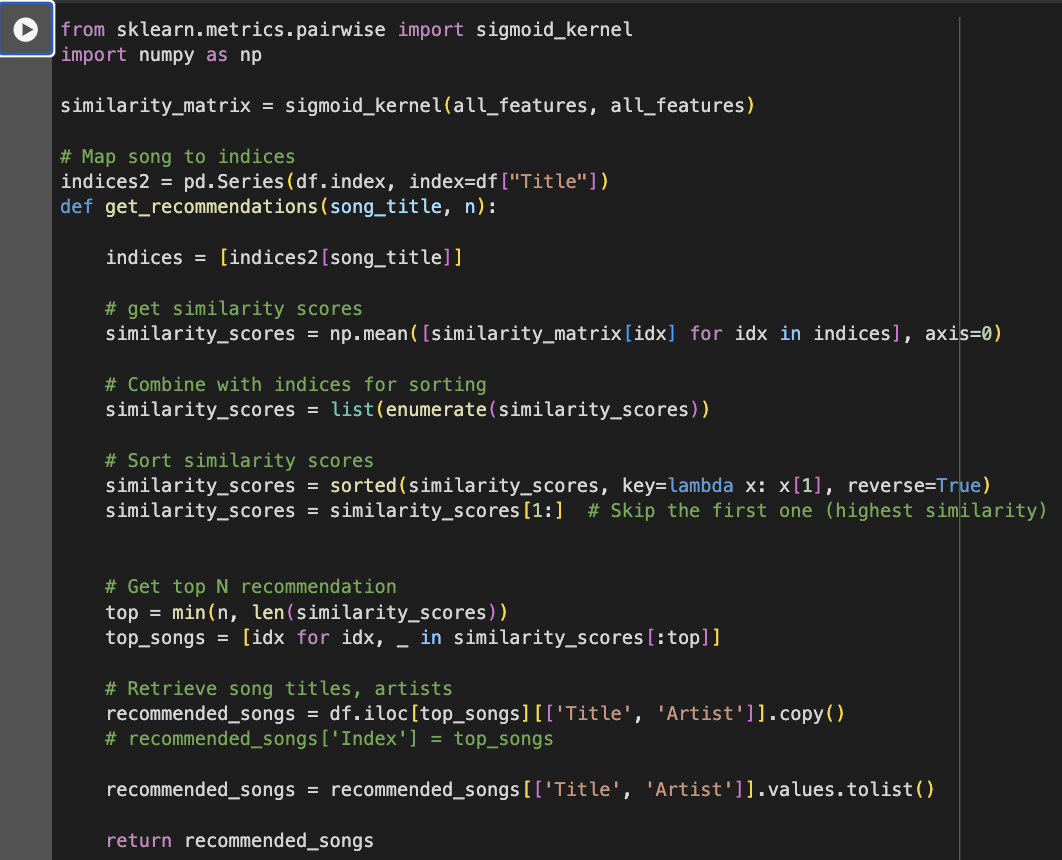
### **Cosine Similarity ML Model:**

* Cosine similarity is a measure of similarity between two non-zero vectors defined in an inner product space.



### **Sigmoid Kernel ML Model:**

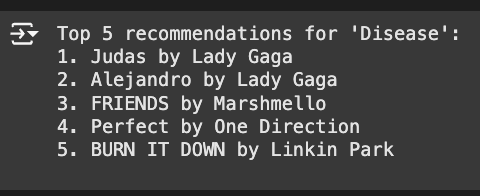
* Sigmoid\_kernel is a non-linear kernel function that computes the similarity between two input vectors.



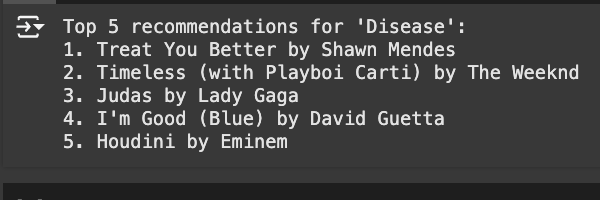
### **Results:**

* Both Cosine Similarity and Sigmoid Kernel, were used to generate the top 5 music recommendations for “Disease By Lady Gaga”.
* Both models performed well, as assessed through manual user testing and listening comparisons, given the absence of a definitive evaluation metric. Notably, both models recommended "Judas" by Lady Gaga, indicating some overlap in their outputs despite differing methodologies.
* These results demonstrate the models' ability to generate meaningful and relevant suggestions.

**For Cosine similarity**



**For Sigmoid Kernel**



### **Limitation and Future Scope**

* Can collect data with user information to create dynamic Collaborative filtering systems that would be more precise and user centric.
* Can implement Neural Networks for creating state of the art Recommendation system
* Having more sophisticated evaluation metrics for Content based recommendation systems would help in improving the systems performance.
* Can integrate the sentiment analysis, summary generated and categorical features created in the above two Objectives to get more accurate results for our Recommendation system.

## 

## **SONG SUCCESS PREDICTION MODEL: A COMPREHENSIVE ANALYSIS**

### Introduction

* In the ever-evolving landscape of the music industry, predicting the success of a song has become increasingly valuable. This project aimed to develop a robust model for predicting song success using a variety of features including audio characteristics, artist metrics, and engineered variables.
* By leveraging data from Spotify's API and applying advanced machine learning techniques, we sought to create a tool that could provide insights into the potential popularity of a track.

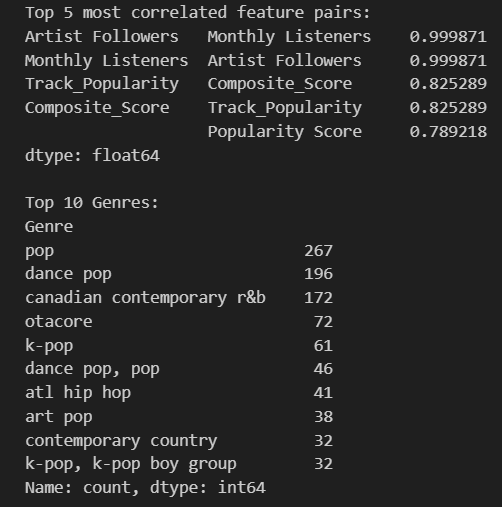
### Data Collection and Initial Exploration

**2.1 Dataset Overview:**

* Our dataset comprised songs from Spotify, containing a rich array of information.
  + Audio features (e.g., Danceability, Energy, Loudness, Speechiness)
  + Track metadata (e.g., Release Date, Duration)
  + Artist information (e.g., Artist Followers, Monthly Listeners)
  + Popularity metrics (Track Popularity, Popularity Score)

**2.2 Initial Data Analysis:**

* We began our analysis by exploring the basic characteristics of our dataset:
* **Basic Statistics**: We examined the distribution and range of our numerical features, identifying potential outliers and understanding the central tendencies of our data.
* **Correlation Analysis**: We performed a correlation analysis between features, visualized through a heatmap. This helped us identify potential relationships between variables and informed our feature selection process.
* **Genre Distribution**: We analyzed the distribution of genres in our dataset, finding that pop music was the most represented, followed by dance pop and Canadian contemporary R&B. This information was crucial for understanding potential biases in our data.
* **Temporal Analysis**: We examined yearly trends in song characteristics from 2000 to 2024, providing insights into how various aspects of popular music have evolved over time.
* **Top Correlations**: We identified the most strongly correlated feature pairs:
  + Artist Followers and Monthly Listeners (correlation: 0.999871)
  + Track Popularity and Composite Score (correlation: 0.825289)
  + Composite Score and Popularity Score (correlation: 0.789218)



* These initial analyses provided a foundation for our feature engineering and model development processes.

### Data Preprocessing and Feature Engineering

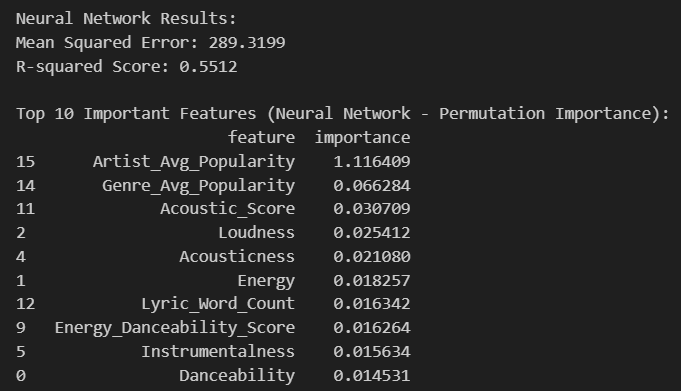
**3.1 Data Cleaning**

* **Handling Missing Values**: We addressed missing values in our dataset using mean imputation. This ensured that we could utilize all available data without significantly biasing our results.
* **Encoding Categorical Variables**: We used Label Encoding to convert categorical variables, such as genre, into a format suitable for our machine learning models.



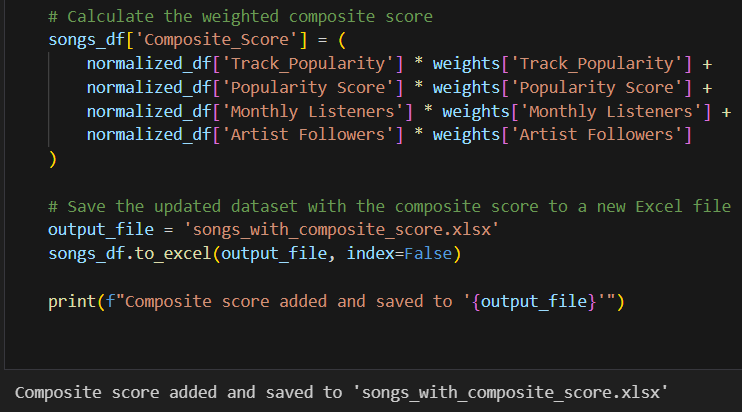
**3.2 Feature Engineering**

* To capture more nuanced aspects of songs and potentially improve our model's predictive power, we engineered several new features:
  + **Composite Scores:**
    - Energy\_Danceability\_Score = (Energy + Danceability) / 2
    - Mood\_Score = (Valence + Energy) / 2
    - Acoustic\_Score = (Acousticness + (1 - Energy)) / 2
  + **Interaction Features**:
    - Energy\_Danceability = Energy \* Danceability
    - Loudness\_Energy = Loudness \* Energy
  + **Artist and Genre Features**:
    - Artist\_Popularity\_Ratio = Artist Followers / Mean(Artist Followers)
    - Genre\_Popularity = Mean Track\_Popularity per Genre
  + **Time-based Features**:
    - We extracted the Release\_Month from the Release\_Date to capture potential seasonal trends.



**3.3 Composite Score**

* We developed a crucial feature called the Composite\_Score, which aimed to encapsulate overall song success in a single metric. This score was calculated as a weighted average of:
  + Track\_Popularity (40% weight)
  + Popularity Score (30% weight)
  + Monthly Listeners (20% weight)
  + Artist Followers (10% weight)
* Before calculation, we normalized these components using Min-Max Scaling to ensure fair comparison. This Composite\_Score became one of our most powerful predictors in the final model.



### Model Development

**4.1 Initial Approach**

Our model development process was iterative, starting with a range of different algorithms to establish a baseline and understand which approaches might be most effective for our data.

**a) Models Tested:**

* Linear Regression
* Random Forest
* Gradient Boosting
* XGBoost
* LightGBM

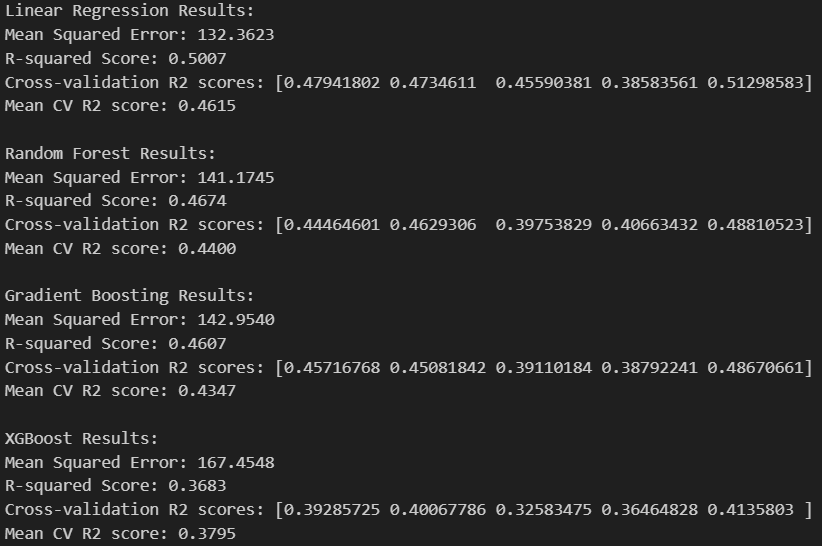
**b) Evaluation Metrics:**

We used several metrics to assess model performance:

* Mean Squared Error (MSE)
* R-squared (R²)
* Cross-validation scores

**c) Initial Results:**

* **Linear Regression:**
  + MSE: 132.3623
  + R-squared: 0.5007
  + Mean CV R² score: 0.4615
* **Random Forest:**
  + MSE: 141.1745
  + R-squared: 0.4674
  + Mean CV R² score: 0.4400
* **Gradient Boosting:**
  + MSE: 142.9540
  + R-squared: 0.4607
  + Mean CV R² score: 0.4347
* **XGBoost:**
  + MSE: 167.4548
  + R-squared: 0.3683
  + Mean CV R² score: 0.3795
* **LightGBM:**
  + MSE: 151.3590
  + R-squared: 0.4290
  + Mean CV R² score: 0.4257



**d) Error Analysis:**

We performed error analysis using residual plots to understand where our models were making mistakes and to identify any patterns in the errors.

**4.2 Feature Selection and Refinement**

* Based on our initial results and error analysis, we refined our approach:
* **Feature Selection**:
* We used the SelectKBest method with f\_regression to identify the most informative features. The top 10 selected features were:
  + Mode
  + Danceability
  + Energy
  + Speechiness
  + Loudness
  + Duration (ms)
  + Artist Followers
  + Popularity Score
  + Monthly Listeners
  + Composite\_Score
* **Feature Importance**:
* We analyzed feature importance, particularly from our Random Forest model. The top 5 important features were:
  + Genre\_Popularity (0.487866)
  + Acousticness (0.056478)
  + Duration (ms) (0.050475)
  + Tempo (0.038741)
  + Speechiness (0.036735)
* **Correlation with Target:**
* We examined the correlation of our features with the target variable (Track\_Popularity). The top correlations were:
  + Composite\_Score (0.764279)
  + Popularity Score (0.601128)
  + Artist Followers (0.076445)
  + Duration (ms) (0.070862)
  + Monthly Listeners (0.055888)

**4.3 Final Model:**

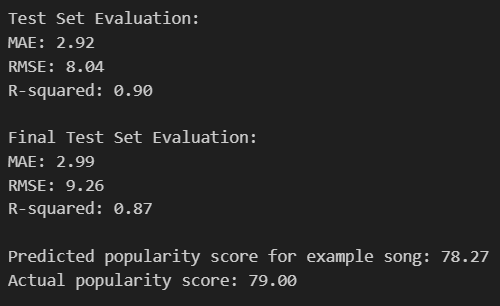
* Ensemble Approach Based on our analysis and the performance of individual models, we decided to create an ensemble model to leverage the strengths of different algorithms:
* **Model Architecture:**
* We used a VotingRegressor, combining:
  + Linear Regression
  + Ridge Regression
  + Random Forest Regressor
* **Rationale**:
  + This ensemble approach allows us to capture both linear and non-linear relationships in the data, potentially leading to more robust predictions across different types of songs and artists.
* **Implementation:**
  + We used scikit-learn's VotingRegressor with equal weights for each model. The features were scaled using StandardScaler before being fed into the model.

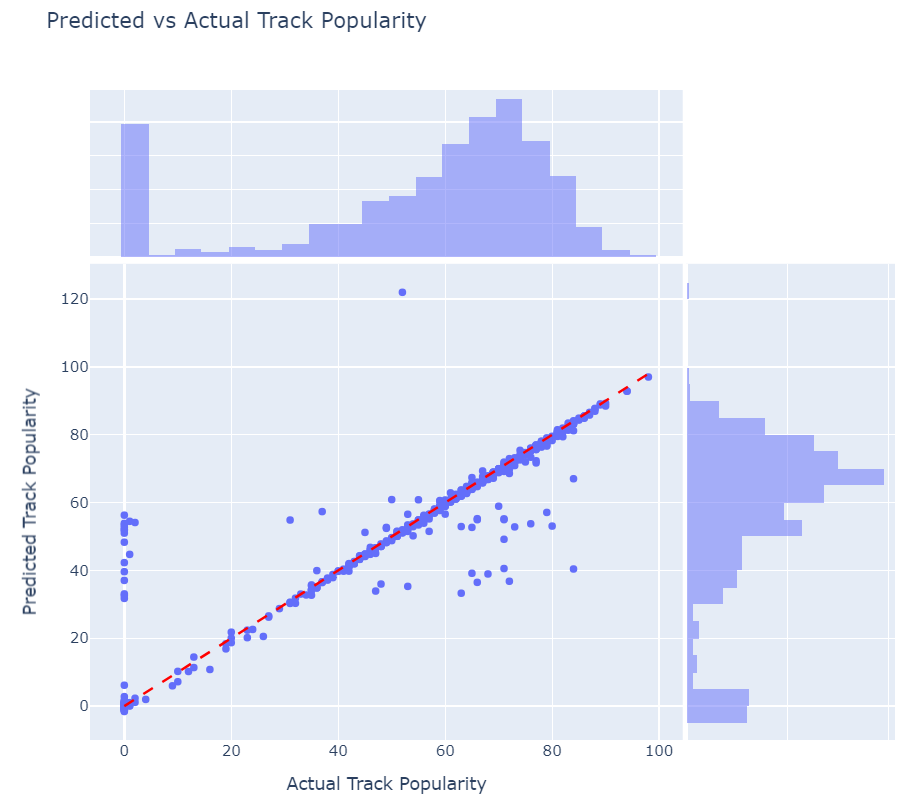
### Model Performance and Evaluation

**5.1 Final Model Results**

Our ensemble model showed significant improvement over the individual models:

* **Test Set Evaluation:**
  + Mean Absolute Error (MAE): 2.92
  + Root Mean Squared Error (RMSE): 8.04
  + R-squared (R²): 0.90
* **Final Test Set Evaluation:**
* We set aside 665 songs as a final test set to evaluate our model's performance on completely unseen data:
  + Mean Absolute Error (MAE): 2.99
  + Root Mean Squared Error (RMSE): 9.26
  + R-squared (R²): 0.87
* **Cross-validation Results:**
* We performed 5-fold cross-validation to ensure our model's performance was consistent:
  + Mean CV R² score: 0.8902
  + CV R² scores: [0.89087408, 0.89284707, 0.90066786, 0.91076638, 0.85560926]





**5.2 Model Interpretation**

* **R-squared Value:**
  + The high R-squared values (0.90 on the test set and 0.87 on the final test set) indicate that our model explains a large portion of the variance in track popularity. This suggests that the features we've selected and engineered are indeed strong predictors of song success.
* **Error Metrics:**
  + The low MAE values (2.92 and 2.99) suggest that, on average, our predictions are within about 3 points of the actual popularity scores on a 0-100 scale. This level of accuracy could be very useful for artists and producers in estimating a song's potential success.
* **Consistency:**
  + The similarity between our test set and final test set results, as well as the consistent cross-validation scores, indicates that our model generalizes well to unseen data and is not overfitting.

**5.3 Feature Importance in Final Model**

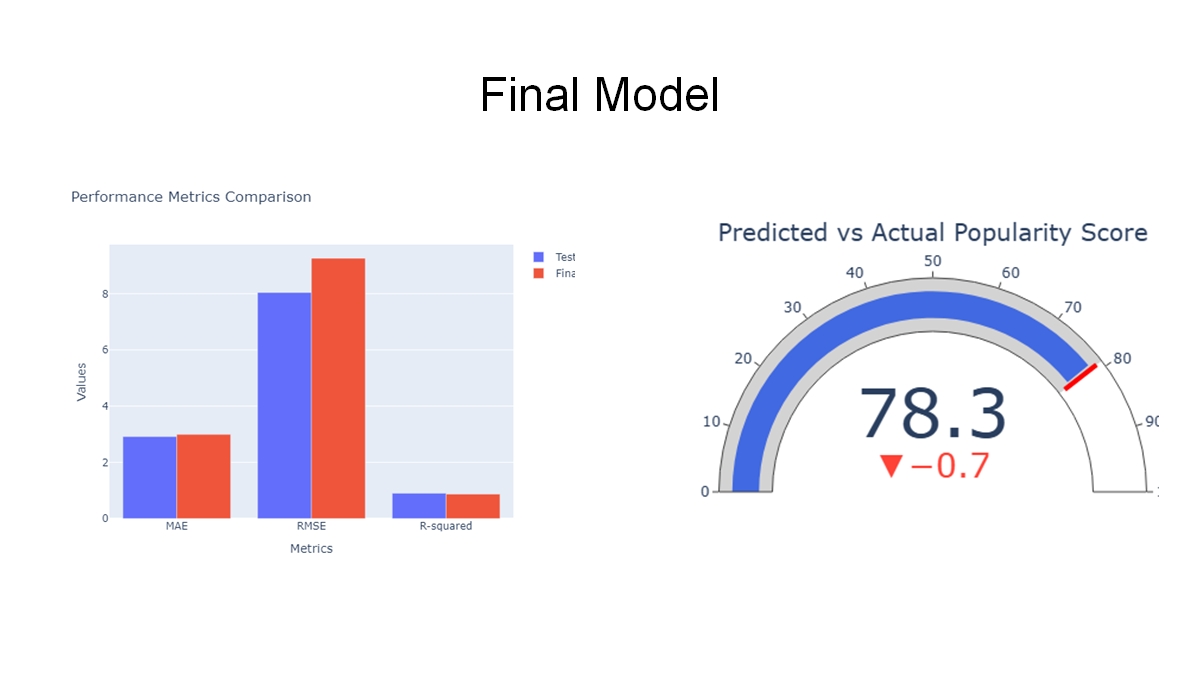
While the ensemble model doesn't provide direct feature importances, we can infer the most influential features based on our earlier analysis and the model's performance:

* Composite\_Score: This engineered feature, combining multiple popularity indicators, proved to be the strongest predictor of track popularity.
* Popularity Score: The artist's overall popularity is a significant factor in predicting an individual track's success.
* Audio Features: While less influential than popularity metrics, features like Danceability, Energy, and Acousticness still contribute to the model's predictions.
* Artist Metrics: Artist Followers and Monthly Listeners provide important context for a track's potential popularity.
* Track Metadata: Features like Duration and Release Month offer additional predictive power.

**5.4 Example Prediction To illustrate the model's performance**

We can look at an example prediction:

* Predicted popularity score: 78.27
* Actual popularity score: 79.00
* This example demonstrates the model's capability to make accurate predictions, with the predicted score being very close to the actual popularity.



## 

## References

* [*Spotify for developers*](https://developer.spotify.com/documentation/web-api)
* [*Genius API*](https://docs.genius.com/)
* [*Scikit-learn - Machine Learning in Python*](https://scikit-learn.org/1.5/index.html)
* [*Matplotlib*](https://matplotlib.org/)
* [*NLTK (Natural Language Toolkit)*](https://www.nltk.org/)
* [*Pandas: Data manipulation and analysis library*](https://pandas.pydata.org/)
* [*NumPy: Fundamental package for scientific computing with Python*](https://numpy.org/)
* [*Seaborn: Statistical data visualization*](https://seaborn.pydata.org/)
* [*XGBoost: Optimized distributed gradient boosting library*](https://xgboost.readthedocs.io/)
* [*LightGBM: A fast, distributed, high performance gradient boosting framework*](https://lightgbm.readthedocs.io/)
* [*TensorFlow: An open-source machine learning framework*](https://www.tensorflow.org/)
* [*Keras: Deep learning API running on top of TensorFlow*](https://keras.io/)
* [*Hugging Face Transformers: State-of-the-art Natural Language Processing*](https://huggingface.co/transformers/)
* [*Gensim: Topic modeling library*](https://radimrehurek.com/gensim/)
* [*Beautiful Soup: Library for web scraping purposes*](https://www.crummy.com/software/BeautifulSoup/)