 ISSN: 2454-4248
22 – 25

Song Recommender System Using Machine Learning

Nikhil Sonavane
Electronics and Telecommunication Department
Vishwakarma Institute of Technology Pune, India nikhil.sonavane16@vit.edu

Rushabh Jangada Instrumentation Department Vishwakarma Institute of Technology
Pune, India
rushabh.jangada16@vit.edu

Ms. Sonali Sonavane, Asst. Professor
Computer Engineering Department
G. H. Raisoni Institute of Engineering and Technology
Pune, India
sonali.sonavane@gmail.com

Abstract— Recommender systems use algorithms to provide users with product or service recommendations..The key to a recommendation system is the prediction of users' preferences. Personalized recommendation for many online music applications depends on the prediction of both long- term as well as the short-term preferences. The project works on the recommender algorithms of Machine Learning (ML) and Data Analytics (DA). This project uses 3 recommender models for 2 different cases. The model looks into each case and accordingly decides which recommender system to use. The three models namely are, Popularity based model, Content based Filtering, and Collaborative filtering. Collaborative filter is divided into two parts further, User based CF and Item based CF. Moreover, it discusses hybrid approach by combining user and item based collaborative filtering model and how it helps us in increasing precision. The million song database was taken from Kaggle.

*****

I. INTRODUCTION

It is a big challenge to provide recommendations to people from the large data available on the internet. E-commerce giants like Amazon, Ebay provide personalized recommendations to users based on their taste and history while companies like Spotify[1] . Pandora use ML and deep learning techniques to provide appropriate recommendations. In this paper, the author has focused on implementing a good personalized recommendation system using user's history. The author first implemented popularity based model which is very simple but is not personalized followed by Content and Collaborative based filtering which provide personalized recommendations based on history. The author has also implemented a hybrid approach in which he combines both content and collaborative techniques to extract maximum accuracy and to overcome drawbacks of both types.

II. LITERATURE SURVEY

Research paper published by David R. Cheriton School of Computer Science University of Waterloo Waterloo, ON, Canada[3], gives a clear view of how recommender systems for multiple applications work. Their published paper , gives an overview of how different algorithms used in ML for making recommender systems work. It gives a theoretical view of which algorithms would work the best, in which cases. The major concept of using the three recommender algorithms along with the hybrid model in this project were understood and applied using this paper.

Another paper published by IIT, Kanpur, India[4], proposes the use of collaborative filtering for song recommendation system. The author uses a unique idea for deciding on how much part to apply user bassed and item based collaborative filtering.

III. DATASET AND PREPROCESSING

The author used the Million song dataset provided by Kaggle[2] . The dataset consist of two file viz triplet file and metadata file. The triplet file is collection of user id, song id and listen count while the metadata file consist of data regarding the songs like the song id, artist, album and year which acts as features of song/feature vector. The author has used only 1 million records from triplet file for the author's recommendation system which has over 40,000 users with

9,000 songs as actual dataset is huge, hence computationally expensive.

Later the author did little preprocessing on the data. After combining the triplet and metadata file he

ISSN: 2454-4248
22 – 25

converted listen counts to ratings between range 1-5 as it's easier to work with ratings. This was done by considering the highest listen count of each user as highest rating 5, for that user and ratings for other songs were calculated accordingly.

IV. ALGORITHMS

A. Popularity Based:

This is the most simple and intuitive model. In this model the author suggests the user the N top most songs – the most popular songs. This is similar to Trending part available on Youtube. Popularity of songs are calculated based on listen count or ratings then the songs are arranged in descending order based on popularity and top N songs are recommended to the user. This model has several problems as it is very naive one. First of all, this doesn't give personalization, that is everyone is recommended with same most popular songs. Also some unpopular songs are never suggested in this model.

B. Content Based Filtering:

Content based Filtering focuses on the user's account in order to give suggestions to the user. User's account has all the information about user's taste and content based filtering uses this aspect for recommendation. User's history plays an important role in this model. The author tries to find songs similar to ones which user has rated positively in his history. Each song can be represented with a feature vector. Similarity between any two songs can be found by using cosine similarity. In cosine similarity the author tries to find angle between two features vectors representing the two songs which is found by dot product of two vectors divided by the norm of the two vectors. Smaller the angle is, closer the feature vectors are and hence more similar the two songs are [5].

C. Collaborative filtering

This algorithm is entirely based on the past behaviour and not on the context. This makes it one of the most commonly used algorithm as it is not dependent on any additional information, that is it doesn't need any meta data/features about the songs for recommendation and hence it allows us predict songs without actually knowing, what the song is about? Who is the singer ? and so on. It only uses only the ratings given by the user for finding recommendation for user. There are two types in this model Item based collaborative filtering and user based collaborative filtering.

1. User based Collaborative filtering

The main intuition behind this model is similar users listen to similar songs. In this algorithm, we assume users having similar history or rating pattern have same taste and hence we can recommend songs from history of users having similar taste to our current user whom we have to recommend songs. Similarity between the users can be calculated by using various techniques like cosine similarity or Pearson correlation. Pearson correlation is similar to cosine similarity with a little variation - mean normalized weights. The need of this normalization is due to difference in perspective of different users[6].

$$P_{u,i} = \overline{r}_u + \frac{\sum_{u' \in u} s(u,u')(\overline{r}_{u',i} - \overline{r}_{u'})}{\sum_{u' \in u} |s(u,u')|}$$

2. Item based Collaborative filtering

The intuition behind this method is items which are rated similarly by the users are similar. In item-based model, it is assumed that songs that are often listened together by some users tend to be similar. It is similar to user based except here we find similar songs by using the column of rating matrix for cosine similarity. User based filtering fails or doesn't perform that well when number of users increase like if number of users are more than number of songs. Although collaborative filtering algorithms are very efficient they have problem. Whenever we come across a new user we don't have any or very little information about his history and hence we won't get good predictions or won't be able to get predictions. This problem is called as cold start problem. As the user listen to some songs history will be generated and now we can easily use collaborative filtering algorithm for that user.

$$P_{u,i} = \overline{r}_u + \frac{\sum_{u' \in N} s(u,u')(\overline{r}_{u',i} - \overline{r}_{u'})}{\sum_{u' \in N} |s(u,u')|}$$

D. KNN Model

When Collaborative filter is applied, we consider contribution of all the users or the items in the user or item based collaborative filters respectively. This reduces the accuracy of the recommender systems. So the author implements a system in which he specifies that the model should consider only the 'n' most closest users or items in their respective models. Here 'n' is specified by the user. This will take into considerations only the the closest n users or items, thereby increasing the accuracy.

V. MODEL

The author has used a hybrid model consisting of both content and collaborative model and even popularity based model.

Let's understand the workflow of our system. First the author splits the data into training and test set. After this splitting, the train data is fed to some learning algorithm like collaborative filtering which learns to make predictions. To evaluate the system the author uses Evaluation metrics in which test data is fed to the

ISSN: 2454-4248
22 – 25

learned algorithm which in return generates prediction ratings. With help of actual user's rating from test set and evaluation metric we check how good our model. Instead of using normal cosine similarity we have used generalized cosine similarity.

Whenever we are generating recommendation for any user u, the norm of u terms remains same while calculating similarity with all the other users. Hence, to emphasize the influence of other user we use this form of cosine similarity.

Now, let's look at our algorithm in which we have combined two or more models by using aggregating method that is combining the end prediction of both the models.

If we take x% recommendation from one model then we have taken 1-x% recommendation from another model. We chose x as 0.5 as it worked better for us and we called it as user item based model.

Algorithm:

input : user id output : list of recommended songs

Get user id for recommendation if user in database:

if no of songs of users > 10 :

apply user based collaborative filtering algorithm to get x% recommendation and item based to get remaining 1-x% recommendation.

else:

apply item based collaborative filtering algorithm to get 80% recommendation apply content based algorithm to recommend remaining

20% recommendation.

else:

add user in database

use popularity based model to get recommendation

VI. EVALUATION METRICS

Evaluation metric is a tool which helps us to understand how well or how good is our recommendation system.

The most common or simple metric is absolute mean error which measures the deviation of the predicted rating from user's actual rating and then averaging over all over all predicted ratings.

The author has used precision and recall which are classical metrics for binary classifiers along with MAE.

But we have ratings in range 1-5, so we need find way to convert this numerical problem into binary problem. We have done this by forming two classes - relevant and not relevant by dividing or splitting the rating into two parts. For ratings equal to 3 and greater, we have assumed it's relevant else it's not relevant. We have used mean average precision in order for evaluating our model. In this metric, first we get top K predictions from our learning algorithm. Then at each rank k,

if the song is relevant we have calculated precision at that k. Let y be the list of predictions given by your system such that y(j)=i means ith song is at rank j in our prediction list.

We have to average all values of precision for each rank k.

$$AP(u,y) = \sum_{i=1}^{k} P_1 \cdot (u,y)$$

Finally we have to average over all the users.

$$mAP = \frac{\sum_{u=1}^{m} AP(u,y)}{m}$$

VII. RESULTS

The author has used mean average precision to test our model as mean absolute error doesn't test our model that efficiently and is superficial. No surprises that popularity based performed the worst as it's not personal with least mAP. User-item model performed best again with highest precision as it combined features of both model. User and item based model performed extremely and were not that far from userite m based model. Item performed better than user due to scalability issues as number of users were more than number of songs.

Sr no	Model	Mean Average Precisio n
1.	Popularity	0.0005451
2.	User based	0.0024897
3.	Item based	0.0067313
4.	User-item based	0.0070884

The author has also used mean average precision to test our model as mean absolute error doesn't test our model that efficiently and is superficial. No surprises that popularity based performed the worst as it's not personal with least mAP. User-item model performed best again with highest precision as it combined features of both model. User and item based model performed extremely and were not that far from userite m based model. Item performed better than user due to scalability issues as number of users were more than number of songs.

VIII. CONCLUSION

We have implemented Popularity based model, Singular Value Decomposition, KNN Algorithm(Content based filtering), user based and item based collaborative filtering algorithms. Popularity based model performed the worst with no doubt as it had no personalization. KNN helped us for content based filtering but the only drawback was no variations in recommendations and it becomes very monotonous.

ISSN: 2454-4248
22 – 25

User based and item based model performed well for us with item based model performing the best as in user based model we faced the scalability problem in which number of users were more than number of songs. SVD too did well although the matrix was too sparse to converge the objective function to global minimum. Hybrid approach outperformed all models as we combined user and item based filtering and helped us to increase precision with some level.".

IX. FUTURE WORK

- Memory models algorithms – collaborative filtering algorithms are very time expensive and hence we can use Hadoop to parallelize the computation.
- To combine the user and item based model by using linear combination and learning the weights.
- gorithms for different aspects like user's mood , current time and day and so on.
- Use Deep learning to process the audio files in order to procure features of songs for recommendation.

X. REFERENCES

[1] Sophia Ciocca, How Does Spotify Know You So Well?,https://medium.com/s/story/spotify-discoverweekly-how- machine-learning-finds-your-new-music19a41ab76efe

[2] Million song dataset triplet file 'https://static.turi.com/datasets/millionsong/10000.txt'

[3] David R. Cheriton School of Computer Science University of Waterloo Waterloo, ON, Canada "The Use of Machine Learning Algorithms in Recommender Systems: A Systematic Review" [4] IIT, Kanpur, India "Music Recommendor Sytem"

[5] Daniar Asanov Berlin Institute of Technology Berlin, Germany, "Algorithms and Methods in Recommender Systems

[6] Z. Sun and N. Luo, "A new user-based collaborative filtering algorithm combining data-distribution," in Proceedings of the 2010 International Conference of Information Science and Management Engineering Volume 02, ser. ISME '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 19–23.