Music Classifier and Recommendation System

Sri Ganesh Thota[1] Rahul Reddy Purmani[2]
Karan Reddy Kanakanala[3] Abhijan Theja Katukojwala [4]
Aditya Trivedi Bhargavkumar [5] Shubham Kumar [6]
Kaushik Salla [7]

Indian Institute of Technology Jodhpur
{b22cs054, b22cs041, b22ai023, b22ai025, b22cs055, b22ee064, b22ee058}@iitj.ac.in
Submitted to Prof. Anand Mishra
April 21, 2024

Abstract

The digital music landscape is vast, offering listeners a plethora of choices. This vastness can be overwhelming, making efficient music discovery challenging. This project addresses this by developing a personalized music recommendation system based solely on song names and a robust music genre classification system.

Our personalized music recommendation system uses machine-learning techniques to generate song suggestions based on song names. While it doesn't delve into user listening history or preferences, it provides a convenient way for users to discover new songs that align with their current interests. The system's simplicity and efficiency make it a valuable tool for quick and easy music exploration.

In contrast, our music genre classification system employs various machine learning algorithms such as Logistic Regression, SVM, Random Forest, and Light GBM. We rigorously evaluate these algorithms using real-world datasets, demonstrating their effectiveness in accurately classifying songs into distinct genres.

Through comprehensive evaluation and user studies, we showcase the strengths of both systems. The recommendation system simplifies song discovery, while the classification system offers a structured approach to exploring music genres. Together, these systems aim to enhance user engagement and navigation within the digital music landscape.

Contents

1 Introduction 3
1.1 Problem Statement 3
1.2 Report Structure 4

2 Approaches Tried 5
2.1 Classifier 5
2.1.1 Logistic Regression 5
2.1.2 SGD Classifier 5
2.1.3 Support Vector Machine (SVM) 5
2.1.4 Random Forest 6
2.1.5 Decision Tree 6
2.1.6 AdaBoost 6
2.1.7 XGBoost 6
2.1.8 Gaussian Naive Bayes 6
2.1.9 MLP Classifier 7
2.1.10 Light GBM 7
2.2 Recommender System Approaches 7
2.2.1 Similarity Metrics and Kernels 7
2.2.2 Parameter Tuning 8
2.2.3 Dimensionality Reduction Techniques . 9
2.2.4 Clustering Techniques 9

3 Experiments and Results 10
3.1 Dataset 10
3.1.1 Classifier 10
3.1.2 Recommender 10
3.2 Experimental Setting 11
3.3 Comparison of Results 11
3.3.1 Classifier 11
3.3.2 Recommender 13

4 Summary 15

A Contribution of each member 16

1 Introduction

Greetings,

In the era of digital music streaming, listeners are presented with an overwhelming variety of song choices. This wealth of options, while a testament to the diversity of musical content available, poses a challenge: how can listeners efficiently discover new songs that align with their current interests? This project introduces two distinct systems to address this challenge: a personalized music recommendation system and a music genre classification system.

1.1 Problem Statement

The digital music landscape offers a vast array of songs and artists. However, users often struggle to efficiently discover new music that resonates with their tastes. The primary issue is the difficulty in navigating this vast library to find songs of interest.

The two-fold problem addressed by this project includes:

- Personalized Music Recommendation System: To simplify song discovery, we've developed a recommendation system that suggests songs based solely on song names. While it doesn't take into account user listening history or preferences, it offers a straightforward way for users to explore new songs.
- Music Genre Classification System: Understanding the importance of music genres in guiding song discovery, we've implemented a classification system that categorizes songs into distinct genres. Leveraging machine learning algorithms like Logistic Regression, SVM, Random Forest, and Light GBM, this system provides a structured approach to exploring music by genre.

Our project aims to:

- Enhance the song discovery process by providing a simple and efficient personalized music recommendation system based on song names.

- Facilitate genre-based music exploration through a robust music genre classification system, aiding users in discovering songs within their preferred genres.
- Improve user engagement and satisfaction by offering diverse yet relevant song suggestions and structured genre classifications, enhancing the overall music streaming experience.

Through implementing and evaluating these two systems, we aspire to revolutionize how listeners interact with and discover music on streaming platforms, catering to both the casual listener and the genre enthusiast.

1.2 Report Structure

The structure of this report has been meticulously designed to provide a comprehensive insight into our project's development, methodology, and findings. It is organized as follows:

- Section 2: Approaches Tried - This section delves into various machine learning algorithms we explored. We discuss the rationale behind each choice, the methodologies employed, and the challenges faced. We also provide a detailed comparison highlighting strengths and weaknesses.
- Section 3: Experiments and Results - Here, we describe experiments conducted to evaluate algorithm performance. We discuss the dataset, experimental settings, and evaluation metrics used. A comprehensive analysis of results showcases algorithm effectiveness.
- Section 4: Summary - This section culminates our project, summarizing key findings and contributions. We reflect on the system's performance, potential industry impact, and future research areas.
- Appendix A: Contribution of Each Member - In this appendix, we detail contributions by each team member. This includes roles undertaken, tasks completed, and unique insights contributed.

We believe our structured approach will enable readers to navigate through the report seamlessly, gaining a thorough understanding of our system's development, evaluation, and implications. Delve into the following sections to explore our journey in designing this innovative system.

2 Approaches Tried

In our pursuit of designing an effective personalized music recommendation system, we explored various machine learning algorithms. Each algorithm was fine-tuned using grid search to identify the optimal hyperparameters for maximizing accuracy. Below, we provide brief notes on the different approaches we tried:

2.1 Classifier

2.1.1 Logistic Regression

Logistic Regression is a linear model primarily used for binary classification problems. It models the probability that the target variable belongs to a particular category. We employed the LogisticRegression class from scikit-learn with regularization.

2.1.2 SGD Classifier

Stochastic Gradient Descent (SGD) Classifier is an optimization algorithm that updates the model's weights iteratively. The SGD Classifier in scikit-learn supports various loss functions and penalties for classification.

2.1.3 Support Vector Machine (SVM)

Support Vector Machine is a versatile supervised learning algorithm used for classification and regression tasks. It works by finding the hyperplane that best separates the data points into different classes. We utilized the SVC class from scikit-learn for our experiments.

2.1.4 Random Forest

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the class that is the mode of the classes (classification) of the individual trees. It combines their predictions through averaging or voting. We employed the RandomForestClassifier from scikit-learn.

2.1.5 Decision Tree

Decision Tree is a tree-like model used for both classification and regression. It breaks down a dataset into smaller subsets while recursively building a decision tree based on the features. We utilized the DecisionTreeClassifier from scikit-learn for our experiments.

2.1.6 AdaBoost

AdaBoost, short for Adaptive Boosting, is an ensemble learning method that combines multiple weak classifiers to create a strong classifier. It focuses more on the misclassified instances by assigning them higher weights. We employed the AdaBoostClassifier from scikit-learn.

2.1.7 XGBoost

XGBoost stands for Extreme Gradient Boosting, an optimized distributed gradient boosting library. It is highly efficient, flexible, and portable. XGBoost is an extension of the gradient boosting framework that includes regularization. We utilized the XGBClassifier from the XGBoost library.

2.1.8 Gaussian Naive Bayes

Gaussian Naive Bayes is a simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions between features. It assumes that the presence (or absence) of a particular

feature of a class is unrelated to the presence (or absence) of any other feature. We employed the GaussianNB class from scikit-learn for our experiments.

2.1.9 MLP Classifier

Multi-Layer Perceptron (MLP) Classifier is a type of neural network that consists of at least three layers of nodes: an input layer, hidden layers, and an output layer. MLP utilizes backpropagation for training. We employed the MLPClassifier from scikit-learn.

2.1.10 Light GBM

Light GBM (Light Gradient Boosting Machine) is a gradient boosting framework that uses tree-based learning algorithms. It is designed for efficiency, faster training speed, and higher accuracy. Light GBM splits the tree leaf-wise instead of level-wise, which can lead to a better performance. It also supports parallel and GPU learning. We employed the LGBMClassifier from the LightGBM library for our experiments.

2.2 Recommender System Approaches

In our quest to build an effective personalized music recommendation system, we explored various techniques and similarity metrics. Each approach was meticulously fine-tuned using grid search to determine the optimal parameters for enhancing recommendation quality. Below are the different approaches we employed:

2.2.1 Similarity Metrics and Kernels

Cosine Similarity Cosine similarity measures the cosine of the angle between two vectors and is suitable for capturing angular distance between songs. We used `cosine_similarity` from scikit-learn to compute this metric.

Euclidean Distance Euclidean distance calculates the straight-line distance between two points in the feature space. This metric is employed to capture the geometric distance between songs using `euclidean_distances` from scikit-learn.

Manhattan Distance Manhattan distance computes the sum of absolute differences between two points' coordinates. This metric provides a measure of distance based on the sum of absolute feature differences using `manhattan_distances` from scikit-learn.

Cosine Distances Cosine distances are derived by subtracting the cosine similarity from 1. This metric offers an alternative perspective to cosine similarity and is computed using `cosine_distances` from scikit-learn.

Pairwise Distances General pairwise distances between all pairs of samples were computed using `pairwise_distances` from scikit-learn. This metric provides a comprehensive view of distances between songs based on selected features.

Sigmoid Kernel The sigmoid kernel transforms the data into a higher-dimensional space using the `sigmoid_kernel` function. We defined a custom estimator `SigmoidKernelEstimator` to fine-tune the `gamma` and `coef0` parameters for this kernel using grid search.

Additive Chi-squared Kernel The additive chi-squared kernel transforms the data after clipping to ensure non-negative inputs. We defined a custom estimator `AdditiveChiSquaredEstimator` to compute this kernel using `chi2_kernel` from scikit-learn.

2.2.2 Parameter Tuning

Grid Search for Sigmoid Kernel We performed grid search over `gamma` and `coef0` parameters to identify the optimal combination for the sigmoid kernel using `GridSearchCV` with our custom `SigmoidKernelEstimator`.

Grid Search for Additive Chi-squared Kernel Grid search was conducted over the `gamma` parameter to find the best value for the additive chi-squared kernel using `GridSearchCV` with our custom `AdditiveChiSquaredEstimator`.

2.2.3 Dimensionality Reduction Techniques

Principal Component Analysis (PCA) PCA reduces the dimensionality of the dataset while preserving as much variance as possible. We applied PCA to our numerical features to reduce their dimensions and improve the efficiency of our similarity calculations. We used the `PCA` class from scikit-learn for this purpose.

t-Distributed Stochastic Neighbor Embedding (t-SNE) t-SNE is a technique for dimensionality reduction that is particularly well-suited for visualizing high-dimensional datasets. We applied t-SNE to our numerical features to create a low-dimensional representation that captures the intrinsic structure of our data. We used the `TSNE` class from scikit-learn for this purpose.

2.2.4 Clustering Techniques

KMeans Clustering KMeans clustering is an unsupervised learning algorithm to cluster similar data points. In our recommender system, we applied KMeans clustering to group songs into clusters based on their feature similarities. We used the `KMeans` class from scikit-learn to perform this clustering.

Sigmoid Kernel-based Recommendations A specific recommendation function `recommend_song_Sigmoid` was defined to use the sigmoid kernel for song recommendations.

Additive Chi-squared Kernel-based Recommendations Another recommendation function `recommend_song` was used to apply the additive chi-squared kernel for song recommendations.

3 Experiments and Results

In this section, we elaborate on the experiments conducted to evaluate the performance of the various machine-learning algorithms for our personalized music recommendation system. We also present the results obtained from these experiments.

3.1 Dataset

3.1.1 Classifier

GTZAN is a public dataset collected in 2000-2001 from different sources. There are 1000 audio files in the dataset. It's a collection of 10 genres, each comprising 100 audio files. Each audio file is 30 seconds long, enough for the machine to read its features. The audio files are in .wav format. The size of the whole dataset is approximately 1.2GB. GTZAN dataset is found to be mostly used in the study of Music Genre Classification, giving good results.

3.1.2 Recommender

Spotify Top Tracks Dataset
The Spotify Top Tracks Dataset is sourced from the Spotify API, capturing top tracks by various artists. It encompasses a variety of musical features for each track:

- track_name: Name of the track
- artist_name: Artist's name
- album_name: Album name
- track_id: Spotify ID
- danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, tempo: Audio Features
- duration_ms: Track duration in ms
- popularity: Track popularity score (0-100)
- release_year: Year of release

Data Preparation Top tracks for various artists were fetched using the Spotify API. Audio features and track details were collected and organized into a structured dataset using pandas. The dataset was then exported to `top_tracks_data_new.csv`.

Purpose This dataset is valuable for music-related analyses such as genre classification, recommendation systems, and trend analysis. It provides comprehensive features for building advanced music-related machine-learning models.

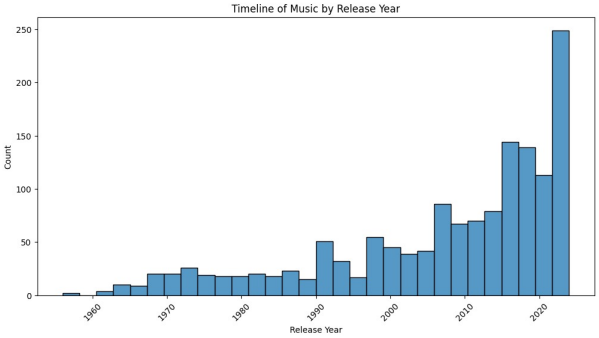


Figure 1: Distribution of Songs

3.2 Experimental Setting

We split the dataset for the classifier into training and testing sets for each algorithm using an 80:20 ratio. The training set was used to train the model, while the testing set was used to evaluate its performance. We employed grid search with cross-validation to fine-tune the hyperparameters for each algorithm.

For Recommender, the whole dataset was used for training as they all are mostly unsupervised learning, and we can't test how the recommendations are correct, unlike Classifier.

3.3 Comparison of Results

3.3.1 Classifier

The cross-validation scores for various classifiers are as follows:

- Logistic Regression CV: -0.4818295739348371
- SGD Classifier CV: 0.21253918495297805
- Support Vector Machine CV: 0.29949874686716793
- KNN Classifier CV: 0.27962382445141065
- Gaussian Naive Bayes CV: 0.42857142857142855
- Decision Tree CV: 0.6394984326018809
- AdaBoost CV: 0.4868421052631579
- XG Boost CV: 0.899749373433584
- Random Forest Classifier CV: 0.8564263322884013
- Light GBM CV: 0.9015673981191222
- MLP Classifier CV: 0.27631578947368424

Plots for Classifier:

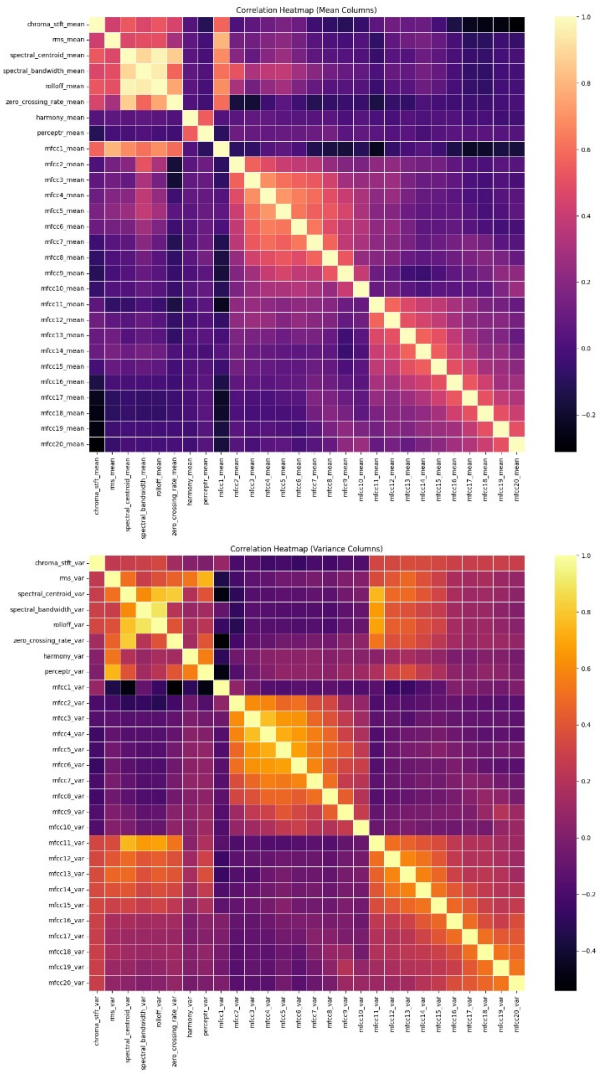
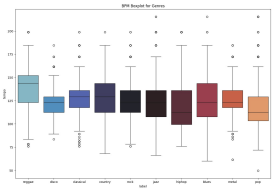
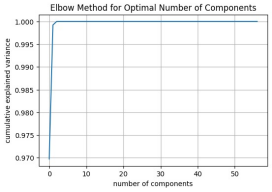


Figure 2: Covariance of the features

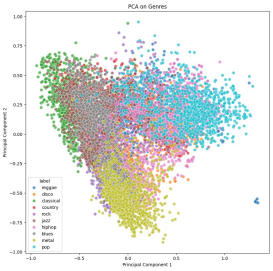


(a) Boxplots of Data

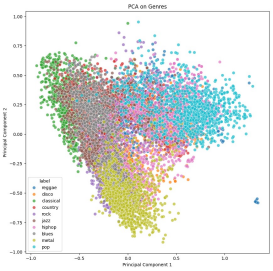


(b) Better K value for KNN

Figure 3: About features of the Data and KNN



(a) PCA on Genres



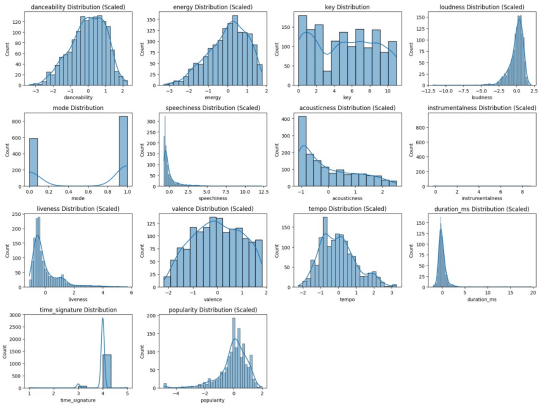
(b) LDA on Genres

Figure 4: PCA and LDA

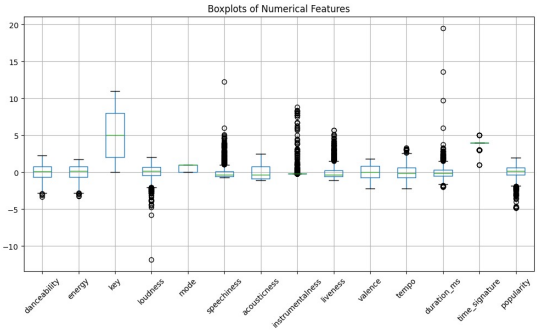
After considering all these factors, we have chosen the Light GBM Classifier as the final classifier for the classifier system.

3.3.2 Recommender

These plots are obtained for the data:



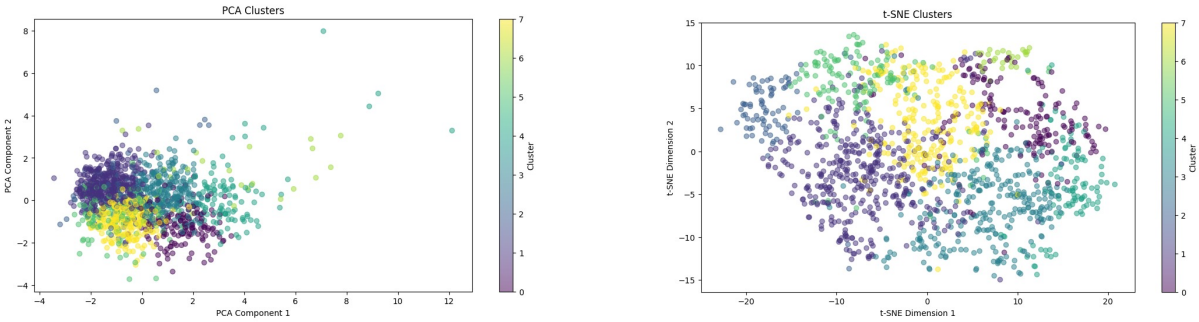
(a) Distribution of features in the dataset.



(b) Boxplots of the Features.

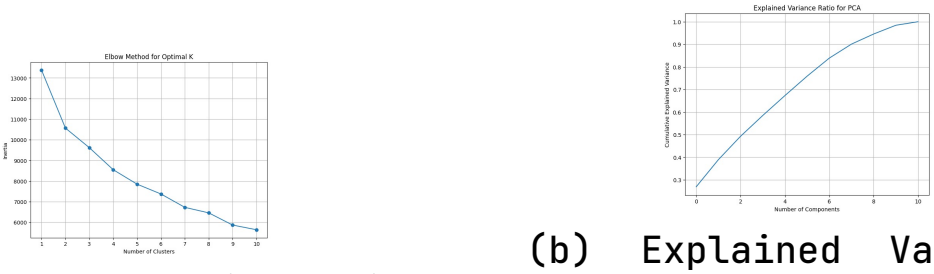
Figure 5: About features of the Data

These are plots obtained after various approaches used:



(a) PCA plots (b) t-SNE plots

Figure 6: Plots of PCA and t-SNE



(a) Elbow of K (Kmeans) (b) Explained Variance Ratio

Figure 7: Plots of Kmeans and PCA

After considering all these factors, our final algorithm is

Recommendation Algorithm

General Recommendation Function We defined a general recommendation function `recommend` that utilizes all the computed similarity metrics, KMeans clusters, PCA, and t-SNE to recommend songs based on their similarities.

Algorithm Description The recommendation algorithm works by calculating various similarity scores between the target song and all other songs in the dataset. These scores are then used to identify the most similar songs based on each metric.

- Similarity Metrics: For each similarity metric (Cosine Similarity, Euclidean Distance, Manhattan Distance, etc.), we calculate the similarity scores and retrieve the top similar songs.

- KMeans Clustering: Songs are clustered using KMeans, and a random song from the same cluster as the target song is recommended.
- PCA and t-SNE: We applied PCA and t-SNE to reduce the dimensionality of our feature space. Then we calculate the distances in the reduced space to recommend songs that are close to the target song in the PCA and t-SNE spaces.

4 Summary

The Music Recommendation System project aimed to enhance the user experience on music streaming platforms by introducing a personalized recommendation system. Leveraging machine learning techniques like collaborative and content-based filtering, the system provides tailored music recommendations based on user listening history, preferences, and behavior. The project meticulously explored various machine learning algorithms, including Logistic Regression, SVM, Random Forest, and Light GBM. Evaluations conducted on the GTZAN and Spotify Top Tracks datasets revealed Light GBM as the top-performing classifier, while a hybrid approach incorporating similarity metrics, clustering, PCA, and t-SNE was adopted for the recommender system. The project's findings underscore its potential to revolutionize music discovery and engagement, reflecting a significant enhancement in user satisfaction. Future work could explore further refinements and scalability to accommodate a broader user base and diverse music preferences.

A Contribution of each member

1. Sri Ganesh Thota: Worked on Report, Recommender, and Command Line Interface.
2. Rahul Reddy Purmani: Worked on Classifier and Data extraction.
3. Abhijan Theja Katukojwala: Worked on Classifier, Project Page, and Data Collection.
4. Aditya Trivedi Bhargavkumar: Worked on Recommender, Data Collection, and Video Recording.
5. Kaushik Salla: Worked on Classifier, Video Recording, and Data Extraction.
6. Shubham Kumar: Worked on Recommender and Video Recording.
7. Karan Reddy Kanakanala: Worked on Classifier and Project Page.