Applications of Multiple Methods for Recommender Systems
illustrated by the case of movie recommandation on *douban.com*

Hao Fu, Maoran Xu, Shuo Liu

June 18, 2017

Contents

1 Introduction

With the development of various kinds of social websites on the Internet, the recommend system based on social network mining becomes attractive among researchers. The system can be applied into multiple cases. Potential acquaintanceship can be detectied and recommended to users on social networking services; recommended products of 'you-may-be-interested-in' can be proposed to users on electronic commerce websites; and possible common preferences of items like music or books can be proposed based on the likelihood among users. Such recommend systems can largely improve the user experience of many websites and services.

Douban is a website that allows users to record and create contents related to films, books, music and several other topics. Besides, recommending potentially interesting movies, books, music is exactly one of the most important functions the website provides to registered users, to whom the website also provides functions of recording and content creating and sharing.

The desicions of recommendation of the website are made largely depending the information registered users openly published on the website.

In this article, we intend to apply several kinds of methods and techniques for recommender systems into the case of movie recommendation on a social networking service namely *douban.com*. We will compare the results from different methods, and furthermore try to combine these results to improve the final outcome.

2 Methodology

There are several current methods like item-based and user-based collaborative-filtering and latent factor model. They have been tested good on recommendation system. Our job is to apply three of these methods on the *douban* movie dataset, however, with some refinement and adaption with respect to the specialty of the dataset. Moreover, we come up with an idea of label regression. In this section we make overview on the basic methodology and raise some adjustment to the current methods.

2.1 Similarity

Similarity function is a real-valued function that quantifies the similarity between two objects. There are various kinds of similarity measures, we introduce some of them as follows.

Cosine similarity is a commonly used measure for real-valued vectors. The definition of cosine similarity is

$$\operatorname{sim}(U_u, U_v) = \cos(U_u, U_v) = \frac{U_u U_v}{\|U_u\|\|U_v\|} = \frac{\Sigma_i r_{u,i} r_{v,i}}{\sqrt{\Sigma_i r_{u,i}^2}\sqrt{\Sigma_i r_{v,i}^2}}. \tag{1}$$

Pearson correlation coefficient $p_{u,v} = \frac{\operatorname{cov}(U_u, U_v)}{\sigma_u, \sigma_v}$ measures the linear correlation between two variables U_u and U_v, which we can transform into a measure of similarity:

$$\operatorname{sim}(U_u, U_v) = \frac{\Sigma_i (r_{u,i} - \overline{r}_u)(r_{v,i} - \overline{r}_v)}{\sqrt{\Sigma_i (r_{u,i} - \overline{r}_u)^2}\sqrt{\Sigma_i (r_{v,i} - \overline{r}_v)^2}}, \tag{2}$$

in which $r_{u,i}$ is the i-th component of vector U_u, and $\overline{r}_u$ stands for the average of components in U_u.

Jaccard similarity measures similarity between finite sample sets, defined as follows:

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}. \tag{3}$$

Adjusted similarity function. To measure the taste of two users, one may come across to the cosine similarity describing the favor of movies. However, there is a problem that can be shown in the table below:

User/Movie	"Lion King"	"Frozen"	"Zootopia"
Shuo	3	5	4
Moran	3	?	?
Howard	3	4	4

The Pearson correlation for Shuo and Moran equals 1 and for Shuo and Howard, apparently less than 1, even if Howard has watched all the same movies as Shuo. We must take the number of community movies into account, by multiplying Jaccard similarity to adjust the Pearson correlation:

$$sim^*(U_u, U_v) = Pearson(U_u, U_v)J(A, B)$$

2.2 Collaborative Filtering (CF)

Collaborative Filtering is based on the similarity on historical ratings of users. The ratings of similar users on certain item can be used to anticipate the rating on the item of the target user.

Assuming $r_{u,i}$ is the rating on item i of user u, we may caucalte the similarity between user u and v by cosine similarity or Pearson correlation coefficient, in which $\overline{r}_u$ stands for the average ratings on items of user u. Let $N(u)$ be the set of neighbors of target user u (who are similar enough with the target user), we then have the prediction of rating on the item i of user u by

$$r_{u,i} = \overline{r}_u + \frac{\Sigma_{v \in N(u)} \text{sim}(u, v) \left(r_{v,i} - \overline{r}_v\right)}{\Sigma_{v \in N(u)} \text{sim}(u, v)}, \tag{4}$$

which method we name it by *User-based CF*. Similiarly, let $N(i)$ be the set of neighbors of target item i (who are similar enough with the target item), we then have the prediction of rating on the item i of user u by

$$r_{u,i} = \overline{r}_i + \frac{\Sigma_{j \in N(i)} \text{sim}(i, j) \left(r_{u,j} - \overline{r}_j\right)}{\Sigma_{j \in N(i)} \text{sim}(i, j)}, \tag{5}$$

which method we name it by *Item-based CF*, in which $\overline{r}_i$ stands for the average ratings of users on item i.

Adjusted CF model. Straightly apply the method on the movie dataset, we may find that there are several movies and users that take charge of the result such that the recommended lists for many users look alike.

For example, there is a rarely watched movie titled "A Scandal in Bohemia". It is rated by only one person in the dataset. Coincidentally, the person is strongly correlated with the rest users in the dataset because he has watched a wide range of movies. Then the result comes out to be totally influenced by him. To reduce the influence of such phenomenon, we add punishment to the films with 10 or less rate records and to those users correlated to more than 900 users in the social net. That is,

$$sim^*(u, v) = \alpha sim(u, v)$$

for $\#sim(u, v) > 0.01 > 900$ and

$$\bar{r}_i^* = \beta \bar{r}_i$$

for items with less than 10 rate records. Here we choose α to be 0.7, β to be 0.8.

2.3 Latent Factor Model (SVD method)

There are some factors behind the movies that determines which group of persons may like the movies. It can be its cast, type, lines, BGM, or anything. The Latent Factor Model(LFM) is able to dig the latent factors without knowing what the factor is. It is implemented with SVD decomposition.

Suppose $\mathbf{M}$ is a $m \times n$ matrix whose entires come from field $\mathbb{R}$. Then there exists a factorization called a SVD of $\mathbf{M}$ of the form $\mathbf{M} = \mathbf{U\Sigma V}^*$. Let $\mathbf{X} = \mathbf{U\Sigma}$, $\mathbf{Y} = \mathbf{V}^*$, and $\mathbf{M} = \mathbf{XY}$ is a full-rank decomposition. SVD can be applied to recommender system for rating prediction. Assuming the existing rating information of users on items are matrix $\mathbf{R}$, in which the unrated items of users are temporarily denoted by a '?'. In applications, $\mathbf{R}$ is a sparse matrix. We already know that $\mathbf{R}$ has a full-rank decomposition

$$\mathbf{R} = \mathbf{PQ},$$

in which $\mathbf{R}$, $\mathbf{P}$ and $\mathbf{Q}$ are respectively $u \times i$, $u \times k$ and $k \times i$, and $k = \text{rank}(\mathbf{R})$, u stands for the number of users and i the number of items. The matrix P and Q has their meaning in reality. P stands for the degree for each latent factor class that a user is interested in. Q stands for the relation between each movie with the latent classes.

Assuming $\hat{r}_{ui} = p_u^T q_i$, in which p_u and q_i are respectively a row of $\mathbf{P}$ and a column of $\mathbf{Q}$, and r_{ui} the true ratings of user u on item i, we have the error $e_{ui} = r_{ui} - \hat{r}_{ui}$. So the square sum of estimation can be calculated by $\text{SSE} = \sum_{u,i} e_{ui}^2 = \sum_{u,i} \left(r_{ui} - \sum_{k=1}^{K} p_{uk} q_{ki} \right)$. We can estimate $\mathbf{R}$ by reduce SSE to the minimum with gradient descent method. Notice that an important parameter in this stage is the number of latent factors, denoted by k.

2.4 Label Regression

Actually, movies and users can be easily classified by taking a look at their favorite types. The *douban* dataset provides us with 29 movie labels. We can establish a movie-type matrix for each user and do some regression with respect to the rate a user gives. It is a matrix which looks like:

Movie/Label	Love	Violence	War	Animation	Adventure	Rate
"Lion King"	0	0	0	1	1	4
"Hacksaw Ridge"	1	1	1	0	0	3
"Your Name"	1	0	0	1	1	5

Table 1: Movie-type matrix for a particular user

The loss function of the regression problem is MAE: the absolute gap between the predicted rate and the ground truth. Since the evaluation function is also MAE, it can perform good with respect to the evaluation function.

2.5 Combination of Models

For user-based and item-based collaborative filtering methods, along with the latent factor model, each yields a $1,000 \times 1,000$ matrix that gives the predicted rate for each user, each movie. We can combine these methods with regression method. The idea is to see ratings given by diverse methods as features and here the task is to assign appropriate weights to them. Intuitively, we select movies which we already have true ratings and implement a regression upon features.

3 Experiment

3.1 Dataset Description

The original *douban* dataset is composed of two datasets: *movie* and *user*, each contains 1,000 records of items. The data was a crawl of https://movie.douban.com.

The *user* dataset has attributes of user id, user name, rates for each movie they have seen(with respect to movie id), their following id, comments for movies. It is worth mentioning that not every movie it contains are in the *movie* dataset and most of the following id are not provided in the *user* dataset.

The *movie* dataset contains movie information like movie title, url, movie id, directors, year, actors, type, countries, rate and the summary. The job is to recommend some movies from the 1,000 movies to every user in the user list.

3.2 Experimental Setup

Tools: MySQL, Python 2.7, D3.js.

We used MySQL to extract the useful columns in the dataset and furthermore, to store the data crawled from web. Most of the analysis and recommendation job is done in python since it can do science computation and crawl job perfectly. Also, some dependence package are useful in this task such as *wordcloud*, *jieba*. Finally, we design the visualization of the unique network of each user with the help of D3.js.

Main Idea: The progress can be abstracted in the following figure.

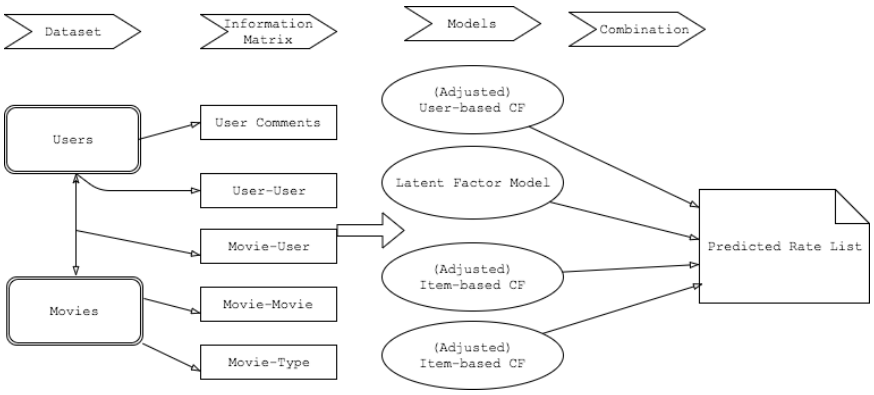

Figure 1: Main Structure of our work.

3.3 Web Spider

There is something outside the provided dataset that is valuable for recommendation. In the dataset, every movie has a set of labels of types, like "dramatic", "romantic", "violence". Analogically, We find that there are labels for users, too. Whenever one user has watched a film, he or she ticks on several labels for the file, and this add a count 1 to the type he or she has watched. Then a user has developed a count of labels as the following picture.

We can use those labels together with the movie labels to construct a "user-movie" similarity matrix. This is an additional method to fit in the appropriate movie recommendation.

We also pay attention to users' wishlists which can also be retrieved from the website. The problem is how to treat the movies in such lists: recommend them to users immediately or mark them with high ratings, etc. Since our system is based on predict ratings, we ignore the wishlists for they provide little information helping make predictions. (Though in a wishlist we can summarize types, actors and other factors attracting a certain user, these information can also be observed from the data we already have.)

Figure 2: Labels for a movie.

Figure 3: Labels for a user.

3.4 Visualization

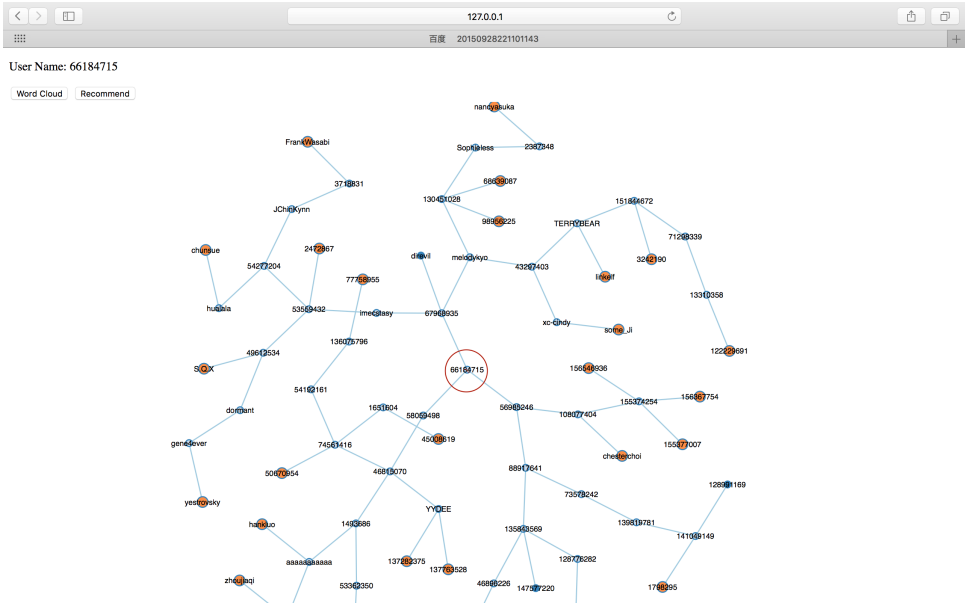

Figure 4: Visualization of the user network.

The visualization list is written in an html web page. Each user has his/her own page. It consists of three parts: the word cloud of the user's comment history, the likelihood relationship with the other users and the recommendation list for the user.

The user social network is construct on their hobby likelihood, by computing the similarity between each pair. However, there are many users closely connecting with the other people, though they may be the community hub, they make the network graph uneasy to see. So we use breath-first search to find the closest three neighbors for a user, and expand out each step of size three. And this turns out to be the best visualization effect, as shown above.

The word cloud is extracted from users' comment so that each person has a unique cloud. *Jieba* package and *wordcloud* package in python are utilized to preprocess the Chinese comments data, extract important words or sentences, and to plot the cloud.

Figure 5: Word cloud, extracted from one user's comments.

Figure 6: Recommendation list to a user.

The recommendation list is made by tracking the 10 highest predicted score of one user.

4 Evaluation

Evaluation in recommendation system is a really big challenge, most because of that there is not ground truth if a user like the recommended movies since he has never watch one of these. We just take on MAE (mean absolute error) as evaluation criteria. Specifically, MAE is defined by

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|\hat{r}_{u,i} - r_{u,i}|,$$

where item i is in the watchlist of a user u.

Method	MAE
1. Traditional user-based CF	0.594
2. Traditional item-based CF	0.589
3. Adjusted user-based CF	0.594
4. Adjusted item-based CF	0.589
5. factor model(k=2)	0.563
6. Latent factor model(k=10)	0.494
7. Latent factor model(k=20)	**0.356**
8. (Comment featured) user-based CF	0.596
9. (Adding crawled user labels) CF	0.547
10. Label Regression Model	**0.199**
11. Combining 3, 4	0.557
12. Combining 3, 4, 8, 9	0.469

Table 2: MAE table with different combination of methods

Discussion:

Different methods deliver different results and better or worse performance. However, care must be taken to see the MAE results. It seems that label regression model outperforms the other methods in MAE, but remember that the way we generate the model is to minimize the MAE value. It is not necessary that low MAE models perform better than those with higher error. For example, comment featured user based CF is a method that deeply mines the linguistic behavior of users. It is an innovative and individual recommendation model, though not revealed by MAE evaluation. It is also worth mentioning that with the crawled user label data, we have reached to a little improvement on MAE. There lies relationship between the user and a movie.

That is, sometimes researchers have to follow the users with the recommendation list if they like it, not totally depend on a solo evaluation standard. It need time and sources to do the experiment.

5 Conclusions

We have come across a wide variety of methods to predict the users' taste for movies. We try the methods separately and in combination. Their performance is worth studying and refining. Among all those methods, we finally choose the combination of user-based, item-based, and our crawled label matrix model to gives a final recommendation. Further work to do may be the case study with these 1000 users with the recommendation list.

6 References

1 项亮. 推荐系统实践[M]. 人民邮电出版社, 2012.

2 Zafarani R, Abbasi M A, Liu H. Social media mining : an introduction[M]. Cambridge University Press, 2014.

3 Furht B. Handbook of Social Network Technologies and Applications[M]. Springer US, 2010.

4 N. Antonopoulus and J. Salter, Cinema screen recommender agent: combining collaborative and content-based filtering, IEEE Intelligent Systems, 2006, pp. 35–41