2005년 한국경영과학회/대한산업공학회 춘계공동학술대회
2005 5월 13일~14일, 충북대학교

사용자의 평가 횟수와 협동적 필터링 성과간의 관계 분석
Analysis of the Number of Ratings and the Performance of Collaborative Filtering

이홍주*, 김종우**, 박성주*
* 한국과학기술원, 테크노경영대학원, {hjlee, sjpark}@kgsm.kaist.ac.kr
** 한양대학교, 경영학부, kjw@hanyang.ac.kr

Abstract

In this paper, we consider two issues in collaborative filtering, which are closely related with the number of ratings of a user. First issue is the relationship between the number of ratings of a user and the performance of collaborative filtering. The relationship is investigated with two datasets, *EachMovie* and *Movielens* datasets. The number of ratings of a user is critical when the number of ratings is small, but after the number is over a certain threshold, its influence on recommendation performance becomes smaller. We also provide an explanation on the relationship between the number of ratings of a user and the performance in terms of neighborhood formations in collaborative filtering. The second issue is how to select an initial product list for new users for gaining user responses. We suggest and analyze 14 selection strategies which include popularity, favorite, clustering, genre, and entropy methods. Popularity methods are adequate for getting higher number of ratings from users, and favorite methods are good for higher average preference ratings of users.

1. Introduction

Recommender systems apply data analysis techniques to the problem of helping customers find which products they would like to purchase at e-Commerce sites [11]. Collaborative Filtering is one of the most successful recommendation techniques, and has been used in a number of different applications to recommend news, movies, music, books, and so on [1; 3; 4; 6; 9; 11].

Early recommender systems were pure collaborative filters that computed pair-wise similarities based on co-rated items among users and recommended items according to a similarity-weighted average [1; 3; 6; 9; 12]. Breese et al. (1998) refer to this class of algorithms as *memory-based* algorithms. In the memory-based collaborative filtering, before starting personalized service, customers are requested to express their preferences for some products, usually on a discrete numerical scale [1]. Using the evaluation ratings, similarities between customers are calculated and they are used to predict preference scores of new products based on the similarities of customers and other customer ratings for the new items.

The memory based algorithm has been successful in several domains, but the algorithm is reported to have some limitations such as scalability, sparsity, and cold start problem[4; 5; 6; 9; 11]..There are 6 parameters which can effect on the scalability, sparsity, and the performance of collaborative filtering. These are product dimension, user dimension, neighbor size, number of ratings of a user, number of ratings on a product, and train/test set ratio. Product and user dimensions can effect on the scalability of collaborative filtering and the performance of recommendations [5; 9; 11]. Reduced dimensional approaches are suggested to alleviate the scalability problem [5; 11]. The number of neighbor for predicting users' expected preferences and train/test set ratio are investigated to find optimum values for performance evaluations [9; 10]. Related to the sparsity problem, number of ratings of a user and number of ratings on a product can have an effect on the performance of collaborative filtering [4]. This paper describes our experimental results on the relationship between number of ratings of a user and the recommendation performance. We use two data sets in our experiments to test the performance of the memory based collaborative filtering: the *EachMovie* and *MovieLens* dataset.

The scopes of this paper are as follows.

- The relationship between the number of ratings of a user and the performance of the memory based collaborative filtering is analyzed where we varied the number of ratings of a user from 1 to 100.
- We try to provide an explanation on the relationship between the number of ratings of a user and the performance in terms of neighborhood formations in collaborative filtering.
- Various methods are suggested and analyzed to provide an initial item list to learn about new users without user efforts.

The rest of the paper is organized as follows: The next section shows a relationship between number of ratings of a user and the performance of collaborative filtering. Section 3 suggests a possible reason of the relationship shown in Section 2. Section 4 experiments various techniques for providing an initial item list to new users. Finally, the implications of the analyses, conclusion and further research issues are discussed in Section 5.

2. Effects of Number of Ratings of a User on the Performance of Collaborative Filtering

In the previous related research of authors [4], an experiment has been carried out to investigate the effects of the number of ratings of a user on the performance of collaborative filtering.

Figure 1 presents the errors of recommendation in terms of MAE (Mean Absolute Error) and RMSE (Root Mean Squared Error) when the number of ratings of a user is varied from 1 to 100. For both datasets, recommendation performances increase as the number of ratings (i) increases from 1 to 20. When the number of ratings (i) increases from 1 to 20, the average MAE of the *EachMovie* and *MovieLens* dataset decrease monotonically from 1.42 to 0.95 and from 1.20 to 0.95 respectively. When i increases from 20 to 30, the average MAEs of both datasets become stable around 0.95 and 0.94. Then the average MAEs of both datasets are lowered to around 0.94 and 0.93 when i is greater than 30 and less than or equal to 100. The results of the experiment show that the number of ratings needs to be over the 20 – 30 range to recommend other relevant products to the user in both the *EachMovie* and *MovieLens* datasets. Errors of recommendation monotonically decrease from 1 to 20 as shown on Figure 1. Also, we can see that the average MAEs of both datasets become stable around 0.95 and 0.93 when i is greater than 30 and less than or equal to 100. The number of ratings of a customer is critical when the number of ratings is small, but after the number is over the threshold (20-30 range in this experiment), its influence on recommendation performance becomes smaller.

3. Explaining the Effect of the Number of Ratings of a User on the Performance of Collaborative Filtering

There can be various reasons to explain the effect of the number of ratings of a user on the performance of collaborative filtering. We classify the reasons into two categories: *user behavior* and *computation algorithms*.

First category is about user behavior. Taste of users can be changed after they rated on various products [2; 7]. Also, consumption patterns and economic conditions of users can be changed as time goes by. Min and Han (2005) applied the concept of time to the collaborative filtering algorithm which can detect customers' time-variant rating or consumption patterns. Kim (2001) varied rating weights which are used for calculating similarities and predicted preferences based on the distance between the rating time and the current time. Holbrook and Hirschman (1982) and Holbrook and Schindler (1994) analyzed that emotional and experiential aspects influenced on consumers' taste for cultural products such as music, actors et al. Therefore, customers' ratings are not always consistent with past rating history.

Second category is about the computation algorithms for the memory based collaborative filtering. Calculating similarities between customers and predicting preference scores are clearly defined as mathematical formulae. When a user rates another product, similarities of a user with other users, average preference score of the user, and neighbors will be changed. However, if he or she has rated on numerous products, when the user rates another product, average preference score of the user may change little. It also has influence on the performance of collaborative filtering, but the effect is not much. Similarities of a user with other users and neighbors are closely related factors. As the number of co-rated products increases, similarities between users increase. Neighbors who are selected based on the similarities of a user tend to be fixed or have similar constitutions of members after the user has rated on a number of products. Therefore, neighborhoods with more stable relationships among neighbors tend to produce similar performance of recommendation because there is not much additional information to make recommendations for the user.

3.1 Experiment procedures

In this paper, we focus on the effect of computation algorithms for explaining the relationship between the number of ratings of a user and the performance in terms of neighborhood formations in collaborative filtering. An experiment is carried out where we varied the number of ratings of a user to investigate the

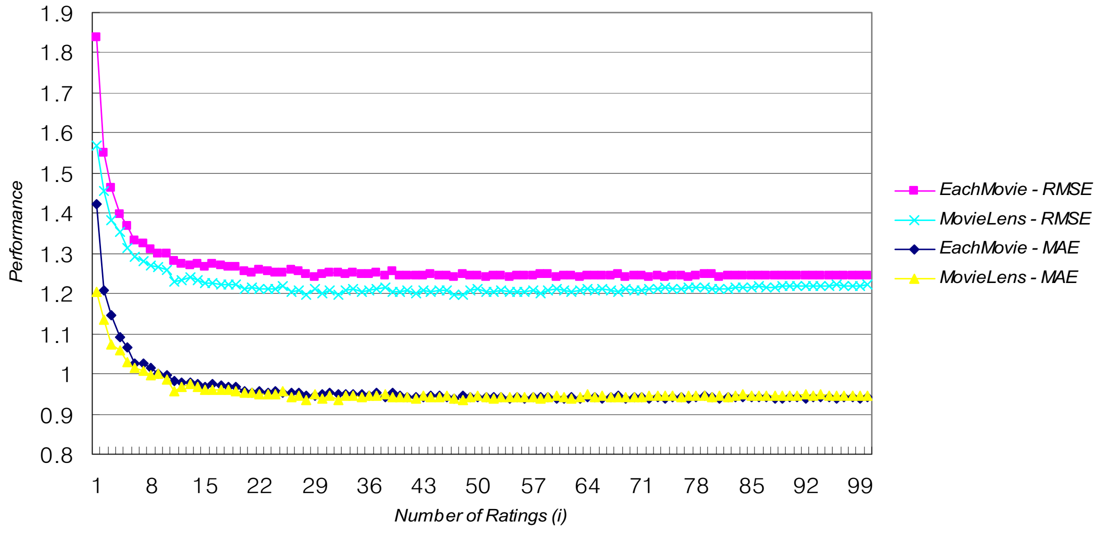

Figure 1. The Number of Ratings of a User and the Performance of Collaborative Filtering

number of overlapped neighbors. The experiment is performed through the following steps:

1) **Selecting testers**: 50 users who rated more than 110 products are randomly selected as testers. All ratings of the remaining users who are not selected as the testers are entered into a training set.

2) **Initializing and stopping**: The value of *i*, which represents the number of ratings of a user, is initialized to one. If this initializing step is visited over 30 times, this experiment procedure is terminated.

3) **Splitting the testers' preferences into the training and testing set**: *i* ratings from the preferences of the tester are randomly chosen to add them into the training set which is used for calculating similarities between users.

4) **Calculating similarities between customers**: With the training set which is built in step 1 and step 3, the similarities between users are measured by computing the Pearson correlation presented.

5) **Formulating neighbors**: Top 100 neighbors in descending correlation coefficient order are selected for each tester. The selected neighbors are stored with their rank, similarity values, tester and number of ratings (*i*).

6) **Repeating and jumping**: If this step is visited, *i* is increased by one. If *i* is less than or equal to 100, jump to step 3 for repeating steps. If *i* is greater than 100, then jump to step 2 for initializing value of *i* to 1.

This experiment is executed for the randomly selected *EachMovie* dataset and *MovieLens* dataset independently.

3.2 Result Analysis

After the experiments are finished, we calculated the average number of overlapped neighbors between *i* ratings and *i+1* ratings for all testers. The average number of overlapped neighbors among 100 neighbors is presented in Figure 2.

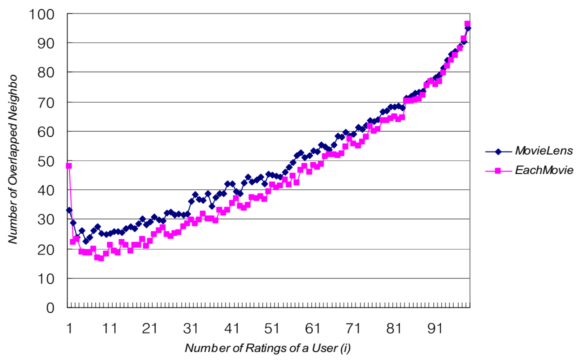

Figure 2. Number of Overlapped Neighbors

When the number of ratings (*i*) increases, the number of overlapped neighbors between *i* ratings and *i+1* ratings increases smoothly. Half of neighbors are overlapped when number of ratings is over 65 in Figure 2. After number of ratings is over 95, 90% of neighbors are overlapped with previous neighbors.

Figure 3 shows the number of overlapped neighbors compared with neighbors who are selected when number of ratings is 1, 11, 21, 31, 41, 51, 61, 71, 81, and 91 in the *EachMovie* dataset.

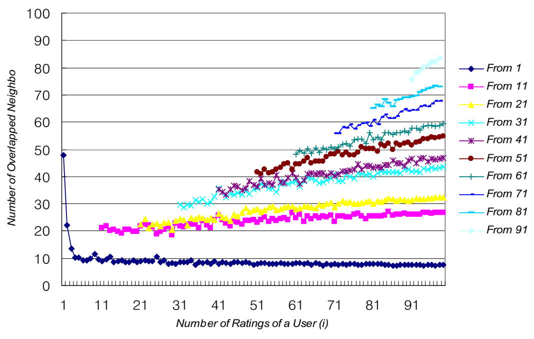

Figure 3. Number of Overlapped Neighbors

The number of overlapped neighbors compared with the neighbors who are selected when the number of rating is 1 is rapidly dropped from 50 to around 10 in the *EachMovie* dataset. Then, the number of overlapped neighbors smoothly decreases as the number of ratings of a user increases. Though the increase rates are different, the number of overlapped neighbors compared with neighbors who are selected when number of ratings is 11, 21, 31, 41, 51, 61, 71, 81, and 91 increases as the number of ratings of a user increases. When the overlapped neighbors are compared with higher number of ratings such as 81 and 91, the increase rates become larger in both datasets.

From Figure 2 and 3, we can conclude that the neighbors tend to be fixed like a family after the number of ratings of a user becomes large. As the number of ratings of the user increases, there is not much additional information to make recommendations for the user because his/her neighbors are almost fixed.

4. Strategies for Selecting Items to Present for New Users

E-Commerce sites should provide an initial product recommendation for gaining new users' preferences and learning about new users. How to select an initial product list with little user information for gaining user responses? This requires a decision of e-Commerce sites to select an appropriate strategy [8]. Schein et al. (2001) and Schein et al. (2002) use the content information – casts of actors – with the pure collaborative filtering for alleviating the cold start problem. Huang et al. (2004) applied an associative retrieval framework and related spreading activation algorithms to explore transitive associations among consumers through their past transactions and feedback. These approaches exploited content information of items or implicit user preferences such as navigations and access patterns. Rashid et al. (2002) applied the popularity of movies and entropy of ratings to select an initial item list for new users.

In this paper, we study 8 techniques that collaborative filtering recommender systems can use to learn about new users. These techniques select a list of items for the collaborative filtering system to present to each new user for rating. In the next section, 8 techniques are suggested to select an initial item list for new customers without any explicit or implicit preferences. We use the *MovieLens* dataset for this experiment. 70% of users are classified into a learning set randomly and other 30% of users are assigned into a testing set.

4.1 Strategies for Selecting Items

1) **Random**: we select 20 products randomly with uniform probability over the universe of items for comparing performances with other techniques [8].

2) **Favorite**: the favorite method selects products which have higher average preference scores. Average preference scores of products are calculated from ratings of the learning set. We select top 20 products in descending average preference order.

3) **Popularity**: the number of ratings of products is calculated from the learning set. We select top 20 products in descending popularity order.

4) **Favorite*Popularity**: the favortites*popularity method considers the favorite and popularity method in same time. The average preference of product i, P_i, and the number of ratings of product i, S_i, are normalized by the min-max algorithm and normalized values are multiplied as the following equation.

$$PS_i = \frac{P_i - P_{Min}}{P_{Max} - P_{Min}} \times \frac{S_i - S_{Min}}{S_{Max} - S_{Min}}$$

PS_i is the final score of product i. P_{Max} is the maximum value of average preference scores and P_{Min} is the minimum value of average preference scores. S_{Max} is the maximum number of ratings and

S_{Min} is the minimum number of ratings. We select top 20 products in descending PS_i order.

5) **Log Popularity*Entropy**: Rashid et al. (2002) suggested the log Popularity*Entropy method. They applied the entropy algorithm to select an initial item list. But the performances of the pure entropy algorithm are poor. Instead popularity-based strategies and pure entropy based strategies, balanced strategies such as popularity*entropy, and log popularity*entropy techniques gained good performances in user effort and accuracy [8]. Entropy of movies can be calculated as following Shannon's formula [8]:

$$H(p) = -\sum_{i=1}^{k} p_i * \log_2 p_i$$

$H(p)$ is the entropy value of product p, p_i is the ratio of i ratings. For example, assume that the total number of ratings on product p is 20. The number of ratings which have rating score 1 is 5, and the number of ratings which have rating score 2 is 15. And then, p_1 is 5/20 (0.25), and p_2 is 15/20 (0.75). The log Popularity*Entropy method multiplies log popularity with entropy $H(i)$ as following equation:

$$\log Popularity * Entropy_i = \log S_i * H(i)$$

According to Rashid et al. (2002), popularity-based strategies tend to get many ratings from users, but each rating may have little information, to the recommender system. Conversely, entropy-based techniques get a lot of value from each rating, but users may find relatively few items to rate.

6) **Genre**: The *MovieLens* dataset classified movies into 19 genres such as comedy, drama and unknown et al. We excluded the unknown genre which has just two movies. We select movies from each genre by using favorite, popularity, favorite*popularity, and log popularity*entropy methods.

7) **Demographical Clustering**: The *MovieLens* dataset has demographic information of users such as age, gender, and occupation. We classify users by their age into 5 clusters such as under 21, 21-30, 31-40, 41-50, and over 50. We select 20 movies from each demographic cluster by using favorite, popularity, favorite*popularity, and log popularity*entropy methods.

After selecting movies for each demographic cluster, we provide a movie list to fit a new user's age. If age of the new user is 35 years old, we provide 20 movies which are selected from the 31-40 age cluster.

8) **K-mean Clustering**: The *MovieLens* dataset is basically composed of user-product matrix, S. If the number of users is n and the number of products is m, matrix S is a $n \times m$ matrix. s_{ij} is the rating of user i on product j. We apply the k-mean clustering to matrix S for classifying movies with users' ratings. Resulted clusters are sets of movies which have similar ratings by users. We choose 5 clusters which have higher number of movies from the resulted clusters. From each cluster, we choose 4 movies which are located nearly from the cluster center. We think that the cluster center represent the cluster's characteristics. So, movies which are located nearly from the cluster center are considered as representatives of the cluster.

4.2 Results and Discussion

Methods		Average Number of Ratings of Users	Average Preference of Users	Compare Avg. Number of Ratings	Compare Avg. Preference
Random		1.421790342	3.510058151	14	13
Favorite		5.055594817	4.316591496	10	1
Popularity		9.213074205	3.872316135	1	7
Favorite*Popularity		7.955359246	4.158127823	7	4
Log Popularity*Entropy		7.45041225	3.488234245	8	13
Genre	Favorite	4.830742049	4.252427475	11	2
	Popularity	8.173144876	3.803396371	5	9
	Favorite*Popularity	7.017903416	4.112091687	9	5
	Log Popularity*Entropy	8.211778563	3.741890588	5	11
Demographic	Favorite	2.113191991	4.220624402	12	2
	Popularity	9.145229682	3.869620541	1	7
	Favorite*Popularity	8.967045236	3.908981563	3	6
	Log Popularity*Entropy	8.835104466	3.767438657	3	9
Clustering (K-mean)		1.800942285	3.593409349	13	12
F – Value				2662.446 (Sig. = 0.000)	388.887 (Sig. = 0.000)

Selected movie lists from suggested techniques in Section 4.1 are given to the testing set. Then, we calculated the average number of ratings of the testing set and the average preference of the testing set. If a tester rated on a movie in the provide movie list, we added number of ratings of the testing set and its rating score also used for calculating average preference of the testing set. Table 1 shows results of each method in terms of the average number of ratings and the average preference of the testing set. The average number of ratings and average preference of 14 methods – 8 methods from the genre and demographic techniques, and 6 methods from other 6 techniques – are statistically tested by the ANOVA test and their average values are ranked by the Duncan's test. Null hypotheses are that the average number of ratings and average preferences resulted from the 14 methods are same. Both null hypotheses are rejected because F-value of average rating number and average preference are 1670.964 and 337.876 respectively.

Results of the Duncan's test are summarized in the right side of Table 1. The random method has the lowest average number of ratings and average preference scores of users. The K-mean clustering method shows poor performances in both measures. In comparison of the average number of ratings, the popularity method and *demographic – popularity* method have the highest average number of ratings. The *demographic – favorite*popularity* and *demographic – log popularity*entropy* shows the 3rd average number of ratings. We can conclude that the popularity methods are adequate for getting higher number of ratings from users. In comparison of the average preference scores of users, the favorite method gains the highest average preference. The second highest average preference is from the *genre – favorite* method and *demographic – favorite* method. The favorite methods are adequate for getting higher average preference scores of users.

5. Discussion and Conclusion

In this paper, we showed an explanation on the relationship between the number of ratings of a user and the performance in terms of neighborhood formations in the collaborative filtering. We can conclude that neighbors tend to be fixed like a family after the number of ratings of a user becomes large. As the number of ratings of the user increases, there is not much additional information to make recommendations for the user because his neighbors are almost fixed.

Various techniques are suggested and analyzed to provide an initial item list to learn about new users. The favorite method gains the highest rating scores from users, but couldn't gain higher number of ratings. This may be caused that art house movies are not popular, but acquire higher rating scores from manias. The popularity method shows the highest average number of ratings, but couldn't gain higher average preference scores. Performances of methods and appropriate methods for e-Commerce sites or recommender systems depend on datasets and users. This requires a decision of e-Commerce sites to select an appropriate strategy. Before applying one of the above methods, e-Commerce sites or recommender systems have to execute similar experiments in Section 4.1 for selecting an appropriate method.

In this paper, we provided an initial item list in a lump-sum way without users' effort. However, user feedback can be helpful to select next movies to present to new customer. A solution of the cold start problem based on user feedback or attributes of products is an interesting issue for further research.

References

[1] Breese, J.S., Heckerman, D., and Kadie, C. "Empirical analysis of predictive algorithms for collaborative filtering," the Fourteenth Conference on Uncertainty in Artificial Intelligence, 1998, pp. 43-52.

[2] Kim, K. "An Algorithmic Framework of Adaptive Collaborative Filtering for Internet Marketing," in: *Graduate School of Management*, Korea Advanced Institute of Science and Technology, Seoul, 2001, p. 48.

[3] Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R., and Riedl, J. "GroupLens: Applying collaborative filtering to usenet news," *Communication of the ACM* (40:3) 1997, pp 77-87.

[4] Lee, H.J., Kim, J.W., and Park, S.J. "Parameter Selection of Collaborative Filtering for e-Commerce Personalized Recommendation," submitted to *Electronic Commerce Research*, 2005.

[5] Linden, G., Smith, B., and York, J. "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing* (7:3) 2003, pp 76-80.

[6] Mild, A., and Natter, M. "Collaborative filtering or regression models for Internet recommendation systems?" *Journal of Targeting, Measurement and Analysis of Marketing* (10:4) 2002, pp 304-313.

[7] Min, S.-H., and Han, I. "Detection of the customer time-variant pattern for improving recommender systems," *Expert Systems with Applications* (28) 2005, pp 189-199.

[8] Rashid, A.M., Albert, I., Cosley, D., Lam, S.K., McNee, S.M., Konstan, J.A., and Riedl, J. "Getting to Know You: Learning New User Preferences in Recommender Systems," the ACM IUI'02, San Francisco, 2002, pp. 127-134.

[9] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. "Analysis of Recommendation Algorithms for e-Commerce," EC'00, Minneapolis, 2000, pp. 158-167.

[10] Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. "Item-Based Collaborative Filtering Recommendation Algorithms," WWW10, Hong Kong, 2001, pp. 285-295.

[11] Sarwar, B.M., Karypis, G., Konstan, J.A., and Riedl, J.T. "Application of Dimensionality Reduction in Recommender System - A Case Study," ACM WebKDD 2000 Web Mining for E-Commerce Workshop, 2000.

[12] Schein, A.I., Popescul, A., Ungar, L.H., and Pennock, D.M. "Methods and Metrics for Cold Start Recommendations," the ACM SIGIR'02, Tampere, Finland, 2002, pp. 253-260.