Music Recommendation System Based on Acoustic Features

Haowei Huang, Qi Ren
College of Engineering
Northeastern University
Toronto, ON
huang.haow@northeastern.edu
ren.qi1@northeastern.edu

Abstract

Recommendation systems are designed to help users discover new content they may like but have never consumed before. In this report, we introduce a new music recommendation system that employs multiple approaches to recommend music based on user input. We described the design of the system, including the methods used to recommend music and the metrics to evaluate the performance. We also conducted experiments using different parameter configurations, and compared the results with the existing music recommendation system.

1 Introduction

1.1 Background

Music recommendation is essential for music streaming services as it helps retain and attract customers. Personalized recommendations allow users to discover new music they enjoy, increasing their likelihood of continued use. In a crowded market, this differentiates a service and helps it stand out. Overall, music recommendation is valuable for music streaming services and for helping individuals discover new music.

1.2 Project summary

This project is aiming to build a content-based music recommendation system using a music genre dataset retrieved from Spotify. One simple idea is to recommend new music with the same genre tag as the one users listened to before. However, this approach might limit the diversity of the recommendations over time, as it narrows the range of selections.

To overcome this drawback, we decided to build a recommendation system based on the acoustic characteristics, which might be similar across different genres of music. By considering a variety of acoustic features, our system will be able to provide personalized and diverse recommendations for users.

In addition, we also created serval metrics to evaluate the effectiveness of the system, including the similarity between user input and our recommendations, the diversity of recommendations, and a comparison of our recommendations with those of Spotify.

1.3 Related methods

In general, there are three kinds of recommendation system:

1. Popularity model：It always recommends to users the most popular content they have never consumed before. Though this kind of model usually provides good recommendations thanks to the wisdom of the public, it is not personalized.
2. Content-based filtering model：The content-based filtering model is what we implemented in this report. It leverages the internal characteristics of previous user input to recommend similar yet still new content. In this report specifically, we used the acoustic features of the music tracks

that users listened to before to make recommendations.

3. Collaborative filtering model：The collaborative filtering model is built on the multiple users' active participation. And the system makes recommendations based on the shared tastes of users.

2 Problem description

The data set we used was originally for music genre classification task. It contains general information about the music tracks like artist, track name and genre, as well as acoustic features like loudness, danceability and key that was measured by Spotify. However, we decided to leverage this dataset to build a music recommender system. To reach our goal, there are two main questions that need to be answered:

1. How to recommend new music tracks based on user input?
2. How to evaluate the effectiveness of our recommendations?

3 Data preparation and data analysis

3.1 Dataset basic features

Table 1: The basic features of Spotify Music datasets.

Data Set Characteristics	Attribute Characteristics	Associated Tasks	Number of Instances	Number of Attributes
Multivariate	Real and Categorical	Classification	50000	18

3.2 Data cleaning and imputation

The dataset contains multiple non-numerical features such as 'key', representing the overall pitch of a music track with values like C# and G, which need to be converted to numerical values for advance processing. To solve the problem, we referred to online resources to gain music domain knowledge. Based on musical scale, we successfully map all the 'key' into integer range from 0 to 11, where higher value stands for higher pitch.

Also, as there are some missing values in 'tempo', we utilized KNN imputer to solve the problem. By comparing the non-missing fields, the imputer can fill the missing field using mean value from nearest neighbors in the dataset.

In addition to handling categorical features and missing values, we also standardize all features as clustering algorithms, which were used to build the system, will be influenced by magnitude greatly.

3.3 Exploratory data analysis

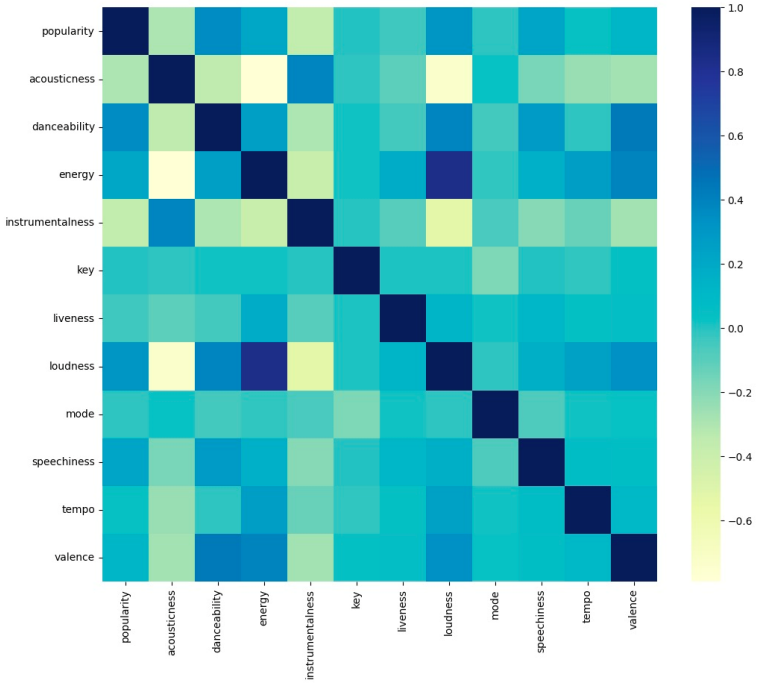

Figure 1: Heatmap of correlation coefficients of features

By looking at the heatmap, we can see that many features are highly related, suggesting that we may apply ICA to achieve maximal independence of the data. We also noticed that energy has a remarkably high positive correlation with loudness, which confirms our usual perceptions. Yet acousticness and instrumentalness showed a negative correlation with loudness, since acoustic music is accompanied by acoustic instruments and without electrical amplification, therefore quieter.

4 Methodology

Here is a flow chart of how we designed the overall process.

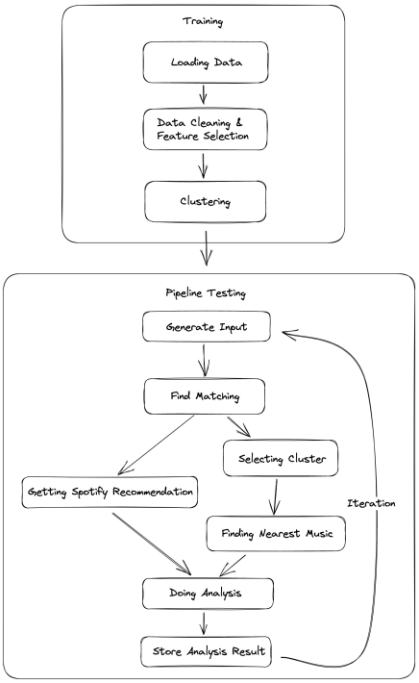

Figure 2: Flow chart of the system design process

4.1 Clustering

First, we used the dataset to train several clustering models based on KMeans and GMM. This step is to cluster the data using their acoustic features while neglecting the genre labels. Later, we can input new data and find its belonging clustering to narrow the range of recommendations. Though in this step we also limited the range of selections, it was done by analyzing the underlying physical features of the tracks instead of human-labeled tags. Due to this reason, the number of clusters will be smaller than the number of genres because music of different genres might share some similarities.

In this step, we also tried to apply PCA and ICA on the original data to reduce feature dimensions and maximize their independence. In later stages, we compared the performance of different models that were built on original data, PCA processed data and ICA processed data.

4.2 Finding nearest music

Next, we calculated the mean Euclidean distance from the centroids of the input music tracks to the centroids of the clusters. We also tried to calculate the distance by farthest and nearest, but did not achieve better results, so they won't be discussed here.

After finding the nearest cluster, we used different algorithms to find the "closest" music. We chose several common distance calculation methods, cosine distance, Euclidean distance and Manhattan distance. We calculated the corresponding distances from the centroids of the input to all points within the clusters, ranked them and selected the music tracks with the closest distance as the recommendation result.

4.3 Making recommendations

In the process above we mentioned that the input goes through two steps, finding the corresponding clusters and finding the nearest music in the corresponding clusters. If we simply look for music tracks with similar acoustic characteristics, then we run into the problem that **we can only ever get the same results if the music tracks we input fall within a "spatial range"**. So, we first imposed clustering, hoping to overcome this problem.

4.4 Evaluation of performance

Finally, after getting recommendations from the system, we need to evaluate its effectiveness. There are three metrics that we used:

1. Genre similarity

$$genre\ similarity = \frac{num\ of\ matched\ genre}{len(tracklist)}$$

2. Acoustic feature similarity

$$feature\ similarity = \frac{1}{1 + dis\tan c\ e}$$

3. Recommendation diversity

$$diversity = \frac{sum(pairwise\ dis\tan c\ e)}{len(dis\tan c\ e)}$$

In the above equations, the distance is calculated by the Euclidean or cosine distance between the features of music. (Data applied with PCA and ICA, also they are recommended based on the new features, their evaluation is based on their original acoustic features, not the features applied with those methods. We would think using original features for evaluation makes more sense because they are natural characteristics of music.)

5 Experiments

5.1 Clustering

First, we used KMeans and GMM to cluster the original data, PCA processed data (with components number = 2) and ICA processed data. We plotted the elbow curve and the silhouette score curve to find the best k value.

Table 2: Clustering experiment result

	KMeans	GMM	KMeans_PCA	GMM_PCA	KMeans_ICA	GMM_ICA
Best k	2	3	3	2	5	2
Silhouette score	0.3	0.06	0.4	0.4	0.115	0.09

The above table shows that using PCA processed data for both KMeans and GMM clustering can improve the Silhouette score, while applying ICA will have a negative effect for KMeans clustering.

5.2 Making recommendations with different configurations

After clustering, first we input a list of tracks' names and the corresponding artist's name. Next, in case input tracks are not in our dataset, **we created a function that can retrieve the acoustic features of the music tracks from our dataset as well as from Spotify**. This was done by leveraging the Spotify python library called "spotipy", through which we can get all relevant data about specific tracks.

Then, we calculated the mean Euclidean distance between the input and centroids of the clusters, and selected new music with the closest distance as recommendations. In the experiment, we used different clusters mentioned, different distance methods to make recommendations. And finally, we evaluated the results using the metrics mentioned in section 4.4.

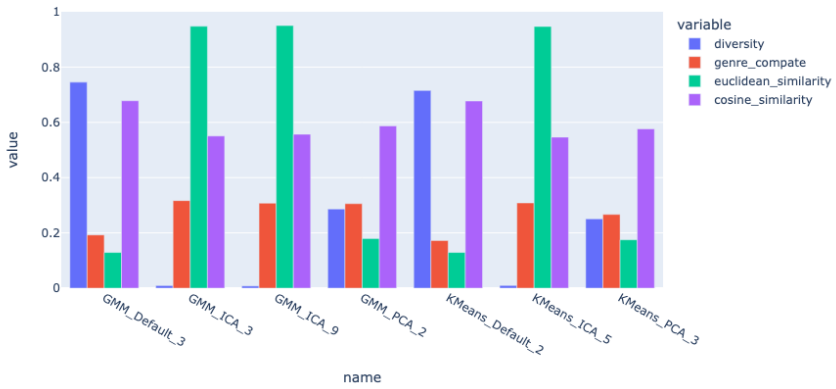

Figure 3: Performance chart of using different clusters for recommendation

First, it is noticeable that using original data to build the system will make recommendations more diverse. However, the feature similarity based on Euclidean and genre similarity is not satisfactory. Second, we can see that ICA processed data can produce recommendations with the highest Euclidean feature similarity, but it has the lowest diversity, which suggests that the recommendations are remarkably similar both among the output itself but also between input and output. **This seems to indicate that when features are independent of each other, we are more inclined to find music that is similar in terms of acoustic (numeric) characteristics**, and it is reasonable that the increase in similarity brings a decrease in diversity. Lastly, the PCA processed data does not achieve the same effect as it was in the clustering stage, as all performance metrics except cosine similarity was low.

This is because we perform clustering only to delineate a general range and avoid always getting the same result when the input falls in the same spatial range. The final search for recommended songs is still calculating the distance within the cluster, which makes the effect of clustering on the final recommendation relatively small. Instead, feature occupies a correspondingly more important role, so when it comes to the impact of PCA and ICA on the recommendation results, we may look more at their impact on features themselves analytically, rather than on the effect of clustering.

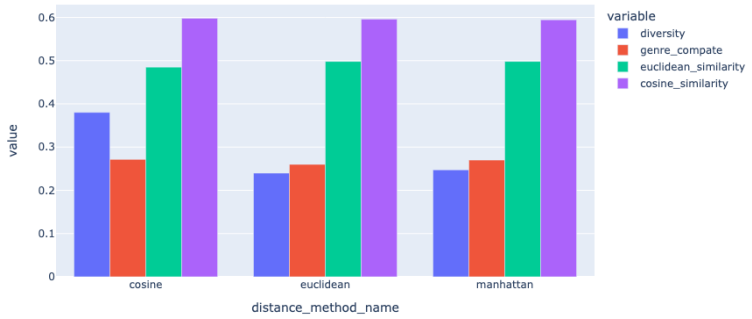

Figure 4: Performance chart of using different distance methods for recommendation

Also, from figure 4, we can see that there is not much difference in the model's performance when we use different distance methods.

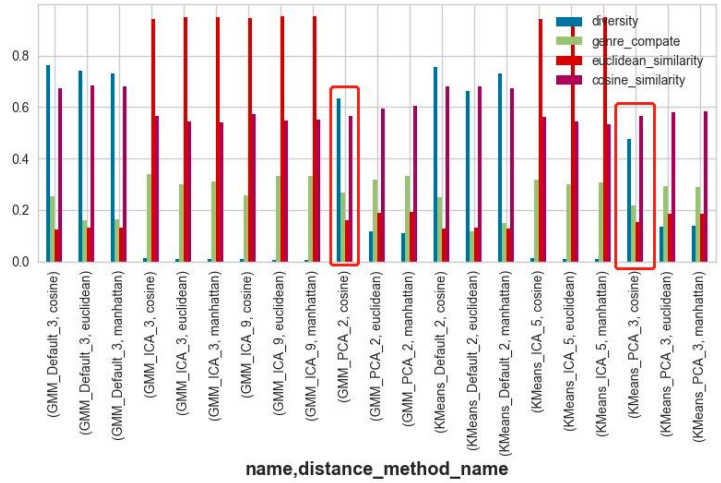

Figure 5: Performance chart of using different clusters and distance methods for recommendation

From the above figure, it also shows that using cosine as distance function to find nearest music within the cluster can greatly improve the diversity for recommendations made by PCA data (Compared to the right two group, the blue bar we circled was much higher). That's because PCA reduced the dimension, and cosine is measuring distance only regardless of scale or "distance", so when it go back to the original dataset, it can be really far away because the dimensions go back.

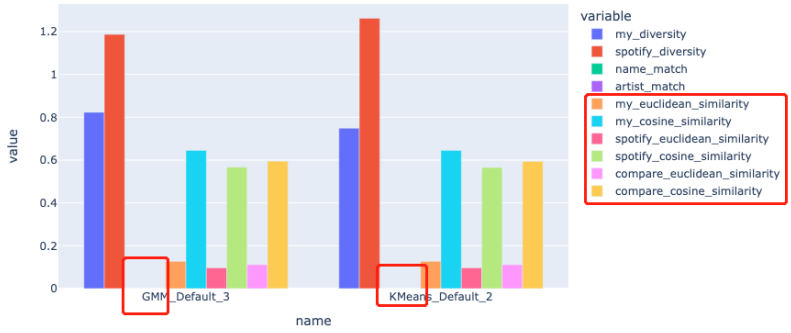

Figure 6: Performance comparison between Spotify recommendations and our recommendations

Lastly, we also retrieved recommendations from Spotify by using its python library spotipy. We used the same metrics to evaluate Spotify's recommendations. **It shows that the diversity of Spotify's recommendations is always higher, it may be because they have much more data of music.** In addition, we tried to match the tracks' name and artist between our output and that of Spotify's, which were marked by name_match and artist_match, but they are both zero. It is quite natural to get this result as there are many similar music tracks made by different artists (though they may share the same producer) and have different names. The last six bars measure the feature similarity between input vs our recommendations, input vs Spotify recommendations, and our recommendations vs Spotify recommendations. It shows that all the recommendations similarities are high when it is measured by cosine. This result indicates the effectiveness of our recommendation system.

7 Conclusion and future works

In this report, we present a new music recommendation system that uses multiple approaches to suggest music to users based on their input. We provide details on the design of the system, including

the techniques used for recommendation and the metrics used to evaluate its performance. We also conducted experiments with different parameter settings and compared the recommendations from our system and from Spotify.

Our recommendation system has several advantages. First, it can make recommendations for input music tracks that aren't in our dataset. This was made possible because we created a function to obtain acoustic features of the input by using Spotify's python library. Second, our comparison with the existing Spotify recommendation system has shown that our recommendations are diverse, and the similarity is close to Spotify's recommendations. Third, our recommendations are content based, so they are not restricted to a particular genre.

There are several ways to improve our system. One is to acquire more training data, which can be done by retrieving data from Spotify. Another is to look for additional evaluation metrics, such as user feedback. We can also explore other clustering algorithms such as DBSCAN and agglomerative clustering. Finally, we can adapt new mythology to enable more flexible adjustment of the existing algorithm. We can apply different weights to different steps, and new elements such as popularity and genre can be added. This will allow the algorithm to provide more personalized and accurate recommendations.

Acknowledgments

The learning code is adapted from: https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre and references therein.
A few distance calculating and data processing algorithms are from:
https://www.kaggle.com/code/vatsalmavani/music-recommendation-system-using-spotify-dataset/notebook
System, pipeline design & implementation: All by ourselves.

References

[1] Mavani, V. (2021, December 17). Music recommendation system using Spotify Dataset. Kaggle. Retrieved December 7, 2022, from https://www.kaggle.com/code/vatsalmavani/music-recommendation-system-using-spotify-dataset
[2] yuxuanliu0626. (2022, January 22). Music genre - clustering in KMEANS and GMM. Kaggle. Retrieved December 7, 2022, from https://www.kaggle.com/code/yuxuanliu0626/music-genre-clustering-in-kmeans-and-gmm/notebook
[3] Moreira, G. (2019, December 8). Recommender Systems in python 101. Kaggle. Retrieved December 7, 2022, from https://www.kaggle.com/code/gspmoreira/recommender-systems-in-python-101
[4] Ashrith. (2019, March 22). What makes a song likeable? Medium. Retrieved December 7, 2022, from https://towardsdatascience.com/what-makes-a-song-likeable-dbfdb7abe404
[5] Niyazov, A., Mikhailova, E., & Egorova, O. (2021). Content-based music recommendation system - IEEE Xplore. IEEE Xplore. Retrieved December 8, 2022, from https://ieeexplore.ieee.org/document/9435533/
[6] Patel, A., & Wadhvani, R. (2018). A Comparative Study of Music Recommendation Systems. IEEE Xplore. Retrieved December 8, 2022, from https://ieeexplore.ieee.org/Xplore/home.jsp
[7] Soleymani, M., Aljanaki, A., Wiering, F., & C. Veltkamp, R. (2015). Content-based music recommendation using underlying music preference ... IEEE Xplore. Retrieved December 8, 2022, from https://ieeexplore.ieee.org/document/7177504
[8] Schedl, M., Zamani, H., Chen, C.-W., Deldjoo, Y., & Elahi, M. (2018, April 5). Current challenges and visions in Music Recommender Systems Research - International Journal of Multimedia Information Retrieval. SpringerLink. Retrieved December 7, 2022, from https://link.springer.com/article/10.1007/s13735-018-0154-2