Movie Rating Prediction

Kangjun Lou
San Jose State University
SID: 013736593

Mandy Wong
San Jose State University
SID: 014558570

Ping Chen
San Jose State University
SID: 013855049

Xiaoming Liu
San Jose State University
SID: 013763646

***Abstract*—Movie rating presents an objective basis for general audiences, cinemas, and advertisers to choose high-quality movies and is also an essential factor to estimate final box office receipts. As a result, predicting the rating of a film is valuable to audiences, cinema, advertisers, and investors. This project is intended to predict the ranks of new movies based on current movie rating information such as MovieLens and IMDb datasets. This project mainly uses content-based prediction and collaborative filtering prediction to predict movie ratings and evaluate the results. For the Content-based method, there are multiple factors chosen as the attributes of the movie to indicate movie profiles. For the collaborative filtering method, multiple ways were used to implement it. Additionally, this project will compare and analyze the results of different prediction methods.**

***Index Terms*—movie rating prediction, content-based, collaborative filtering, MovieLens, IMDb**

I. Introduction

Nowadays, With the connection of the world, the film resources available to people are getting richer, and audiences can easily share their ratings about a movie on the Internet. For ordinary viewers, the increase in options also means that the difficulty of choosing the right movie increases. In the increasingly busy modern society, the time cost of choosing the wrong movie becomes more expensive. For movie theaters, it is a challenge to arrange appropriate schedules for many movies to maximize revenue. For advertisers, whether they can choose the right film directly determines whether their investment can be rewarded. Therefore, accurately predicting the ratings of movies is of considerable significance to audiences, cinemas, and advertisers. Besides, for online video service providers, it is also one of the critical functions to recommend proper movies to customers' preferences. This is the motivation of our project.

The first problem this project wants to solve is how to predict the unknown rating based on the existing movie rating data. This includes comprehensive ratings for movies provided to cinemas and advertisers, as well as targeted ratings for ordinary audiences. The second problem this project wants to solve is how to deal with the cold start problem. For newly released movies, there is no user data to make predictions. But the prediction of new movie ratings is essential for movie theaters and advertisers. Then it is about movie recommendation, especially for online video service providers, such as Netflix, it is important to accurately recommend favorite movies to viewers. Additionally, the movie rating data set used in this project comes from multiple databases, which have different content and format, and there are duplicates and invalid items. Thus, how to merge and clean up different datasets and select the required information from them is also a problem to be solved in this project.

The main methods used in this project are content-based prediction and collaborative filtering prediction. The content-based approach focuses on the properties of items (movies in this project). Similarity of movies is determined by measuring the similarity in their properties. This project uses two factors: the specific types of movies such as drama, action, and comedy; the crew behind movies such as director, actor, and producer, to construct each movie a profile. This project mainly uses this method to solve the cold start problem. The collaborative filtering approach focuses on the relationship between users and items. Similarity of movies is determined by the similarity of the ratings of those movies by users who have rated both movies. This project uses four kinds of collaborative filtering: baseline only, user-user, item-item, and latent factors (SVD). This project uses collaborative filtering as the main prediction method.

II. Literature Review

Rating predictions are closely related to issues like recommending products to users. Extensive research has been carried out on such problems, and three basic prediction methods have been derived: content-based methods, collaborative methods, and hybrid methods [1].

Content-based methods predict user ratings using the item's features and the user's affinities, while collaborative methods predict ratings using the ratings of other similar users. Hybrid methods combine the two approaches mentioned above. These approaches can additionally be subdivided into heuristic and model-based approaches [1].

The content-based method relies on the item's features and needs to conduct a profile for each item. This feature of this method makes it suitable to apply to object-oriented models. People have created a rating prediction system based on decision trees. For each item, this system will build a separate decision tree based on the item features [3].

The collaborative method relies on similar items' ratings and needs to find the most similar items, which is just like a k nearest neighbor algorithm (K-NN). People have built a distributed system for news filtering based on predicted ratings by using a collaborative heuristic method via the K-NN algorithm [3].

The hybrid method has been used to build up a model that contains user and movie features, as well as all available ratings. This model uses the singular value decomposition method (SVD) and the k-nearest neighbor algorithm (K-NN) to predict ratings, which is called the SVD-kNN method [1].

Neural networks can be used to predict a user's movie preferences by applying both the content-based and collaborative approach. Neural networks using a content-based approach were built in [5] to predict movie ratings and achieved an accuracy rate close to 88

[4] proposed an artificial-intelligence-based approach for movie rating prediction, which used six different parameters and along with their individual ratings to train an artificial neural network (ANN). This approach used the Levenberg-Marquardt backpropagation algorithm to train the neural algorithm. It achieved an accuracy of 97.33% [4], which is higher than the contemporary techniques.

[2] proposed an improved method for movie rating prediction based on Gaussian Mixture Model (GMM). It can give an accurate movie rating while avoiding malicious ratings by decomposing an object into several normal distributions and calculating the final probability.

III. Technical Approach

A. *Content-Based Prediction*

The content-based prediction is used to predict the movie rating based on its genres, actors, directors and writers. Those attributes are keywords for the movie rating prediction. Instead of using TF-IDF to pick the important features, all keywords are considered as necessary in this case.

Before performing the cosine similarity, all keywords were vectorized by the CountVectorizer function from scikit-learn python library. Based on vectorized keywords, cosine similarity is used to calculate the similarity of each movie. Mathematically, it is represented as (1) and Fig. 1:

$$\cos(\mathbf{A},\mathbf{B}) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n}\mathbf{A}_i\mathbf{B}_i}{\sqrt{\sum_{i=1}^{n}(\mathbf{A}_i)^2}\sqrt{\sum_{i=1}^{n}(\mathbf{B}_i)^2}} \tag{1}$$

The predicting function is taking the movie name as the input. Note that keywords of the predicting movie which are the movie types, directors, writers and actors should already be included in the dataset before performing the prediction.

Within the function, the movie name is used to find the index of the movie in datasets. The next step is to locate

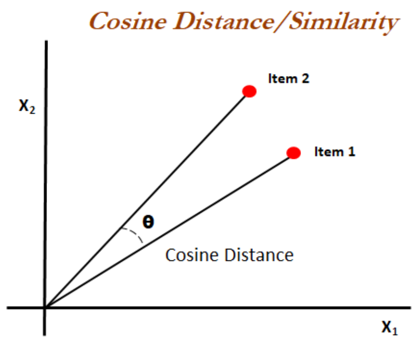


Fig. 1. Cosine Similarity

the predicting movie in cosine similarity matrix and sort the similarity inversely. The top 10 similarities were picked as the recommended movie. The recommendation list includes top 10 similarities with its movie title, number of votes and average ratings.

The predicted score is the mean of the weighted rating. Each movie has a different number of votes and it will affect the average rating badly. Therefore, the weighted rating can balance the rating with regards to the number of votes. Mathematically, it is represented as (2):

$$WeightedRating(wr) = \frac{\mathbf{v}}{(\mathbf{v}+\mathbf{m})} * \mathbf{R} + \frac{\mathbf{m}}{(\mathbf{v}+\mathbf{m})} * \mathbf{C} \tag{2}$$

where,

- v is the number of votes of the movie
- m is the required minimum number of votes that list in recommendation list
- R is the average rating of the movie
- C is the mean vote of whole dataset

B. *Collaborative Filtering Prediction*

In this project, the collaborative filtering is also performed to predict ratings based on similar users or similar items. Four models were used in collaborative filtering: Baseline Estimation; user-user collaborative filtering; item-item collaborative filtering; singular value Decomposition (SVD).

1) Baseline estimation: This model predicts the baseline estimate for given user and item. The baseline will use the overall mean rating, deviation of user and deviation of item to estimate the baseline:

$$\hat{r_{ui}} = b_{ui} = \mu + b_u + b_i \tag{3}$$

where,

- u is overall mean rating
- b_u is average rating of user x minus u
- b_i is average rating of movie i minus u

Below shows the baseline optimization function that try to minimize b_u , b_i. The following regularized squared error:

$$\sum_{r_{ui} \in R_{train}} \left(r_{ui} - (\mu + b_u + b_i)\right)^2 + \lambda \left(b_u^2 + b_i^2\right) \tag{4}$$

2) User-user collaborative filtering: This model will define the similarity sim(u,v) of user u and v. Then it will select k nearest neighbors $N_i^k(u)$ who rated this movie i. The prediction rating r_{ui} is set as:

$$\hat{r_{ui}} = b_{ui} + \frac{\sum_{v \in N_i^k(u)} sim(u,v) \cdot (r_{vi} - b_{vi})}{\sum_{v \in N_i^k(u)} sim(u,v)} \tag{5}$$

Generally, the similarity will be cosine similarity or Pearson correlation coefficient. The shrunk pearson-baseline correlation coefficient is used here, which uses baseline for centering instead of means. The shrinkage parameter helps to avoid overfitting when only few ratings are available.

3) Item-Item collaborative filtering: The Item-Item CF is similar to User-user CF. It will select k similar items for user u. The prediction rating r_{ui} is set as:

$$\hat{r_{ui}} = b_{ui} + \frac{\sum_{j \in N_u^k(i)} sim(i,j) \cdot (r_{uj} - b_{uj})}{\sum_{v \in N_u^k(i)} sim(i,j)} \tag{6}$$

This model also uses pearson-baseline correlation coefficient to computer similarity.

4) Singular value decomposition: SVD was first introduced into the recommendation system domain by Simon Funk during the Netflix Prize. What this model does is that it decomposes the user-item matrix into vectors to reduce the number of features.

Each item can be represented by a vector q_i. Similarly each user can be represented by a vector p_u . The predicted rating is the dot product of those 2 vectors plus the baseline:

$$\hat{r_{ui}} = \mu + b_u + b_i + q_i^T p_u \tag{7}$$

In order to solve over fitting, the optimization function will introduce regularization to penalize for large values of p_x and q_i:

$$\sum_{r_{ui} \in R_{train}} \left(r_{ui} - \hat{r_{ui}}\right)^2 + \lambda \left(b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2\right) \tag{8}$$

The minimization is performed by Stochastic Gradient descent:

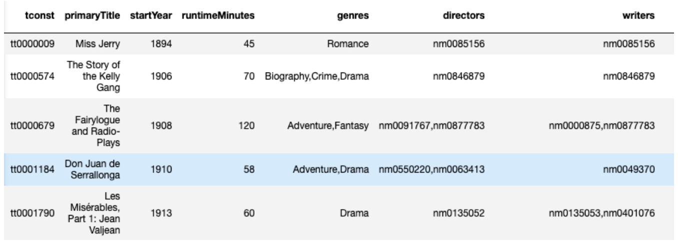
$$\begin{aligned}
b_u &\leftarrow b_u + \gamma \left(e_{ui} - \lambda b_u\right) \\
b_i &\leftarrow b_i + \gamma \left(e_{ui} - \lambda b_i\right) \\
p_u &\leftarrow p_u + \gamma \left(e_{ui} \cdot q_i - \lambda p_u\right) \\
q_i &\leftarrow q_i + \gamma \left(e_{ui} \cdot p_u - \lambda q_i\right)
\end{aligned}$$

where $e_{ui} = r_{ui} - \hat{r}_{ui}$

IV. Simulation and Experimentation

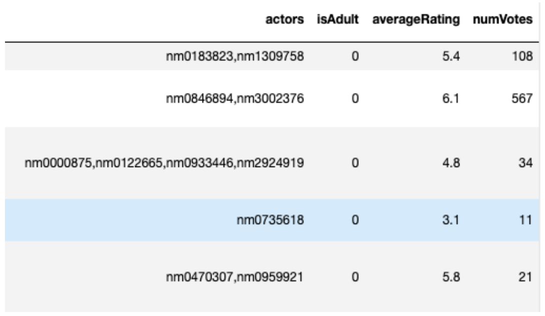
A. Content-Based Prediction

The IMDb datasets are mainly used for content-based prediction. All IMDb datasets are first merged into one dataset with relevant content. All rows that include null or missing value in either genres, directors, writers and actors columns were removed. The top 5 clean data rows are showing in Fig. 2 and Fig. 3:



tconst	primaryTitle	startYear	runtimeMinutes	genres	directors	writers
tt0000009	Miss Jerry	1894	45	Romance	nm0085156	nm0085156
tt0000574	The Story of the Kelly Gang	1906	70	Biography,Crime,Drama	nm0846879	nm0846879
tt0000679	The Fairylogue and Radio-Plays	1908	120	Adventure,Fantasy	nm0091767,nm0877783	nm0000875,nm0877783
tt0001184	Don Juan de Serrallonga	1910	58	Adventure,Drama	nm0550220,nm0063413	nm0049370
tt0001790	Les Misérables, Part 1: Jean Valjean	1913	60	Drama	nm0135052	nm0135053,nm0401076

Fig. 2. Top 5 clean data rows part 1



actors	isAdult	averageRating	numVotes
nm0183823,nm1309758	0	5.4	108
nm0846894,nm3002376	0	6.1	567
nm0000875,nm0122665,nm0933446,nm2924919	0	4.8	34
nm0735618	0	3.1	11
nm0470307,nm0959921	0	5.8	21

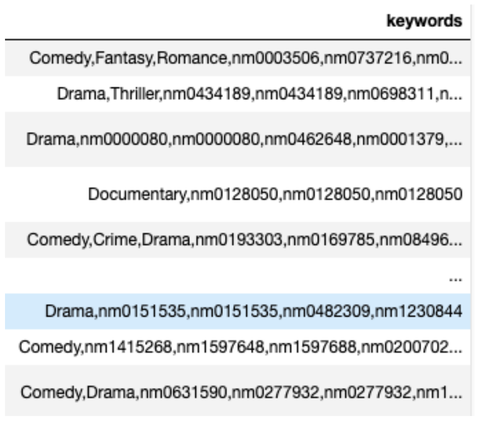
Fig. 3. Top 5 clean data rows part 2

Dataset is prepossessed into different sizes due to the issue of computer power. The detailed is showing in Table 1.

TABLE I
Description of Datasets

Filename	***Size***	***Description***
imdb.csv	(247043, 11)	All movie data from 1894 to 2019 with \N or NaN
imdb_clean_all.csv	(82368, 11)	All movie data from 1894 to 2019 **without** \N or NaN
imdb_clean_1990.csv	(54054, 11)	All movie data from **1990** to 2019 **without** \N or NaN

Before performing the content based prediction, genres,directors, writers and actors columns are all combined into one and store it in the new column called Keywords. The example of keyword column is shown in Fig. 4.

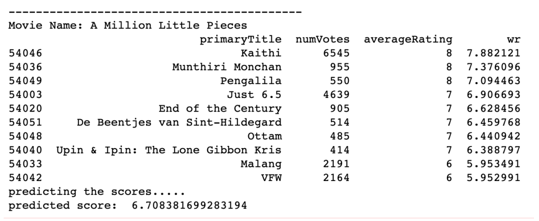


keywords
Comedy,Fantasy,Romance,nm0003506,nm0737216,nm0...
Drama,Thriller,nm0434189,nm0434189,nm0698311,n...
Drama,nm0000080,nm0000080,nm0462648,nm0001379,...
Documentary,nm0128050,nm0128050,nm0128050
Comedy,Crime,Drama,nm0193303,nm0169785,nm08496...
...
Drama,nm0151535,nm0151535,nm0482309,nm1230844
Comedy,nm1415268,nm1597648,nm1597688,nm0200702...
Comedy,Drama,nm0631590,nm0277932,nm0277932,nm1...

Fig. 4. Example of keyword column

At this point, the dataset is ready to use. The function is taking the movie name as the input, and the output is the list of similar movies or the recommended movie either they have similar movie types, same directors/writers or some actors in both movies. Those recommended movies are used to calculate the predicted score. Note that the predicted score is the mean of the weighted rating. The weighted rating is used to balance the average rating with regards to the different number of votes for each movie.

There are two results of content-based prediction shown in Fig.5 and Fig.6.



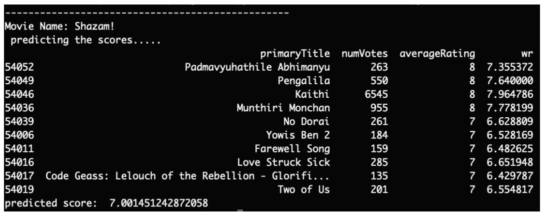
```
--------------------------------------------
Movie Name: A Million Little Pieces
                               primaryTitle  numVotes  averageRating        wr
54046                                Kaithi      6545              8  7.882121
54036                      Munthiri Monchan       955              8  7.376096
54049                             Pengalila       550              8  7.094463
54003                              Just 6.5      4639              7  6.906693
54020                    End of the Century       905              7  6.628456
54051       De Beentjes van Sint-Hildegard       514              7  6.459768
54048                                 Ottam       485              7  6.440942
54040  Upin & Ipin: The Lone Gibbon Kris        414              7  6.388797
54033                                Malang      2191              6  5.953491
54042                                   VFW      2164              6  5.952991
predicting the scores.....
predicted score:  6.708381699283194
```

Fig. 5. result of content-based prediction 1

In this example (Fig. 5), it is predicting the movie called “the million little pieces”, it shows that the IMDb score is 6.2/10 and the predicted score is 6.7 which is a little bit higher.

Here is another example (Fig. 6). It is predicting the movie called “shazam!”, the IMDb score is 7.1/10 and the predicted score is 7 which is closer to the imdb score.

According to the results shown above, TF-IDF is necessary to determine the important features before calculating the cosine similarity. Cosine similarity shouldn't perform for whole datasets, it cannot fit in RAM, and it kills the driver.



```
------------------------------------------------
Movie Name: Shazam!
 predicting the scores.....
                                        primaryTitle  numVotes  averageRating        wr
54052                        Padmavyuhathile Abhimanyu       263              8  7.355372
54049                                        Pengalila       550              8  7.640000
54046                                           Kaithi      6545              8  7.964786
54036                                 Munthiri Monchan       955              8  7.778199
54039                                         No Dorai       261              7  6.628809
54006                                       Yowis Ben 2       184              7  6.528169
54011                                    Farewell Song       159              7  6.482625
54016                                  Love Struck Sick       285              7  6.651948
54017  Code Geass: Lelouch of the Rebellion - Glorifi...       135              7  6.429787
54019                                        Two of Us       201              7  6.554817
predicted score:  7.001451242872058
```

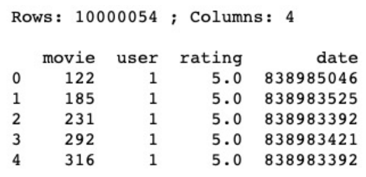
Fig. 6. result of content-based prediction 2

B. Collaborative Filtering Prediction

The movielens 10M dataset is used in collaborative filtering prediction.

1) Data preprocessing: Some data preprocessing are performed like remove missing value and remove duplicated items. Then the whole dataset is split into 80% training set and 20% test set.

2) Data visualization: This data set contains 10000054 ratings and 95580 tags applied to 10681 movies by 71567 users of the online movie recommender service MovieLens (Fig. 7).



```
Rows: 10000054 ; Columns: 4

   movie  user  rating       date
0    122     1     5.0  838985046
1    185     1     5.0  838983525
2    231     1     5.0  838983392
3    292     1     5.0  838983421
4    316     1     5.0  838983392
```

Fig. 7. Top 5 rows of dataset

Below shows the histogram (Fig. 8) of the number of movies under all ratings. The histogram below (Fig. 8) shows that most ratings fall between 3-4.

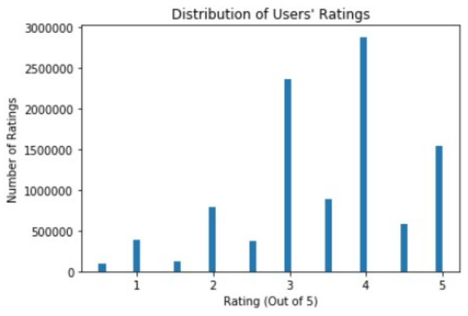


Fig. 8. Distribution of users' ratings

3) Predict movie ratings: Four models are implemented to predict ratings:

- BaseLine Estimation
- User-User Collaborative Filtering: The number of similar users is 20. Use SGD to find parameters.

- Item-Item Collaborative Filtering: The number of similar items is 20. Use SGD to find parameters.
- SVD: The number of latent factors is 100.

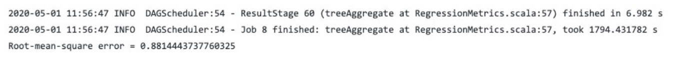
The RMSE and MAE of test data for each model show in the table below. It can be seen that BaseLine estimation got the highest root mean square error. The User-User CF got the results quite similar with Item-Item CF. SVD model works best on training data.

	Model	***RMSE***	***MAE***
0	BaseLine	0.995483	0.786751
1	User-User CF	0.990302	0.783754
2	Item-Item CF	0.990320	0.783772
3	SVD	0.989015	0.787322

The following table shows the randomly picked ratings for comparing the predicted results from each model:

	user	***movie***	***rating***	***baseline***	***knn-user***	***knn-item***	***svd***
0	1883	2301	2	3.303331	2.417889	2.684213	2.265347
1	1567	1299	4	4.193610	4.018281	4.084649	3.976425
2	592	3258	3	3.179302	2.945151	3.122728	3.344425
3	4009	3174	4	3.034792	3.678274	3.454418	3.405793
4	690	954	4	4.837191	4.212733	4.235366	4.303308
5	5	41	4	3.470849	3.619949	4.000806	3.822184
6	615	2571	4	4.502376	4.274269	4.462415	4.625705
7	4152	3363	5	3.998745	4.608908	4.527895	4.568134
8	2330	2160	1	3.170795	1.730769	1.799452	1.597867
9	2432	1721	5	3.631089	4.494586	4.313615	4.416390

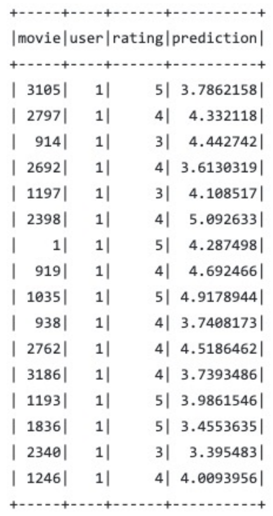
4) Recommendation System using ALS Spark: Another recommendation system is performed by using pyspark in HPC. Spark uses the alternating least squares(ALS) algorithm to learn latent factors. ALS algorithms is the implementation of the matrix factorization models in Netflix Prize competition, this model also allows the incorporation of additional information such as implicit feedback, temporal effects and confidence levels[7]. An ALS model trained in HPC by treating the ratings as explicit. The recommendation model is evaluated by measuring the root mean square error of the rating prediction. The RMSE of this model is 0.88, which is much smaller than the four models implemented above.



```
2020-05-01 11:56:47 INFO  DAGScheduler:54 - ResultStage 60 (treeAggregate at RegressionMetrics.scala:57) finished in 6.982 s
2020-05-01 11:56:47 INFO  DAGScheduler:54 - Job 8 finished: treeAggregate at RegressionMetrics.scala:57, took 1794.431782 s
Root-mean-square error = 0.8814443737760325
```

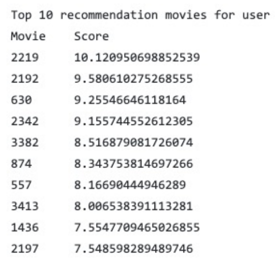
Some prediction examples are shown in Fig. 9. The third column shows the original ratings in the dataset and the last column shows the predicted ratings.

Based on this model, a recommendation system was built. It can recommend 10 movies for each user and 10 users for each movie. The recommendation information are saved in MovieRec.csv and UserRec.csv. For example showing in Fig. 10, it recommends 10 movies for user1 based on the recommendation score.



```
+-----+----+------+----------+
|movie|user|rating|prediction|
+-----+----+------+----------+
| 3105|   1|     5| 3.7862158|
| 2797|   1|     4|  4.332118|
|  914|   1|     3|  4.442742|
| 2692|   1|     4| 3.6130319|
| 1197|   1|     3|  4.108517|
| 2398|   1|     4|  5.092633|
|    1|   1|     5|  4.287498|
|  919|   1|     4|  4.692466|
| 1035|   1|     5| 4.9178944|
|  938|   1|     4| 3.7408173|
| 2762|   1|     4| 4.5186462|
| 3186|   1|     4| 3.7393486|
| 1193|   1|     5| 3.9861546|
| 1836|   1|     5| 3.4553635|
| 2340|   1|     3|  3.395483|
| 1246|   1|     4| 4.0093956|
+-----+----+------+----------+
```

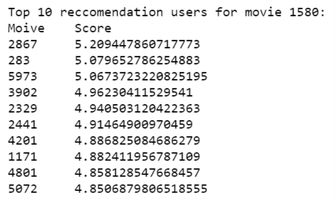
Fig. 9. Examples of prediction



```
Top 10 recommendation movies for user
Movie   Score
2219    10.120950698852539
2192    9.580610275268555
630     9.25546646118164
2342    9.155744552612305
3382    8.516879081726074
874     8.343753814697266
557     8.16690444946289
3413    8.006538391113281
1436    7.5547709465026855
2197    7.548598289489746
```

Fig. 10. Top 10 recommendation movies for user

Another example showing in Fig. 11 is that it recommends 10 users for movie 1580:



```
Top 10 reccomendation users for movie 1580:
Moive   Score
2867    5.209447860717773
283     5.079652786254883
5973    5.0673723220825195
3902    4.96230411529541
2329    4.940503120422363
2441    4.91464900970459
4201    4.886825084686279
1171    4.882411956787109
4801    4.858128547668457
5072    4.8506879806518555
```

Fig. 11. Top 10 recommendation users for movie 1580

V. Conclusion

This project helped us to understand the mechanism behind the moving rating prediction. By combining two methods of content based prediction and collaborative filtering, we compare the prediction accuracy and RMSE of these two methods. We found that content based prediction performed worse than its counterpart, which means it is not the ideal candidate for movie rating prediction. With respect to collaborative filtering, by testing on small and medium size samples, It is proved that if we blend all the models (such as SVD and Knn), with XGBoost, we can further reduce the rmse. We are confident that by adopting this strategy, the final rmse would be lower than 0.8814, though still far away from the threshold set by Nexflix.

Acknowledgment

We would like to thank Dr. Wencen Wu for teaching Large Scale Analytics and providing guidance for our project. We would also like to express our sincere thanks to teaching assistant Chao Pi for answering our questions.

Authors' Contributions

Mandy Wong and Ping Chen designed the model and the computational framework. Xiaoming Liu and Kangjun Lou were responsible for the data collection and preprocessing. Mandy Wong and Kangjun Lou carried out the implementation while Xiaomingliu and Ping Chen performed the testing and provided critical feedback. All team members participated in writing the proposal, presentation slides, manuscript and report. All the code is published on github [8].

References

[1] M. Marović, M. Mihoković, M. Mikša, S. Pribil and A. Tus, ”Automatic movie ratings prediction using machine learning,” 2011 Proceedings of the 34th International Convention MIPRO, Opatija, 2011, pp. 1640-1645.

[2] Jiaxin Zhu, Yijun Guo, Jianjun Hao, Jianfeng Li, and Duo Chen. 2016 2nd IEEE International Conference on Computer and Communications (ICCC). IEEE, 2016. 2016 2nd IEEE International Conference on Computer and Communications (ICCC). Web.

[3] J. Tarvainen, M. Sjöberg, S. Westman, J. Laaksonen and P. Oittinen, ”Content-Based Prediction of Movie Style, Aesthetics, and Affect: Data Set and Baseline Experiments,” in IEEE Transactions on Multimedia, vol. 16, no. 8, pp. 2085-2098, Dec. 2014.

[4] Basu, Somdutta, Kamal, Raj, Henshaw, Michael, and Nair, Pramod S. International Conference on Advanced Computing Networking and Informatics ICANI-2018 /. 1st Ed. 2019. ed. Vol. 870. Singapore :: Springer Singapore :, 2019. International Conference on Advanced Computing Networking and Informatics. Web.

[5] YuMin, Su, Yuan, Zhang, and JinYao, Yan. Proceedings of the 2018 International Conference on Big Data and Computing - ICBDC ’18. New York NY: ACM, 2018. The 2018 International Conference. Web.

[6] Hu, Ting, and Song, Ting. ”Research on XGboost Academic Forecasting and Analysis Modelling.” Journal of Physics. 1324 (2019): 012091. Web.

[7] Koren, Yehuda, Robert Bell, and Chris Volinsky. ”Matrix factorization techniques for recommender systems.” Computer 42.8 (2009): 30-37.

[8] K. Lou, M. Wong, P. Chen and X. Liu. ”Movie rating Prediction.” 2020. Available: https://github.com/wmymandy/Movie_Rating_Prediction.git