An Automatic Weighting Scheme for Collaborative Filtering

Rong Jin
Dept. of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824
rongjin@cse.msu.edu

Joyce Y. Chai
Dept. of Computer Science and Engineering
Michigan State University
East Lansing, MI 48824
jchai@cse.msu.edu

Luo Si
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
lsi@cs.cmu.edu

ABSTRACT
Collaborative filtering identifies information interest of a particular user based on the information provided by other similar users. The memory-based approaches for collaborative filtering (e.g., Pearson correlation coefficient approach) identify the similarity between two users by comparing their ratings on a set of items. In these approaches, different items are weighted either equally or by some predefined functions. The impact of rating discrepancies among different users has not been taken into consideration. For example, an item that is highly favored by most users should have a smaller impact on the user-similarity than an item for which different types of users tend to give different ratings. Even though simple weighting methods such as variance weighting try to address this problem, empirical studies have shown that they are ineffective in improving the performance of collaborative filtering. In this paper, we present an optimization algorithm to automatically compute the weights for different items based on their ratings from training users. More specifically, the new weighting scheme will create a clustered distribution for user vectors in the item space by bringing users of similar interests closer and separating users of different interests more distant. Empirical studies over two datasets have shown that our new weighting scheme substantially improves the performance of the Pearson correlation coefficient method for collaborative filtering.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval]:** Information Search and retrieval—*Information Filtering*

Keywords
Collaborative filtering, memory-based approach, leave one out method, item weighting scheme.

1. INTRODUCTION
Collaborative filtering predicts the utilities of items for a particular user based on the rating information for the same set of items given by many other users. In the past years, many collaborative filtering algorithms have been developed [1, 2, 3, 4, 5, 8, 12]. Generally, they can be categorized into two classes: memory-based algorithms and model-based algorithms [1]. To obtain a prediction for a particular user (i.e., a test user), the memory-based algorithms first identify users from the training database that are most similar to this user in terms of the rating patterns, and then combine those ratings together. This category includes the Pearson-correlation based approach [4], the vector similarity based approach [1], and the extended generalized vector space model [3]. Model-based approaches group together different users in the training database into a small number of classes based on their rating patterns. In order to predict the rating from a test user on a particular item, these approaches first categorize the test user into one of the predefined user classes and use the rating of the predicted class on the targeted item as the prediction. Algorithms within this category include Bayesian network approaches [1] and the aspect model [5]. Compared to the memory-based approaches, the model-based approaches have an advantage that only the profiles of models need to be stored. However, the memory-based approaches are usually much simpler than the model-based approaches and require little offline computation whereas model-based approaches usually have to spend many computation cycles on creating model profiles. Furthermore, the model-based approaches tend to assume that a small number of user classes are sufficient for modeling the rating patterns of many different users, and thus may lose the diversity of users. Finally, model-based approaches tend to perform worse than the memory-based approaches when the number of training users is small [15]. This is because ratings by only a small number of users are usually insufficient to create reliable clusters of users. To combine the strength of both approaches, hybrid methods such as 'Personality Diagnosis' approach [12] is developed, which outperforms several model-based and memory-based approaches.

Because of the simplicity and robustness, the memory-based approaches have been widely used in many real world applications. The key to the memory-based approaches is to identify the users within the training database that are most similar to the test user. The similarity between two different users is usually computed by matching the ratings given by the two users over the same set of items. For many memory-based approaches, items are treated with equal importance. Apparently, this is undesirable because the discrepancies in different items have not been taken into account. Some items may be highly favored by most users while others are rated significantly differently by different users. Intuitively, items with similar ratings should have smaller impact in determining the user-similarity than those with

SIGIR'04, *JULY 25–29, 2004, SHEFFIELD, SOUTH YORKSHIRE, UK.*

different ratings. In other words, items with a large variance in their ratings tend to be more important than items with a small variance in ratings. However, this is not necessarily true because a large variance in the ratings of an item can also arise from the difficulty in rating such an item by many users. As shown in Herlocker et al. [2], weighting items using their rating variance leads to slightly worse results than no weighting. In addition to variance, other weights such as inverse user frequency [1], entropy, and mutual information [13] have been studied in the previous literature. The results in [13] indicate that few weighting schemes for items are able to improve the performance of collaborative filtering. One of the reasons, in our opinion, is that most of the current weighting schemes are usually hand crafted and computed by predefined functions. It is unclear what global objectives those weighting schemes try to achieve.

To address this problem, in this paper, we present a new weighting scheme based on the leave-one-out (LOO) method. This work is built upon the intuition that the rating behavior of an individual user should be similar to the rating behaviors of *some but not all* other users. Therefore, a good weighting scheme for items should bring users of similar interests closer and meanwhile separate users of different interests further apart. Another way of describing this idea is to look at the user "distribution" over the item space. Consider the vector space spanned by different items and each user is a point in this space with the projection on each axis representing his rating of the corresponding item. The above intuition leads to a clustered distribution of user points in the item space, i.e., each user point is surrounded closely by several user points and is distant from others. In other words, the rating behavior of each user is well "explained" by several users, but is totally different from many others. A good weighting scheme for items should shape the original user "distribution" (i.e., a user distribution without using any weights for items) into such a clustered distribution. In this paper, we formalize this idea into a probabilistic optimization problem, in which the appropriate weights of items are found by maximizing the likelihood for each user to be similar to at least one of other users. Unlike most previous work on weighting schemes where item weights are determined by predefined functions, our approach automatically computes the appropriate weights for different items using the observed ratings provided by the training users.

The assumption of having a clustered distribution for user points in the item space is similar to the clustering assumption embedded in most model-based approaches. One important distinction is that our approach makes a less explicit assumption about the clustered distribution of user rating behaviors. Unlike many model-based approaches that separate users into several disjoin classes, our algorithm only requires that for each user, there always exists at least one user that is similar to that user. As a result, our algorithm does not have to specify the exact number of clusters, while most model-based approaches have to. Another view of the difference between our method and the model-based approaches is that most model-based approaches are generative models whose goal is to explain the observed ratings by different training users, while the new approach is a discriminative model whose goal is to explain why some training users are similar to each other and different from the others. As a result, most model-approaches search for the seeds of clusters that can be used to generate the ratings of different users. Our algorithm tries to find the weights for different items that bring each user close to the similar users and distant from the dissimilar ones**.** The disadvantage of a generative model versus a discriminative model is that a generative model has to explain the observed ratings of all items, even the ones that are useless in distinguishing user's interests. In contrast, the discriminative model only focuses on the important items by assigning them much higher weights. In effect, various studies of classification problems have shown that in general a discriminative model performs better than a generative model [18].

The rest of this paper is arranged as follows: Section 2 provides a brief description of several major approaches for collaborative filtering. Section 3 discusses the related work on item weighting. Section 4 describes the details of our weighting scheme for collaborative filtering. The results from empirical studies are presented in Section 5, followed by the conclusion in Section 6.

2. Background

In this section, we briefly describe several major approaches that have been used for collaborative filtering. First, we introduce the notations that are used throughout this paper. Let $X = \{x_1, x_2, ..., x_M\}$ be a set of items, $Y = \{y_1, y_2, ..., y_N\}$ be a set of training users, and y_t be the test user. Let $\{(x_{(1)}, y_{(1)}, r_{(1)}), ..., (x_{(L)}, y_{(L)}, r_{(L)})\}$ be all the ratings information in the training database. Each triple $(x_{(i)}, y_{(i)}, r_{(i)})$ indicates that item $x_{(i)}$ is rated as $r_{(i)}$ by the user $y_{(i)}$. For each user y, $X(y)$ denotes the set of items rated by him, $R_y(x)$ denotes the rating of item x by him, and $\bar{R}_y$ denotes his average rating. The rating scale goes from 1 to r_{max}.

2.1 Memory-based Approaches

Two commonly used memory-based algorithms are the Pearson Correlation Coefficient (PCC) algorithm (Resnick et al., 1994) and the Vector SPACE Similarity (VS) algorithm (Breese, Hckerman & Kadie, 1998). The main idea of these two algorithms is to calculate the similarities of a set of training users to a test user. To predicate the rating of an item given by the test user, the ratings from the training users for the item are averaged and weighted by their similarities to the test user. These two approaches differ in the computation of similarity. More specifically, the PCC method defines the similarity between two users $w_{y_t,y}$ as:

$$w_{y_t,y} = \frac{\sum_{x \in X(y) \wedge X(y_t)} (R_y(x) - \bar{R}_y)(R_{y_t}(x) - \bar{R}_{y_t})}{\sqrt{\sum_{x \in X(y) \wedge X(y_t)} (R_y(x) - \bar{R}_y)^2} \sqrt{\sum_{x \in X(y) \wedge X(y_t)} (R_{y_t}(x) - \bar{R}_{y_t})^2}}$$

while the VS method defines the similarity as:

$$w_{y_t,y} = \frac{\sum_{x \in X(y) \wedge X(y_t)} R_y(x) R_{y_t}(x)}{\sqrt{\sum_{x \in X(y)} R_y(x)^2} \sqrt{\sum_{x \in X(y_t)} R_{y_t}(x)^2}}$$

More details can be found in [1].

2.2 Model-based Approaches

Two popular model-based algorithms are the aspect model (AM) [5, 14] and the Personality Diagnosis model (PD) [12].

Aspect model is a probabilistic latent space model, which models individual preferences as a convex combination of preference factors [5, 14]. The latent class variable $z \in Z = \{z_1, z_2, ..., z_K\}$ is associated with each observed pair of a user and an item. The aspect model assumes that users and items are independent from each other given the latent class variable. Thus, the probability for each observation tuple (x, y, r) is calculated as follows:

$$p(r \mid x, y) = \sum_{z \in Z} p(r \mid z, x)\, p(z \mid y)$$

where $p(z|y)$ stands for the likelihood for the user y to be the class z and $p(r|z,x)$ stands for the likelihood of assigning the item x with the rating r by the class z of users. In [14], the ratings of each user are normalized to be norm distribution with zero mean and one variance. A Gaussian distribution is used for the parameter $p(r|z,x)$ and a multinomial distribution for $p(z|y)$.

Personality diagnosis approach treats each user in the training database as an individual model. To predicate the rating of an item by a test user, this approach first computes the likelihood for the test user to be in the 'model' of each training user and then uses the aggregate average of ratings for the item by the training users as the estimator. By assuming that the observed rating of the test user y_t on an item x is drawn from an independent normal distribution with the mean as the true rating $R_{y_t}^{True}(x)$, we have

$$p(R_{y_t}(x) \mid R_{y_t}^{True}(x)) \propto e^{-(R_{y_t}(x) - R_{y_t}^{True}(x))^2 / 2\sigma^2}$$

Then, the probability for the test user y_t to be in the model of any user y in the training database can be written as:

$$p(y_t \mid y) \propto \prod_{x \in X(y_t)} e^{-(R_y(x) - R_{y_t}(x))^2 / 2\sigma^2} \qquad (1)$$

Finally, the likelihood for the active user y' to rate an item x as r is computed as:

$$p(R_{y_t}(x) = r) \propto \sum_{y} p(R_{y_t} \mid R_y) e^{-(R_y(x) - r)^2 / 2\sigma^2}$$

Previous empirical studies have shown that the PD method is able to outperform several other approaches for collaborative filtering [5], including the PCC method, VS method and the Bayesian network approach.

3. Related Work

Automatically assigning appropriate weighting to items has not been well studied in the previous literatures. To our knowledge, there are only two major approaches, i.e., inverse user frequency weighting and the variance weighting.

The first approach borrows the TF.IDF weighting schemes [19] from information retrieval. In [1], the authors proposed the inverse user frequency (IUF) for weighting different items. This is an analogy to inverse document frequency (IDF) for information retrieval. More specifically, the IUF weight for an item x_k is defined as $w_k^{IUF} = \log(N / N_k)$ where N stands for the number of training users and N_k stands for the number of training users that have rated item x_k. Similar to the IDF weighting, the IUF weights favor the items that have been rated by only a few training users. Apparently, this may not be a good idea since items rated by fewer training users may not necessarily be useful in distinguishing users of different interests. In the empirical study conducted by Yu et al. [13], the IUF weighting has degraded the performance of the PCC method.

The second approach weights different items proportionally to their variance in ratings given by different users [2]. It is based on the intuition that an item with a large variance in its ratings is more valuable in discerning a user's interest than an item with a small variance. According to [2], the weight for item x_k is computed as $w_k^{VW} = \frac{\mathrm{var}_k - \mathrm{var}_{\min}}{\mathrm{var}_{\max}}$ where $\mathrm{var}_k = \frac{\sum_{i=1}^{N} (R_{y_i}(x_k) - \bar{R}_k)^2}{N-1}$. $\bar{R}_k$ represents the averaged ratings for item x_k. As discussed in Section 1, an item with a large variance in its ratings may not necessarily be a good item in discerning the user's interests because the large variance can be caused by the fact that the item is difficult to rate for most users.

4. Automatic Weighting Scheme for Items

The main idea of our new automatic weighting scheme is to find appropriate weights for items such that each user is brought closer to the users that share the similar interests and separated apart from the users that have different interests. Viewing from the item space, a good weighting scheme will result in a clustered distribution for different user points, in which each user point is closely surrounded by several neighbors and meanwhile distant from the points of dissimilar users. In the following sections, we first describe a probabilistic model for measuring the similarity between different users that can incorporate weights for different items. With the probabilistic description for user-similarity, we then formulate the problem of finding appropriate weights into an optimization problem and describe the solution.

4.1 A Conditional Exponential Model for Measuring User-Similarity

First, let $p(y_i \mid y_j)$ denote the likelihood for a user y_j to be similar to a user y_i. This likelihood can be viewed as the similarity between the users y_j and y_i. The larger the likelihood $p(y_i \mid y_j)$ is, the more similar is the user y_j to y_i. However, different from the standard similarity measurement, this likelihood is asymmetric, i.e., $p(y_i \mid y_j) \neq p(y_j \mid y_i)$. In other words, even though the user y_j may find that y_i is the most similar one among all the training users, user y_i can still find himself more similar to others than the user y_j. This is because $p(y_i \mid y_j)$ measures the similarity of user y_j to user y_i from the view point of y_j, while $p(y_j \mid y_i)$ measures the same similarity but from the view point of y_i.

Second, we introduce a pseudo user O to account for the users that are outside the set of training users. Therefore, the complete user set should be $\{y_1, y_2, ..., y_N, O\}$. Each user y_j is similar either to at least one of the remaining users in the training database or to the pseudo user O. Therefore, the sum of the probability $p(y_i \mid y_j)$ over different users has to be one, or,

$$p(O \mid y_j) + \sum_{\{i|i \neq j\}} p(y_i \mid y_j) = 1 \qquad (2)$$

Now, we need to express the likelihood $p(y_i \mid y_j)$ more explicitly. As indicated in Equation (1), the Personality Diagnosis method uses a generative Gaussian model to express a similar likelihood. This could be problematic since it requires the ratings of *every* item by the user y_i to be well "explained**"** by rating behavior of the user y_j without putting more emphasis on the items that are useful in discerning user's interests. As a result, a user can be determined as dissimilar to another user only because he does not provide ratings for the items that most users share a similar opinion. In order to incorporate the weights of items into the probabilistic model, we use the conditional exponential model [16] to describe the likelihood $p(y_i \mid y_j)$, i.e.

$$p(y_i \mid y_j) = \frac{1}{Z_j} \exp\left(\sum_{k=1}^{M} w_k v_{j,k} v_{i,k} \right) \qquad (3)$$

The variable $v_{i,k}$ denotes the normalized rating for an item x_k by a user y_i. It has a definition similar to the one used in Pearson Correlation Coefficient method, i.e.,

$$v_{i,k} = \begin{cases} \dfrac{R_{y_t}(x_k) - \bar{R}_{y_i}}{\sqrt{\sum_{x \in X(y_i)} (R_{y_i}(x) - \bar{R}_{y_i})^2}} & \text{if item } x_k \text{ is rated by user } y_t \\ 0 & \text{otherwise} \end{cases}$$

Z_j is a normalization factor that ensures the sum of conditional probability $p(y_i \mid y_j)$ to be one, which is further written as:

$$Z_j = \rho + \sum_{\{i|i \neq j\}} \exp\left(\sum_{k=1}^{M} w_k v_{j,k} v_{i,k} \right) \qquad (4)$$

Here we introduce a constant ρ to account for the probability mass $p(O \mid y_j)$. The likelihood for a user y_j to be similar to the pseudo user O $p(O \mid y_j)$ equals to ρ / Z_j. This expression indicates that when the user y_j shares similar ratings as many other training users over a large number of items, the likelihood $p(O \mid y_j)$ will be small because the normalization factor Z_j is large in this case. On the other hand, likelihood $p(O \mid y_j)$ will be large when the user y_j cannot find a similar rating behavior among other training users.

Note that Equation (3) is similar to the similarity measurement used for the Pearson Correlation Coefficient method in that both of them measure the user-similarity using the dot product of the normalized ratings by different users. However, they differ in the following two aspects:

1) Equation (3) introduces weights $\{w_i\}$ to determine the importance of different items and the weights can be automatically estimated using the leave-one-out method. The PCC method does not use any weights for items.

2) In Equation (3), likelihood $p(y_i \mid y_j)$ is asymmetric, i.e. $p(y_i \mid y_j) \neq p(y_j \mid y_i)$ while the similarity measurement used in the PCC method is symmetric.

4.2 Learning Weights for Items

According to the aforementioned idea, we need to adjust weights such that each training user is similar to at least one of the rest training users and distant from other dissimilar ones. To realize this goal, we take the leave-one-out approach. More specifically, for each training user, we measure how well the rating behavior of the training user can be "explained" by the rating behaviors of the rest training users. To measure such a quantity, for every training user y_i, we compute the following expression

$$\frac{1}{N-1} \sum_{\{j|j \neq i\}} p(y_i \mid y_j) \qquad (5)$$

This quantity can be interpreted as the average similarity of other training users to the user y_i. It is important to note that we use $p(y_i \mid y_j)$ in Equation (5) instead of $p(y_j \mid y_i)$. In other words, this average similarity is measured from the perspective of other training users (i.e., y_j), not the user y_i itself. This is important because, when replacing $p(y_i \mid y_j)$ in Equation (5) with $p(y_j \mid y_i)$, the expression will be simplified as $\left(1 - \rho / Z_j\right)/(N-1)$. This quantity is maximized when ρ is zero and therefore has nothing to do with the weights for items.

Equation (5) measures how well the rating behavior of a user y_i can be "explained" by the rest of training users. Since our goal is not just targeting one single training user, but rather ensuring that the rating behavior of any training user can be "explained" by the rating behaviors of others, we apply this leave-one-out approach to every training user. The overall measurement becomes the product of the expression in Equation (5) for all training users. The following equation gives an overall measurement:

$$\prod_{j=1}^{N} \left\{ \frac{1}{N-1} \sum_{\{j|j \neq i\}} p(y_i \mid y_j) \right\} \qquad (6)$$

The above quantity indicates globally how well the rating behavior of a single training user is "explained" by the rest training users. By maximizing this measurement, we will find the optimal weights for different items. Formally, the optimization problem is stated as follows:

$$\vec{w}^* = \arg\max_{\vec{w}} \prod_{j=1}^{N} \left\{ \frac{1}{N-1} \sum_{\{j|j \neq i\}} p(y_i \mid y_j; \vec{w}) \right\} \qquad (7)$$

In the above, the notation $\vec{w}$ stands for weights $\{w_1, w_2, ..., w_M\}$. We put weights $\vec{w}$ on the conditional side of the likelihood $p(y_i \mid y_j)$, i.e., $p(y_i \mid y_j; \vec{w})$, to emphasize that the likelihood is parameterized by weights $\vec{w}$.

To understand how the optimization problem in Equation (7) is able to twist the weights for items, consider the extreme case when an item is rated identically by all the training users. If a much larger weight is assigned to this item than other items, most likelihood $p(y_i \mid y_j)$ will be on the order of 1/*N*, which is a small value. As a result, the summation $\sum_{\{j|j\neq i\}} p(y_i \mid y_j; \vec{w})$ tends not to be large and the measurement in Equation (6) is not maximized. As a result, it is not optimal to assign high weights to the items that are rated similarly by most users.

In addition to the formulation in Equation (7), we can improve the robustness of the learning algorithm by introducing constraints on the weights. First, we restrict weights to be positive, or $w_k \geq 0$ for all *k*. This is because when a negative weight is assigned to an item, users with same ratings on the item will have a smaller similarity than users with different ratings on the item. Apparently, this contradicts our intuition. Second, to prevent a single weight from being too large, we introduce an upper bound on each weight, i.e., $w_k \leq \frac{100}{M}\sum_{m=1}^{M} w_m$. This constraint indicates that no weight is allowed to be one hundred times larger than the average of all weights. As a result, no item will be dominative in the similarity measurement.

Putting the constraints into the optimization problem in Equation (7), the final optimization problem becomes:

$$\vec{w}^* = \arg\max_{\vec{w}} \prod_{j=1}^{N} \left\{ \frac{1}{N-1} \sum_{\{j|j\neq i\}} p(y_i \mid y_j; \vec{w}) \right\}$$

subject to (8)

$$\forall k, 0 \leq w_k \leq \frac{100}{M}\sum_{m=1}^{M} w_m$$

Finding the optimal solution to Equation (8) is rather difficult due to the non-concave objective function and the constraints. We derive an optimization strategy for Equation (8) that uses the idea of auxiliary function. The main idea is to divide the optimization procedure into many steps. For each step, a simple auxiliary function, which is a lower bound of the original objective function, is used for optimization. Details about the optimization procedure are described in a separate paper.

Finally, to predict the ratings for the test user, we will simply add the weights to the standard memory-based approach. In our experiments, we used the Pearson Correlation Coefficient method as our basis. With the computed weights, the similarity in PCC method is computed as:

$$w_{y_t,y} = \frac{\sum_{x\in \tilde{X}(y)\wedge\tilde{X}(y_t)} w_x (R_y(x)-\bar{R}_y)(R_{y_t}(x)-\bar{R}_{y_t})}{\sqrt{\sum_{x\in \tilde{X}(y)\wedge\tilde{X}(y_t)} (R_y(x)-\bar{R}_y)^2}\sqrt{\sum_{x\in \tilde{X}(y)\wedge\tilde{X}(y_t)} (R_{y_t}(x)-\bar{R}_{y_t})^2}}$$

5. Experiment

We conducted a set of experiments to examine the effectiveness of our new weighting scheme. Particularly, we address the following three issues:

1) *How does the constant ρ influence the prediction accuracy*? The constant ρ controls the likelihood for a training user to be similar to a user outside the training database. The larger ρ is, the less likely it is for a training user to be similar to any other training users. Experiments are conducted to examine the impact of this constant on the final performance of the new weighting scheme.

2) *How is our weighting scheme compared to the existing weighting scheme for items*? Our approach is compared to two commonly used weighting schemes: the inverse user frequency (IUF) [1] and the variance of ratings [2], together with a detailed comparative analysis of the computed weights.

3) *How is the weighted memory-based approach compared to other approaches*? We compare the weighted memory-based approach (by incorporating our weighting scheme) to standard memory-based approach including the Pearson Correlation Coefficient (PCC) method, the Vector Similarity (VS) method, the Aspect Model (AM), and the Personality Diagnosis (PD) method.

5.1 Experiment Design

Two datasets of movie ratings are used in our experiments: MovieRating[1] and EachMovie[2]. For the EachMovie dataset, we extracted a subset of 2,000 users with more than 40 ratings. The global statistics of these two datasets as used in our experiments are summarized in Table 1.

Table 1: Characteristics of MovieRating and EachMovie.

	MovieRating	EachMovie
Number of Users	500	2000
Number of Items	1000	1682
Avg. # of rated Items/User	87.7	129.6
Number of Ratings	5	6

To compare algorithms more thoroughly, we experimented with several different configurations. For MovieRating, we altered the training size to be the first 20, 100, or 200 users, and for EachMovie we used the first 20, 200 or 400 users for training. The rest of users in both cases were used for testing. Furthermore, for each testing user, we varied the number of rated items provided by the test user from 5, 10, to 20. By varying the number of training users and the number of given items, we are able to test our weighting scheme for different configurations. Particularly, the experiment with only 20 training users is able to evaluate the robustness of our weighting scheme given a small number of training users.

The evaluation metric used in our experiments is the mean absolute error (MAE), which is the average absolute deviation of

[1] http://www.cs.usyd.edu.au/~irena/movie_data.zip

[2] http://research.compaq.com/SRC/eachmovie

the predicted ratings to the actual ratings on items the test user has actually voted.

$$MAE = \frac{1}{L_{Test}} \sum_{l} | r_{(l)} - R_{y_{(l)}}(x_{(l)}) | \qquad (9)$$

where L_{Test} denotes the number of the test ratings.

Finally, to predict ratings for a test user, the computed weights are incorporated into the Pearson Correlation Coefficient method as described in Section 3.

5.2 Experiments (1): Impact of Constant ρ

In the first experiment, we vary the constant ρ from 0.1, 0.5, to 2. The results of using different constant ρ for computing item weights over dataset MovieRating and EachMovie are presented in

Table 2: MAE using different ρ on MovieRating. A smaller value means a better performance.

Training Users Size	ρ	5 Items Given	10 Items Given	20 Items Given
20	0.1	0.852	0.817	0.796
	0.5	***0.851***	***0.816***	***0.795***
	2	0.855	0.831	0.812
100	0.1	0.849	0.813	0.789
	0.5	***0.846***	***0.812***	***0.785***
	10	0.849	0.818	0.791
200	0.1	0.846	0.810	0.779
	0.5	***0.842***	***0.807***	***0.777***
	2	0.850	0.815	0.778

Table 3: MAE using different ρ on EachMovie. A smaller value means a better performance.

Training Users Size	ρ	5 Items Given	10 Items Given	20 Items Given
20	0.1	1.208	1.156	1.136
	0.5	***1.207***	***1.155***	***1.133***
	2	1.217	1.164	1.150
200	0.1	1.175	1.115	1.086
	0.5	1.175	***1.114***	***1.085***
	2	***1.174***	1.115	1.090
400	0.1	1.163	1.114	1.087
	0.5	***1.162***	***1.106***	***1.075***
	2	1.168	1.114	1.089

Table 2 and 3, respectively. We can see that ρ=0.5 performs slightly better than ρ=0.1 and ρ=2 for most configurations. Since the constant ρ controls the likelihood for a training user to be similar to all other training users, these results indicate that best weights for items are obtained when we assign a medium chance for every training user to find itself to be dissimilar to other training users.

5.3 Experiment (2): Comparison to Existing Weighting Schemes

In this experiment, we compare our weighting scheme to two commonly used weighting schemes, i.e., inverse user frequency weighting (IUF) and variance weighting (VW). Both our

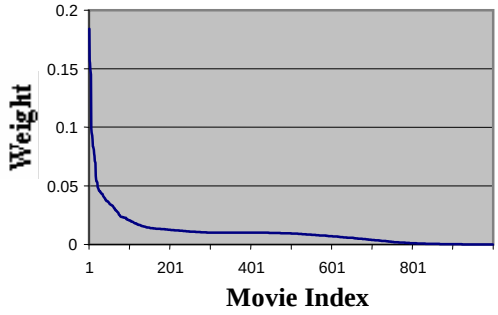

Figure 1: The distribution of weights computed for the movies in dataset 'MovieRaing' under the configuration of 100 training user.

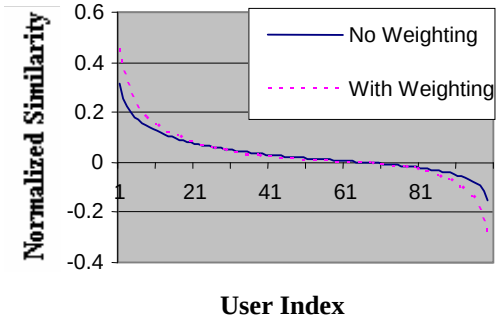

Figure 2: Distributions of normalized similarities for 'MovieRaing' database using 100 training users. The solid line represents the distribution computed without weights and the dot line represents the distribution computed with weights.

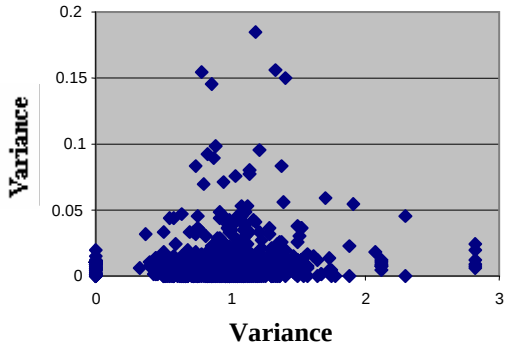

Figure 3: Distribution of weights for items versus the variance of ratings.

weighting scheme and the two weighting schemes to be compared are incorporated into the Pearson Correlation Coefficient method to predict ratings for test users. The results are listed in Table 4 and 5, together with the results for the Pearson Correlation Coefficient method without using any weighting scheme. The first observation is that, both the inverse user frequency weighting and the variance weighting do not improve the performance from the baseline method that does not use any weighting for items. This is actually consistent with the founding in [2, 13]. In contrast, our

Table 4: MAE using different weighting scheme on MovieRating. Title 'No' refers to Pearson Correlation Coefficient using no weighting schme; title 'VW' refers to the variance weighting; title 'IUF' refers to using inverse user frequency for weighting items; title 'New' refers to the new weighting scheme. A smaller value means a better performance.

Training Users Size	Weight	5 Items Given	10 Items Given	20 Items Given
20	No	0.878	0.835	0.808
	VW	0.875	0.842	0.817
	IUF	0.925	0.915	0.913
	New	***0.851***	***0.816***	***0.795***
100	No	0.875	0.829	0.804
	VW	0.881	0.845	0.818
	IUF	0.910	0.880	0.879
	New	***0.846***	***0.812***	***0.785***
200	No	0.874	0.829	0.796
	VW	0.879	0.845	0.812
	IUF	0.899	0.875	0.855
	New	***0.842***	***0.807***	***0.777***

Table 5: MAE using different weighting scheme on MovieRating. Title 'No' refers to Pearson Correlation Coefficient using no weighting schme; title 'VW' refers to the variance weighting; title 'IUF' refers to using inverse user frequency for weighting items; title 'New' refers to the new weighting scheme. A smaller value means a better performance.

Training Users Size	Weight	5 Items Given	10 Items Given	20 Items Given
20	NO	1.242	1.178	1.143
	VW	1.258	1.200	1.180
	IUF	1.302	1.262	1.251
	New	***1.207***	***1.155***	***1.133***
200	NO	1.226	1.153	1.119
	VW	1.258	1.196	1.174
	IUF	1.238	1.206	1.183
	New	***1.175***	***1.114***	***1.085***
400	NO	1.218	1.154	1.111
	VW	1.253	1.1981	1.176
	IUF	1.258	1.218	1.212
	New	***1.162***	***1.106***	***1.075***

new weighting scheme is able to boost the prediction accuracy for all configurations.

To better understand why our weighting scheme improves the performance of Pearson Correlation Coefficient method, we first examine the distribution of weights for different movies. Figure 1 plots the computed weight distribution for the MovieRating dataset given 100 training users. As indicated in Figure 1, the distribution of weights for different movies is rather skewed: only 2% of items are weighted larger than 0.05, and only 20% of items are weighted larger than 0.015; more than 50% of items are weighted equal or less than 0.01. This skewed weight distribution indicates that only 2% of items are very important in determining the user-similarity and around 50% of items are almost ignorable in the computation of user-similarity. This is similar to the phenomenon in text categorization, in which people found that only a small number of words are important in determining the category for a document [20].

Second, we examine how the computed weights shape the distribution of user-similarities. Figure 2 displays the distribution of normalized similarities calculated based on the weights. The

Table 6: MAE on MovieRating for the Vector Similarity (VS) method, Pearson Correlation Coefficient using the proposed weighting scheme (PCC+), Personality Diagnosis (PD) method, and the Aspect Model (AM) method. A smaller value means a better performance.

Training Users Size	Methods	5 Items Given	10 Items Given	20 Items Given
20	VS	0.912	0.840	0.812
	PD	0.888	0.882	0.875
	AM	0.982	0.976	0.958
	PCC+	***0.851***	***0.816***	***0.795***
100	VS	0.859	0.834	0.823
	PD	0.839	0.826	0.818
	AM	0.882	0.856	0.836
	PCC+	***0.846***	***0.812***	***0.785***
200	VS	0.862	0.950	0.854
	PD	0.835	0.816	0.806
	AM	0.891	0.850	0.818
	PCC+	***0.842***	***0.807***	***0.777***

Table 7: MAE on EachMovie for the Vector Similarity (VS) method, Pearson Correlation Coefficient using the proposed weighting scheme (PCC+), Personality Diagnosis (PD) method, and the Aspect Model (AM) method. A smaller value means a better performance.

Training Users Size	Methods	5 Items Given	10 Items Given	20 Items Given
20	VS	1.24	1.19	1.17
	PD	1.25	1.24	1.23
	AM	1.28	1.24	1.23
	PCC+	***1.21***	***1.16***	***1.13***
200	VS	1.25	1.24	1.26
	PD	1.19	1.16	1.15
	AM	1.27	1.18	1.14
	PCC+	***1.18***	***1.11***	***1.09***
400	VS	1.32	1.33	1.37
	PD	1.18	1.16	1.15
	AM	1.28	1.19	1.16
	PCC+	***1.16***	***1.11***	***1.08***

distribution of the similar similarities without using weights is shown in Figure 2. Here, a normalized similarity of a user y_i to a user y_j is computed as $sim(y_i, y_j) / \max_{k \neq i}(sim(y_i, y_k))$. Both distributions are computed under the configuration of 100 training users. Observed from Figure 2, we can see that by introducing weights for items, a large similarity becomes even larger and a small similarity becomes even smaller. In other words, the incorporation of weights makes similar users more similar and dissimilar users more dissimilar. This effect is consistent with the intuition that stated in Section 4, namely a good weighting scheme for items should bring similar users closer and separate dissimilar users further away.

Third, we examine the correlation between the computed weights and the variance of ratings. The distribution of weights versus the variance of ratings is plotted in Figure 3. Unlike the variance weighting scheme, where the weight of an item is proportional to the variance of ratings for the item, the plot in Figure 3 indicates that the items with medium variances are assigned with large weights and items with both large and small variances are actually assigned with only small weights. Again, this is consistent with

the studies in information retrieval [17], where words with medium frequency are usually most informative.

5.4 Experiments (3): Comparison with Other Approaches for Collaborative Filtering

This experiment compares the Pearson Correlation Coefficient approach using our weighting scheme to the other three methods: the Vector Similarity (VS) method, the Aspect Model (AM) approach, and the Personality Diagnosis (PD) method. Table 6 summarizes the results for these three methods. Clearly, the Pearson Correlation Coefficient method using our weighting scheme (referred as 'PCC+') outperforms the other three methods in all configurations. This experiment shows that our new weighting scheme is effective in improving the prediction accuracy for collaborative filtering.

6. Conclusion

In this paper, we present a novel algorithm to automatically determine appropriate weights for different items for collaborative filtering. Unlike the previous weighting schemes where weights are computed using a predefined function, our algorithm automatically learns weights for different items from the ratings given by training users. The main idea behind this weighting scheme is to adjust weights of items to bring similar users closer and separate dissimilar users further apart. Based on this idea, an optimization approach is developed to efficiently search for a weighting scheme. Empirical studies have shown that our new weighting scheme can be incorporated to improve the performance of Pearson Correlation Coefficient method substantially under many different configurations. Comparison with two existing weight schemes and three different approaches also indicates the effectiveness of our approach for collaborative filtering.

7. REFERENCES

[1] J. S. Breese, D. Heckerman and C. Kadie, Empirical Analysis of Predictive Algorithms for Collaborative Filtering, Proceeding of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI), 1998.

[2] J. L. Herlocker, J. A. Konstan, A. Brochers and J. Riedl. An Algorithm Framework for Performing Collaborative Filtering. In Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR), 1999.

[3] I. M. Soboroff and C. Nicholas. Collaborative Filtering and the Generalized Vector Space Model. In Proceedings of the 23rd Annual International Conference on Researech and Development in Information Retrieval (SIGIR), 2000.

[4] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom and J. Riedl, Grouplens: An Open Architecture for Collaborative Filtering of Netnews. In Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work, pages 175-186, 1994.

[5] T. Hofmann and J. Puzicha, Latent Class Models for Collaborative Filtering, In Proceedings of International Joint Conference on Artificial Intelligence (UAI), 1999.

[6] I. M. Soboroff and Charles K. Nicholas. Combining Content and Collaboration in Text Filtering. In Proceedings of the IJCAI'99 Workshop on Machine Learning for Information Filtering, 1999

[7] P. Melville, R. J. Mooney, R. Nagarajan, Content-Boosted Collaborative Filtering for Improved Recommendations. In Proceedings of the the Eighteenth National Conference on Artificial Intelligence (AAAI), 2002.

[8] SWAMI: a framework for collaborative filtering algorithm development and evaluation. In Proceedings of the 23rd Annual International Conference on Researech and Development in Information Retrieval (SIGIR), 2000.

[9] http://www.cs.usyd.edu.au/~irena/movie_data.zip

[10] http://research.compaq.com/SRC/eachmovie

[11] A. Dempster, N. Laird and D. Rubin. Maximum Likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, B 39:1-38, 1977.

[12] D. M. Pennock, E. Horvitz, S. Lawrence and C. L. Giles, Collaborative Filtering by Personality Diagnosis: A Hybrid Memory- and Model-Based Approach, in Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI), 2000.

[13] K. Yu, Z. Wen, X.W Xu, and M. Ester, Feature Weighting and Instance Selection for Collaborative Filtering, in Proceedings of 2nd International Workshop on Management of Information on the Web: Web Data and Text Mining (MIW'01), 2001

[14] T. Hofmann, Gaussian Latent Semantic Models for Collaborative Filtering, In the Proceedings of the 26th Annual International ACM SIGIR Conference, 2003

[15] L. Si and R. Jin, A Flexible Mixture Model for Collaborative Filtering, In the Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003), 2003.

[16] A. Berger, V. Pietra and S. Pietra, A Maximum Entropy Approach to Natural Language Processing. In Computational Linguistics, 22:39--71, 1996.

[17] V. Rejisbergen, Information Retrieval, 1979

[18] A.Y. Ng and M.I. Jordan. On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. NIPS 14,2002

[19] G. Salton and C. Buckley, Term-weighting Approaches in Automatic Text Retrieval. Information Processing and Management, 24, 513-523, 1998

[20] D.D. Lewis, Y. Yang, T. Rose and F. Li. RCV1: A New Text Categorization Test Collection. Journal of Machine Learning Research 2003.