A Movie Recommendation System

Background

In our weekends, almost everyone likes watching movies with family and friends to spend leisure time. We may all have experienced this: wanted to watch a movie in the next two hours, but sat on the couch for 20 minutes did not know what to see, choice phobia disorder happened again, the good mood has become frustrated. In this time, we need a computer agent, to provide recommendations for the selection of movies. Therefore, Now, movie intelligent recommendation system has become a part of everyday life. According to recent report, experts familiar with the matter estimate that the recommendation system will generate as much as 10% to 25% revenue growth for large E-commerce platforms like Amazon and Netflix. According to Amatriain (2013), "Recommender Systems are a prime example of the main- stream applicability of large scale data mining."(p.1)

Recommendation Method

Roughly speaking, there are two types of recommendation systems except for a simple rating method. The first one is Content-based Recommendation. According to Gomez, Carlos, and Neil (2016), Historically, the Netflix recommendation problem has been thought of as equivalent to the problem of predicting the number of stars that a person would rate a video after watching it, on a scale from 1 to 5. "(p. 2).It is a regression problem, and we treat movie content as a feature to predicts users' ratings of movies. For example, we give 5 features action, comedy, adventure, music, and romance to each movie as table 1.If the movie has the same rate at features, they are treated as similar movies. For instance, Titanic and Roman Holiday can be considered as similar because of the same rating on all features.

Figure 1.

Content-based Recommendation

Movie	Action	Comedy	Adventure	Music	Romance
Titanic	1	1	1	1	5
Roman Holiday	1	1	1	1	5
Transformers	5	1	1	1	1
Avatar	3	1	5	1	1

User	Titanic	Roman Holiday	Transformers	Avatar
Tom		5		
Kun			3	
Jerry	5	5		
Lily				4

When the user has same interesting in different movies, similar movies will be provided as a recommendation. As the result in table 2, Tom and Jerry give the same scores on Roman

Holiday, so it can be assumed that they also give same rate on Titanic as the system treat these two users have same interesting.

Another widely used method is collaborative filtering. In the a collaborative filtering recommendation system, movies content characteristics are generally not obtained in advance. According to Sarwar, Karypis, Konstan and Riedl (2001), “The goal of a collaborative filtering algorithm is to suggest new items or to predict the utility of a certain item for a particular user based on the users’ previous likings. “(p. 287). It is learned by potential characteristics from the similarity between users (users give the same rating with one movie) and the similarity between movies (movies with similar user ratings). By doing so, the system can predict the user's rating of the movie.

For example in table 3, Tome and Jerry have the same ratings on Roman Holiday and Transformers, so it can be predicted that they Tom may have similar rates on Titanic and Avatar as Jerry.

Figure 2.

Collaborative filtering recommendation

User	Titanic	Roman Holiday	Transformers	Avatar
Tom		5	2	
Kun	1	1	3	5
Jerry	5	5	2	1
Lily	4	3	2	4

User	Titanic	Roman Holiday	Transformers	Avatar
Tom	5	5	2	1
Kun	1	1	3	5
Jerry	5	5	2	1
Lily	4	3	2	4

In addition, after studying the characteristics of the movie, the system measures the similarity between the movies and recommend the most similar movie to him / her based on the historical viewing information of the user.

In this paper, we will use collaborative flirting to predict in the major part of the recommendation system, but we also use contend-based thought to deal with the first time user. In addition, we can also combine "content recommendations" with "collaborative filtering," using content as side information to improve predictive accuracy. This hybrid approach can be implemented using the "Low-rank matrix factorization" algorithm.

System Design Process

We use the movie dataset from MovieLens website in this project which includes 1000,000 ratings and 1300 tags, 9000 movies rated by 700 users. We can use two matrices to represent the user interesting rating and movies appeal ratings. The product of this matrix can be the rate of the movies. We use python and using numpy, scipy, pandas to create the program.

First, we separate these data into two parts, 70% training data and 30% testing data, we use a matrix to represent user rating. Normalize the rating to avoid some user with all blank data. Using factorization algorithm to find movies; latent features. The predicted rating can be obtained by multiplying the matrix of User and matrix of Movie.

Figure 3.

LRMF Algorithm

Input data M and U,
Pick random matrix for M and U
  cost = KnownMatrix - U * M
x∈ U, y∈ M
Repeat
  x+Δ, y+Δ check the cost
Until by to minimize the cost
  PredMatrix = U * M

To predict the movie rating matrix, the low-rank matrix factorization algorithm will be used instead of SVD, because there are many missing data in this sparse matrix. According to Koren, Bell, and Volinsky (2009), "Matrix factorization models map both users and items to a joint latent factor space of dimensionality f, such that user-item interactions are modeled as inner products in that space. "(p. 44). Matrix factorization broke one large matrix down into smaller matrices. We set matrix U and M randomly and calculate the cost of real matrix value and this random matrix. By iteration, until the cost is close to zero, we get the predict matrix.

The approach to finding the similar movies, first, we load features data from products feature data produced by training dataset. Then we load movie titles and choose one to find the similar movies to it. Next, we utilize matrix factorization to obtain the features of choosed movie. The primary method to find the similar movies is to compare the movie features difference which is the sum of several features absolute difference between the target movie and our movie database. By sorting the different scores, we can find the top similar movies to our target movie.

Root mean square error will be used to measure the accuracy of the prediction result and real data in the testing dataset. First, we load data both from the raw testing and training dataset. Using matrices to represent user rating and movie rating. In real data, NaN will signify the missing value. The RMSE can be got by calculating the root mean square error between the raw training data and testing data and predicted matrix which obtained by low rank matrix factorization algorithm before.

As there are no historical data for the first-time user, normally there are three methods to deal with the first-time user: no recommendation, the highest average rating movie, recommend similar movies based on the first time click. We choose to recommend highest rating movie in this project, it is highly possible the random user also like the most popular movie in the database, so we calculate the average rating of each movie and offer the number 1 from sorted data to the first-time user.

Program Result

First-time User Case

First, for the first-time user, as mentioned above, we will use provide the highest rating move to the user without historical data. The function display as below. When we choose how many recommended movie we want to obtain, the result is as below, there are top 5 rating movies that are provided to this first-time user.

Figure 4

Solution result to first-time user

movie_id	title	genre	rating
6	Attack on Earth 1	sci-fi, action	4.900000
10	Surrounded by Zombies 1	horror, zombie fiction	4.882353
3	The Sheriff 2	crime drama, western	4.818182
12	Horrorfest	horror	4.800000
5	The Big City Judge 2	legal drama	4.785714

Similar Movie Finding

Before we get the recommendation result, we also want to have a sub-function to find the similar movies indecently, after input the move id and numbers of similar movie we want, the result is as below when we input movie id 19 and 3 similar movies we want to find. Difference score 0 means the movie is itself.

Figure 5.

Similar movies finding result

The 3 similar movies are as below:

movie_id	title	difference_score
19	Fake News about Fake News	0.000000
10	Surrounded by Zombies 1	3.271244
9	Biker Gangs	3.332729
26	Mafia Underground	3.529711

Recommendation Result

When we select the user ID and the number of movies we want to get as a recommendation, there are most 5 similar movies to move 77 as Fig 7 as below.

Figure 6.

Recommendation result to certain user

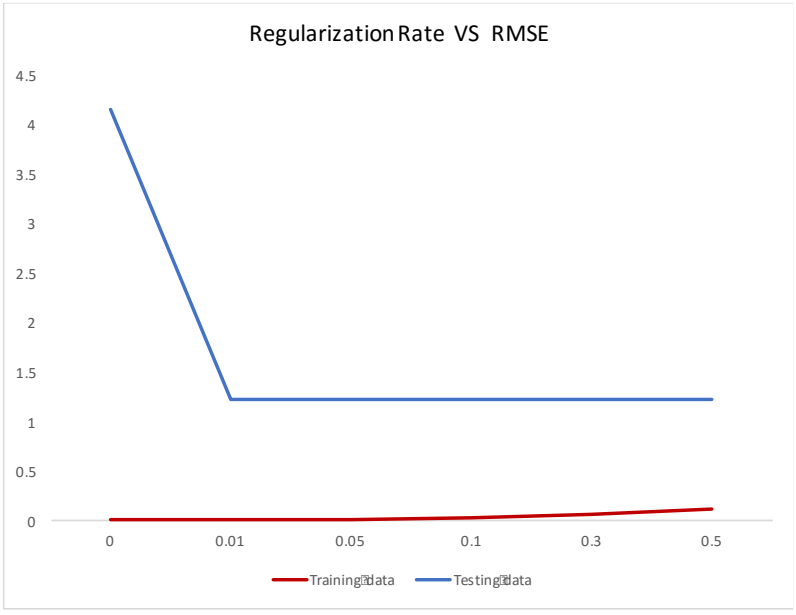
```
Select a User:
77
These Movies are recommended:
                          title                         genre    rating
movie_id
16               Master Criminal  thriller, horror, crime drama  5.213842
5           The Big City Judge 2                   legal drama  5.095540
32             Behind the Scenes                 comedy-drama  4.955643
26             Mafia Underground         crime drama, thriller  4.940760
10       Surrounded by Zombies 1        horror, zombie fiction  4.937442
```

Statistic Result

As we need to set the regularization to avoid the result overfitting which means the model focus on some certain pattern too much instead of the overall pattern. As the iteration times increases, the RMSE decrease to a stable level. Based on the fig 8, we can set the regularization rate to 0.01 to remain the RMSE to 1.21 of testing data and 0.02 of training data.

Figure 7

Regularization Rate VS RMSE

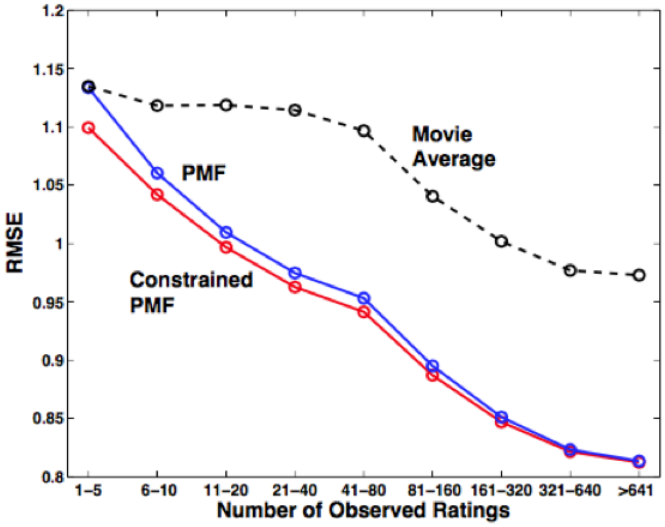


According to Mnih and Salakhutdinov (2008), Fig 8 shows the RMSE of PMF algorithm and Netflix baseline. Compared with these two accuracy ratings, we can see the RMSE of our

algorithm is relatively low and performance is accepted within a reasonable range. The accuracy of the result is also related to the dataset scale and data quality.

Figure 8.

The accuracy compared with previous work



Method	RMSE
My Recommendation RMSE	1.21
Netflix Baseline	0.95
Constrained PMF	1.05

Conclusion

Content-Based Referral and Collaborative Filtering were state-of-the-art technologies over a decade ago. Obviously, there are many models and algorithms that can improve the prediction. For example, implicit matrix factorization can be used to replace user movie scoring with preference and confidence. In addition, clicks numbers of a movie also can assist collaborative filtering in the absence of prior user movie rating information.

Future work should user interface, dataset and API parts for this recommendation system. The database also can be improved by using MongDB or Hadoop. The algorithm, python packages which used in the project also can be updated to TensorFlow to improve the accuracy.

Reference

Amatriain, X. (2013, August). Big & personal: data and models behind netflix recommendations.

In *Proceedings of the 2nd international workshop on big data, streams and heterogeneous source Mining: Algorithms, systems, programming models and applications (pp. 1-6).* ACM.

Gomez-Uribe, C. A., & Hunt, N. (2016). The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, *6*(4), 13.

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001, April). Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web* (pp. 285-295). ACM.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42(8).

Mnih, A., & Salakhutdinov, R. R. (2008). Probabilistic matrix factorization. In *Advances in neural Information processing systems* (pp. 1257-1264).