CSE/ECE 343: Machine Learning - Project Final Report
Music Recommendation System Using Machine Learning

Ashutosh Gera ashutosh21026@iiitd.ac.in
Aniket Kanojia aniket21377@iiitd.ac.in
Tushar Chandra tushar21211@iiitd.ac.in
Piyush Kumar piyush21180@iiitd.ac.in

Abstract

We all love listening to music, don't we? But no amount of music is ever enough for us. We always want to discover new artists, new music and new tones to satisfy our music cravings. The motivation behind selecting the music recommendation system as our project stems from its unique blend of ***practical utility and intellectual exploration****. While acknowledging its prevalence, we recognize the profound opportunities it presents for* ***harnessing our theoretical machine learning knowledge into a tangible application****. By crafting a system that aspires to match the quality of established platforms like YouTube and Spotify, we aim to decipher the intricacies of recommendation algorithms, diving deep into content-based filtering, and real-time learning. The recommendation algorithms that giants like Youtube and Spotify use are made to cater the masses, our aim is to make a personalised recommendation system for a user to enhance the music listening experience for everyone. We believe that this project is a tool that resonates with* ***genuine utility*** *and will give us a chance to apply all our machine learning knowledge to real-world use.*

1. Introduction

Our project revolves around the need to develop an efficient and accurate music recommendation system. Our aim is to develop a system that can intelligently analyze users' listening behavior and preferences to recommend music that they would potentially like. The existing solutions have many shortcomings such as:

- **Limited Diversity:** Existing recommendation systems tend to recommend popular or mainstream items, leading to a lack of diversity in recommendations. This hinders users from discovering new and niche content.
- **Popularity Bias:** Systems often recommend popular items to users, perpetuating a popularity bias.
- **User Feedback Loop:** Existing Systems struggle to incorporate user feedback effectively, leading to recommendations that do not align with user preferences over time.

To overcome these problems, our ultimate goal is to develop a completely unbiased music recommendation system that heavily focuses on the audio features of music and analyze user preferences to strike the right balance between providing personalized recommendations and introducing diversity.

2. Literature Survey

The literature review that follows is a summary of the studies that influenced and helped to shape our attempt to create a music recommendation system. These papers provide useful information on a variety of areas of music recommendation systems, such as data processing, feature extraction, user preference models, classification techniques, clustering techniques, and real-time adaptation.

1. A Music Recommendation System with a Dynamic K-means Clustering Algorithm suggests a method for personalized music recommendation services. They propose a **dynamic K-means clustering algorithm** which clusters the pieces in the music list dynamically adapting the number of clusters. They recommend pieces of music based on the clusters. The Existing recommendation systems analyze a user's preference by simply averaging the properties of music in the user's list. So those cannot recommend correctly if a user prefers several genres of music. By this proposed dynamic K-means clustering algorithm, they aimed to recommend pieces of music which are close to user's preference even though he likes several genres.

2. M. G. Galety, R. Thiagarajan, R. Sangeetha, L. K. B. Vignesh, S. Arun and R. Krishnamoorthy, "Personalized Music Recommendation model based on Machine Learning," 2022 8th International Conference on Smart Structures and Systems (ICSSS), Chennai, India, 2022, pp. 1-6, doi: 10.1109/ICSSS54381.2022.9782288.
used a library of songs to uncover connections across individuals and music so that a hit album might be offered to individuals derived from history. The main focus of their study is on the context-aware recommendation process's

insufficient integration of context data with the emergence of new attractions. They primarily focused on **Count Vectorizer (CV), and Cosine similarity (CS)** machine learning techniques delivering a complete end-to-end interface which, when a piece of given music, processes it and provide with the suggested tracks.

3. Varsha Verma, Ninad Marathe, Parth Sanghavi, Dr. Prashant Nitnaware , "Music Recommendation System Using Machine Learning ", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 7, Issue 6, pp.80-88, November-December-2021: They use a sample data set of songs to find correlations between users and songs so that a new song will be recommended to them based on their previous history. They have also used **Cosine similarity along with CountVectorizer.**

4. **D. Lin and S. Jayarathna, "Automated Playlist Generation from Personal Music Libraries," 2018 IEEE International Conference on Information Reuse and Integration (IRI), Salt Lake City, UT, USA, 2018, pp. 217-224, doi: 10.1109/IRI.2018.00039.** Given an arbitrary collection of music recordings, they aim to automatically sort songs with similar musical qualities into playlists. They utilize various clustering algorithms such as **K-Means, Affinity Propagation and DBSCAN.**

3. Dataset:

3.1 Dataset Preparation:

We are using a Spotify provided dataset which contains audio features of over **1, 204, 012 (1.2 million+) songs** obtained with the **Spotify API (spotipy)**.

3.2 Dataset Features:

The original dataset comprised of 24 columns (features). After performing data-preprocessing and eliminating non-numeric variables, variable with little change across the dataset and highly correlated variables, we have the below 15 features on which we have performed our analysis.

Feature	**Datatype**
explicit	int
danceability	float
energy	float
key	int
loudness	float
mode	int
speechiness	float
acousticness	float
instrumentalness	float
liveness	float
valence	float
temp	float
duration(in ms)	int
time_signature	float
year	int

Example-: {'name': 'Testify', 'album': 'The Battle Of Los Angeles', 'artists': "['Rage Against The Machine']", 'track_number': 1, 'disc_number': 1, 'explicit': False, 'danceability': 0.47, 'key': 7, 'loudness': -5.399, 'mode': 1, 'speechiness': 0.0727, 'acousticness': 0.0261, 'instrumentalness': 1.09e-05, 'liveness': 0.356, 'valence': 0.503, 'tempo': 117.906, 'duration_ms': 210133, 'time_signature': 4.0, 'year': 1999}

We observed the trend of the songs across the years and plotted the trend in various audio features across generations and observe that overall, songs increased in danceability and energy, while decreasing in acousticness. Also, use of minor keys became more common. There was also a reversal in the overall trend during the 80s. We also plotted a word cloud of

the most popular words in songs.

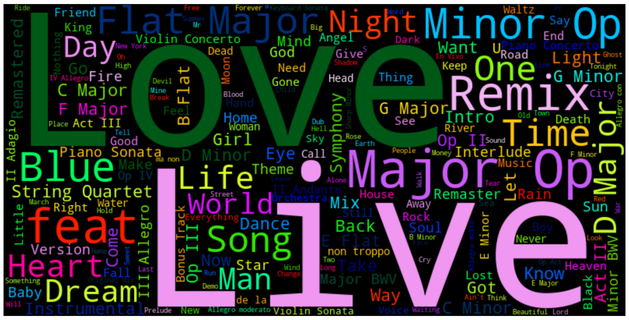


Figure 1: Word cloud of Song names

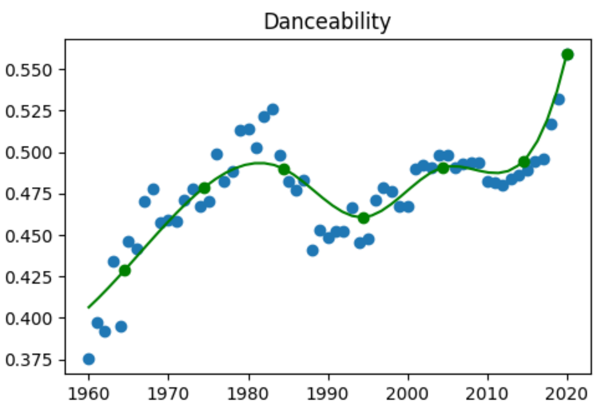
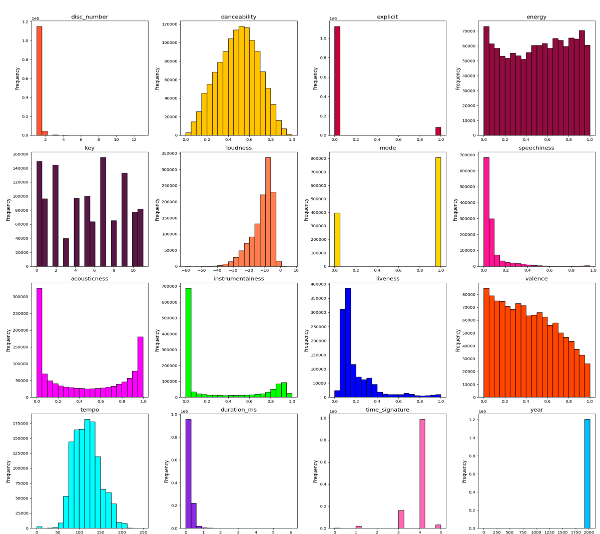


Figure 2: The Danceability of a song is increasing with time

3.3 Dataset Preprocessing and Feature Extraction

We have made minor adjustments dataset features, which were necessary for further analysis.



We began with 21 features in our dataset. To better understand their distributions, we visualized each feature using histograms. We observed that certain features, namely 'disc_number', 'explicit', and 'mode', exhibited minimal variation across the dataset. As these features only added to the complexity of our model without significant benefit, we opted to remove them. Furthermore, we converted boolean type variable into integer (0, 1) and eliminated non-numeric features such as 'artist_id', etc.

Further, we generated correlation heatmaps to analyze the interrelationships between features. We observe that acousticness is inversely related to loudness and energy. Danceability is directly correlated with valence

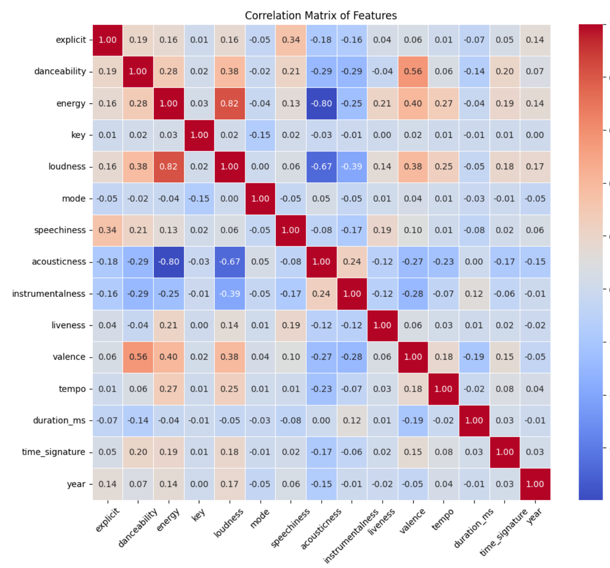


Figure 3: CORRELATION HEATMAP OF FINAL FEATURES

4. Methodology:

4.1 Filtering techniques

These are the most common techniques used in music recommendation systems

- **Collaborative filtering** recommends music based on the actions of comparable users.
- **Content-based filtering** offers suggestions based on the features of the music, such as

genre, tempo, and mood.

- **Hybrid filtering** combines both approaches to provide more precise recommendations.

Our project focuses on content based filtering using machine learning algorithms on various audio features available.

4.2 Models and Classification

We used the following machine learning algorithms throughout our project:

K Nearest Neighbours using Euclidean distance
Cosine similarity to find songs with similar features
K Means clustering algorithm
Gaussian mixture model clustering algorithm
DBSCAN clustering
Agglomerative clustering
Spectral clustering

4.2.1 Kmeans clustering

We employed the K-Means Clustering method, a prevalent unsupervised Machine Learning technique. K-Means operates on the principle of centroids or by gauging distances to determine the appropriate cluster for a point. It pinpoints 'k' centroids, and subsequently, every data point is grouped with the closest cluster, ensuring that the centroids remain minimal.

As unsupervised data doesn't contain labels, we used the k-means algorithm to assign labels to our data.To find the best value of k we used elbow method and silhouette analysis.

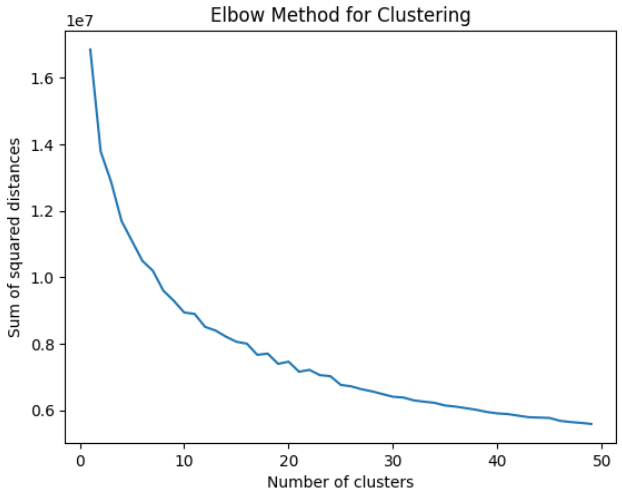


Figure 5: Elbow Methos for Optimal K

Using the elbow method, the appropriate number of clusters is roughly 20 for our dataset.

Upon analyzing with k=20, it provided a more realistic and better data interpretation for recommending songs. Our recommendation strategy will be based on cluster matching and song similarity, adopting a content-based approach.

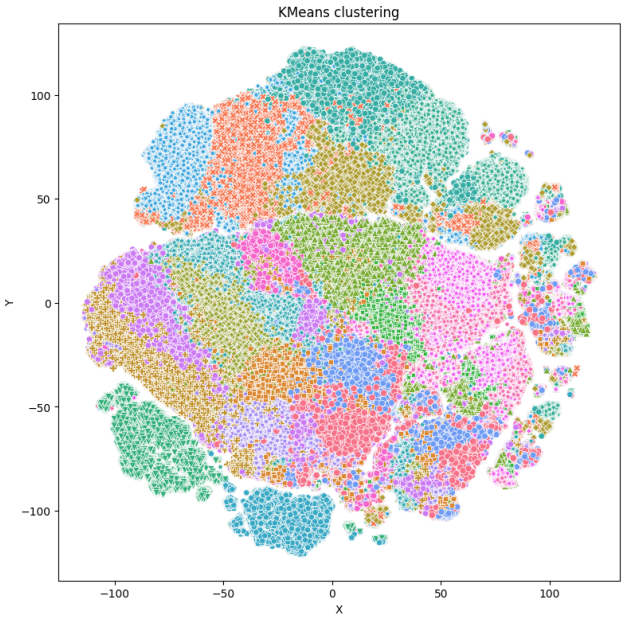


Figure 6: K MEANS CLUSTER REPRESENTATION USING TSNE

4.2.2 Gaussian Mixture Model

Gaussian mixture models are a probabilistic model for representing normally distributed subpopulations within an overall population. Mixture models generally don't require knowing which subpopulation a data point belongs to, allowing the model to learn the subpopulations automatically. Since subpopulation/label assignment is not known, this constitutes a form of unsupervised learning.

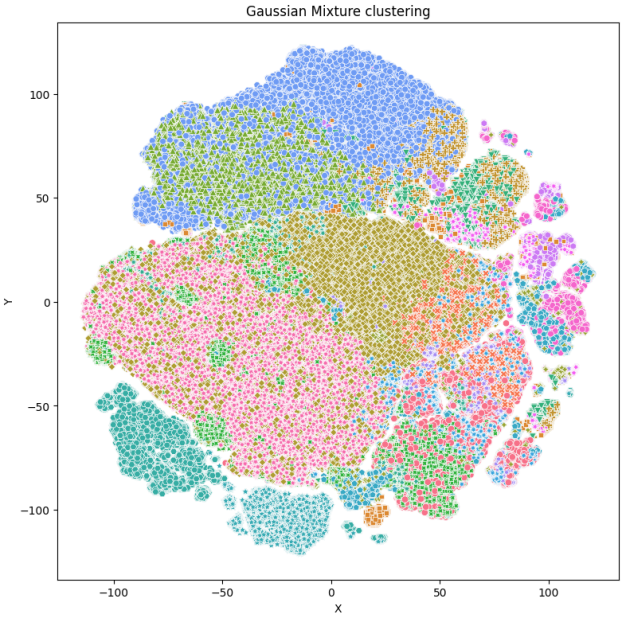


Figure 7: Gaussian Mixture Model cluster Representation

4.2.3 DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) models are used as unsupervised machine learning algorithms for clustering spatial data points. It identifies clusters based on density, defining clusters as areas where there are many data points close to each other while also detecting outliers as noise. DBSCAN is adept at handling irregularly shaped clusters and doesn't require the number of clusters to be predefined. We can observe the clusters formed in the graph given below. Total of 4 clusters are formed on the basis of density at specific values of epsilon and minimum samples. (Value of epsilon used here is 0.5 and the minimum samples is 15).

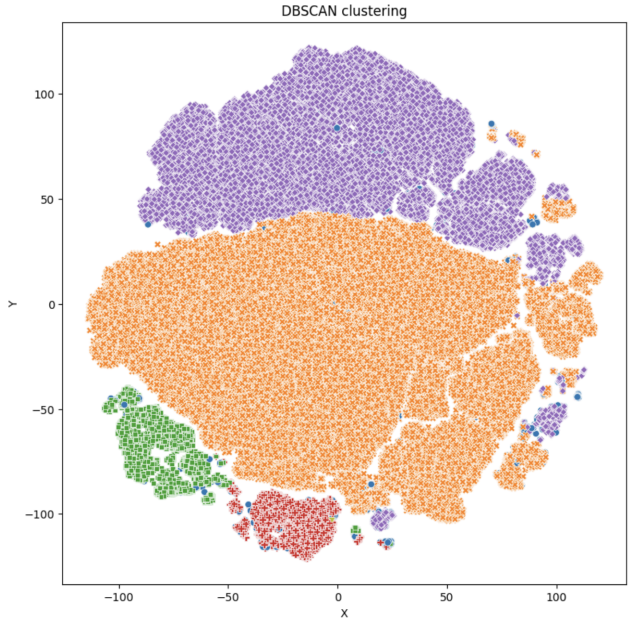
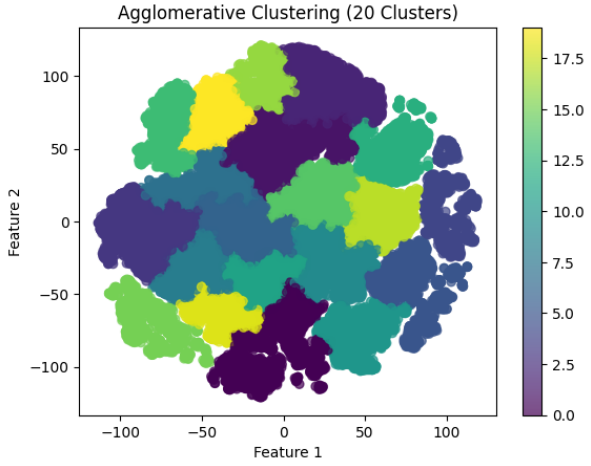


Figure 8: Density-Based Spatial Clustering of Applications with Noise cluster Representation

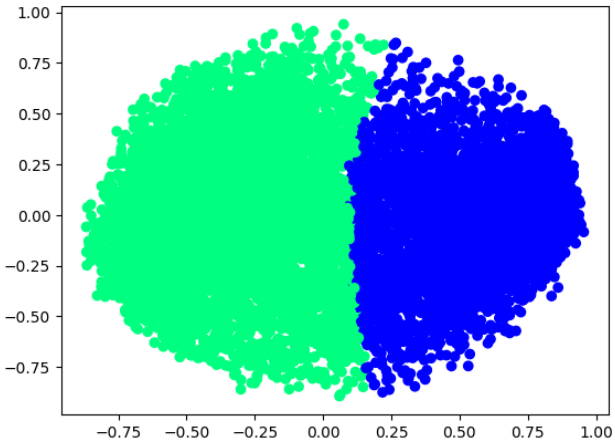
4.2.4 Agglomerative Clustering (AC)

Agglomerative clustering is a hierarchical clustering technique widely used in data analysis and machine learning for grouping similar data points into hierarchical structures. The algorithm begins by treating each data point as an individual cluster and iteratively merges clusters based on the similarity between them, ultimately forming a hierarchy or tree-like structure known as a dendrogram. Below is agglomerative clustering for N = 20.



4.2.5 Spectral Clustering (SC)

In SC, data points are treated as nodes of a graph. Thus, spectral clustering is a graph partitioning problem. The nodes are then mapped to a low-dimensional space that can be easily segregated to form clusters. No assumption is made about the shape/form of the clusters. The goal of spectral clustering is to cluster data that is connected but not necessarily compact or clustered within convex boundaries. SC is very computationally extensive so we performed it on a small dataset comprising of 10,000 random songs and got 2 clusters using nearest neighbours as affinity.



4.3 Playlist Generation using recommendation algorithms

4.3.1 Using Cluster Similarity:

Songs are represented in an N-dimensional space based on their features. By leveraging various clustering algorithms, songs with similar attributes are grouped together. For a given song, the system scans the cluster to which this song belongs and recommends songs from the same cluster.

4.3.2 Using Feature Similarity and Content-Based Filtering:

Here, each song is represented by a series of features, The similarity between the given song and other songs in the database is determined using the cosine similarity(CS) metric.

Formula Used: Similarity = (A.B) / (||A||.||B||)

5. Results and Analysis:

Input Song: Testify
Top 4
Output Songs using cosine similarity:

- Who I Am
- Trisha Please Come Come (Live)
- What Keeps You Up At Night
- Your Love - Radio EdiT

Output Songs using KMeans:

- Veillée spatiale
- The Ride
- Seekir
- Low Tide

Output Songs using GMM:

- El Camino de la Noche
- Minimal Slee
- Tombei The Mist
- Indigo Aerial

Output Songs using DBSCAN:

- If You Could Hold Your Woman
- Osaka
- Veillée spatiale
- This Side of Paradise

Output Songs using AC:

- Calm Like a Bomb
- Born as Ghosts
- Voice of the Voiceless
- New Millenium Homes

6. Conclusion:

We have implemented various unsupervised machine learning algorithms to recommend songs based on a given input songs. There is a partial overlap between the recommended songs across all the algorithms we have used, given the same input, which justifies the fact that our algorithms are recommending the songs which have highest feature similarity with the input songs.
All the references that we have used are mentioned in section 2 (Literature Survey) of this document.