CS674 Final Report:
Movie Data Analysis and Movie Recommender

Siyuan Zhong **Kaifei Lei**

Abstract

In this project, we did data analysis and built a movie recommender based on the movielens dataset. We analyzed the dataset on in different ways and also used the machine learning methods collaborating filter and kmeans to recommend movie and friends to users.

1 Introduction

In 2006, Netflix made a contest: If any team could improve the prediction accuracy bar that is 10% better than what Cinematch (which developed by Netflix itself) can do on the same training data set, the best one would be offered a $50,000 Progress Prize. In 2009, a team which contains 7 people won the big prize and Netflix made another contest immediately. It shows us that a great recommend system is such important as well as difficult.

And in this project, we used the dataset from movielens(mov,), which is also a dataset contains lots of movie and user rating data information. At first, we only take the movielens-10k which contains 100,000 ratings from 1000 users on 1700 movies as our data. And we also use the bigger size of dataset movielens-20M which contains 20 million ratings from 138000 users on 27000 movies in our system after we finished small dataset.

We first did the data pre-process on the dataset and then developed such a recommendation system. In our system, we can recommend several movies to the users based on their rating record. What's more, we can recommend some people according to the similarity about preferred movie genres and personal informations. Because of the computer resource limited, we ran the large dataset on Amazon web service EMR. We made use of ALS model of Collaborative filtering and Kmeans cluster on Spark to complete the function which mentioned before.

2 Data Process and Analysis

In this section, we used Apache Zeppeplin as developing tool to deal with data. In addition, we did some visualizations on data.

2.1 Data Content

The dataset we used in this part is movielens 100k data. The key data we used from this dataset are in the follows:

- u.data: The full u data set, 100000 ratings by 943 users on 1682 items. Each user has rated at least 20 movies. Users and items are numbered consecutively from 1. The data is randomly ordered. This is a tab separated list of user id | item id | rating | timestamp.

- u.item: Information about the items (movies); this is a tab separated list of movie id | movie title | release date | video release date | IMDb URL | unknown | Action | Adventure | Animation | Children's | Comedy | Crime | Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci- Fi | Thriller | War | Western.
  The last 19 fields are the genres, 1 indicates the movie is of that genre, 0 indicates it is not; movies can be in several genres at once.
  The movie ids are the ones used in the u.data data set.

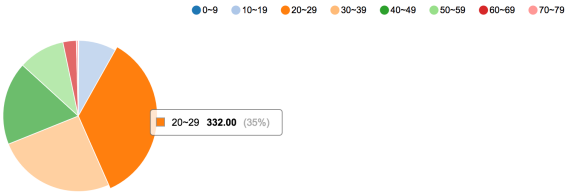
- u.user: Demographic information about the users; this is a tab separated list of user id | age | gender | occupation | zip code.
  The user ids are the ones used in the u.data data set.

2.2 Data Analysis

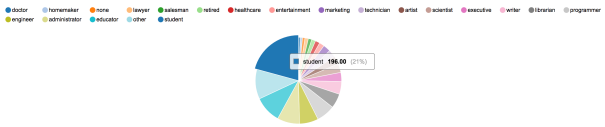
2.2.1 Data statistics

In this dataset, we did several statistics about users' and movies' attributes. According to users,

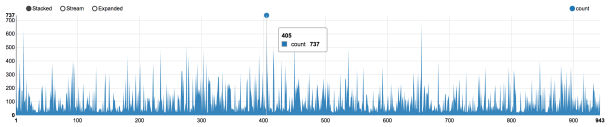
we selected their ages, occupations, and the number of movies rated by each person. The follows are their visualizations.



(a) Age distribution



(b) Occupation distribution



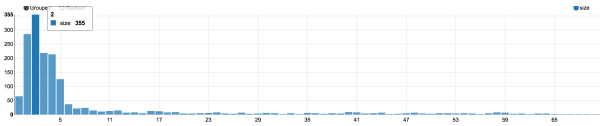
(c) The number of movies rated by users

Figure 1: Users statistics

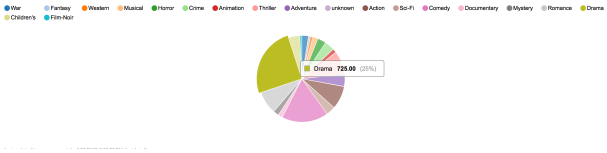
From these figures, we found that the most watching movie age period focuses on 20-29 because they are young and be more into movie. The most part occupation is student as the most time they have. What's more, the user id which rated the most movies is 405 rated 737 times. From dataset, we knew this person is female and 22 years old. She works in health care area and lives in New York city.

According to movies, we selected their release periods, genres, the number of users rating each movie, the top 9 most rating movies and the top 9 highest rating movies. Because this dataset collected the movies which issued before 1998, we used 1998 as our based to calculate periods. What's more, in counting the highest rating movies, we removed the movie which rating times is less than 50. The visualizations are shown in figure 2.

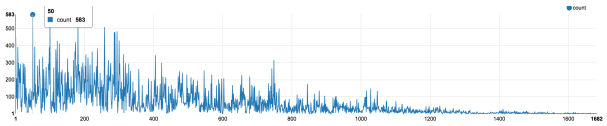
From these figures, we knew that the in the latest 5 years before 1998, there were the most movies released, the biggest part of genre is drama. In addition, the most rating movie is Star Wars(1977) and the highest rating movie is A Close Shave(1995).



(a) Release Period



(b) Genre distribution



(c) The number of users rating each movie

name	rating
Star Wars (1977)	583
Contact (1997)	509
Fargo (1996)	508
Return of the Jedi (1983)	507
Liar Liar (1997)	485
English Patient, The (1996)	481
Scream (1996)	478
Toy Story (1995)	452
Air Force One (1997)	431

(d) Top 9 most rating movies

name	rating_times	average_rating
Close Shave, A (1995)	112	4.4911
Schindler's List (1993)	298	4.4664
Wrong Trousers, The (1993)	118	4.4661
Casablanca (1942)	243	4.4568
Wallace & Gromit: The Best of Aardman Animation (1996)	67	4.4478
Shawshank Redemption, The (1994)	283	4.4452
Rear Window (1954)	209	4.3876
Usual Suspects, The (1995)	267	4.3858
Star Wars (1977)	583	4.3585

(e) Top 9 highest rating movies

Figure 2: Movies statistics

In rating data, we found out that the maximum rating number is 5 and the minimum rating number is 1, the mean is 3.52986, median is 4, the rating times per user is 106 and the rating times per movie is 59. The most rating number is 4 which takes 37%. Here is the rating distribution.

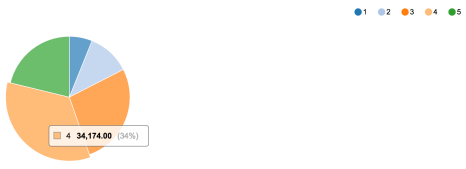


Figure 3: Rating distribution

2.2.2 Data relationship

In this part, we worked on data relationships, the attitude about the movie among different age period as well as different genders.

First, we calculated the average rating of different age period. The result shows that age 10-19

has the lowest rating for movies and the age 0-9 gives the highest points(There is only one person in this period).

age_period	average
0~9	3.7674
10~19	3.4788
20~29	3.5677
30~39	3.5866
40~49	3.6061
50~59	3.7124
60~69	3.5972
70~79	3.6862

Figure 4: Average rating of different age period

What's more, here we take four movies as examples to show different rating in different age. The movies are Toy Story (Comedy), Star Wars (Adventure), The Godfather (Crime) and Titanic (Romance).

(a) Toy Story (b) Star Wars

(c) The Godfather (d) Titanic

Figure 5: Four Examples

For the relationship between rating and genders, we found out the top 5 most difference rating movie between male and female which is shown as below,

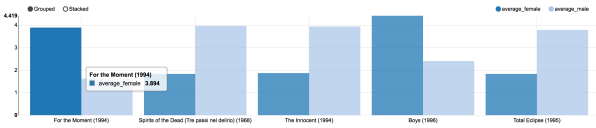


Figure 6: Most difference rating movie between male and female

According to difference figure, we can find out the most difference movie is For the Moment which is the romance movie. It obtains the 3.8936 points from women and 1.622 from men. On the contrary, the second one, Spirits of the Dead which is a horror, thriller movie, gains 1.8345 from women and 3.9668 from men. Thus, we can conclude female may prefer romance movies than thriller movies, and vice versa.

3 Movie Recommender

3.1 ALS Model based Collaborative Filtering

Collaborative filtering(Breese et al., 1998) is commonly used for recommender systems. These techniques aim to fill in the missing entries of a user-item association matrix.spark.ml currently supports model-based collaborative filtering, in which users and products are described by a small set of latent factors that can be used to predict missing entries.spark.ml uses the alternating least squares (ALS)(Koren Y, 2009) algorithm to learn these latent factors.

The original user-movie rating matrix is very sparse looks like as below.

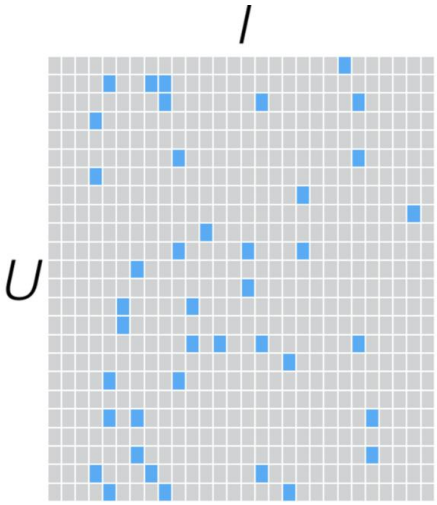


Figure 7: Sparse Rating Matrix

ALS is Matrix Factorization Algorithm. Matrix Factorization decomposes a large matrix into products of matrices.

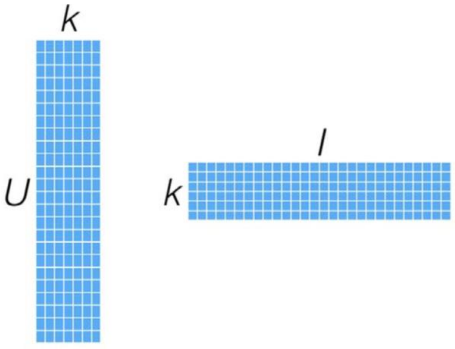


Figure 8: After Matrix Factorization

In our recommendation systems, let R as a matrix of User (Rows) and Movies (Columns). Matrix factorization will allow us to discover the latent features that define the interactions between User and Ratings. U is m * k matrix and I is n * k matrix, which m is number of users, n is number of movies and k is number of latent features.
In order to make $U \cdot I^T$ approximates to R as much as possible, need to minimized the equation as followed.

$$L(U, I) = \sum_{ij} (R_{ij} - U_i I_j^T)^2 \tag{1}$$

Use regularization to avoid overfitting,

$$L(U, I) = \sum_{ij} (R_{ij} - U_i I_j^T)^2 + \lambda(|U_i|^2 + |I_j|^2) \tag{2}$$

Then, first fix V and solve I by minimization of the function L(U,V) by resolving the least square error function.Second, fix U and solve I by minimization of the function L(U,V) by resolving the least square error function. Finally, we can get the approximate value by $U_i^T \dot{I}_j$. Then according to this equation, we can predict the $user_i$ will rate $item_j$ for $u_i \dot{i}_j$. Then we can select the highest rate for item and recommend this item to user.

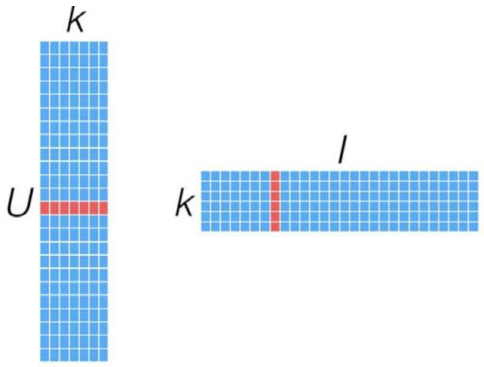


Figure 9: Calculating the approximate rating value

3.2 Kmeans Cluster for Users and Items

After we have built the ALS model, we can extract the feature vectors for users and items. And then based on these feature vectors, cluster users and items individually.

The basic kmeans algorithm can be compute in this steps.

1. Randomly initialize K centroids c_K where $C_K = 1, 2, 3, \ldots, K$
2. For every point p, Compare every Euclidian distances between p and C_K, label it according the nearest centroid.
3. Centroid upgrade: Compute the average of all the points in every category, and replace the former centroid with it.
4. Repeat step 2 & 3 until the centroids barely change (less than threshold)

The ALS model can only recommend user to item or item user. It can't recommend the user to user. So we use this kmeans cluster algorithm to find the user may have similar features. Then based on their age, job information to recommend friends. For the movies(items), we can also find similar product based on their cluster and genres.

3.3 Experiment Results

In the our experiment, the dataset is the user's rate to different movies. We first randomly split the data to training and testing with $60\% : 40\%$. And then build the ALS model on different parameters based on training data. The goal is to use training data to build model then to minimize the rating error for testing dataset. Once we find the minimum error, we can use this model as pre-defined model to recommend movie for users.

rank	lamda	iteration	RMSE
10	1.0	5	1.3632
10	0.001	5	0.9030
10	1.0	10	1.3616
10	0.001	10	0.9263
50	1.0	5	1.3619
50	0.001	5	0.8469
50	1.0	10	1.3617
50	0.001	10	0.8515

Figure 10: movielens-10k dataset for different parameter, when rank = 50, lamda=0.001, iterations = 5, it has smallest root mean square error

Here11 is some recommendation example for users.

Because of Kmeans is an unsupervised learning method. So we can't check if the cluster result is ideal or not. But in here we can see that the cluster result is good in some cluster. For example, figure12 showed that in cluster 2, the "Amitybille" series movie are in the same cluster.

For recommendation friends, we use data frame to select the right friends. We use the user favorite

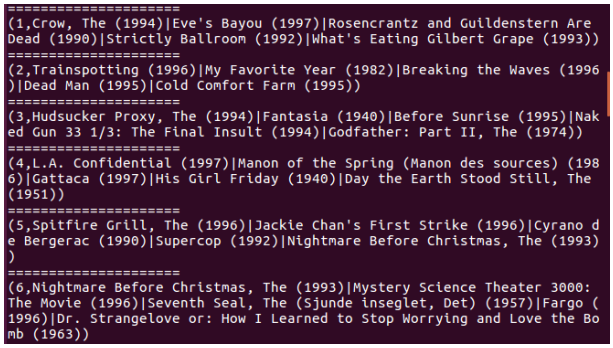


Figure 11: Recommend Movie for users

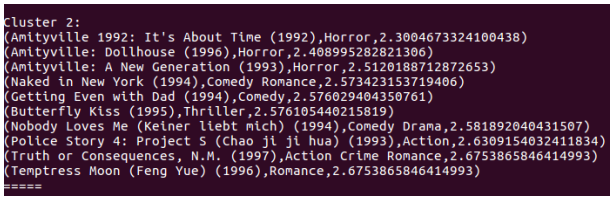


Figure 12: Kmeans cluster results

movie, age and job information to recommend the friends to users. Here is an example for recommend friends.

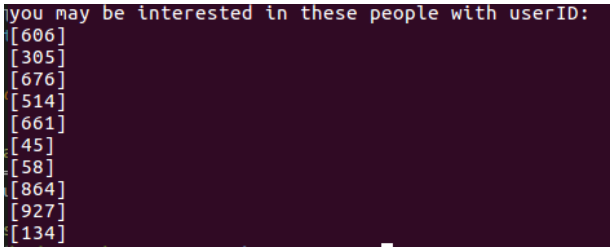


Figure 13: recommend fiends

4 Conclusion

From this project, we learned how to use spark process data.Besides, we also made the reliable movie recommender system which can recommend movie and friend with lower by using ALS model and kmeans algorithm. These make us have deeper understanding on spark and machine learning.

In the future, maybe we can add some deep learning methods on the movie recommender system, which may let the system be stronger.

Acknowledgment

This paper is based on an outline prepared by Prof. Gerard de Melo for CS674 Big Data Analytics and Text Mining. We gratefully acknowledge with Prof. Gerard de Melo of the CS department. We used zeppelin to visualize data. We referred ALS library on spark official documentation, and we also use the jblas library for vector computation.

References

[Breese et al.1998] John S Breese, David Heckerman, and Carl Kadie. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc.

[Koren Y2009] Volinsky C. Koren Y, Bell R. 2009. Matrix factorization techniques for recommender systems. *Computer*, 42(8).

[mov] Movielens dataset. https://grouplens.org/datasets/movielens/.