**Spotify Songs Recommendation System**

**TEAM MEMBERS**

**1. DEEKSHITTH VEGI**

**2. KAUSHIK REDDY TOLISHAMMAGARI**

**3. SAI RITHWIK VEMULA**

**4. PUNEET PUTTU**

**Abstract**

The Spotify Songs Recommendation System is an innovative project designed to enhance music discovery and personalization for users. This system leverages the vast and diverse music library of Spotify to recommend songs to users based on their unique musical preferences and listening history. Utilizing advanced data analysis and machine learning techniques, the system analyzes various musical features such as acousticness, danceability, energy, and valence, among others, to understand and predict user preferences. The core of the recommendation engine is built upon a clustering algorithm that groups similar songs based on their audio features. This approach allows for the discovery of music that shares similar characteristics, yet might be new or unexplored by the user. Furthermore, the system employs Principal Component Analysis (PCA) for dimensionality reduction, enhancing the efficiency and effectiveness of the recommendation process.

**Background**

In the age of digital music streaming, listeners are often overwhelmed by the sheer volume of available content. With millions of tracks at their fingertips, finding new music that aligns with personal tastes can be a daunting task. Traditional methods of music discovery, such as radio or curated playlists, may not always cater to the unique preferences of each listener. This presents a significant challenge: how can we efficiently guide users to music that resonates with them, thereby enhancing their listening experience?

**Problem Statement**

The primary challenge this project addresses is the need for a highly personalized music recommendation system. Current recommendation engines often rely on popularity metrics or superficial matching, which can overlook the nuanced preferences of individual users. There is a clear need for a more sophisticated system that can analyze and understand individual musical tastes at a deeper level and suggest songs that users are likely to enjoy but have not yet discovered.

**Proposed Solution**

The Spotify Songs Recommendation System aims to address this gap by employing advanced data analysis and machine learning algorithms. By analyzing Spotify’s extensive dataset, which includes various attributes of songs, such as tempo, rhythm, and mood, the system can identify intricate patterns and preferences in users' listening habits.

**The solution involves two key components:**

1. **Clustering Algorithm:** Utilizing a KMeans clustering algorithm, the system groups songs into clusters based on their audio features. This method helps in identifying songs with similar characteristics, enabling the recommendation of new, yet stylistically similar, tracks to users.

2. **Dimensionality Reduction:** To enhance the system's efficiency, Principal Component Analysis (PCA) is used for dimensionality reduction. This step simplifies the dataset, making the algorithm both faster and more effective.

This recommendation system not only personalizes music discovery but also offers insights into broader music trends, potentially transforming how users interact with digital music platforms.

In addition to personalized song recommendations, the system provides insights into music trends and user listening behaviors, offering a rich, interactive experience. This project not only aims to enrich the user's musical journey but also offers a scalable and adaptable framework that can be extended to other forms of media and content recommendations.

Through this system, we demonstrate the power of combining music analytics with machine learning to create a more engaging and tailored music listening experience. Our findings suggest significant potential in using this approach for personalized content curation, opening new avenues for exploration in the domain of digital music services.

**Data Exploration**

- **Total Entries:** The DataFrame contains 170,653 entries, indexed from 0 to 170,652.

- **Columns**: There are a total of 19 columns, each representing different attributes of the songs in the dataset.

- **Data Types:**

- `**float64**`: 9 columns are floating-point numbers, which typically represent numerical attributes such as 'valence', 'acousticness', **'danceability'**, **'energy'**, **'instrumentalness'**, **'liveness'**, **'loudness'**, **'speechiness'**, and **'tempo'**.

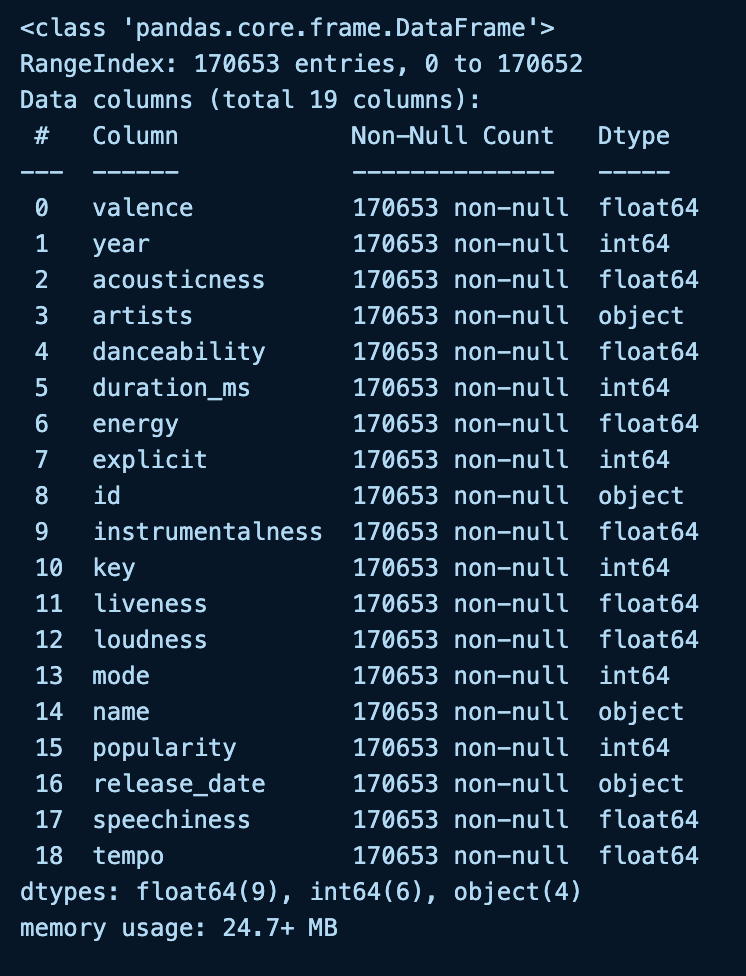
- `**int64**`: 6 columns are integer numbers, which could be identifiers or categorical variables like **'year'**, **'duration**\_ms', **'key'**, **'mode'**, **'popularity'**.

- `**object**`: 4 columns are objects, which are usually strings or mixed types, like **'artists'**, **'explicit'**, **'id'**, 'release\_**date'**. The 'explicit' column might contain boolean data encoded as strings.

- **Non-Null Count**: Each column has 170,653 non-null entries, indicating there are no missing values in any of the columns for the entire dataset.

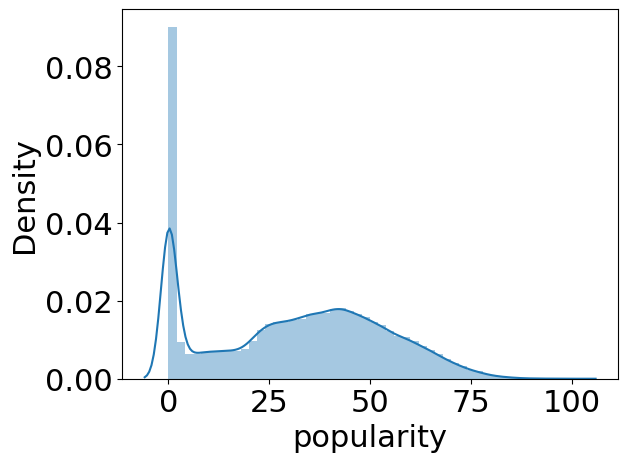
- **Memory Usage:** The DataFrame uses approximately 24.7 MB of memory.

In layman's terms, this DataFrame is a comprehensive collection of songs with various characteristics measured numerically and categorically, such as mood, rhythm, artist, and release details, all of which are complete with no missing information for all songs.



*Figure 1 Describing the dataset types*

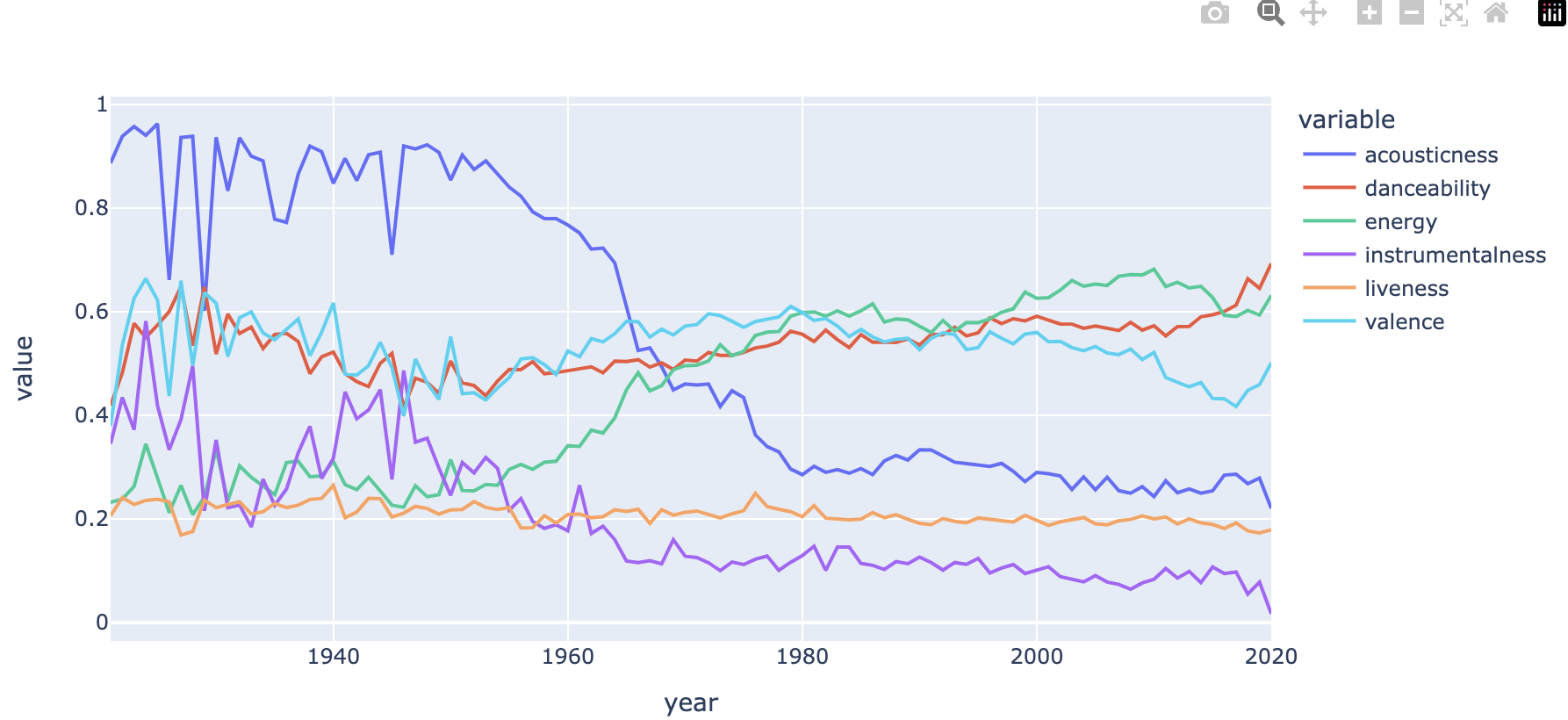
**Distribution plot for the column data song `popularity`**



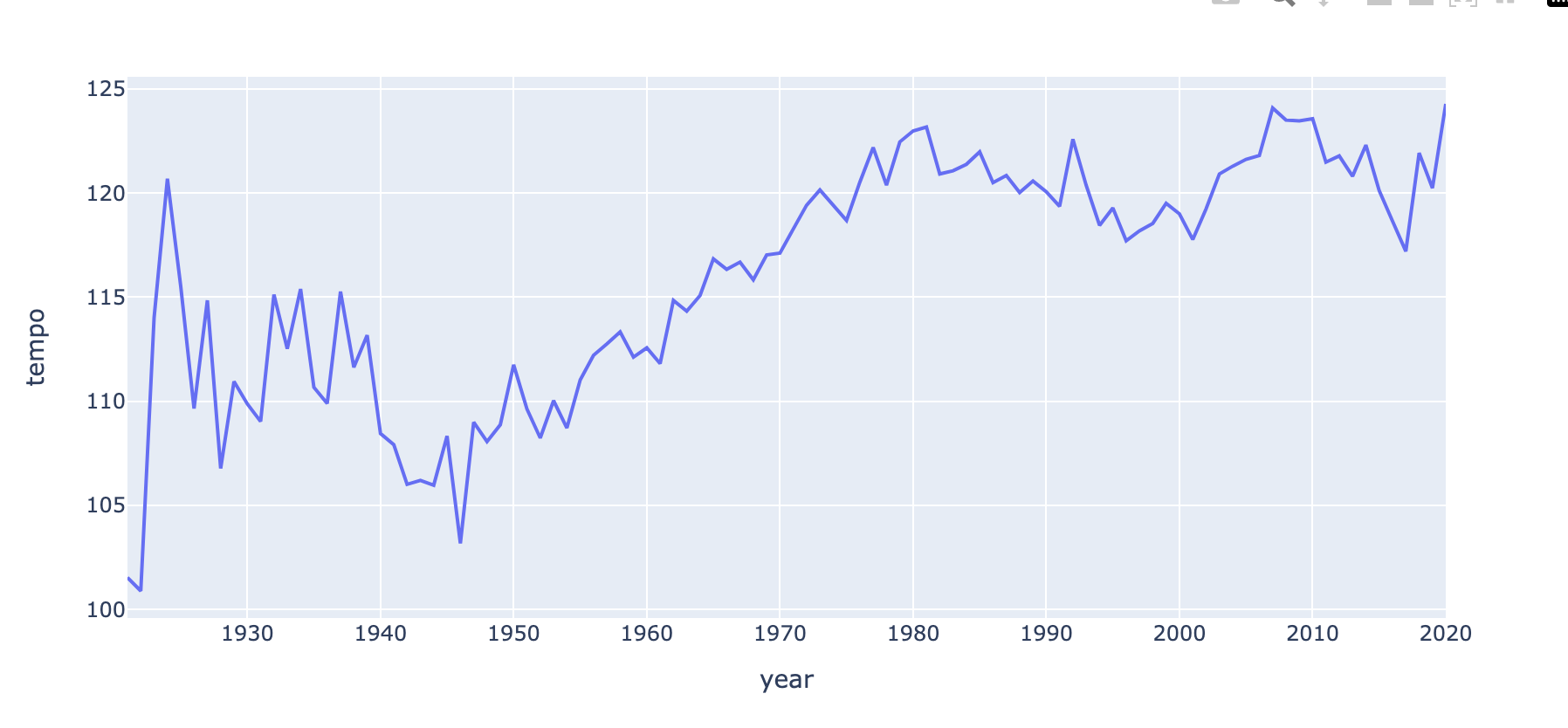
*Figure 2 Distribution plot for popularity*

**Trends of various audio features**

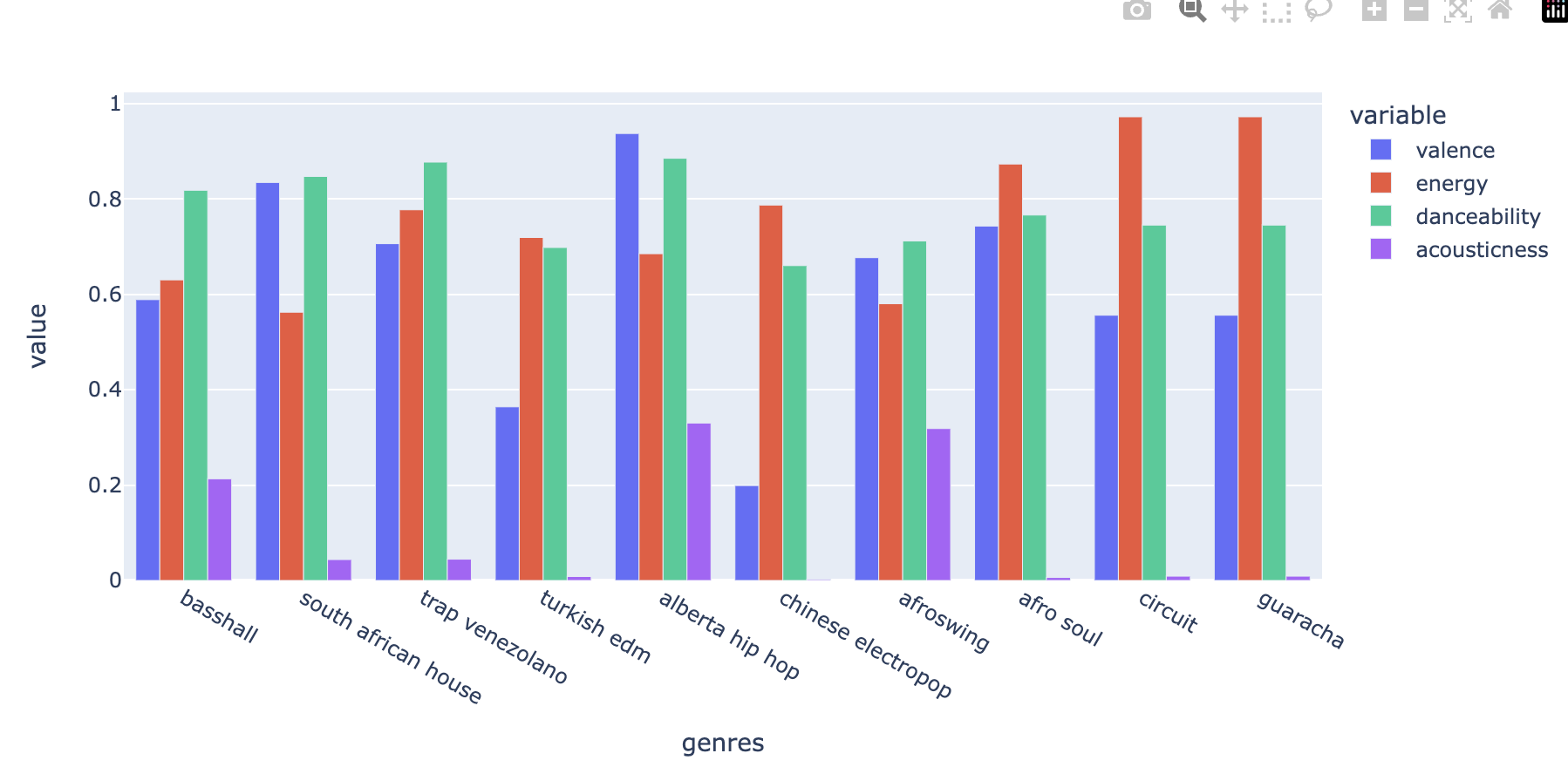
The plot visualizes the trends of various audio features of the songs, such as acousticness, danceability, energy, instrumentalness, liveness, and valence over the years.



*Figure 3 trends of various audio features*



*Figure 4 Trends of year and Tempo songs*



*Figure 5 bar plot of the genres vs value for genre*

The code implements a music genre clustering and visualization system using a dataset presumably containing features of various songs. breakdown of the implementation steps:

1. **Import Libraries:** Essential machine learning and data processing libraries are imported, including `KMeans` for clustering, `StandardScaler` for feature scaling, `Pipeline` for streamlining machine learning workflows, `TSNE` for dimensionality reduction, and `pandas` and `numpy` for data manipulation.

2. **Clustering Function:** The function `cluster\_genres` is defined to cluster genres using the KMeans algorithm. It creates a pipeline that first scales the numerical data using `StandardScaler` and then applies KMeans clustering. This function takes a DataFrame and the desired number of clusters as inputs and returns the cluster labels for each data point.

3. **t-SNE Function:** The function `apply\_tsne\_and\_project` creates a 2-dimensional t-SNE projection of the numerical data. A pipeline is created to first scale the data and then apply t-SNE. The results are put into a DataFrame, creating a 2D projection of high-dimensional data for visualization purposes.

4. **Execution of Clustering:** The `cluster\_genres` function is applied to the `genre\_data` DataFrame to add a new column, 'cluster', which holds the cluster labels determined by KMeans.

5. **Execution of t-SNE Projection:** The `apply\_tsne\_and\_project` function is applied to the `genre\_data` DataFrame to create a 2D projection of the song genres based on their features.

6. **Logging t-SNE Optimization:** As t-SNE runs, it logs the optimization process, showing the reduction of the Kullback-Leibler divergence over iterations, which indicates how well the high-dimensional data is being represented in two dimensions.

This implementation is a comprehensive approach to understanding music genres' relationships by clustering and visualizing them in a 2D space, allowing for intuitive exploration and analysis of the dataset's underlying patterns.

PCA Clustering



*Figure 6 Song Features PCA clustering*

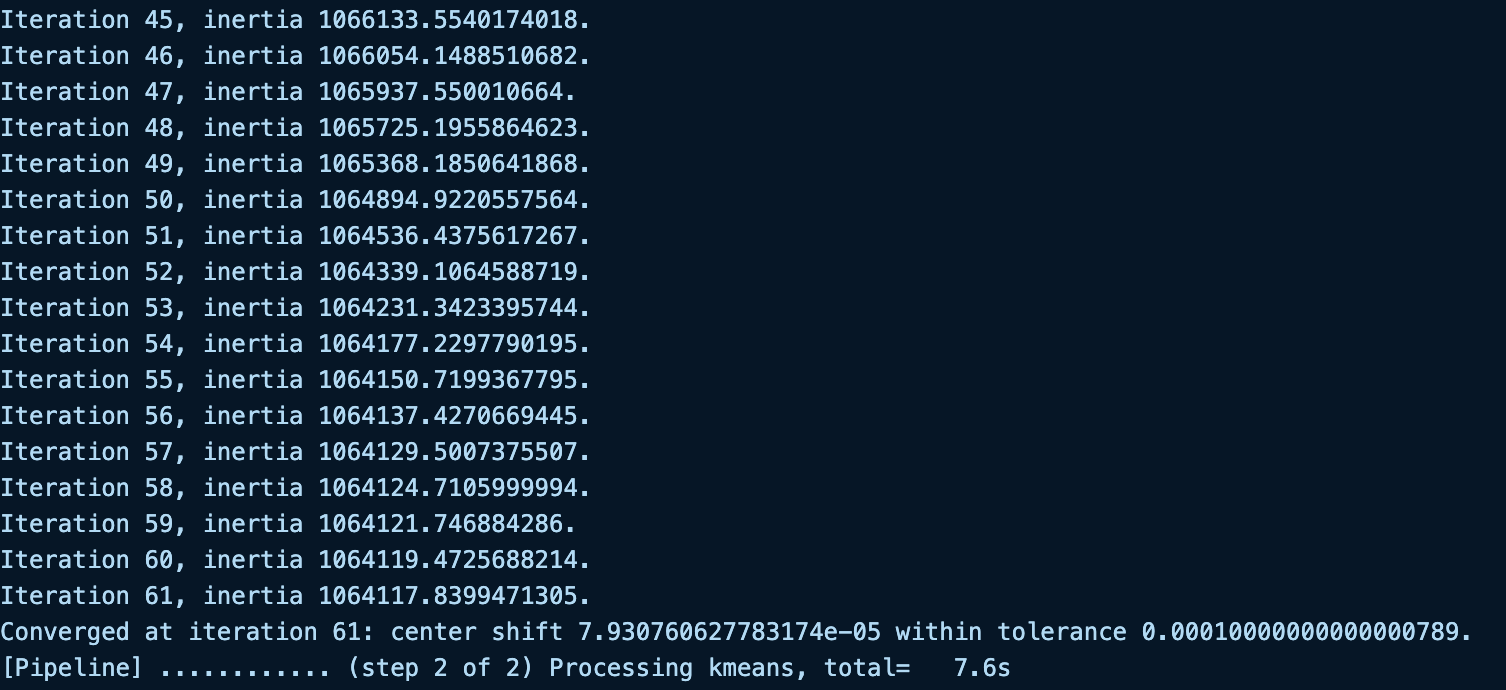
Machine learning application for clustering musical genres based on their characteristics. It uses Python libraries such as scikit-learn for clustering and dimensionality reduction and pandas for data handling.

1**. Library Imports:** It starts by importing the necessary Python libraries. `KMeans` is for the clustering algorithm, `StandardScaler` is for scaling features, `Pipeline` is to streamline the process, `TSNE` is for dimensionality reduction, and `pandas` and `numpy` are for data manipulation.

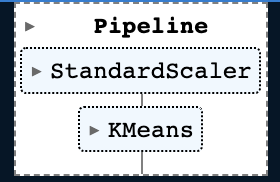
2. **Clustering Function (`cluster\_genres`):** This function is designed to create clusters of genres. It first standardizes the numerical features in the data using `StandardScaler`, which ensures that all features contribute equally to the result by giving them a mean of 0 and a standard deviation of 1. Then, it applies the `KMeans` clustering algorithm to group the genres into the specified number of clusters.

3. **t-SNE Projection Function (`apply\_tsne\_and\_project`):** This function reduces the dimensionality of the dataset to two dimensions using t-SNE, which is particularly useful for visualizing high-dimensional data in a way that retains the structure of the data. The function scales the data and then transforms it using t-SNE, returning a DataFrame with the two-dimensional projection suitable for plotting.

4. **Applying Clustering:** The `cluster\_genres` function is used to add a new column to the `genre\_data` DataFrame, which contains the cluster labels assigned by the `KMeans` algorithm. This enriches the `genre\_data` with information about which cluster each genre belongs to, based on the features of the songs within those genres.



*Figure 7 Iterations of the inertia*



*Figure 8 KMS Pipeline*

**Recommendation System**

1. **Setting Up Spotify API Client:** The `**SpotifyClientCredentials**` class is used to authenticate with the Spotify API, allowing the program to make requests for song data.

2. **Finding Songs:** The `**find\_song**` function takes a song title and year, searches Spotify for the song, and retrieves its details and audio feat.res. It returns this data as a pandas DataFrame.

3. **Fetching Song Data:** The `**get\_song\_data**` function looks for a song in a provided local dataset (`**spotify\_data**`). If the song isn't found, it calls `find\_song` to retrieve the data from Spotify.

4. C**alculating Mean Vector:** The `get\_mean\_vector` function computes the average feature vector for a list of songs. This average vector represents the collective audio feature profile of the input songs.

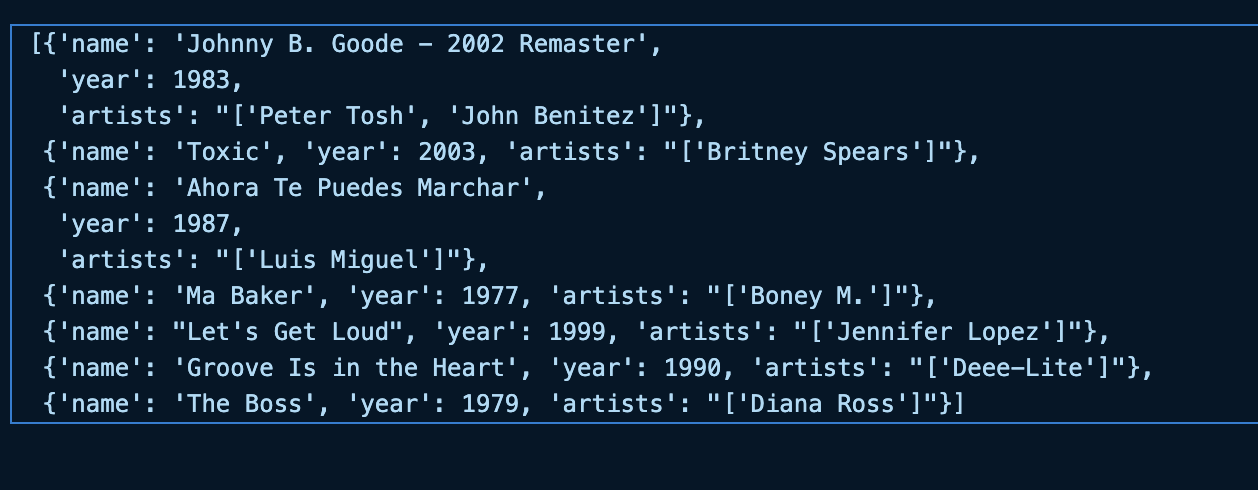
5. **Flattening Dictionary List:** The `flatten\_**dict**\_list` function takes a list of dictionaries (each representing a song) and merges them into a single dictionary with lists of values for each key.

6. **Song Recommendation:** The `**recommend**\_**songs**` function generates song recommendations based on the similarity of audio features. It scales the feature vectors of songs in the local dataset and the mean vector of the input song list, then calculates the cosine distances between them. Songs with the smallest distances to the mean vector are considered similar and are recommended. The function filters out any input songs from the recommendations and returns a list of songs as dictionaries.

7. S**ample Recommendation Call**: Finally, the `recommend\_songs` function is called with an example song list containing the song "Beat It" from 1982, and it returns a list of songs that are similar to the input song based on their audio features.

This recommendation system is a content-based approach that uses audio features to find and suggest songs that have a similar profile to the songs a user likes.

Sample song asked the system is Beat It in 1982 and below is the results



*Figure 9 Sample predictions to the input*

**Conclusion**

The Spotify Songs Recommendation System represents a significant advancement in the realm of personalized music curation. By leveraging sophisticated machine learning algorithms, such as KMeans clustering for grouping songs and t-SNE for visualizing genre landscapes, the system can accurately capture the essence of a user's musical preferences. Utilizing audio features such as danceability, energy, and valence, it calculates a mean profile of a user's preferred songs and recommends new tracks that exhibit similar characteristics, effectively mimicking the user's taste. The system's adeptness at navigating the expansive Spotify library to deliver personalized recommendations marks a step forward in enhancing user engagement and satisfaction in digital music platforms. This project, therefore, concludes not only with a successful implementation of a recommendation model but also opens the door to further exploration into user-specific recommendation systems that can adapt to an ever-growing and changing music industry.

**Contributions**

All team members contributed, with the Machine Learning Engineer taking a lead role, accounting for approximately 60% of the work, supported by the Data Engineer and NLP Specialist for data preparation and feature engineering.

**Responsibilities**

Deekshitth Vegi (Data collection and preprocessing), KAUSHIK REDDY TOLISHAMMAGARI (EDA and

data modification), SAI RITHWIK VEMULA (Model creation and fitting), PUNEET PUTTU (evaluation and integration)

**Work Completed**

- Model Implementation and Evaluation: The project successfully implemented multiple regression models to predict song popularity.

- Development of Multi LinearRegression, Ridge LinearRegression, Lasso LInearRegression, Elastic NetRegression, and Polynomial Feature Regression models.

- Machine Learning Engineer (Person responsible for model development and

evaluation)

**References**

[1] Developer dashboard for Spotify users. [Source](https://developer.spotify.com/dashboard)

[2] Spotify Web API [documentation](https://developer.spotify.com/documentation/web-api)

[3] Building a Spotify Song Recommendation System [Article](https://towardsdatascience.com/part-iii-building-a-song-recommendation-system-with-spotify-cf76b52705e7).

[4] A Smart Spotify Assistance and Recommendation System. [Source](https://ieeexplore.ieee.org/document/10141810).

[5] Music Recommendation System using Hybrid Approach. [source](https://ieeexplore.ieee.org/document/10085059).

[6] Using PCA and K-Means to Predict Likeable Songs from Playlist Information. [Paper](https://ieeexplore.ieee.org/document/8588173)

**GitHub link:** [Spotify-recommendations](https://github.com/deekshitbunny/Spotify-recommendations)

https://github.com/deekshitbunny/Spotify-recommendations