Survey of Recommender Systems

(Collaborative Filtering)

Jong Seo Lee

{jslee@calpoly.edu}

California Polytechnic State University - San Luis Obispo

In these days, as the e-commerce industry is growing and becoming complex, the information about the products is increasing with exponential rate. In such environment, customers have difficulty to find optimal information about product items from the tremendous amount of information. To help the buyers, the major e-business companies are also developing their recommender systems (RS) to help their customers choose product items more efficiently [Gil, A. and García, F. 2003]. This is a kind of Win-Win strategy in E-commerce. The customers can get benefits by receiving some useful information about the products which they are likely to buy. At the same time, the business can get benefits with an increase of its sales [Sotomayor et al. 2005]. Now, the Recommender System technologies are still metamorphosing day by day. The main object of this survey is to study the existing concepts of Recommender Systems, look over the real Recommender System examples, and scrutinize the background theories and algorithms in detail.

1. Introduction

Recommender system is a "specific type of information filtering (IF) technique" that can be used to present and recommend commercial items that could be interesting to the each specific user. Those items are like movies, music, books, news, images, web pages, and so on. Most widely used techniques for Recommender system are *Knowledge-based recommender* and *Collaborative Filtering* (CF). *Knowledge-based recommender* works by asking a user about the requirement of the products and finds out what products meet the user's requirements [Chun, I. and Hong, I. 2001]. *Collaborative filtering* decides recommendable items that a user might be interested in based on the information about many users' preference. *Collaborative filtering* is the most successful recommendation technique to date.

In this paper, study about collaborative filtering technique will be mainly discussed.

The contributions and organization of this survey are as follows:

- General information about CF is mentioned (Section 2)
- Observation about 4 real recommender systems are mentioned (Section 3)
- Memory-based CF algorithms and problems (Section 4.1, 4.2)
- Model-based CF algorithms and problems (Section 4.3)
- Detailed application of Model-based CF in Item-based system (Section 5.1, 5.2, 5.3)
- Performance and Quality comparison between user-based and item-based algorithms (Section 6)

2. Collaborative Filtering

The underlying assumption of *Collaborative Filtering Approach* is that "those who agreed in the past tend to agree again in the future". It works by building a database which contains users' preferences.

Here are the simple basic two steps for traditional Collaborative Filtering:

1. Look for like-minded users ("*neighbors*")
2. Use the ratings from those "neighbors" to calculate a prediction for the active user

There are several kinds of collaborative filtering approaches:

1. Active Filtering
2. Passive Filtering
3. Item based filtering

General features of these approaches are mentioned as follows.

2.1. Active Filtering

Active filtering uses a peer-to-peer approach. It is a system where people with similar interests rate products, and also shares the information with other people in the web. In cases where people who don't know well about the detailed structure of database want to find useful information, the active filtering can be useful and efficient method for them.

2.1.1. Exmaples

Here are examples of data collection in active filtering:

The system asks users to

i) rate an item on a sliding scale

ii) rank a collection of items from favorite to least favorite

iii) ask users to choose the best one between two items

iv) create a list of items that they like

2.1.2. Advantages

One of the advantages is an actual rating by a person who has already experienced the product. By the Active Filtering, the system can get a reasonable explanation and rank from a reliable source. Another benefit is that as the data is straightforward, the system doesn’t need to have complex functions to analyze the data.

2.1.3. Disadvantages

There are some disadvantages of active filtering. Firstly, it is highly probable that at the beginning time when there are not many enrolled users in the system, the opinion could be biased by the existing users. Maybe in this case the systems could make wrong expectation for new users based on the biased data from existing users. Secondly, requiring users to do something is bothersome, and, therefore, the users could avoid to participating in the rating work. This situation can cause that the system could have insufficient data, and, therefore, the expectations from system could be incorrect. The other problem is that the system knows nothing about the items' true contents or what they are about. The system only depends on preference ratings. So, the users could get suggestion only about products with high rating values.

Additionally, there are two more issues, the *First-Rater Problem* and *Cold-Start Problem*. The *First-Rater problem* is caused by new items that have not received any ratings from users. The system can recommend items which have rating scores. Therefore, the system is unable to suggest these items until they get ratings. Similarly, the *Cold-Start Problem* is caused by new users who have not submitted any ratings. Without any information about the user, the system is not able to guess the user's preferences and generate recommendations. Therefore, the new user cannot get any suggestion until she rates a few items.

2.2. Passive Filtering

Passive Filtering has great potential in the future. In this filtering, to gather information implicitly, a web browser is used to record a user’s preferences by observing their actions. The data from implicit filters are then used to determine what else the user will like and recommend potential items of interest.

2.2.1. Exmaples

Here are the data collection examples in passive filtering:

The system relies on the actions of users

i) Purchasing an item

ii) Repeatedly using, saving, printing an item

iii) Referring or linking to a site

iv) Number of times queried

v) Analyzing the user's social network and discover the like-minded users.

2.2.2 Features & Advantages

An important feature of passive filtering is using the time aspect to detect the user’s action. Therefore, the time record helps determining whether a user is scanning a document or fully reading the material [Parsons et al. 2004]. The greatest strength of the passive filtering is that it doesn’t require anything to users, and, at the same time, the systems can get every data from every user automatically. By such advantage, passive filtering can gather much more data than the previous active filtering.

2.3 Item-based filtering

In item based filtering, items are rated and used as parameters instead of users. Item-based algorithm mainly depends on the relationships between items. The system discriminates between interesting or uninteresting products for the user by examining the user’s purchase history. After then, recommendations for users are computed by finding items that are similar to other items the user has liked. This technique is well known by Amazon.com. In the Amazon.com, the systems recommends items to users by showing the message, "users who bought *something* also bought *some other things*".

Here are the basic steps of item-based filtering:

1) First is building item-relationships.

2) Using the item-relationships, the system can predict the user’s taste.

More detailed techniques will be mentioned in the following section 5.

3. Recommender Systems in real E-commerce

In here, introducing the real recommender systems of the representative e-business companies could be helpful to understand the basic backgrounds, the purpose, and the effectiveness of Recommender Systems. In addition, some predominant collaborative filtering (CF) techniques will be mentiond to help the understanding of the mechanisms of RS.

3.1. Music Recommender System in Apple iTunes

The Music Recommender System for iTunes is one of the most popular recommender systems. It is a software for iTunes, which is used for the integrated rating system, not for music download. It uses collaborative filtering technique to generate music recommendations. To see the simple steps how it works, the system takes ratings from each user's iTune play lists and compares the ratings with those of other iTune users who also have rated for their own musics.

One of the merits for this system is that generally each iTune users have large amounts of music files. Based on the abundant file information, the system can easily find out the taste of the user, predict what else the user would like, and recommend potential music items of interest. One interesting fact is that although the services is primarily for iTunes users, the system was invented not by Apple Computer, but by researchers at the University of Illinois at Urbana-Champaign. Currently, explicit development for improving the recommend system is still done by the student researchers.

3.2. Recommendation System in Amazon.com

At Amazon.com, they use recommendation algorithms to personalize the online store for each customer. The recommendation items are continuously being changed based on customer interests. It easily detects what the customer's job is, in what situation the customer is, and to what area the customer's taste is changed, and then, it suggests the precisely adequate book items to the customer. For now, the recommend system of Amazon.com is famous for its item-based filtering, whose computational algorithm scales independently of the number of customers and number of items [Linden et al. 2003].

To speak about the organization of the recommendation system of Amazon.com, it is comprised with 4 information collecting parts -- "Your Browsing History", "Rate These Items", "Improve Your Recommendations (Items you own)", and "Your Profile". The system determines customer's interest by examining the items that they have bought (Improve Your Recommendations), they already have, and they rated (Rate These Items). On the other hand, the system compares user's activity (Your Browsing History) on the site with those of other customers. On the basis of both the results of examination and comparison, system decides and recommends the product items that would be interesting to the customer.

One more new promising aspect, featuring passive filtering character, is the system offers the users to organize their social network through the site. It is assumed that the system could analyze the user's social network (Your Profile) and discover similar likes and dislikes of users. To analyze the user's social network, the system will needs to observe user's actions such as emails, coupons, friends & interesting people, and recommendations about products [Weigend, A. S. 2003]. The greatest strength of such passive filtering method is that it doesn’t require anything to users apparently. It interprets user behavior implicitly [John et al. 1998]. And, at the same time, the systems can get every data from every user automatically. So, it can gather much more data.

When considering with the three perspectives of collaborative filtering techniques -- active (Improve Your Recommendations, Rate These Items), passive (Your Browsing History, Your Profile), and item-based (Items you own) filterings, the Amazon.com's recommendation system is gradually being evolved by combining those filtering techniques [Good et al. 1999; Lee et al. 2008].

3.3. Recommendation System in like-i-like

"like-i-like.org" is a web site for movie recommendation system. The underlying assumptions of this recommendation system are "Those who agreed to one thing tend to agree to the similar thing again." and "People with similar taste can be advisor for each other." Based on those assumptions, the system tries to figure out if a new user has a similar taste with the existing user by detecting users who rated similar numbers (1~10) for specific movies, and categorizes them by their preferences. After then, users with similar interests rate products, and also share the information with other people through the web site continuously. It is a typical Active Filtering (User-based) System [Kim et al. 2004]. But, as mentioned in the Active Filtering, here are some anticipating problems to be studied more. If most of the participating users have biased opinion, the system would make undesirable results. Two other issues are the First-Rater Problem and Cold-Start Problem.

3.4. MusicSurfer beyond Existing Music Recommendation Systems

There is a trial to raise the accuracy of Music Recommendation System. To speak frankly, the existing music recommender systems like RS of iTunes know nothing about the nature of music files. They only rely on the rating values or the action of users. Only depending on such statistic data has limitations in finding more accurately matching music files.

In the existing music recommender system, the users can find musics in only two ways. First case is when the user knows the name of song or artist, and second case is when the system suggests a music which got high rating score from the other users who are like-minded.

As a new attempts to overcome such limitation of existing music recommender system, there is a MusicSurfer, content manage system based on music descriptions which were retrieved directly from the music file. To find similar musics, this system relies on the musical aspects like Timbre, Rhythm, Tempo, harmony, melody and so on [Cano et al. 2005]. To get those data, of course, we need techniques to analyze audio signals, identify the instrument, figure out the mood and so on. It really requires so huge amount of information and analysis skills. However, in the near future, this kind of technique will offer further higher accuracy of recommendation.

4. Algorithms of CF

In E-Commerce sites, recommendations can be made from statistics of the user population, overall top selling items, or past purchase history of users as a predictor or future items. The purpose of collaborative filtering is to predict the utility of items to a particular user based on a database of user ratings from a sample or population of other users. Collaborative algorithms mainly depend on the database. Of course, those data related to items are from the Active and Passive filtering techniques. To manipulate those data, there are two collaborative filtering algorithms, memory-based algorithm (User-based) and model-based algorithm (Item-based).

4.1. Memory-based algorithms

Memory-based algorithms operate over the entire user database to make predictions. The system utilizes statistical methods to find a set of users (known as *like-minded users* or *neighbors*) who have similar history of tastes with the active user [Sarwar et al. 2001].

To see how to predict the votes of a particular active user, there are some assumptions. $V_{i,j}$ is a set of votes. $V_{i,j}$ is corresponding to the vote for user i on item j. I_i is the set of items on which user i has voted. V_i, the mean vote for user i, becomes as shown below [John et al. 1998]:

$$\overline{v}_i = \frac{1}{|I_i|} \sum_{j \in I_i} v_{i,j}$$

Function [1]

To predict the votes of the active user (V_a), it is based on some partial information regarding the active user and a set of weights calculated from the user database. $\boldsymbol{P_{a,j}}$, the predicted vote of the active user for item j, is a weighted sum of the votes of the other user [John et al. 1998]:

$$p_{a,j} = \overline{v}_a + \kappa \sum_{i=1}^{n} w(a,i)(v_{i,j} - \overline{v}_i)$$

Function [2]

n: the number of users in the collaborative filtering database with nonzero

w(a,i): weights which reflect distance, correlation, or similarity between the active user and each user i

Ķ: a normalizing factor such that the absolute values of the weights sum to unity

Once a *neighborhood of users* is formed, the system uses different algorithms to combine the preferences of neighbors to produce a prediction or top-N recommendation for the active user. The technique, also known as nearest-neighbor or user-based collaborative filtering, is predominant and widely used in practice. The various collaborative filtering algorithms are distinguished according to the weight (*w*(*a*,*i*)) calculation.

4.1.1 Weight from Correlation

In the GroupLens project, the Pearson correlation coefficient was defined as the basis for the weights [Konstan et al. 1997]. The correlation between active user (*a*) and each user (*i*) can be expressed as shown below [John et al. 1998; Cöster and Svensson 2005]:

$$w(a,i) = \frac{\sum_j (v_{a,j} - \overline{v}_a)(v_{i,j} - \overline{v}_i)}{\sqrt{\sum_j (v_{a,j} - \overline{v}_a)^2 \sum_j (v_{i,j} - \overline{v}_i)^2}}$$

Function [3]

The summations over *j* are over the items for which both users *a* and *i* have recorded votes.

4.1.2 Weight from Vector Similarity

Another weight calculation function is based on the formalism that "The similarity between *two documents* is often measured by treating each document as a vector of word frequencies and computing the cosine of the angle formed by the two frequency vectors" [Salton and McGill 1983]. In here, users replace the role of two documents, titles replace the role of words, and votes take the role of word frequencies [John et al. 1998].

$$w(a,i) = \sum_j \frac{v_{a,j}}{\sqrt{\sum_{k \in I_a} v_{a,k}^2}} \frac{v_{i,j}}{\sqrt{\sum_{k \in I_i} v_{i,k}^2}}$$

Function [4]

4.2. Limitations of Memory-based CF Algorithm

Although the Memory-based collaborative filtering algorithm has been embraced by many recommender systems successfully, on the other hand, it has shown some problems.

4.2.1. Problems

1. Problem with Sparsity – As previously mentioned in the disadvantages of active filtering, the users could get narrow-ranged recommendations from the system. In the real E-commerce environment, although the system has tremendous number of product items, active users tend to purchase only small percentages of the total items. The system can only consider evaluated items for

recommendation. Therefore, the system based on nearest neighbor algorithms may be unable to make any good item recommendations for a specific user who has unique character and particular taste [Kim et al. 2004].

2. Problem with Scalability – As the number of items and users are growing, the typical collaborative algorithms should take much more time to check and compare each item and user to make unique recommendation list for every user.

4.2.2 Countermeasures for Problems

To overcome the sparsity problem, *semi-intelligent agents* were tried. These agents worked on evaluating and rating each item using syntactic features. By providing a dense ratings set, they helped alleviate coverage and improved quality [SARWAR et al. 1998]. But this solution couldn't figure out the fundamental problem of poor relationships among like-minded but sparse-rating users.

Another trial is *model-based approach* which is for the scalability problem [Sarwar et al. 2001]. The main idea of model-based approach is to analyze the user-item representation matrix to identify relations between different items and then to use these relations to compute the prediction score for a given user-item pair. The basic concept is that "*a user would be interested in purchasing items that are similar to the items the user liked earlier and would tend to avoid items that are similar to the items the user didn't like* *earlier"*. As this technique doesn't need to identify the neighborhood of similar users when a recommendation is requested, it tends to produce much faster resulting recommendations. More detailed algorithm is presented in the next section 4.3.

4.3. Model-based (Item-based) algorithms

Generally, it is regarded that model-based CF is directly related to item-based CF. Item-based CF is sometimes called model-based CF because its online recommendation is based on the prediction models that have been built in an offline batch process. *Model-based collaborative filtering*, in contrast to memory-based algorithms, uses the user database to estimate or learn *a model*, which is then used for predictions. Model-based collaborative filtering algorithms provide item recommendation by first developing a model of user ratings. *Model-based algorithms* take a probabilistic approach and execute the collaborative filtering process as computing the expected value of a user prediction with the user's ratings on the items.

On the assumption that the votes are integer valued with a range from 0 to *m*, the function of calculating the expected value of a vote is as shown below [John et al. 1998]:

$$p_{a,j} = E(v_{a,j}) = \sum_{i=0}^{m} \Pr\left(v_{a,j} = i | v_{a,k}, k \in I_a\right) i$$

Function [5]

In this function, the *probability expression* is the probability that the active user will have a particular vote value ($v_{a,j}$) for item *j*, given the previously observed votes (*$v_{a,k}$*). On the basis of the probability calculation, the model building process is performed by different *machine learning* algorithms. There are some algorithms such as *Bayesian network*, and *clustering* explained in detail as follows.

4.3.1 Clustering models approach

The clustering model treats collaborative filtering as a classification problem and works by clustering similar users in same class and estimating the probability that a particular user is in a particular class *C*, and from there computes the conditional probability of ratings. The probability model, the standard "naive" *Bayes formulation*, relates *joint probability of class and votes* to *a tractable set of conditional and marginal distributions* [John et al. 1998].

$$\Pr(C = c, v_1, \ldots, v_n) = \Pr(C = c) \prod_{i=1}^{n} \Pr(v_i | C = c)$$

Pr(*C*=*c*): the probabilities of class membership

Pr(*v_i*|*C*=*c*): conditional probabilities of votes

Function [6]

The left part of equal symbol (=) is the probability of observing an individual of a particular class and a complete set of vote values. To calculate the probability in "Function [5]" simply, the "**Pr**" part of Function [5] can be substituted by the "**Pr**" in the right side of Function [6].

In Function [6], Pr(C=c) and Pr(v_i|C=c) are estimated from a training set of user votes, the user database.

4.3.2 Bayesian Network model approach

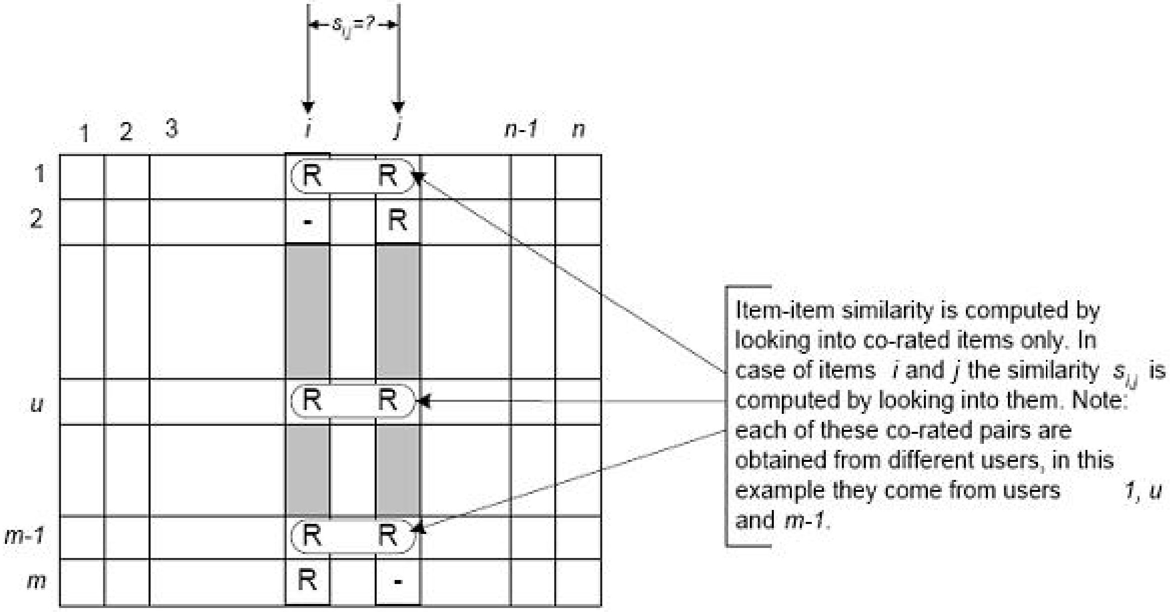
The Bayesian network model [John et al. 1998] is also for a probabilistic model for collaborative filtering problem. Bayesian network has a node corresponding to each item in the domain. The possible vote values for each item are related to the states of each node. A state involved with "no vote" is also included in those domains where there is no natural interpretation for missing data. Then an *algorithm for learning Bayesian networks* is used for the training data. The missing votes in the training data are indicated by the "no vote" value. The *learning algorithm* examines various model structures related to each item. After then, finally, each item will find and have a set of items that are the optimal predictors of its votes. Each conditional probability table is represented by a decision tree encoding the conditional probabilities for that node.

5. Item-based (Model-based) CF Algorithm

Unlike the memory-based (user-based) collaborative filtering algorithm, the item-based approach looks into "*the set of items" which* the target user has rated and computes *how similar they are to the target item i*, and then selects "*k most similar items* {i_1, i_2, ..., i_k}". At the same time their "*corresponding similarities* {s_{i1}, s_{i2}, ..., s_{ik}}" are also computed [Sarwar et al. 2001]. At first, if the system finds the *most similar items*, it computes *the* by calculating *a weighted average* of the target user's ratings about the similar items. *The similarity computation* and *the prediction generation* are very important aspects. The methodologies how to deduce those two aspects are explained as follows.

5.1. Item Similarity Computation

In the item-based collaborative filtering algorithm, the first step is to compute the similarity between items. The basic idea in similarity computation between two items *i* and *j* is to select the users who have rated both of these items, and then to apply a similarity computation technique to determine the similarity $s_{i,j}$.



Item-item similarity is computed by looking into co-rated items only. In case of items *i* and *j* the similarity $s_{i,j}$ is computed by looking into them. Note: each of these co-rated pairs are obtained from different users, in this example they come from users *1*, *u* and *m-1*.

Figure [1] [Sarwar et al. 2001]

In the Figure [1], we can easily see how it works. The matrix rows represent users and the columns represent items. To calculate the similarity between items, there are various methods like cosine-based method, correlation-based method, and adjusted-cosine method.

5.1.1 Cosine-based Method

The first method is cosine-based Method. In here, two items are regarded as two vectors in the *m* dimensional user-space. The similarity between items *i* and *j* is measured by computing the cosine of the angle between these two vectors as shown below.

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

"·": denotes the dot-product of the two vectors.

Function [7]

5.1.2 Correlation-based Method

The second, correlation-based method seeks correlation of items. In this method, the *Pearson-r* correlation $corr_{i,j}$ is applied to calculate the similarity between two items *i* and *j*. At first, the co-rated cases should be isolated (i.e., cases where the users rated both *i* and *j*) as shown in Figure [1]. This job is necessary for making the correlation computation accurate. In the Function [8], *U* is the set of users who both rated *i* and *j*.

$$sim(i,j) = corr_{i,j} = \frac{\sum_{u \in U}(R_{u,i} - \bar{R_i})(R_{u,j} - \bar{R_j})}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R_i})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R_j})^2}}.$$

R$_{u,i}$: denotes the rating of user *u* on item i

R¯$_i$: the average rating of the *i*-th item

Function [8]

5.1.3 Adjusted Cosine Method

The third, Adjusted cosine method is modified from the correlation-based method. Computing similarity using basic cosine measure in item-based case has one important drawback - the differences in rating scale between different users are not taken into account. The adjusted cosine similarity offsets this drawback by subtracting the corresponding user average from each co-rated pair. Formally, the similarity between items *i* and *j* using this scheme is expressed as Function [9]

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \bar{R_u})(R_{u,j} - \bar{R_u})}{\sqrt{\sum_{u \in U}(R_{u,i} - \bar{R_u})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \bar{R_u})^2}}.$$

R¯$_u$: the average of the *u*-th user's ratings.

Function [9]

5.2 Prediction Computation

Once the system makes a model using one of the similarity measures described above, it predict the rating for any user-item pair by using the idea of weighted sum. At first, all the items similar to the target item should be obtained, and from those similar items, the system picks items which the active user has rated. The next step is to look into the target user's ratings and use a technique to obtain predictions. Here are two such techniques for prediction.

5.2.1 Weighted Sum Approach

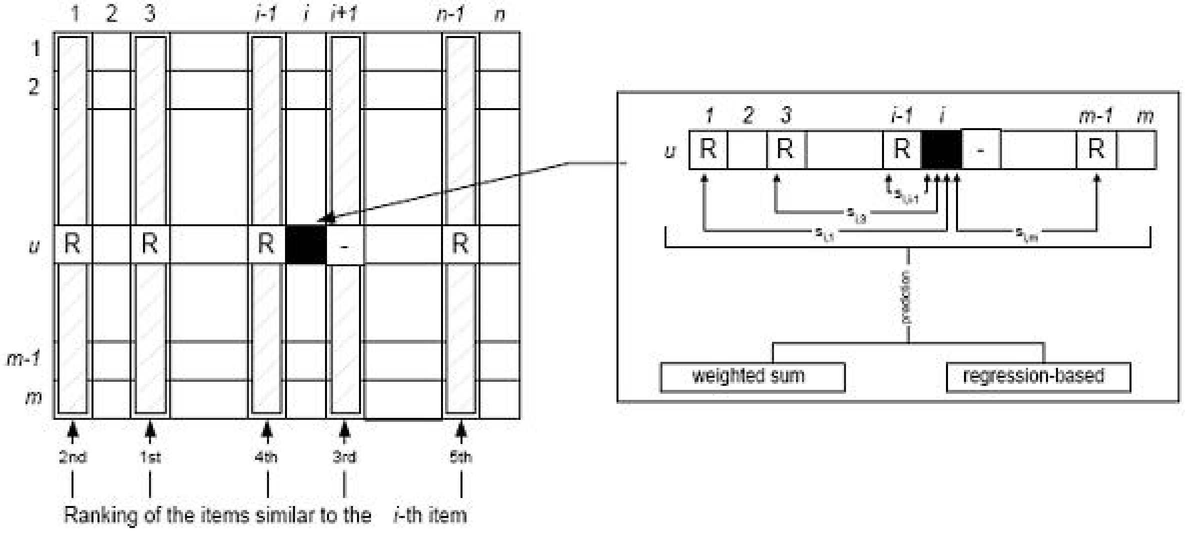


Figure [2] [Sarwar et al. 2001]

This method computes 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 ratings is weighted by the corresponding similarity $s_{i,j}$ between items *i* and *j*.

Formally, using the notion shown in Figure [2], we can denote the prediction $P_{u,i}$ as Function [10] [Sarwar et al. 2001]:

$$P_{u,i} = \frac{\sum_{\text{all similar items, N}} (s_{i,N} * R_{u,N})}{\sum_{\text{all similar items, N}} (|s_{i,N}|)}$$

Function [10]

Basically, this approach tries to figure out how the active user rates the similar items. The weighted sum is scaled by the sum of the similarity terms to make sure the prediction is within the predefined range.

5.2.2 Regression Approach

Although this approach is similar to the weighted sum method, instead of directly using the ratings of similar items, it uses an approximation of the ratings based on regression model. Often it happens that the similarities computed using cosine or correlation measures may be misleading in the sense that two rating vectors may be distant (in Euclidean sense) yet may have very high similarity. In that case using the raw ratings from similarity computation may cause incorrect prediction [Sarwar et al. 2001]. To overcome such situation, this model uses their approximated values $R^{-\prime}_{u,N}$ based on a linear regression model. The linear regression model function is shown in the Function [11] [Sarwar et al. 2001]:

$$\bar{R}'_N = \alpha \bar{R}_i + \beta + \epsilon$$

R_i and R_N: vectors of the target item *i* and the similar item *N*

ɑ and β: determined by going over both of the rating vectors

Є: the error of the regression model.

Function [11]

5.3. Item-to-item CF in Amazon.com

Item-to-item CF in Amazon.com is the most renowned item-based collaborative filtering. In here, we will see the how the system algorithm is implemented in Amazon.com.

As same as with the item-based algorithm shown before, to determine the most-similar matching item, the algorithm builds a *similar-items table* by finding items that customers tend to purchase together.

In the Amazon.com system, there is a *product-to-product matrix* by iterating through all item pairs and computing a similarity metric for each pair. The following iterative algorithm provides a better approach by calculating the similarity between a single product and all related products [Linden et al. 2003]:

<Algorithm>

```
For each item in product catalog, $I_1$
    For each customer C who purchased $I_1$
        For each item $I_2$ purchased by customer C
```

```
                Record that a customer purchased $I_1$ and $I_2$
        For each item $I_2$
            Compute the similarity between $I_1$ and $I_2$
```

It's possible to compute the similarity between two items in various ways, but a common method is to use the cosine measure described earlier, in which each vector corresponds to an item rather than a customer, and the vector's *M* dimensions correspond to customers who have purchased that item.

6. Conclusion

In here, we need to think about the performance and quality features of collaborative filtering algorithms. To consider about the performance, let's say that M is the number of customers and N is the number of product catalog items. In the user-based CF, its online computational complexity is O(MN) in the worst case because it needs to scan all the users once to find similar users and scan all the items to find the items that the similar users have selected [Linden et al. 2003]. In the item-based CF, this theoretical offline computation of the similar-items table requires so much time because of the complexity, $O(N^2M)$ as worst case. In real situation, it's closer to O(NM) because most customers have very few purchases. Moreover, customers who purchase best-selling titles save runtime much more. With calculated similar-items table, the algorithm can finds items similar to each of the user’s purchases and ratings, aggregates those items, and then recommends the most popular or correlated items very quickly [Sarwar et al. 2001]. Its online computation complexity is O(N) in the worst case, but usually the average is O (constant) [Kim et al. 2004]. Therefore, it can be said that item-based CF is more time efficient than user-based (memory-based) CF in the online.

In the other hand, although item-based (model-based) algorithm have better performance than the user-based (memory-based) algorithm, for the quality aspect, user-based can get better quality because larger model size can get better resulting recommendation. Therefore technique which combines the user-based and item-based effectively should be studied and developed in the future.

REFERENCES

CANO, P., KOPPENBERGER, M., AND WACK, N. 2005. An industrial-strength content-based music recommendation system. In Proceedings of the 28th Annual international ACM SIGIR Conference on Research and Development in information Retrieval (Salvador, Brazil, August 15 - 19, 2005). SIGIR '05. ACM, New York, NY, 673-673. DOI= http://doi.acm.org/10.1145/1076034.1076185

CÖSTER, R. AND SVENSSON, M. 2005. Incremental collaborative filtering for mobile devices. In Proceedings of the 2005 ACM Symposium on Applied Computing (Santa Fe, New Mexico, March 13 - 17, 2005). L. M. Liebrock, Ed. SAC '05. ACM, New York, NY, 1102-1106. DOI= http://doi.acm.org/10.1145/1066677.1066926

CHUN, I. AND HONG, I. 2001. The implementation of knowledge-based recommender system forelectronic commerce using Java expert system library, Industrial Electronics, Proceedings. ISIE 2001. IEEE International Symposium on Volume 3, Issue

GIL, A. AND GARCÍA, F. 2003. E-commerce recommenders: powerful tools for E-business. Crossroads 10, 2 (Dec. 2003), 6-6. DOI=http://doi.acm.org/10.1145/1027328.1027334

GOOD, N., SCHAFER, J. B., KONSTAN, J. A., BORCHERS, A., SARWAR, B., HERLOCKER, J., AND RIEDL, J. 1999. Combining collaborative filtering with personal agents for better recommendations. In Proceedings of the Sixteenth National Conference on Artificial intelligence and the Eleventh innovative Applications of Artificial intelligence Conference innovative Applications of Artificial intelligence (Orlando, Florida, United States, July 18 - 22, 1999). American Association for Artificial Intelligence, Menlo Park, CA, 439-446.

JOHN S., BREESE, DAVID HECKERMAN AND CARL KADIE. 1998. (July), Empirical analysis of predictive algorithms for collaborative filtering. In Proceedings of the Fourteenth Annual Conference on Uncertainty in Artificial Intelligence.

KIM, D., ATLURI, V., BIEBER, M., ADAM, N., AND YESHA, Y. 2004. A clickstream-based collaborative filtering personalization model: towards a better performance. In *Proceedings of the 6th Annual ACM international Workshop on Web information and Data Management* (Washington DC, USA, November 12 - 13, 2004). WIDM '04. ACM, New York, NY, 88-95. DOI= http://doi.acm.org/10.1145/1031453.1031470

KONSTAN, J. A., MILLER, B. N., MALTZ, D., HERLOCKER, J. L., GORDON, L. R., AND RIEDL, J. 1997. GroupLens: applying collaborative filtering to Usenet news. *Commun. ACM* 40, 3 (Mar. 1997), 77-87. DOI= http://doi.acm.org/10.1145/245108.245126

LEE, T. Q., PARK, Y., AND PARK, Y. 2008. A time-based approach to effective recommender systems using implicit feedback. Expert Syst. Appl. 34, 4 (May. 2008), 3055-3062. DOI= http://dx.doi.org/10.1016/j.eswa.2007.06.031

LINDEN, G., SMITH, B., YORK, J. 2003, Amazon.com recommendations: item-to-item collaborative filtering., Internet Computing, IEEE, Volume 7, Issue 1, Jan/Feb 2003 Page(s): 76 - 80., Digital Object Identifier 10.1109/MIC.2003.1167344

Parsons, J., Ralph, P., AND Gallagher, K. 2004. (July), Using viewing time to infer user preference in recommender systems., AAAI Workshop in Semantic Web Personalization, San Jose, California

Salton, G. and McGill, M. 1983. Introduction to Modern Information Retrieval. McGraw-Hill, New York.

SARWAR, B. M., KONSTAN, J. A., BORCHERS, A., HERLOCKER, J., MILLER, B., AND RIEDL, J. 1998. Using filtering agents to improve prediction quality in the GroupLens research collaborative filtering system. In Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work (Seattle, Washington, United States, November 14 - 18, 1998). CSCW '98. ACM, New York, NY, 345-354. DOI= http://doi.acm.org/10.1145/289444.289509

SARWAR, B., KARYPIS, G., KONSTAN, J., AND REIDL, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international Conference on World Wide Web* (Hong Kong, Hong Kong, May 01 - 05, 2001). WWW '01. ACM, New York, NY, 285-295. DOI= http://doi.acm.org/10.1145/371920.372071

Sotomayor, R., Carthy, J., AND Dunnion, J. 2005. The Design an Implementation of an Intelligent Online Recommender System., American Association for Artificial Intelligence. Technical Report FS-05-07 Published by The AAAI Press, Menlo Park, California.

WEIGEND, A. S. 2003. Analyzing customer behavior at Amazon.com. In Proceedings of the Ninth ACM SIGKDD international Conference on Knowledge Discovery and Data Mining (Washington, D.C., August 24 - 27, 2003). KDD '03. ACM, New York, NY, 5-5. DOI= http://doi.acm.org/10.1145/956750.956754