The International Journal of Engineering and Science (IJES)
|| Volume || 8 || Issue || 11 || Series II || Pages || PP 01-05|| 2019 ||
ISSN (e): 2319 – 1813 ISSN (p): 23-19 – 1805

Smart Music Recommendation System

Vinay Ambre, Rohit Ramaswamy, Rishabh Gajra
Department of Computer Engineering
Sies Graduate School of Technology, Mumbai, India
Corresponding Author: Vinay Ambre

----------ABSTRACT----------
The smart music recommendation system curates the music library using machine learning, dynamically and creates smart playlists based on a myriad of factors. The recommendation system takes away all the distraction and unnecessary elements that complicate the users listening experience. The smart music system analyzes and takes into consideration a variety of data, such as: users listening behaviour the music library data, time and anonymous analytics information from other users. This data would help the smart music system to give better recommendations. The recommendation system continuously teaches itself about the users music taste and would help the user to explore new songs that are similar and what he/she might like. We use both content based as well as collaborative filtering taking the user ratings as well considering what type of songs are listened by the user.

Index Terms—Machine Learning, Recommender system, Collaborative filtering.

Date of Submission: 28-10-2019 Date Of Acceptance: 16-11-2019

I. INTRODUCTION

RECOMMENDATION systems are one of the most successful and widespread applications of machine learning technologies in business. A recommendation is a content suggestion to users implying that if the user was looking to try some new content out, this particular recommendation would be well suited to their tastes with the content pertaining to several areas like movies, news, books, music, etc. A recommendation is called good when upon trying out the suggested content, the user ends up liking it as intended by the recommendor. A recommendation system, thus, is designed to provide content for users designed to suit individual needs by working on variables like their preferred content, their personal history, etc. Such systems are mostly classified into collaborative filtering and contentbased filtering. Collaborative filtering systems present content that is based on similar users also using the system. If two users tend to like similar content, the system proposes their favorite content to the other user and avoiding content they did not like. Content-based filtering works more on the individual user's tastes and finding out similar new content to that of the user's history, the similar content is the focus. Content-based methods are based on similarity of item attributes and collaborative methods calculate similarity from interactions.

II. LITERATURE SURVEY

We have compared the three aforementioned music streaming services and have found a few shortcomings in their working. SoundCloud misses out on automatic playlist generation. SoundCloud does allow the user to create his own playlist and share them with fellow users but does not create one for the user. Saavn and Spotify both create playlists for their users. Saavn, however created a "For You" playlist at the time of registration for the user to begin with, but it not being dynamic is a let down. Spotify is the leader in this aspect. Spotify created "Daily Mixes" for all of its users. These automatically generated playlists keep changing everyday as per what the user listens to. SoundCloud has the advantage of being a social network for artists as well as a popular music streaming service. This serves to be a great platform for people who like experimenting with their music. Some of the songs can be downloaded for free from the platform. This also has its cons, it is good for discovering remixes of songs made by users just like you, but the original track posted gets lost at times. However, being open is a good thing for new artists to come in the limelight of the music industry. Spotify and Saavn do not have a very good profile management system. The songs one listens to is uploaded by verified artists and not by users. Spotify allows the user to make their playlist public which helps the user to gain popularity on the platform, however user made playlists on Saavn are kept private. After studying the three services, none of the services are perfect but are the best in their own way. SoundCloud being a social network helps the user to discover more variety of artists and songs. Spotify helps you to discover new music and has

curated playlists for each mood of yours, whether you are relaxing, working out or partying. Saavn provides songs from top artists in the country, custom radio stations and exclusive releases.

III. DATASET AND SCHEMA

The metadata of songs is taken from a dataset found on kaggle. It contains characteristics about a particular song which includes energy, danceability, acousticness, etc. The dataset contained two separate csv files - songdata and songinfo. songdata contained the characteristics of the song (acousticness, energy, liveness, speechiness, danceability, instrumentalness, etc), while songinfo contained the metadata of the song (album, artist, genre). The two datasets were concatenated using the song name column and stored in the database. A schema is maintained for storing the user ratings from the user for a specific song which contains the song id, user id and rating in numerical type. The user profile contains the data for generating metrics for the user. For every time a user plays a song, a new record is created in the database denoting that the song has been played, if it is streamed again then the count attribute is updated by one.

IV. PROPOSED SYSTEM

4.1 Data Flow

There are a plethora of parameters to be considered before recommending a song to the user. User data, music library and data analytics are important sources of data necessary for the algorithms to provide the recommendation. The major sources for generating a favourable song would be the music library and the analytics data of the user collected over a period of time from all the users using or having used the system.

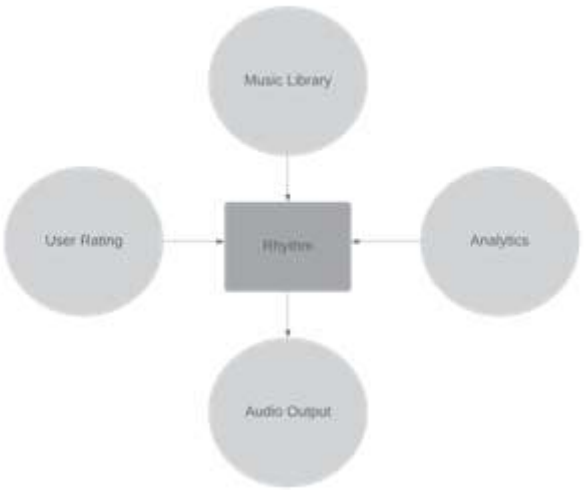

Fig 3.1.1 Proposed System

4.2 Architecture

The backend used for hosting the metadata of the songs as well as the API used for the mobile app is made up of Express, Mongoosejs and MongoDB. Express is a node.js framework for building web applications. With a myriad of HTTP utility methods and middleware at disposal, creating a robust API is quick and easy. The apis will be called from the mobile app and the express js controllers will respond with the response. The Mongoosejs module will be used as a driver to access the data storage of MongoDB. Hence, mongoose queries will be used to access the data. The queries would be made from the mobile app and the hosted backend would give the appropriate response to be shown in the app. A machine learning server would also be hosted that would be responsible for all the computations performed on the metadata of songs. Flask is used for this purpose. Machine learning models will be trained when major changes will be identified in the schema and models will be trained and stored on the server. Whenever a request is made to the python server, these models are imported and predictions are made upon them.

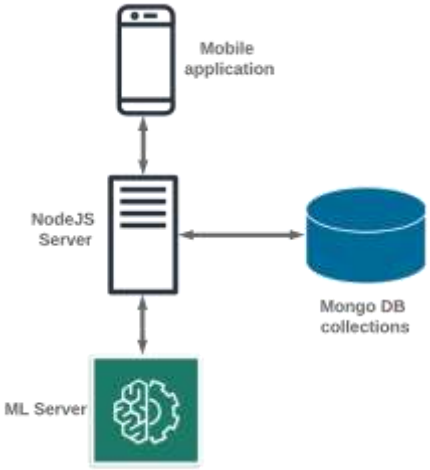

Fig 3.2.1 Architecture

V. MODELS

Similar Songs

The metadata of the songs in the songs database is used to recommend similar songs to a particular song. This is used for the radio feature in the system where similar songs are suggested. The significant features of a song were selected i.e. acousticness, danceability, energy and the different genre of the song. Out of which the genre feature was hot encoded for clustering purpose. K nearest neighbour was used to find nearest neighbours in the dataset which represents the similar songs to a given song. These songs are similar in the features provided above. A similarity score is provided for each song showing the measure of similarity with the neighbour song.

Rating Prediction

Collaborative filtering approach was used in predicting the rating the user might give to a given unheard song. Collaborative filtering models which are based on assumption that people like things similar to other things they like, and things that are liked by other people with similar taste. User based collaborative filtering was used wherein the similar users were identified and the rating given by that similar user was used to determine the predicted rating[3]. Surprise library contains knn inspired algorithms to perform collaborative filtering. Out of these, KNNBasic is a basic collaborative algorithm which performs cf. It clusters similar users using cosine similarity. The formula for the same goes as, The ratings of the similar users are observed and the ratings are predicted for unrated songs.The mathematical model for KNNBasic is as follows:

$$\hat{r} = \frac{\sum_a (u,v) \cdot r_{vi}}{\sum (u,v)} \quad (1)$$

The user ratings are stored in the database in document form which is fetched later to train the model and group similar users together. Thereafter, ratings are predicted for unrated songs for a specific user and is arranged in descending order of predicted rating. The user must rate more and more songs in order to get good predictions.

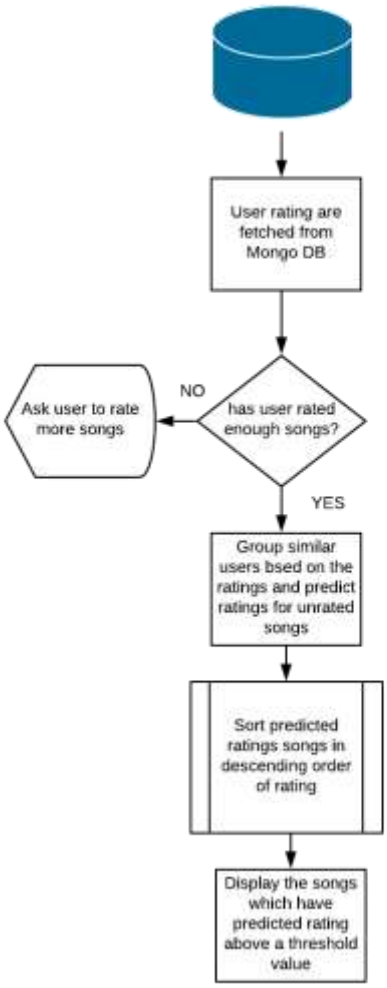

Fig 4.2.1 Rating Prediction flow diagram

5.3 Automatic Playlist

The automatic generated playlist will contain the recently played songs by the users depending on the number of plays by the user. This playlist will depend on the time of the day. User playing specific songs during the specific period of the time will be noted and playlist will be made accordingly. A record of songs played by the user will be stored in the database which will then be processed to create playlist according to the time of the day. This reduces the job of the user to create his own playlist which contains the songs that he/she listens to everyday during different time of the day. i.e. Morning, Evening, Late Night.

VI. USER METRICS

User metrics can be generated for a specific user. User metrics will include basic user liking like most streamed song, favourite artist or album, etc. Song characteristic which the user listens to will also be calculated like acousticness, danceability, energy. This metric can be displayed in the form of percentage. Genre diversity the user listens to can also be calculated to predict how diverse the user is in his choice of song. The time of the day during which the user listens to the song can also be noted which will be helpful in creating automatic playlists. In general, streaming habits of a user can be calculated using these metrics and similar users can be identified. Everytime a song is played the three primary characteristics i.e. energy, danceability, acousticness are observed. The value of the attribute with the higher score is modified in the user schema. For generating the user metrics this data is used to show analytics.

VII. ACCURACY AND RESULTS

7.1 Accuracy Metrics

The collaborative filtering model (KNNBasic) for predicting the user ratings for non rated songs was tested on a movielens dataset of user ratings given to movies which consisted of 100,000 ratings by 600 users on 9000 movies. The traditional method for testing the accuracy of recommendation systems include finding the RMSE (Root mean square error) or MAE (Mean Absolute Error).

Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors).

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\left(\frac{d_i - f_i}{\sigma_i}\right)^2} \quad (2)$$

Mean absolute error (MAE) is a measure of difference between two continuous variables.

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|e_t| \quad (3)$$

We used K fold method for evaluating accuracy metrics with k=5, The results are as shown in Table 6.1.1,

fold no.	1	2	3	4	5	Mean
RMSE	0.9746	0.9701	0.9717	0.9853	0.9617	0.9727
MAE	0.7515	0.7443	0.7499	0.7568	0.7426	0.7490

Table 6.1.1 RMSE and MAE Accuracy metrics

As we can see, the mean RMSE and mean MAE for 5 folds was 0.9727 and 0.7490 respectively which is acceptable as there is an average deviation of less than 1 rating value for error.

7.2 Results and Conclusion

Consider a sparse matrix as below containing users in rows and songs in columns.

	song1	song2	song3	song4	song5	song6
user1	5		5	5		5
user2	5	3	5	5	0	5
user3			5		5	

Table 6.2.1: user-song rating sparse matrix

As we can see in Table 4.2.2.1, User 1 and User 2 have listened to similar songs and User 1 havent rated song 2 and Song 5 yet. i.e. never listened to song 2 and song 5. In this case, the prediction for User 1 for song 2 will be 3 and for song 5 it will be 0. The same result was obtained by using the rating prediction model (KNNBasic). In such manner the predicted ratings for not listened songs are arranged in descending order and those songs are recommended to the user. This feature needs user ratings for the user in order to suggest proper songs that the user may like. Hence, the more the user rates the songs, the more accurate are the predicted rating for any song.

REFERENCES

[1]. https://www.kaggle.com/edalrami/19000-spotify-songs
[2]. ZsoltMezei, Carsten Eickho. Evaluating Music Recommender Systems for Groups. VAMS17, August 27, Como, Italy.
[3]. https://arxiv.org/pdf/1707.09790.pdf
[4]. Rahman, M. S., Rahman, M. S., Chowdhury, S. U., Mahmood, A., Rahman, R. M. (2016). A personalized music recommender service based on Fuzzy Inference System. 2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS). doi:10.1109/icis.2016.7550757
[5]. Lee, J., Lee, J. (2018). Music Popularity: Metrics, Characteristics, and Audio-Based Prediction. IEEE Transactions on Multimedia,20(11), 3173-3182.
[6]. doi:10.1109/tmm.2018.2820903

Vinay Ambre" Smart Music Recommendation System" The International Journal of Engineering and Science (IJES), 8.11 (2019): 01-05