User-Based and Item-Based Collaborative Filtering Recommendation Algorithms Design

Guanwen Yao
A53049615
guyao@eng.ucsd.edu

Lifeng Cai
A53045464
l6cai@eng.ucsd.edu

ABSTRACT

Offering online personalized recommendation services helps to improve customers' satisfaction and needs. Conventionally, a recommendation system is considered as a success if customers purchase the recommended products. However, the act of purchasing itself does not guarantee satisfaction and a truly successful recommendation system should be one that maximizes the customer's after-use gratification. In this paper, we build the recommendation system based on collaborative filtering. Two models are tested: item-based and user-based. The dataset we use is one of the Amazon datasets [1].

Keywords

User-based, Item-based, Collaborative filtering

1. INTRODUCTION

Personalization of product information has become one of the most important factors that impact a customer's product selection and satisfaction in today's competitive and challenging market. Personalized service requires firms to understand customers and offer goods or services that meet their needs. Successful firms are those provide the right products to the right customers at the right time and for the right price.

Recommendation systems are widely used by e-commerce practitioners and have become an important research topic in information sciences and decision support systems. Recommendation systems are decision aids that analyze customer's prior online behavior and present information on products to match customer's preferences. Through analyzing the customer's purchase history or communicating with them, recommendation systems employ quantitative and qualitative methods to discover the products that best suit the customer. Most of the current recommendation systems recommend products that have a high probability of being purchased. They employ content-based filtering (CBF), collaborative filtering (CF), and other data mining techniques, for example, decision tree, association rule, and semantic approach. Some of them have been used by large companies, like Amazon and Dell. In this paper, we build our recommendation system based collaborative filtering (CF) and we use two models: item-based and user-based. The dataset we used is one of the Amazon datasets [1]. Lastly, we test our two models and compare the performance of two models.

2. DATASET EXPLORATORY ANALYSIS

We use the Amazon dataset $'Gourmet\_Foods.txt.gz'$ on the link [1]. This dataset includes 154635 Gourmet Food reviews on Amazon. Each review contains review/profileName, product/price, review/time, product/productId, review/summary, review/helpfulness, review/userId, product/title, review/score and review/text. However, 898 reviews show 'unknown' for the field of 'review/userId'. We discard these reviews because we can get less useful information from them. So, there are actually 153737 reviews in our dataset.

We do exploratory analysis on the whole dataset because in our experiments of recommendation system, we will change the size of the training set and the test set. Also, doing exploratory analysis on the whole dataset can offer us entire impression on how the whole dataset looks like and help us to split data into training set and test set efficiently.

1) In the $'Gourmet\_Foods.txt.gz'$ dataset, there are actually 153737 valid reviews, discarding the reviews with 'unknown' for the field of 'review/userId'.

2) There are 23368 distinct items and 112543 distinct users in the dataset.

3) The average rating for gourmet foods is 4.23886897754, which shows the degree of satisfaction is relatively high.

4) The most popular item is Tuscan Whole Milk, 1 Gallon, 128 fl oz, and it has 1384 reviews in the dataset. 9282 items has only 1 review in the dataset, which indicates 39.7210% of the items have been bought for only 1 time.

5) The user who has written the most reviews is Rebecca of Amazon "The Rebecca Review", and the number of the reviews she has written is 203. Also, 94733 users has written only 1 review in the dataset, which indicates 84.1749% of the users have bought only 1 item.

3. PREDICTIVE TASK

We will split the whole dataset into the training set and the test set. We will build a recommendation system which filters information (the behaviors of his or her similar users) of a user based on a collection of user profiles in the training set, and predict and give recommendations on what items he or she will buy in the future. We will test our predictions by searching whether the items that we recommend to a user according to the training set, are in the item list the user have bought in the test set.

4. LITERATURE

4.1 Related Work

The GroupLens [2] is one of the earliest implementation of collaborative filtering recommend system based on ratings. The GroupLens research system provides a pseudonymous collaborative solution for Usenet news and movies. Later on, the item-based and user-based collaborative filtering recommendation systems are proposed by [3] [4] [5], which are widely used now by large companies such as Amazon and Dell. We implement both the item-based and user-based collaborative filtering recommend systems in this assignment.

4.2 Dataset

We use Amazon dataset *'Gourmet_Foods.txt.gz'* on the link [1] to build a recommend system for customers. This dataset along with other datasets on the link [1] have been used in the research of 'Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text' by Julian McAuley and Jure Leskovec. We use one of the categories: gourmet foods among the datasets on the link above for our item-based and user-based collaborative filtering recommendation system. There are more than 25 other categories such as books, health, music, software and so on, which can be used and compared in the future.

5. FEATURES

Each review in the dataset contains information: review/profileName, product/price, review/time, product/productId, review/summary, review/helpfulness, review/userId, product/title, review/score and review/text. There are four fields we can use in the dataset. They are review/score, review/helpfulness, product/price and review/text. However, most of the reviews show 'unknown' for the filed of product/price. Then for recommendation system, there are two things useful in dataset. One is customers' ratings and the other is customers' reviews. While, to extract feature from reviews is quite complex and the accuracy of feature is hard to be guaranteed. On the other hand, customers' ratings can directly reflect what customers think about this item. Directly using ratings as feature will no doubt give best result. So we choose our recommendation system based on customer's ratings, and the feature we use is reviewers' rating: review/score.

The pre-processing we do on the feature is to create a two-key hash map by using review/userId, product/productId and review/score. For user-based collaborative filtering recommendation system, the first key is review/userId, the second key is product/productId and the value is review/score. The two-key hash map can be regarded as a 2D matrix where the row is user and the column is item. For item-based collaborative filtering recommendation system, the first key is product/productId, the second key is review/userId and the value is review/score. Similarly, the two-key hash map can also be regarded as a 2D matrix where the row is item and the column is user.

To justify the features selected to use as recommendation system, we can see from the section of results and conclusions, which present our result of recommendation system.

6. RECOMMEND SYSTEM STRUCTURE

The first step is to collect the preferences of the users. Our Collaborative Filtering (CF) implementation stores the data in two 2D matrices. So for each user in a row we have columns for each item that he or she has rated. The matrix is quite sparse, since a lot of users only buy one item. After getting the 2D dataset matrix, we implement two kinds of recommendation system models: item-based and user-based correlative filtering. For both models, we need to compute the similarity and prediction score. In the following part, we will explicitly show how we build our item-based and user-based recommendation system, and compare different models in the following subsection. Figure 1 shows the collaborative filtering process.

6.1 Item Similarity Computation

Computing the similarity between items is the fundamental step of our recommendation system, since we want to recommend similar items to customers based on what they have bought before. The basic idea of similarity computation between two items i and j is to firstly isolate the users who have rated both of these items and then to apply a similarity computation technique to determine the similarity $s_{i,j}$. Figure 2 shows the isolation of the co-rated items and similarity computation. We present three ways to compute similarity. In our recommendation system, we use the second method. The reason is stated below.

6.1.1 Cosine-based Similarity

In this case, two items are thought of as two vectors in m dimensional user-space. The similarity between them is measured by computing cosine angle between two vectors. Similarity between items i and j, denoted by $sim(i,j)$ is given by

$$sim(i,j) = cos(\overrightarrow{i}, \overrightarrow{j}) = \frac{\overrightarrow{i} \cdot \overrightarrow{j}}{||\overrightarrow{i}||_2 * ||\overrightarrow{j}||_2} \quad (1)$$

6.1.2 Correlation-based Similarity

In this case, the similarity between two items is measured by computing correlation $corr_{i,j}$. Denoting the set of users who both rate i and j as U, the correlation similarity is given by

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R}_i)(R_{u,j} - \overline{R}_j)}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R}_i)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R}_j)^2}} \quad (2)$$

6.1.3 Adjusted Cosine Similarity

Computing similarity by using basic cosine measure in item-based case has one obvious drawback, and it is the difference

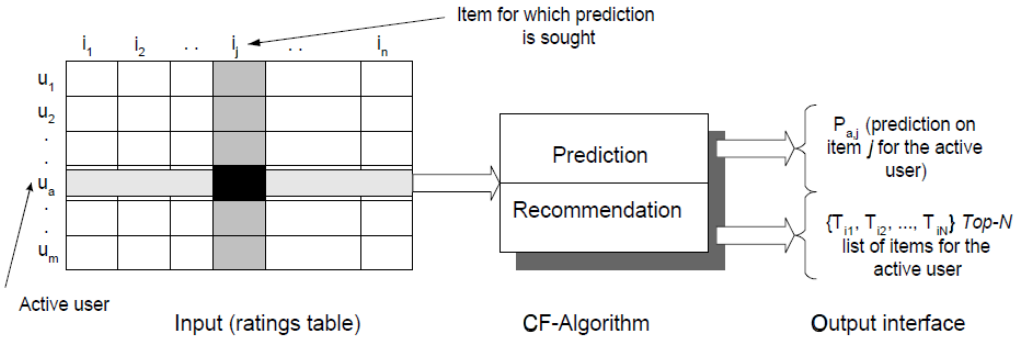

Figure 1: The collaborative filtering process.

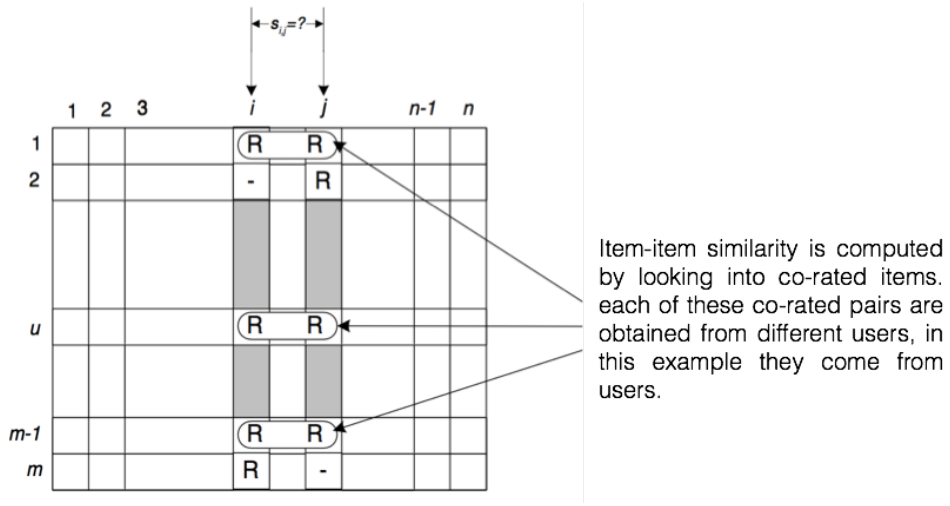

Figure 2: Isolation of the co-rated items and similarity computation.

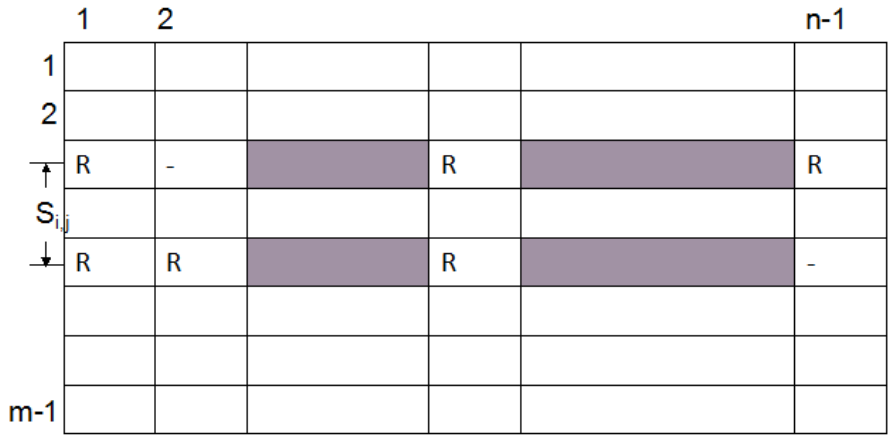

Figure 3: Isolation of the co-rated users and similarity computation.

in rating scale between different users. We can subtract this kind of bias, so the similarity using this scheme is given by

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R}_u)(R_{u,j} - \overline{R}_u)}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R}_u)^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R}_u)^2}} \tag{3}$$

The reason why we choose the second method to compute similarity is that the range of ratings in our dataset is discrete number from 1 to 5 and large amount of users buy only one items. In this case, if we choose the third method, denominator will be zero, which will happen frequently. Based on this consideration, we choose the second method.

6.2 Prediction Computation

After getting similarity between two different items, we then compute the prediction on an item i for a user u by computing the sum of the ratings given by the user on the items similar to i. Each rating is weighted by the corresponding similarity $s_{i,j}$. And final weighted sum is divided by the sum of similarity to get normalized prediction value. The prediction $P_{u,i}$ is given by

$$P_{u,i} = \frac{\sum_N (s_{i,N} * R_{u,N})}{\sum_N (|s_{i,u}|)} \tag{4}$$

This approach tries to capture how the active user rates the similar items. The weighted sum is then scaled by the sum of the similarity terms to make sure the prediction is in the specific range.

6.3 User-based CF

Figure 3 shows the isolation of the co-rated users and similarity computation. The user-based CF is quite like item-based CF. Instead of computing the similarity between two items, we focus on the similarity between two customers. We use correlation-based method of computing similarity between two customers u, v, which is the same as item-based method. We denote that similarity as $s_{u,v}$. The prediction on an item for a user u is calculated by computing weighted sum of different users ratings on item i. The prediction $P_{u,i}$ is given by

$$P_{u,i} = \frac{\sum_v (r_{v,i} * s_{u,v})}{\sum_v s_{u,v}} \tag{5}$$

where $r_{v,i}$ is the rating of user v on item i.

6.4 Different Model Comparison

The user-based CF has some limitations. One is its difficulty in measuring the similarities between users, and the other is the scalability issue. As the number of customers and products increases, the computation time of algorithms grows exponentially.

The item-based CF was proposed to overcome the scalability problem as it calculates item similarities in an offline basis. It assumes that a user will be more likely to purchase items that are similar or related to the items that he or she has already purchased. Another one is the ratings, which are some discrete values, can not provide us much information about relationship between users and items.

The content-based filtering (CBF) method applies content analysis to target items. Target items are described by their attributes, such as color, shape, and material. The user's profile is constructed by analyzing his/her responses to questionnaires, his/her rating of products, and navigation history. The recommendation system proposes items that have high correlations with a user's profile. However, a pure CBF system also has its shortcomings. One is that users can only receive recommendations similar to their earlier experiences. The other is that some items, such as music, photographs, and multimedia, are hard to analyze. We want to use this model, but for this assignment dataset, it's hard to get the such information from dataset. We decide not to use it.

7. RESULTS AND CONCLUSIONS

From the exploratory analysis in section 2, we have found that 84.1749% of the users have bought only 1 item. So we decide to put all these data into the training data, because it is less meaningful to predict what items the users will buy with less or even no correlated information in the training set. We vary the size of the training set and the test set by splitting randomly with equal probability the reviews written by the users who have bought over n items in the whole dataset. So about half of the reviews written by the users who have bought over n items in the whole dataset will be put into the training set and the other half will be put into the test set.

In our experiments, we choose n (over n items bought by a particular user) to be 3, 4, 6, 8, 10, 15 and 20. Note that one particular n we choose corresponds to one particular training/test set ratio. So n can represent the training/test set ratio in our project and they are the same thing. We will analyze the impact of different n (different training/test set ratio) on the recommendation results.

We will then fix n (over n items bought by a particular user) equal to 5, and limit our recommendation items to at most m top ranked items. In other words, we will recommend the top m ranked items to the users in the test set according to the correlation information we obtain in the training set. We will analyze the impact of choosing top m recommendation items on the recommendation results.

We have implemented both user-based and item-based collaborative filtering recommendation systems in our experiment and the results are exactly the same. For item-based collaborative filtering recommendation system, we will calculate the related items or recommendation items beforehand, which are stored in one large matrix (database) for each user in the training set. And for a particular user in the test data, we just look up this matrix (database) and offer the recommendations. For user-based collaborative filtering recommendation system, we do not have this large matrix (database) and we will calculate the similarity and rating scores for related items each time when we need to recommend items to users. User-based collaborative filtering recommendation system consumes much time on calculating the similarity and rating scores for related items and it works well when the scale is not so large and data refreshment is relatively frequent. Item-based collaborative filtering recommendation system needs space and works well when the scale is pretty large and data refreshment is relatively infrequent. It consumes much less time when providing recommendations to users than user-based collaborative filtering recommendation system does because it calculates the relat-

Table 1: The impact of splitting the dataset by reviews written by users who have bought over n items on the recommendation results

n	Training Set Size	Test Set Size	# of Users in Test Set	# of Successful Recommendations	# of Recommendations
3	135072	18665	6703	1287	33196
4	140161	13576	3743	1199	19485
6	145458	8279	1414	618	8944
8	147577	6160	766	568	7616
10	148728	5009	497	583	7853
15	150199	3538	245	547	6694
20	150912	2825	163	532	5165

Table 2: The impact of choosing top m recommendation items on the recommendation results

m	# of Successful Recommendations	# of Recommendations
3	151	1167
5	258	2019
8	368	3072
10	450	3669
15	621	4929
20	698	5899
25	731	6654
30	744	7299
50	792	9302

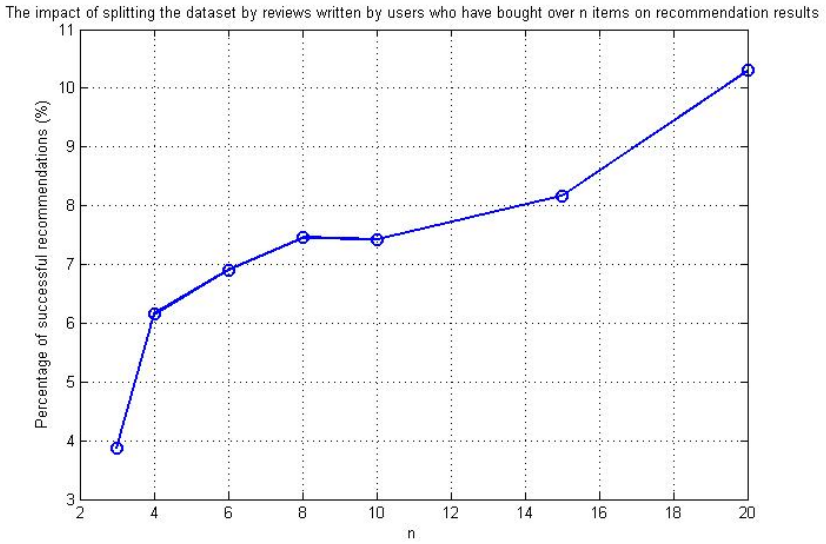

Figure 4: The Impact of splitting the dataset by reviews written by users who have bought over n items on the recommendation results.

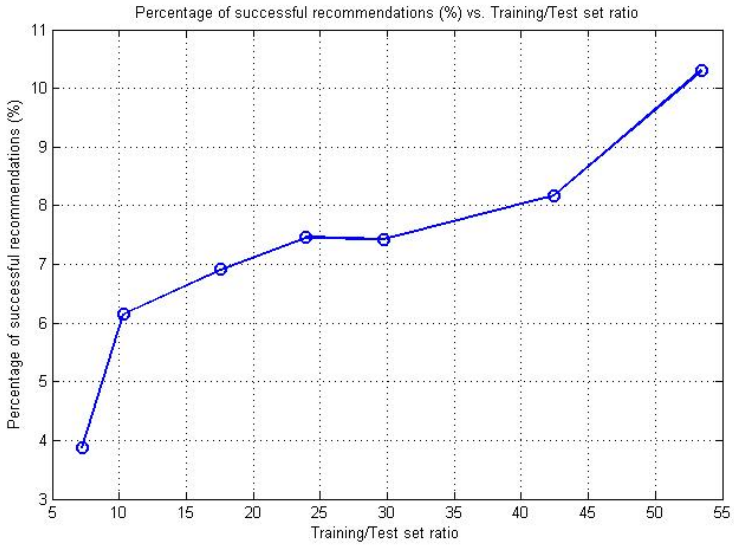

Figure 5: Percentage of successful recommendations (%) vs. Training/Test set ratio.

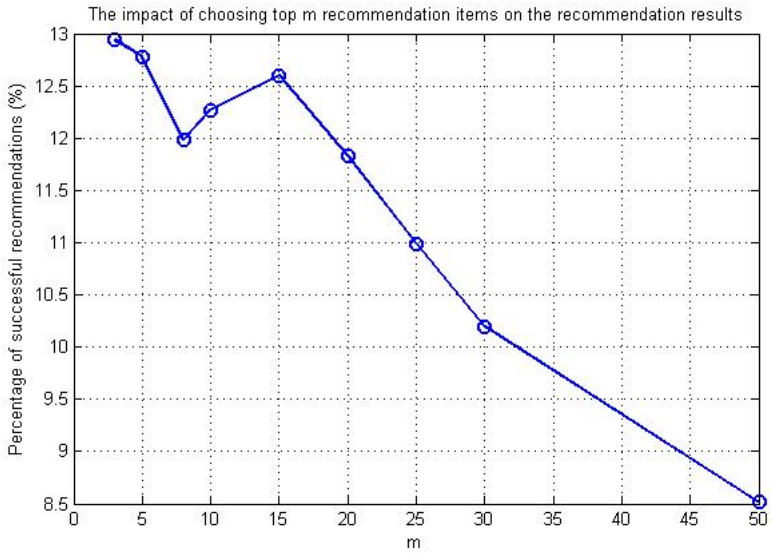

Figure 6: The impact of choosing top m recommendation items on the recommendation results.

ed items or recommendation items beforehand. Some large companies like Amazon have their own large database, so space for them is not a big problem and they usually use item-based collaborative filtering recommendation system.

7.1 Percentage of Successful Recommendations (%) vs. Training/Test Set Ratio

Table 1 shows the impact of splitting the dataset by reviews written by users who have bought over n items on the recommendation results. From table 1, it can be found that as n increases, the size of training set increases and the size of test set decreases. Also as n increases, the number of distinct users in test set decreases. Figure 4 shows the impact of splitting the dataset by reviews written by users who have bought over n items on the recommendation results. Figure 5 shows the impact of training/test set ratio on the recommendation results. Figure 4 and figure 5 are the same because each one particular n corresponds to one particular training/test set ratio. From the two figures, we can see that as n and training/test set ratio increase, the percentage of successful recommendations increases because the more items the users have bought, the more correlated information we can obtain to give recommendations.

7.2 Percentage of Successful Recommendations (%) vs. Top m Recommendation items

Table 2 and figure 6 show the impact of choosing top m recommendation items on the recommendation results. From the table and the figure, it can be found that when we choose m smaller than 15, the percentage of successful recommendations remains higher than 12%, which indicates good recommendations. As m increases and becomes larger than 20, the percentage of successful recommendations decreases. In the reality, we should always recommend suitable number of items to the potential customers. If we recommend too many items, the customers will feel bored and show no interest in even glancing at them. On the other hand, if we recommend too few items, the users will not have enough choices. As a result, a suitable number of recommendation items should be selected carefully for recommendation system in reality.

8. REFERENCES

[1] http://snap.stanford.edu/data/web-Amazon-links.html
[2] Konstan, Joseph A., et al. "GroupLens: applying collaborative filtering to Usenet news." Communications of the ACM 40.3 (1997): 77-87.
[3] http://en.wikipedia.org/wiki/Collaborative_filtering.
[4] Sarwar, Badrul, et al. "Item-based collaborative filtering recommendation algorithms." Proceedings of the 10th international conference on World Wide Web. ACM, 2001.
[5] Linden, Greg, Brent Smith, and Jeremy York. "Amazon.com recommendations: Item-to-item collaborative filtering." Internet Computing, IEEE 7.1 (2003): 76-80.