Song Recommendation

Shima Azizza dehRoodpish (U00500360)

Department of Computer Science, University of Memphis, Memphis, TN, szzzdhrd@memphis.edu

Semester Project Report for Data Mining, Professor Xiaofei Zhang Instructing, Spring 2020

ABSTRACT

In this project, I have implemented and analysed two song recommendation systems to predict a user's interest rating of unheard songs. In the following report, I will introduce the problem of song recommendation, discuss the available methods, and present the selected methods including popularity-base and collaborative-base plus, their corresponding results. I got my dataset from Million Song Dataset [1] as a subset consisting of 10,000 songs which the website provided as a random selection as 1% of their data. My analysis shows that the best result for the data comes from the collaborative filtering algorithm.

Keywords- Recommendation Systems, Song, Popularity Base, Content Base, Collaborative Base.

1 INTRODUCTION

Rapid development of mobile devices and internet provides access to different music resources freely. There are more songs than any individual can make time for it. It sometimes feels difficult to choose from millions of songs. In addition, there are many music streaming services providers who need an efficient way to manage songs and help their costumers to discover music by quality recommendation. Thus, there is a strong thriving market for good recommendation systems [2].

Music was one of the first application fields of collaborative filtering (CF) recommender systems. The Ringo system [5] was designed to recommend albums and musical artists to users, initially as an email-based service. It was based on explicit rating information provided by users for artists to construct preference profiles. Recommendations in that system were then made based on a user-to-user or item-to-item neighbourhood scheme and sent via email to Ringo's users. Today, the social web has led to new dimensions of social information filtering as well as new ways to listen to music and discover new artists or albums. As a result, almost all of today's major online music services, provide some form of music recommendation functionality [4].

A machine learning recommendation system (ML recommender) considers a user's past activity along with a large library of activity from many other users. Various algorithms are available to try to build an effective recommender system. Some of them will be explained in the next sections.

In this project, I want to build a song recommendation system to predict a user's interest rating of unheard songs. The goal here is to recommend a song to a user given his/her listening history and other user's history. I will try different systems and make a comparison between them. As it is mentioned before, this kind of recommendation system can become useful in many different applications, such as social media, music and video players and stores.

2 DATASET

Initially I need to gather my data and create my data table. I got my dataset from Million Song Dataset [1]. This website is a collection of audio features and metadata for a million contemporary popular music tracks, freely available, with the purpose of encouraging research algorithms and providing reference dataset to evaluate those algorithms. My downloaded data include a subset consisting of 10,000 songs and 10000 users, which the website provided as a random selection as 1% of their data.

The Million Song dataset is released by Columbia University Laboratory for the Recognition and Organization of Speech and Audio and contains around 48 million triplets collected from histories of over one million users and metadata (280 GB) of millions of songs.

3 RECOMMENDATION SYSTEM

In this section, I describe several algorithms designed for tackle song recommendation task. The simplistic approach is the popularity-based approach which is technically just finding the highest rating songs without personalization. It can still bring out useful information from dataset. In addition, there are 3 types of recommendation system: content-based systems, collaborative filtering systems (sometimes known as similarity-based system), and hybrid systems.

3.1 Popularity-Based System

This model is used to recommend songs which are popular or say, trending in data. Basically, this model works based by only the songs which are popular or listened by almost every user in the system. It is the most basic and simple algorithm. The popularity of each song is determined by looking into the training set and calculating the number of users who had listened to this song. Songs are then sorted in the descending order of their popularity. For each user, the topmost popular songs except those already in his/her profile is recommended. This method involves no personalization and some songs may never be listened in future.

3.2 Content-Based System

Content-based filtering systems make recommendations based on the characteristics of the items themselves. Due to the sparsity of readily available user feedback data, music recommendation techniques tend to rely more upon content descriptions of items than techniques in other domains. Content-based music recommendation techniques are strongly tied to the broader field of

music information retrieval (MIR), which aims at extracting semantic information from or about music at different representation levels (e.g., the audio signal, artist or song name, album cover, or score sheet). Content information includes any information describing music items that can be extracted from the audio signal, as well as metadata provided by external sources (e.g., web documents, discography data, or tags). In this section, we overview research on content-based approaches to music recommendation and categorize the existing approaches with respect to the employed information sources [6].

3.3 Collaborative Filtering Systems

Collaborative filtering systems make recommendations based on other user interactions. They use a database about user preferences to predict additional topics or products a new user might like. The task in collaborative filtering is to predict the utility of items to a particular user (the active user) based on a database of user votes from a sample or population of other users (the user database) [3].

Collaborative filtering involves collecting information from many users and then making predictions based on some similarity measures between users and between items. That is why it sometime is called similarity-based model. In general, they can either be user based or item based (figure 1). User based collaborating filtering uses the patterns of users similar to target user to recommend a product. Item based collaborative filtering uses the patterns of users who browsed the same item as target user to recommend a product [8].

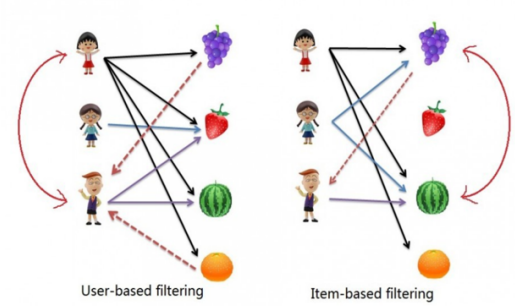

Figure 1: *User-based vs Item-based [1].*

3.4 Hybrid Music Recommendation

Since music preference is a complex and multi-faceted concept, it is a logical step to incorporate multiple aspects of musical similarity into recommendation. Hybrid music recommenders are systems that "combine two or more recommendation techniques to gain better performance with fewer of the drawbacks of any individual one" [7].

4 METHODOLOGY

4.1 Software

I am using python in Jupyter notebook for coding purpose of the project. The Jupyter Notebook is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

4.2 Data Processing

As my data, the file song_data.csv is a spreadsheet including songs features (song id, title, release, artist name and year). The other file named as history.txt, has 3 columns with 1450934 rows (user ID, song id, number of repeat). Triplets data contain:

1: User ID (anonymous, so no information about their demography and timestamps of listening events);

2: Song ID;

3. Play count (instead of rating, the feedback is implicit the number of repeat) [1].

Note that the all the songs in the original list are not necessarily appearing in the triples file. The first part of the work is to load the data, clean it, merge it and remove repetitions and combine them together. The length of the dataset is now 1450932.

[9]:

	user_id	song_id	listen_count	title	release	artist_name	year
0	fd50c4007b68a3737fe052d5a4f78ce8aa117f3d	SOEGIYH12A6D4FC0E3	1	Horn Concerto No. 4 in E flat K495: II. Romanc...	Mozart - Eine kleine Nachtmusik	Barry Tuckwell/Academy of St Martin-in-the-Fie...	0
1	fd50c4007b68a3737fe052d5a4f78ce8aa117f3d	SOFLJQZ12A6D4FADA6	1	Tive Sim	Nova Bis-Cartola	Cartola	1974
2	fd50c4007b68a3737fe052d5a4f78ce8aa117f3d	SOHTKMO12AB01843B0	1	Catch You Baby (Steve Pitron & Max Sanna Radio...	Catch You Baby	Lonnie Gordon	0

Figure 2: *Merged Database*

4.3 Algorithms

I applied popularity-based model and collaborative-based model on data. Here, I will describe the details of implementation.

The first method used is popularity-based method:

The song and artist name are merged into one column, aggregated by number of repeats in general by all users. The code group the song by number of repeats ascending. It calculates the group sum next. Then it calculates this percentage by dividing the repeats by the total sum and then multiply by 100. The ascending order of will goes from most popularity to less popularity. I can also get the number of unique users (110000) and songs (163206). The common practice of dividing data to 80% training and testing is followed. The recommender class [9] is used for training the model. Based on the popularity of each song, the code creates a recommender that accept a user id as input and out a list of recommended song of that user.

```
Training data songs suggested for the user:
91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62
--------------------------------------------------------------
Love Letters - Dario Marianelli
Pastorale - Secret Garden
In The Waiting Line - Zero 7
Until The Morning - Thievery Corporation
The Richest Man In Babylon - Thievery Corporation
Un Simple Histoire - Thievery Corporation
Big Yellow Taxi - Counting Crows / Vanessa Carlton
Kryptonite - 3 Doors Down
Passacaglia - Secret Garden
Illumination - Secret Garden
...
```

Figure 3: *Recommended songs for a specific user by popularity-based method. Only the few top recommendations are presented.*

The second method is collaborative method:

Here I build a song recommender with personalization. It is item based collaborative filtering model that allows personalized recommendations to each user. Item-item filtering approach involves defining a co-occurrence matrix based on a song a user likes. The goal is to answer a question, for each song, what a

number of time a user, who have listened to that song, will also listen to another set of other songs. To implement that, first I create an instance item similarity-based recommender class and feed it with the training data. The function used is item_similarity_recommender. I am using similar training and testing data as before. Thea considered features are song id and user id. I can get the recommended songs for a specific user id. I can also get a similarity for a given id song.

```
Training data songs for the user userid:
91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62
----------------------------------------------------------------------------------------------------
Love Letters - Dario Marianelli
Pastorale - Secret Garden
In The Waiting Line - Zero 7
Until The Morning - Thievery Corporation
The Richest Man In Babylon - Thievery Corporation
Un Simple Histoire - Thievery Corporation
Big Yellow Taxi - Counting Crows / Vanessa Carlton
Kryptonite - 3 Doors Down
Passacaglia - Secret Garden
Illumination - Secret Garden
...
No. of unique songs for the user: 10
no. of unique songs in the training set: 6070
Non zero values in cooccurence_matrix :216
```

	user_id	song	score	rank
0	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Banzai Pipeline - Henry Mancini & His Orchestra	0.050000	1
1	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Your Red Dress (Wedding Song At Cemetery) - Al...	0.050000	2
2	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Je Suis Malade - Lara Fabian	0.050000	3
3	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Stay Forever - Ween	0.050000	4
4	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Spoken Introduction: Timber - Odetta	0.050000	5
5	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Neighborhoods - Matthew Dear	0.050000	6
6	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Movin' On (2006) - Damu The Fudgemunk	0.033333	7
7	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Summer Long - Kathleen Edwards	0.033333	8
8	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	To Have And To Hold - Deftones	0.033333	9
9	91b8fac7dc5e03f6cfaf6e2aa7171f14a8354d62	Start Here (2007) - Damu The Fudgemunk	0.033333	10

Figure 4: *Recommended songs by collaborative filtering method for a specific user id. Only the few top recommendations are presented.*

The applied recommender system calculated a weighted average of the scores in cooccurence matrix for all user song. It's worth to note that this method is not Deep Learning but purely based on linear algebra. This cooccurence matrix will tend to be sparse matrix because it's not possible to predict if a user like a particular song, whether or not he/she will like a million other song. The method is based on some similarity measure to compare between two songs or between two users. Cosine similarity weighs each of the users equally which is usually not the case. User should be weighted less if he has shown interests to many varieties of items. Likewise, user is weighted more if listens to very limited set of songs [9].

5 RESULTS AND DISCUSSION

The Analysis are done for both similarity base model and collaborative filtering model. To see how well the models performed, I need some kind of evaluation for the models. The evaluation method selected here is using precision-recall graph. And to quantitatively measure the performance of these two systems, I use two different metrics: Precision (Eq.1) and Recall (Eq.2). F-1 Score (Eq.3) can also be calculated from these parameters if needed.

$$Precision = \frac{TP}{TP + FP} \qquad (Eq.1)$$

$$Recall = \frac{TP}{TP + FN} \qquad (Eq.2)$$

$$F1\ Score = \frac{2.Precision.Recall}{Precision + Recall} \qquad (Eq.3)$$

Where, TP means true positive, FP is false positive, and FN is false negative. According to Marcel Caraciolo, *Precision* is "the proportion of top results that are relevant, considering some definition of relevant for your problem domain". In this project, *precision* seeks to measure the relevancy of songs in relation to the top ten results of recommended song, whereas *recall* seeks to measure the relevancy of songs in relation to all the songs [9]. The curve is plotted using precision_recall_calculator library.

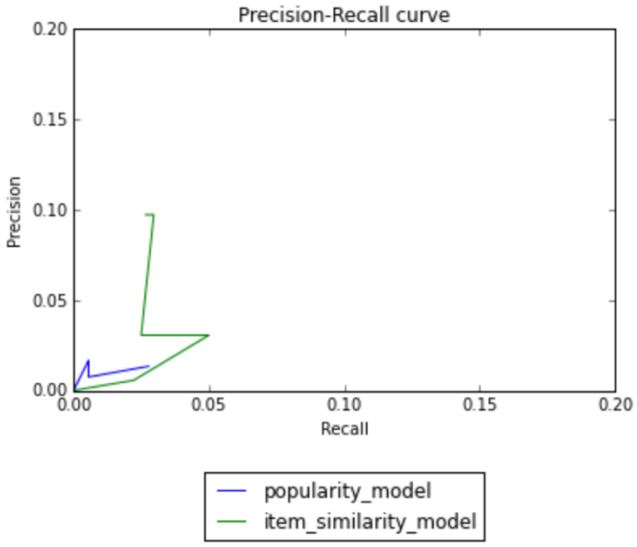

Figure 5: *Precision-Recall graph [9]*

Observing the precision recall curve of both popularity-based model and item similarity model, shows that item similarity model has higher recall and precision values, so it performs betters up to certain point in precision-recall curve.

6 CONCLUSIONS

There are many different approaches to the song recommendation problem, of which I have tried popularity-based and collaborative-based method in this project and concluded that collaborative method gives more accurate result. Here I have the features of only 10,000 songs, which is about 1% of the whole dataset so only some of these 10,000 songs could be recommended. The huge lack of information leads to the bad performance for some methods, such as popularity-based method here. Also the popular method is not personalized as collaborative method, so it will leads to less accurate results.

Building a recommender system is not a trivial task. The fact that we have to deal with large scale dataset makes it difficult in many aspects. Technically we will need more recourses in memory an CPU. It is certainly not easy to get a high precision as it can be inferred from the results. It is also difficult to extract relevant features. But that is what make this task challenging and interesting for future works.

7 FUTURE WORK

Music Recommender System is a wide and complicated subject that can take some initiatives and do a lot more tests in future. Adding more dataset might be helpful in reaching to better results.

Also making improvement on data for example using SVD model to reduce some features and make some improvement on data can be helpful. I can also try other models like KNN or combine different models as hybrid.

ACKNOWLEDGMENTS

I like to thank professor Zhang for encouraging us to do a practical project for the data mining course. I also appreciate the MillionSongs website for making their data available to public for research work.

REFERENCES

[1] http://millionsongdataset.com/, April 1st, Mahiux Ellis, http://labrosa.ee.columbia.edu/millionsong/tasteprofile

[2] Garg, S. and Sun F. 2014, Music Recommender System, Indian Institute of technology.

[3] Breese, J.S., Heckerman, D. and Kadie, C., 2013. Empirical analysis of predictive algorithms for collaborative filtering. arXiv preprint arXiv:1301.7363.

[4] McFee, B., Bertin-Mahieux, T., Ellis, D.P. and Lanckriet, G.R., 2012, April. The million song dataset challenge. In Proceedings of the 21st International Conference on World Wide Web (pp. 909-916).

[5] Shardanand, U. and Maes, P., 1995, May. Social information filtering: algorithms for automating "word of mouth". In Proceedings of the SIGCHI conference on Human factors in computing systems (pp. 210-217).

[6] Schedl, M., Knees, P., McFee, B., Bogdanov, D. and Kaminskas, M., 2015. Music recommender systems. In Recommender systems handbook (pp. 453-492). Springer, Boston, MA.

[7] Burke, R., 2002. Hybrid recommender systems: Survey and experiments. User modeling and user-adapted interaction, 12(4), pp.331-370.

[8] https://dev.to/nickymarino/how-to-build-a-song-recommender-using-create-ml-mlrecommender-45h1, April 1st.

[9] https://towardsdatascience.com/how-to-build-a-simple-song-recommender-296fcbc8c85, April 1st

[10] https://medium.com/@cfpinela/recommender-systems-user-based-and-item-based-collaborative-filtering-5d5f375a127f, April 24th.