Socializing Platform based on Music Interests of Users

Tanishq Chugh
Dept of Computer Science
PES University
Bengaluru, India
tanishq_chugh@yahoo.co.in

SUNDEEP A
Dept of Computer Science
PES University
Bengaluru, India
sundeepannabathula0@gmail.com

Jimish Mrug
Dept of Computer Science
PES University
Bengaluru, India
mrugjimish@gmail.com

Abstract— Match(IT) is a socializing platform built to connect people based on their music interests.
In this ever- evolving world, people feel more comfortable to interact with each other virtually as compared to in-person and having some sorts of similarities, makes the conversation to flow smoothly and people vibe much quicker.
Music is one of the most important metrics which binds people together as it brings out the inner sentiments and personality of a person.
Recommending people music alike to their taste and using the same to help them connect with people who have a similar taste would be a great way to socialize. Hence, a product that uses Music as the metric to help people meet similar minded people can prove to be the most efficient one in its industry.
So, we built a model which will predict the top 'k' similar songs of a User and find other Users who Listen to any of the top 'k' similar Songs. Based on this information, we display the top 3 Users to the Current User. This way we are connect different Users based on their Music Taste's.

Keywords— Data Mining; Machine Learning; Recommender Systems; Web Development;

I. INTRODUCTION:

Music taste is a unique, peculiar and characteristic quality of a person. Of all the millions of songs and sounds that exist, the fact that many people decide to develop a liking for a particular style, a "subset" of music is something that We believe is not by mere chance.
We built a model, to connect different people based on their music tastes. We will be collecting information from the Users about their music preferences like artist, Song to which they listen regularly.
Based on this information provided by the User, we will produce a list of similar songs. These Similar songs are predicted using a variety of Machine Learning models like: K-nearest-neighbours, DBSCAN, Using Cosine Similarity on varies parameters of the song to find the top 'k' songs.
Based on these Similar songs generated, we are going to find the list of Users who listen to any one of the songs from the similar song list. And we are going to pass the information of the top 3 Uses to the User on whose preferences we are making the recommendations.
The Users can also share their Instagram ID. So that they can keep in touch with each other.

Similarity is a measure that calculate how much two objects have similar distances. If we observe our songs as data points, we can measure the similarity of two songs using a distance function.
we applied the k-nearest neighbours (k-NN) algorithm. The k-NN searches for the k similar elements based on a query point at the centre; i.e., searches for k similar songs from a selected song
DBSCAN is a density-based clustering algorithm. That is, the clusters it discovers consist of elements that in the input data forms a dense subset of the input.
In order to find these clusters, parameters must be passed to the algorithm which are used to determine which areas are to be considered dense and which are not. These fundamental parameters are and min_samples. Due to its properties DBSCAN can discover clusters of arbitrary shape.
Cosine similarity measures the similarity between two vectors by calculating the cosine of the angle between them. For Our model, vector A remained constant as our user profile. We then looped through all the songs and plugged in each as Vector B to calculate the cosine similarity of each song to my profile. It then prints the top 'k' Similar Songs.

II. LITERATURE REVIEW / RELATED WORK:

The Author, **Vuong Khuat. [1]** attempt to build a recommendation system for music tracks using collaborative filtering with two optimization methods, dataset rescaling and automatic halting. The Author Uses the dataset, Echo Nest Taste Profile Subset, which provides play counts of more than 300,000 users for approximately 1 million songs.
The function takes in the dataset d, a user U, a positive integer N and returns a list that contains N songs that best match U interest. For this, f determines the users who are similar to U. Now, Recommendations are made from a list of Songs to which a proportion of similar users have listened to but U has not. To prevent the model from producing too many recommendations, we set the maximum number of recommendations to be 10k, and the songs chosen are the most frequent songs among the list of similar users.
To reduce the Runtime, the author has implemented 2 features:
Randomly select a subset of the dataset, Halt after finding enough similar users at s = 0.4 and m = 1000, we attain the highest values for precision and recall rates at 0.0392 and 0.426, respectively.

The Author, Sheela Kathavate [2] uses a hybrid approach which involves individual implementation of collaborative and content-based methods and aggregation of their predictions to generate recommendations.
A generic consolidative model that is the assimilation of both content based and collaborative characteristics is proposed
The main objective of this work was to develop an application for music recommendations. The application allows users to select and listen to the songs available in the device. Whenever a user listens to a particular song, a log is created.

In order to suggest songs to the users, they used various strategies to implement recommendation engine. The main motive of this Proposed System is extending the capabilities of the traditional recommendation System. The experimentation was done using twenty artists. In this experiment, the author was able to make a music recommendation system using a hybrid approach of collaborative and content filtering. They were able to play and recommend songs in four languages covering more than forty artists. To improvise the system. They asked the users about their preferences and we also provided them with playlists of popular and latest songs. The author has tested the system with at least twenty users and the results shown were quite promising. We received an accuracy of 96% on the music recommendation system.

The Authors, Adi Riyan Prasetya and Desi Ramayanti [3] have created a taste profile for each user with the individual taste in music. Taste profile was created based on the frequency of playing to a song, whether the user skipped tracks or not which works as thumbs up and down and with explorations.
The Author built a model which leads to an alternating least squares optimization process, where the item and user vectors are initialized at the beginning. Then the item vectors are fixed and solved for optimal user vectors, by taking the derivate of the min function, setting equal zero and solving. (2). The user vectors are fixed and solved for optimal item vectors. (3). This procedure will be repeated until it converges.
They have also Implemented the feedback loop, where the collected output information is sent back to the input and adjusts by the behaviors. This process is called feedback control by error, because the system controls the behavior and eliminates errors.
The system detects and analyses information after learning research and reflects the given information.
For the last literature review, we decided to look at People – People Recommendation,
Here The Authors, Xiongcai Cai, Michael Bain, Alfred Krzywicki [4] have developed CollabNet, a novel algorithm that uses gradient descent to learn the relative contributions of similar users or items to the ranking of recommendations produced by a recommender system, using weights to represent the contributions of similar users for each active user. The authors propose to learn a model for the weights rather than use heuristic similarity, where the weights are defined as the reliability of similar users. The same pair of similar users can have different contributions for different active users. Evaluation of CollabNet recommendations on datasets from a commercial online social network shows improved performance over standard Collaborative Filtering. CollabNet outperforms others methods in both Precision and Recall.

III. PROPOSED SYSTEM:

A. *Dataset*

We have Used a Spotify Songs Dataset Which had around 3731 rows and 14 columns. The Dataset had Attributes like: SongName, AritstName, Popularity, Danceability, Energy, Key, Loudness, Mode, Speechiness, Acousticness, Liveness, Valence, Tempo, Duration_ms.
On this dataset, after performing Data-Cleaning operation like removing Duplicates, etc. we Finally have only 1084 rows to work with .
Since each attribute had data of different Ranges, we decided to Normalize the data. For Normalizing the data, we used **The min-max feature scaling.** The min-max approach (often called normalization) rescales the feature to a hard and fast range of [0,1] by subtracting the minimum value of the feature then dividing by the range.

From the Users Data, We Used to Take the SongName as the Input to our Recommender System and we were using the Features of other attributes like: Popularity, Danceability, Energy, Key, Loudness, Mode, Speechiness, Acousticness, Liveness, Valence, Tempo, Duration_ms to find the Similarity between various Songs and given Song from the User data.

B. Methodology

Traditionally, People are recommended to other people based on the Mutual friends that exist between the two Users and since Mutual Friends, the recommended people all stay in the same City/State. So, people won't be able to meet with different people around the world who have similar taste like them.
To solve this issue, we are going to make recommendations to Users based on their music taste. Since music has no boundary, even a person staying in the USA can also have the same music taste as an individual who stays in INDIA. So, using our model we can make recommendation to an USA guy about an INDIAN guy.
And there is a certain probability that those two people would have never interacted with each other if it wasn't for our Model.

The main objective of this work is to develop an application for Bringing Different People together based on their music taste's.
We will be using a web application to collect all the user information like Name, Age, Gender, the song to which they mostly listen and the Uses are also free to give their Instagram ID. Now, we have the User details along with the song to which they often listen.
And we also have a dataset which contains a list of songs with various features which are used to uniquely identify each song.
Now we have all the details which are required for computing similar songs and computing similar Users based on similar songs data.
To find the similar songs, we will be using a list of models which work recursively one after the other till we get the names of
at least 3 Users who are similar to the given User.
Before we move on to the model which were used, we need to clean the data and normalize the data and all the model predictions will be done based on this normalized data.

The first Model what we are going to Use is Cosine Similarity to find the list of songs which are similar to the song specified by the User.
Cosine similarity measures the similarity between two vectors by calculating the cosine of the angle between them.
For my model, vector ui remained constant as my user profile. I then looped through all the songs and plugged in each as Vector uj to calculate the cosine similarity of each song to my profile. It then prints the top 'k' Similar Songs.

$$cos(\vec{u_i}, \vec{u_j}) = \frac{\vec{u_i} \cdot \vec{u_j}}{\|\vec{u_i}\|\|\vec{u_j}\|}$$

This is one of the standard techniques for calculating similarity in collaborative filtering algorithms. The dot product of ui and uj can be calculated by finding the components in which both vectors have non-zero values, then sum up the products of the corresponding values.

the similarity of two users, ui and uj can be determined by the cosine of the angle between the corresponding vector:

The next Model that we are going to use is DBSCAN.
I use the DBSCAN algorithm from the "SKLearn" library to help me cluster the songs the DBSCAN algorithm does not have to give a pre-defined "k" value since it automatically clusters the songs based on the values that is assign in the "eps" and "min_sample." "min_sample" is set to 24 since the "min_sample" is usually twice the number of the feature columns and a total of my feature columns is twelve after scaling. For finding the value of 'eps' , we are finding the nearest neighbor's values by setting n_neighbors=24 and sort the distances obtained. We plot the sorted distance values to visualize, and find that it has a sudden spike corresponding to 0.57 on the y-axis. So, we set the value of eps ='0.57'.
Finally, we had 3 cluster, where cluster 0 had 368 items , cluster 1 has 583 items, cluster 2 had 33 items and there were 100 outliers.

The next Model that we going to use is KNN:
We are using Elbow method to find the ideal number of clusters based on the normalized data, we are also finding the ideal number of clusters using Reed method. In both the cases we get the same value, that is the ideal number of clusters is 7.
After finding the ideal number of clusters, we are finding the value of the cluster in which the User's SongName is present.
After finding the cluster number, we are computing the distance of each element of that cluster from the required SongName and sorting based on the distance. Finally, we are displaying the top 'k' Songs which are nearest to the required SongName.

The next Model that we going to use is Euclidian Distance.
We are going to compute the mean of each Song using all the attributes of that song. Now, we have mean values of each song.

We are going to take down the value of the mean of the required song. And we are going to use Euclidian Distance formula to find the distance value between the mean of the required song and all other values iteratively. Now we are going to sort the distance values and print the top k nearest SongNames.

IV. RESULTS:

In this experiment, we were able to make connections between different Users based on their Music Taste. To find the similarity between Users, since we had a limited User base. So, there were limited songs to match with in the first place. In order to get a minimum of 3 Users we needed to get more similar Song Names. To deal with this issue we are using multiple models to predict similar songs. And we are iteratively traversing through each model till we get at least 3 Users. For predicting the songs we are using 4 Models and these models are: Cosine Similarity, DBSCAN, KNN, Euclidian Distance which are used to find the similarity between songs using various metrices.

V. CONCLUSION:

We made use of multiple models to build a platform that allows the users to find and connect with people
with alike music taste based on the preferences entered by the user.
We provide users flexibility to update their preferences so that they can choose to find and get in touch
with people according to their current music taste and not be fixed onto one which enables the user to
interact with people of various music taste.
We believe this platform is effectively serving the purpose of helping users socialize in a unique and a
more practical approach as well.

VI. FUTURE SCOPE:

Deployment & Testing of the product.
Developing an interface to take feedback from users to develop a dataset relating users to his/her music interests as well as testing the application based on match-predictions.
Develop this application to be a fully functional social network including features like - chat interface, user posts, etc.
Currently we are run these models on a limited User data which we have manually entered. In the future, we would like to check the performance of our models on a larger User Database and Song Database.

VII. REFERENCES:

1. https://portfolios.cs.earlham.edu/wp-content/uploads/2019/08/final_paper_draft_Vuong.pdf

2. **https://www.ijert.org/research/music-recommendation-system-using-content-and-collaborative-filtering-methods-IJERTV10IS020071.pdf**

3. **https://www.ijcaonline.org/archives/volume181/number36/prasetya-2019-ijca-918320.pdf**

4. **https://ieeexplore.ieee.org/abstract/document/5694032**