Music Recommender System

Krithika Ragothaman
Computer Science Engineering
PES University
Bengaluru, India
krithika.ragoth@gmail.com

Anchal Sharma
Computer Science Engineering
PES University
Bengaluru, India
anchalsharma31@gmail.com

Ankita.V
Computer Science Engineering
PES University
Bengaluru, India
ankita.v.2001@gmail.com

***Abstract*—Music Recommender systems, which are a subset of MIR, have become increasingly relevant as well as required today. In this paper a content-based music recommender is built using the concepts of K-Means Clustering, Principal Component Analysis and Cosine Similarity. Illustrations of the process flow and implementation flow for the proposed system have been included. The Spotify 160k dataset is obtained from Kaggle. The proposed model was first used to cluster the data twice. Once the user enters the name of a song, the cluster and sub-cluster to which the song belongs to is found, followed by the 10 most similar songs in that sub cluster. These 10 songs are provided as output to the user. While the proposed model is slower than the baseline models, on subjective evaluation it provides better recommendations.**

***Keywords*- Recommender system, Content-based, Unsupervised Machine Learning, Within Cluster Sum of Squares, K Means Clustering, Cosine Similarity**

I. Introduction

The internet has become more popular than ever in the last few decades. The percentage of the world population on the internet has grown from 5% in 2000 to 62% as of 2020. With the rise of this capacity, there has also been a boom of content on the internet. The music industry has profited off this boom, with the digital music services being at the forefront. With more than 50 million songs available on Spotify, and more than 40,000 added daily, music recommender systems have never been more relevant.
A recommender system is a system that aims to model users' behaviour and preferences, and recommend products (in this case, songs) that are very likely to be interesting to them. There are three major approaches to building a recommender system, namely:

i. Collaborative Filtering,
ii. Content-based, and lastly
iii. Hybrid systems which combine the above two approaches

[8]Collaborative Filtering is the process of filtering items based on the opinions of other people. Collaborative filtering technology brings together the opinions of large interconnected communities on the web, and uses it to support filtering of substantial quantities of data.
[4]Content-based recommendation systems are systems that recommend an item to a user based upon a description of the item and a profile of the user's interests. These recommendations are free from the opinions of other like-minded users.
[9]Hybrid recommenders use both user-item interaction data and the item descriptions to produce recommendations for the user.

The techniques employed in this project include unsupervised machine learning using K-Means Clustering, Principal Component Analysis and Cosine Similarity.

This paper provides a detailed explanation of the project, starting with I. Introduction section discussing the basics of recommender systems, II. Literature survey which discusses the latest work done by authors in this field, III. Methodology which describes our approach to this field and the techniques employed, IV. Algorithm which shows the final algorithm used to accomplish our research, V. Implementation which describes in detail how the system has been implemented using python, VI. Results which discuss the outcome of our work, VII. Conclusion which discusses the future of this project, our shortcomings and how we would improve upon it in the future.

II. Literature Review

Content based, collaborative and hybrid are the different approaches used by past researchers for the development of recommender systems.[6] In 2017, Juuso Kaitila developed a recommender system that used K-nearest neighbors (KNN) as the classifier to easily obtain the top-N recommendations for a given query song using euclidean distance as the measure. This recommender did not incorporate any user profiling and was based purely on objective similarity.
[1]In 2018, Hardik Jain, Jeevanjot Singh and Prashant Singh Rana built a hybrid recommender system using XgBoost, Neural Networks and K-Means Clustering. K fold cross validation was then applied to it to recommend a playlist of songs.

[5]In 2020, V Sri Nandhini and Dr. Thaiyalnayagi proposed an algorithm to build a Convolutional Neural Network (CNN) that classifies music in different genres based on the music's audio signal beats. Collaborative filtering algorithm is used in the customized music recommendation system to combine CNN output with logfiles to suggest music to the user.
[7]In 2016, Ja-Hwang Su and Ting-Wei Chiu proposed a recommender system that fused user ratings and music low-level features to enhance recommendations.
[3]In 2019, Rishabh Ahuja, Arun Solanki and Anand Nayyar developed a movie recommender system which employed Within Clustered Sum of Squares to find the right number of clusters so that K-means clustering can be applied and then Utility clustered matrix was used to define the average rating the user gives to each cluster.

III. Methodology

This section explains the methodology used by the proposed system.

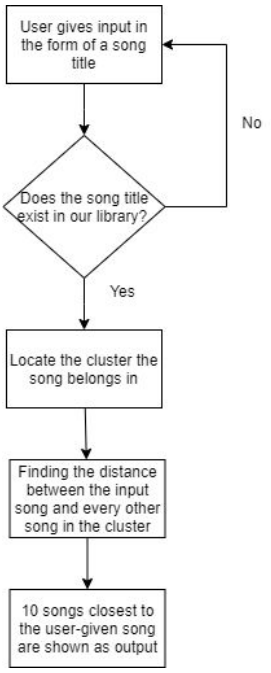

Fig1. Process Flow Diagram

The techniques involved in this project include:

WCSS

The Within Clustering Sum of Squares is a measure of the variableness of the observations within each cluster. It is the sum of squares of the distances of each data point in the cluster to its respective centroid. A cluster with a low WCSS score will be more compact than one with a high WCSS score. The WCSS score is used to find the number of clusters that will provide the minimal WCSS score. The WCSS score is calculated using the following formula,

$$WCSS = \sum_{i=1}^{m} (x_i - c_i)^2$$

K-Means Clustering

[2]K Means Clustering is one of the most simple and popular unsupervised machine learning algorithms. 'K', which is the number of clusters to be formed or more specifically, the number of centroids required in the clustering of the data, is found using WCSS.The centroid is the imaginary centre of the cluster. The data points are then allocated to the closest/nearest cluster. The algorithm starts off with randomly selected centroids which are used as the starting points for the chosen clusters. Then repetitive iterations are performed until we are provided with optimal positions of centroids. This process is finally stopped when either the centroids are stable or the defined number of iterations have been reached. $J = \sum_{i=1}^{m} \sum_{k=1}^{K} \omega_{ik} \| x_i - \mu^k \|^2$ where J is the objective function.

PCA - Principal Component Analysis

[12]Principal components are essentially variables that have been constructed as linear combinations or mixtures of multiple variables. These combinations are done in a way that the principal components are uncorrelated and most of the information within the original variables are compressed into fewer components. PCA is a form of unsupervised machine learning To perform PCA, the data is first normalized. A covariance matrix is then constructed to find out the correlations of the variables. Eigenvectors and eigenvalues are then calculated which is the direction of the axes of variation and the amount of variance in each component respectively.

$$\overline{X} = \frac{1}{N} \sum_{n=1}^{N} X_n$$

$$S = \frac{1}{N} \sum_{n=1}^{N} \left\{ u_1^T x_n - u_1^T \overline{x} \right\}^2 = u_1^T S u_1$$

Cosine Similarity

Cosine similarity is the cosine of the angle between vectors. The vectors are typically non-zero and within an inner product space. Mathematically, the cosine similarity is described as the division between the dot product of vectors and the product of the magnitude of each vector. This similarity is calculated for an array containing each of the principal components to produce a similarity score for a song, with 0 being the most similar and 1 being the least similar.

$$similarity(A,B) = \frac{A.B}{|A| \times |B|} = \frac{\sum_{i=1}^{N} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

IV. Algorithm

The steps involved in the working of the model.

Step I : The first step is the inclusion of the dataset with its features as mentioned above.Python libraries like pandas , numpy and sklearn are imported.

Step II : CSV file is read as a dataframe and the data.Data is preprocessed with dropping of unnecessary columns and normalization of columns that are out of range.

Step III : WCSS method was used to find the optimal number of clusters.

Step IV : The data is then clustered into 4 clusters and then further into 4 sub clusters.

Step V : The model then takes a song input from the user and calculates the similarity of the song to the clusters and the sub clusters present in the dataframe.

Step VI : It then returns the 10 most similar songs to the provided input as the result.

V. Implementation

A. Dataset

The spotify 160k+ tracks dataset was used in this project, which was taken from kaggle website. The dataset consists of the following features: artists, acousticness, danceability, duration_ms, energy, instrumentalness, liveness, loudness, speechiness, tempo, valence, popularity, key, mode, genres.

B. Data Preparation

The next step involves data cleaning and preprocessing. Certain features like 'duration_ms', 'explicit', 'mode', 'release_date', and 'Unnamed: 0' were dropped from the dataset since they were irrelevant to the project. Normalization of the features was implemented with the help of the sci-kit learn library. The features normalized were: 'tempo', 'loudness', 'key' and 'popularity'. The 'Genres' variable which was present in a different dataset

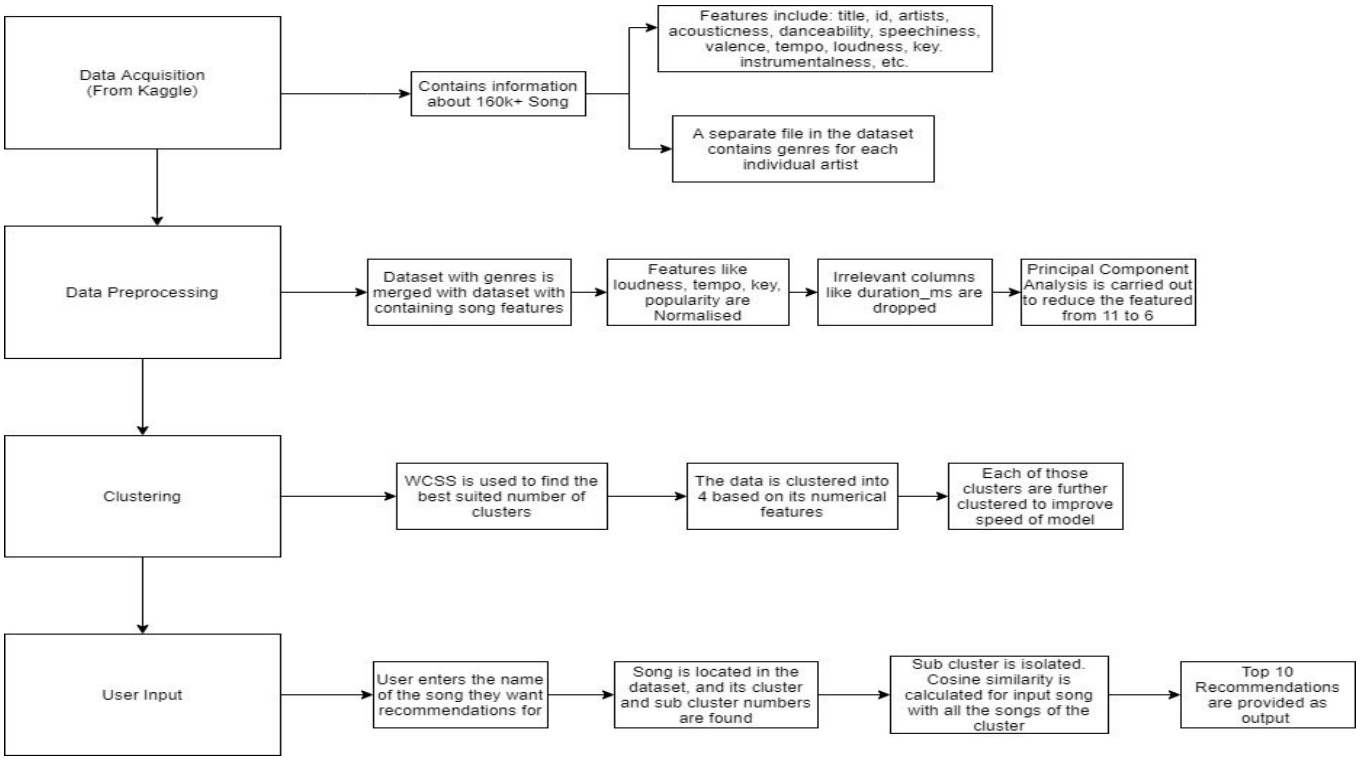

Fig2. Implementation Flow

was merged with the current dataset. Duplicate songs of the same name from the same artist were dropped to ensure no confusion. The numerical features were then used to carry out principal component analysis, which allowed us to reduce the number of features to be processed.

C. Data Clustering

In this step, the songs are clustered. The 160,000 songs were first clustered into one of four clusters using the K-means clustering function that was built from scratch, following the K-means algorithm. The optimal number of clusters was determined using the WCSS method. Each cluster was then further clustered into four different sub-clusters. Thus each song had a 'cluster' and 'sub-cluster' attribute added to it. Columns containing corresponding cluster and sub cluster numbers were added to the dataset.

D. Computing Similarity and Generating Recommendations

In this step, the user is asked to input a song name, for which the model provides 10 similar songs. This is done by first locating the input song in the dataset. The cluster and the sub cluster number of the input is used to create a new dataset with just the songs belonging to the corresponding clusters. The input song is then compared to each and every song of that dataset. This comparison is done by finding the cosine similarity for each of the features achieved by PCA. The similarities are added and a score is generated for each song. The songs are then arranged in order of their scores from least to most. The first 10 songs are considered to be the most similar to the input song and therefore are given as output to the user.

TABLE I . PACKAGES INCORPORATED

Packages Installed	**Implementation**
Pandas	Reading ,saving csv
Numpy	Matrix/Array multiplication
Sklearn	Normalization, KMeans Clustering, KNeighborsClassifier, NearestNeighbor

VI. Results

Since the proposed model is neither classification or regression, the evaluation is not very straightforward. For evaluating this model, we compared our clustering algorithm to that of sklearn. To do this comparison, we used KNeighborsClassifier function from sklearn to classify songs to particular cluster and subcluster

```
clf= neighbors.KNeighborsClassifier(weights='distance',n_neighbors=10, leaf_size=15)

clf.fit(X_train, y_train)

KNeighborsClassifier(algorithm='auto', leaf_size=15, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=10, p=2,
                     weights='distance')

clf.score(X_test,y_test)

0.9828852417140302
```

Fig3. K Means from Sklearn showed 98.288% accuracy

```
[14]: clf= neighbors.KNeighborsClassifier(weights='distance',n_neighbors=10, leaf_size=15)

[15]: clf.fit(X_train, y_train)

Out[15]: KNeighborsClassifier(algorithm='auto', leaf_size=15, metric='minkowski',
                              metric_params=None, n_jobs=None, n_neighbors=10, p=2,
                              weights='distance')

[16]: clf.score(X_test,y_test)

Out[16]: 0.9812248547161377
```

Fig4. K Means algorithm written from scratch showed 98.122% accuracy.

Music taste is subjective, and since an updated music history of users could not be found, the recommendations produced by our model were compared to those produced by NearestNeighbor from sklearn library.

The song chosen for this comparison was “She” - dodie.

```
Recommendations for she dodie:

turning page sleeping at last
tepid rainscape fizzonaut
ocean lady a
desperado - 2013 remaster eagles
don't let me down - remastered 2009 the beatles
stairway to heaven - 1990 remaster led zeppelin
no matter what calum scott
dreamland glass animals
golden thing cody simpson
the greatest lana del rey
Time:  0.3492621999999983
```

Fig5. Recommendations from NearestNeighbors using clustered data

```
Enter Song Name: She dodie
she dodie
(11144, 14)
10902
                                              name     score
10810                  cherry wine - live hozier  0.013079
11035                     touch sleeping at last  0.014053
11046  boys like you - acoustic anna clendening  0.016616
8520   above the clouds of pompeii "bear's den"  0.024007
10390                 alone with you canyon city  0.025971
10841               perfectly wrong shawn mendes  0.026764
11056   fly me to the moon the macarons project  0.027572
10544                   real estate adam melchor  0.029399
9178                two of us on the run lucius  0.033445
10756             first fuck 6lack & jhené aiko  0.034653
Time:  8.038433899999987
```

Fig6. Recommendations from the proposed model

Time taken by the proposed model to generate recommendations (8.03 seconds) is significantly greater than that of NearestNeighbor (0.35 seconds). But on evaluation of the music recommended, the proposed model generates songs more similar to the query song - “She”-dodie.

A basic website using the flask library was created to display the output of the model in a user-friendly format.

VII. Conclusions

Unsupervised machine learning algorithms find patterns in a dataset without any reference to outcomes that may be known. Concepts of supervised machine learning cannot be used in this recommendation system because this is neither a classification or regression problem. We use K Means Clustering and Principal Component Analysis to instead discover any underlying structure of the data. These techniques also allowed our model to be more time efficient, since a smaller sample was generated to perform the next step on. Cosine Similarity with a few external tweaks was used to generate a similarity score. Upon comparison of our model with the likes of a simple KNearest Neighbor model, our model is less time efficient even with the inclusion of PCA and Kmeans clustering, but gives better recommendations (subjective).

The proposed work can be improved by using Spotify’s API features to collect user information like recently played songs and the features of the songs directly. This would allow the dataframe to grow, considering the current dataset only contains 160k songs which is a very small percentage of the Spotify library. Songs which are not found in the datasets could be assigned to a particular cluster and sub cluster using K Neighbors Classifier. The work can further be improved by using users’ Spotify information to incorporate a collaborative approach to the recommender system.

References

[1] Hardik Jain, Jeevanjot Singh, Prashant Singh Rana ,”Music Recommender System” ,https://www.ijtra.com/,October 2018

[2] Hemanth Sharma ,”K Means Clustering Algorithm and how it works”, October 2020

[3] R. Ahuja, A. Solanki and A. Nayyar, "Movie Recommender System Using K-Means Clustering AND K-Nearest Neighbor," 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), Noida, India, 2019, pp. 263-268, doi: 10.1109/CONFLUENCE.2019.8776969.

[4] Pazzani M.J., Billsus D. (2007) Content-Based Recommendation Systems. In: Brusilovsky P., Kobsa A., Nejdl W. (eds) The Adaptive Web. Lecture Notes in Computer Science, vol 4321. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-72079-9_10.

[5] Vajjala Srinandhini and Thaiyalnayaki Krishnan ,”Music Recommender System”, 2020

[6] Juuso Kaitila,University of Tampere , Natural Sciences Computer Science, ”A content-based music recommender system”,2017

[7] Su JH., Chiu TW. (2016) An Item-Based Music Recommender System Using Music Content Similarity. In: Nguyen N.T., Trawiński B., Fujita H., Hong TP. (eds) Intelligent Information and Database Systems. ACIIDS 2016. Lecture Notes in Computer Science, vol 9622. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-49390-8_17

[8] Schafer J.B., Frankowski D., Herlocker J., Sen S. (2007) Collaborative Filtering Recommender Systems. In: Brusilovsky P., Kobsa A., Nejdl W. (eds) The Adaptive Web. Lecture Notes in Computer Science, vol 4321. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-72079-9_9

[9] E. Aslanian, M. Radmanesh and M. Jalili, "Hybrid Recommender Systems based on Content Feature Relationship," in IEEE Transactions on Industrial Informatics, doi: 10.1109/TII.2016.2631138.

[10] Arjun Raj Rajanna, Kamelia Aryafar, Ali Shokoufandeh, and Raymond Ptucha. Deep neural networks: A case study for music genre classification. In Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on, pages 655– 660. IEEE, 2015

[11] Dr. Michael Garbade https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1.

[12] Mishra, Sidharth & Sarkar, Uttam & Taraphder, Subhash & Datta, Sanjoy & Swain, Devi & Saikhom, Reshma & Panda, Sasmita & Laishram, Menalsh. (2017). Principal Component Analysis. International Journal of Livestock Research. 1. 10.5455/ijlr.20170415115235.