# A social network-based recommender system (SNRS)

**Chapter** · June 2010

DOI: 10.1007/978-1-4419-6287-4_4



**2 authors:**

Jianming He
University of California, Los Angeles
**3** PUBLICATIONS **465** CITATIONS
SEE PROFILE

Wesley W. Chu
University of California, Los Angeles
**281** PUBLICATIONS **7,185** CITATIONS
SEE PROFILE

# A Social Network-Based Recommender System (SNRS)

Jianming He and Wesley W. Chu

Computer Science Department
University of California, Los Angeles, CA 90095

jmhek@cs.ucla.edu, wwc@cs.ucla.edu

**Abstract.** Social influence plays an important role in product marketing. However, it has rarely been considered in traditional recommender systems. In this paper we present a new paradigm of recommender systems which can utilize information in social networks, including user preferences, item's general acceptance, and influence from social friends. A probabilistic model is developed to make personalized recommendations from such information. We extract data from a real online social network, and our analysis of this large dataset reveals that friends have a tendency to select the same items and give similar ratings. Experimental results on this dataset show that our proposed system not only improves the prediction accuracy of recommender systems but also remedies the data sparsity and cold-start issues inherent in collaborative filtering. Furthermore, we propose to improve the performance of our system by applying semantic filtering of social networks, and validate its improvement via a class project experiment. In this experiment we demonstrate how relevant friends can be selected for inference based on the semantics of friend relationships and finer-grained user ratings. Such technologies can be deployed by most content providers.

## 1 Introduction

In order to overcome information overload, recommender systems have become a key tool for providing users with personalized recommendations on items such as movies, music, books, news, and web pages. Intrigued by many practical applications, researchers have developed algorithms and systems over the last decade. Some of them have been commercialized by online venders such as Amazon.com, Netflix.com, and IMDb.com. These systems predict user preferences (often represented as numeric ratings) for new items based on the user's past ratings on other items. There are typically two types of algorithms for recommender systems -- *content-based methods* and *collaborative filtering*. Content-based methods measure the similarity of the recommended item (target item) to the ones that a target user (i.e., user who receives recommendations) likes or dislikes [25, 22, 30] based on item attributes. On the other hand, collaborative filtering finds users with tastes that are similar to the target user's based on their past ratings. Collaborative filtering will then make recommendations to the target user based on the opinions of those similar users [3, 5, 27].

Despite all of these efforts, recommender systems still face many challenging problems. First, there are demands for further improvements on the prediction ac-

curacy of recommender systems. In October 2006, Netflix announced an open competition with the grand prize of $1,000,000 for the best algorithm that predicts user ratings for films (http://www.netflixprize.com). The improvement in the prediction accuracy can increase user satisfaction, which in turn leads to higher profits for those e-commerce websites. Second, algorithms for recommender systems suffer from many issues. For example, in order to measure item similarity, content-based methods rely on explicit item descriptions. However, such descriptions may be difficult to obtain for items like ideas or opinions. Collaborative filtering has the *data sparsity* problem and the *cold-start* problem [1]. In contrast to the huge number of items in recommender systems, each user normally only rates a few. Therefore, the user/item rating matrix is typically very sparse. It is difficult for recommender systems to accurately measure user similarities from those limited number of reviews. A related problem is the cold-start problem. Even for a system that is not particularly sparse, when a user initially joins, the system has none or perhaps only a few reviews from this user. Therefore, the system cannot accurately interpret this user's preference.

To tackle those problems, two approaches have been proposed [3, 29, 21, 23]. The first approach is to condense the user/item rating matrix through dimensionality reduction techniques such as *Singular Value Decomposition (SVD)* [3, 29]. By clustering users or items according to their latent structure, unrepresentative users or items can be discarded, and thus the user/item matrix becomes denser. However, these methods do not significantly improve the performance of recommender systems, and sometimes make the performance even worse.

The second approach is to "enrich" the user/item rating matrix by 1) introducing default ratings or implicit user ratings, e.g., the time spent on reading articles [23]; 2) using half-baked rating predictions from content-based methods [21]; or 3) exploiting transitive associations among users through their past transactions and feedback [12]. These methods improve the performance of recommender systems to some extent. In this paper we try to solve these problems from a different perspective. In particular, we propose a new paradigm of recommender systems by utilizing information in social networks, especially that of *social influence*.

Traditional recommender systems do not take into consideration explicit social relations among users, yet the importance of social influence in product marketing has long been recognized [32, 35]. Intuitively, when we want to buy a product that is not familiar, we often consult with our friends who have already had experience with the product, since they are those that we can reach for immediate advice. When friends recommend a product to us, we also tend to accept the recommendation because their inputs are trustworthy. Many marketing strategies that have leveraged this aspect of human nature have achieved great success. One classic example is the Hotmail's free email service. The marketing strategy of Hotmail is to attach a promotion message at the bottom of every outgoing email: “Get your private, free email at http://www.hotmail.com.” People who receive the email will sign up and then further propagate this promotion message. As a result, the number of Hotmail user accounts grew from zero to 12 million in 18 months on only a $500,000 advertising budget—thereby out-performing many conven-

tional marketing strategies [14]. Thus, social influences play a key role when people are making decisions of adopting products.

Additionally, the integration of social networks can theoretically improve the performance of current recommender systems. First, in terms of the prediction accuracy, the additional information about users and their friends obtained from social networks improves the understanding of user behaviors and ratings. Therefore, we can model and interpret user preferences more precisely, and thus improve the prediction accuracy. Second, with friend information in social networks, it is no longer necessary to find similar users by measuring their rating similarity, because the fact that two people are friends already indicates that they have things in common. Thus, the data sparsity problem can be alleviated. Finally, for the cold-start issue, even if a user has no past reviews, recommender system still can make recommendations to the user based on the preferences of his/her friends if it integrates with social networks. All of these intuitions and observations motivate us to design a new paradigm of recommender systems that can take advantage of information in social networks.

The recent emergence of online social networks (OSNs) gives us an opportunity to investigate the role of social influence in recommender systems. With the increasing popularity of Web 2.0, many OSNs, such as Myspace.com, Facebook.com, and Linkedin.com have emerged. Members in those networks have their own personalized space where they not only publish their biographies, hobbies, interests, blogs, etc., but also list their friends. Friends or visitors can visit these personal spaces and leave comments. Note that in this paper we define *friends* as any two users who are connected by an explicit social link. We define *immediate friends* as those friends who are just one hop away from each other in a social network graph, and *distant friends* as friends who are multiple hops away. OSNs provide platforms where people can place themselves on exhibit and maintain connections with friends. As OSNs continue to gain more popularity, the unprecedented amount of personal information and social relations improve social science research where it was once limited by a lack of data.

In our research, we are interested in the role of explicit social relations in recommender systems, such as how user preferences or ratings are correlated with those of friends, and how to use such correlations to design a better recommender system. In particular, we design an algorithm framework which makes recommendations based on user's own preferences, the general acceptance of the target item, and the opinions from social friends. We crawl a real online social network from Yelp.com, and perform extensive analysis on this dataset. Some of the key questions, such as whether or not friends tend to select the same item, and whether or not friends tend to give similar ratings, have been studied in this dataset. We also use this dataset to evaluate the performance of our proposed system on the prediction accuracy, data sparsity, and cold-start. The experimental results of our system show significant improvement against traditional collaborative filtering in all of those aspects. For example, the prediction accuracy has improved by 17.8% compared to traditional collaborative filtering. Furthermore, we propose to use the semantics of friend relationships and finer-grained user ratings to improve the prediction accuracy.

The remainder of the paper is organized as follows. First, in Section 2 we give a background of traditional collaborative filtering algorithms. Then we formally propose a social network-based recommender system in Section 3. In Section 4 we introduce the dataset that we crawled from Yelp, and present some analytical studies on this dataset. Following that, we evaluate the performance of the proposed system on the Yelp dataset in Section 5. In Section 6 we propose to further improve the prediction accuracy of the system by applying semantic filtering of social networks, and validate its improvement via a class experiment. In Section 7 we review related studies, and conclude in Section 8.

## 2 Background

After the pioneering work in the Grouplens project in 1994 [27], collaborative filtering (CF) soon became one of the most popular algorithms in recommender systems. Many variations of this algorithm have also been proposed [2, 21, 11, 36, 13]. In this paper we will use the traditional CF as one of the comparison methods. Therefore, the remainder of this section will focus on this algorithm.

The assumption of CF is that people who agree in the past tend to agree again in the future. Therefore, CF first finds users with taste similar to the target user's. CF will then make recommendations to the target user by predicting the target user's rating to the target item based on the ratings of his/her top-$K$ similar users. User ratings are often represented by discrete values within a certain range, e.g., one to five. A one indicates an extreme dislike to the target item, while a five shows high praise. Let $R_{UI}$ be the rating of the target user $U$ on the target item $I$. Thus, $R_{UI}$ is estimated as the weighted sum of the votes of similar users as follows.

$$R_{UI} = \overline{R_U} + Z \sum_{V \in \Psi} w(U,V) \times (R_{VI} - \overline{R_V}), \tag{1}$$

where $\overline{R_U}$ and $\overline{R_V}$ represent the average ratings of the target user $U$ and every user $V$ in $U$'s neighborhood, $\Psi$, which consists of the top-$K$ similar users of $U$. $w(U, V)$ is the weight between users $U$ and $V$, and $Z = \frac{1}{\sum_V w(U,V)}$ is a normalizing constant to normalize total weight to one. Specifically, $w(U, V)$ can be defined using the Pearson correlation coefficient [27].

$$w(U,V) = \frac{\sum_I (R_{UI} - \overline{R_U})(R_{VI} - \overline{R_V})}{\sqrt{\sum_I (R_{UI} - \overline{R_U})^2 \sum_I (R_{VI} - \overline{R_V})^2}} \tag{2}$$

where the summations over $I$ are over the common items for which both user $U$ and $V$ have voted.

Other variations to this algorithm include different weighting techniques. For example, when two users have less than 50 co-rated items, [11] proposed to insert a significance weighting factor of $n/50$ to the original weight, where $n$ is the number of co-rated items. As we can see, traditional collaborative filtering and its variations do not utilize the semantic friend relations among users in recommender systems; however, this is essential to the buying decisions of users. In the following sections, we are going to present a new paradigm of recommender systems which improves the performance of traditional recommender systems by using the information in social networks.

## 3 A Social Network-Based Recommender System

Before we introduce the system, let us first show a typical scenario. Angela wants to watch a movie on a weekend. Her favorite movies are dramas. From the Internet, she finds two movies particularly interesting, "Revolutionary Road" and "The Curious Case of Benjamin Button." These two movies are all highly rated in the message board at Yahoo Movies. Because she cannot decide which movie to watch, she calls her best friend Linda whom she often hangs out with. Linda has not viewed these two movies either, but she knew that one of her officemates had just watched "Revolutionary Road" and highly recommended it. So Linda suggests "Why don't we go to watch Revolutionary Road together?" Angela is certainly willing to take Linda's recommendation, and therefore has a fun night at the movies with her friend. If we review this scenario, we can see at least three factors that really contribute to the Angela's final decision. The first factor is Angela's own preference for drama movies. If Angela did not like drama movies, she would be less likely to pick something like "Revolutionary Road" to begin with. The second factor is the public reviews on these two movies. If these movies received horrible reviews, Angela would most likely lose interest and stop any further investigation. Finally, it is the recommendation from Angela's friend, Linda, that makes Angela finally choose "Revolutionary Road." Interestingly, Linda's opinion is also influenced by her officemate. If we recall the decisions that we make in our daily life, such as finding restaurants, buying a house, and looking for jobs, many of them are actually influenced by these three factors.

Figure 3.1 further illustrates how these three factors impact customers' final buying decisions. Intuitively, a customer's buying decision or rating is decided by both his/her own preference for similar items and his/her knowledge about the characteristics of the target item. A user's preference, such as Angela's interest in drama movies, is usually reflected from the user's past ratings to other similar items, e.g. the number of drama movies that Angela previously viewed and the average rating that Angela gave to those movies. Knowledge about the target item can be obtained from public media such as magazines, television, and the Internet. Meanwhile, the feedbacks from friends are another source of knowledge regarding

the item, and they are often more trustworthy than advertisements. When a user starts considering the feedbacks from his/her friends, he/she is then influenced by his/her friends. Note that this influence is not limited to that from our immediate friends. Distant friends can also cast their influence indirectly to us; e.g., Angela was influenced by Linda's officemate in the previous scenario. Each one of these three factors has an impact on a user's final buying decision. If the impact from all of them is positive, it is very likely that the target user will select the item. On the contrary, if any has a negative influence, e.g., very low ratings in other user reviews, the chance that the target user will select the item will decrease. With such an understanding in mind, we are going to propose a social network-based recommender system *(SNRS)* in the following subsections. As we mentioned, social influences can come from not only immediate friends but also distant friends. The techniques for handling these types of influences are different. We shall begin with the *immediate friend inference*, in which we only consider influences from immediate friends. Then, in the *distant friend inference*, we will describe how we incorporate influences from distant friends via leveraging the immediate friend inference.

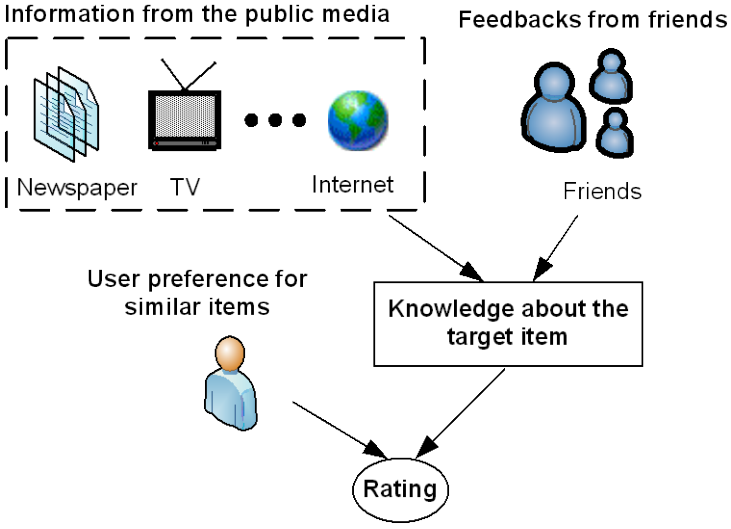



Figure 3.1: The three factors that influence a customer's buying decision: user preference for similar items, information regarding the target item from the public media, and feedbacks from friends.

## *3.1 Immediate Friend Inference*

We introduce the following naming conventions for the variables used in this paper. We use capitalize letters to represent variables, and use capitalize and bold

letters to represent the corresponding sets. The value for each variable or variable set is represented by the corresponding lowercase letter.

Formally, let us consider a social network as a graph $G = (\boldsymbol{U}, \boldsymbol{E})$ in which $\boldsymbol{U}$ represents nodes (users) and $\boldsymbol{E}$ represents links (social relations). Each user $U$ in $\boldsymbol{U}$ has a set of attributes $\boldsymbol{A}_U$ as well as immediate neighbors (friends) $\boldsymbol{N}(U)$ such that if $V \in \boldsymbol{N}(U)$, $(U, V) \in \boldsymbol{E}$. The values of attributes $\boldsymbol{A}_U$ are represented as $\boldsymbol{a}_U$. Moreover, a recommender system contains the records of users' ratings, which can be represented by a triple relation of $T = (\boldsymbol{U}, \boldsymbol{I}, \boldsymbol{R})$ in which $\boldsymbol{U}$ is the users in the social network $G$; $\boldsymbol{I}$ is the set of items (products or services), and each item $I$ in $\boldsymbol{I}$ has a set of attributes $\boldsymbol{A}'_I$. $\boldsymbol{R}$ stands for the ratings such that each $R_{UI}$ in $\boldsymbol{R}$ is user $U$'s rating on item $I$. $R_{UI}$ has a numeric value $k$ (e.g. $k \in \{1, 2, \ldots 5\}$). Moreover, we define $\boldsymbol{I}(U)$ as the set of items that user $U$ has reviewed, and refer to the set of reviewers of item $I$ as $\boldsymbol{U}(I)$. The goal of this recommender system is to predict $Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I, \boldsymbol{A} = \boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\})$; i.e., the probability distribution of the target user $U$'s rating on the target item $I$ given the attribute values of item $I$, the attribute values of user $U$, and the ratings on item $I$ rated by $U$'s immediate friends. Once we obtain this distribution, $R_{UI}$ is calculated as the expectation of the distribution. Items with high estimated ratings will be recommended to the target user, and users with high estimated ratings on the target item are the potential buyers.

In order to estimate $Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I, \boldsymbol{A} = \boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\})$, we adopt the naive Bayes assumption which assumes that the influences from item attribute values, user attribute values, and immediate friends' ratings are independent. Although this assumption simplifies the correlations among these variables, the naive Bayes model has been shown to be quite effective in many applications including textual document classification [16]. By making this assumption, the original conditional probability can be factorized as follows,

$$\begin{aligned} &\Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I, \boldsymbol{A} = \boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\}) \\ &= \frac{1}{Z} \Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I) \times \Pr(R_{UI} = k \mid \boldsymbol{A} = \boldsymbol{a}_U) \times \\ &\quad \Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\}) \end{aligned} \tag{3}$$

First, $Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I)$ is the conditional probability that the target user $U$ will give a rating $k$ to an item with the same attribute values as item $I$. This probability represents $U$'s preference for items similar to $I$. Because this value depends on the attribute values of items rather than an individual item, we drop the subscript $I$ in $R_{UI}$ for simplification. Second, $Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_U)$ is the probability that the target item $I$ will receive a rating value $k$ from a reviewer whose attribute values are the same as $U$. This probability reflects the general acceptance of the target item $I$ by users like $U$. For the same reason, because this value depends on the attribute values of users rather than a specific user, we drop the subscript $U$ in $R_{UI}$. Finally, $Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\})$ is the probability that the target user $U$ gives a rating value $k$ to the target item $I$ given the ratings of $U$'s immediate friends on item $I$. This is where we actually take social influences into con-

sideration in our system. In addition, $Z$ is a normalizing constant. We shall present the methods to estimate each of the factors in the following subsections.

### 3.1.1 User Preference

As we pointed out, $Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I)$ measures the target user $U$'s preference for the items similar to item $I$. For example, if we want to know how high Angela will rate "Revolutionary Road," $Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I)$ gives us a hint of how likely it is that Angela will give a rating $k$ to a drama movie which is also casted by Kate Winslet. To estimate this probability, we adopt the naive Bayes assumption again. We assume that the item attributes in $\boldsymbol{A}'$, e.g., category and cast, are independent of each other. Therefore, we have

$$\Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I) = \frac{\Pr(R_U = k) \times \Pr(A'_1, A'_2, ..., A'_n \mid R_U = k)}{\Pr(A'_1, A'_2, ..., A'_n)} \tag{4}$$

$$= \frac{\Pr(R_U = k) \times \prod_{j=1}^{j=n} \Pr(A'_j \mid R_U = k)}{\Pr(A'_1, A'_2, ..., A'_n)}, \boldsymbol{A}' = \{A'_1, A'_2, ..., A'_n\}$$

where $Pr(A'_1, A'_2, ..., A'_n)$ can be treated as a normalizing constant, $Pr(R_U = k)$ is the prior probability that $U$ gives a rating $k$, and $Pr(A'_j \mid R_U = k)$ is the conditional probability that each item attribute $A'_j$ in $\boldsymbol{A}'$ has a value $a'_j$ given $U$ rated $k$; e.g., $Pr($movie type = drama $\mid R_U = 4)$ is the probability that a movie will be a type of drama movie, given that $U$ gives a rating 4. The last two probabilities can be estimated from counting the review ratings of the target user $U$. Specifically,

$$\Pr(R_U = k) = \frac{\left|\boldsymbol{I}(R_U = k)\right| + 1}{\left|\boldsymbol{I}(U)\right| + n}, \text{ and} \tag{5}$$

$$\Pr(A'_j = a'_j \mid R_U = k) = \frac{\left|\boldsymbol{I}(A'_j = a'_j, R_U = k)\right| + 1}{\left|\boldsymbol{I}(R_U = k)\right| + m}, \tag{6}$$

where $|\boldsymbol{I}(U)|$ is the number of reviews of user $U$'s in the training set, $|\boldsymbol{I}(R_U = k)|$ is the number of reviews that user $U$ gives a rating value $k$, and $|\boldsymbol{I}(A'_j = a'_j, R_U = k)|$ is the number of reviews that $U$ gives a rating value $k$ while attribute $A'_j$ of the corresponding target item has a value $a'_j$. Notice that we insert an extra value 1 to the numerators in both equations, and add $n$, the range of review ratings to the denominator in Eq. (5), and $m$, the range of $A'_j$'s values, to the denominator in Eq. (6). This method is also known as Laplace estimate, a well-known technique in estimating probabilities [7], especially on a small size of training samples. Because of Laplace estimate, "strong" probabilities, like 0 or 1, from direct probability computation can be avoided.

Moreover, in some cases when item attributes are not available, we can approximate $Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I)$ by the prior probability $Pr(R_U = k)$. Even though $Pr(R_U = k)$ does not contain information specific to certain item attributes, it does take into account $U$'s general rating preference; e.g., if $U$ is a generous person, he/she gives high ratings regardless of the items.

### 3.1.2 Item Acceptance

$Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_u)$ captures the general acceptance of item $I$ from users like user $U$. For example, for a reviewer who is similar to Angela (e.g., the same gender and age), how likely is it that "Revolutionary Road" will receive a rating of 5 from her. Similar to the estimation in user preference, we use the naive Bayes assumption and assume user attributes are independent. Thus, we have

$$\Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_U) = \frac{\Pr(R_I = k) \times \Pr(A_1, A_2, ..., A_m \mid R_I = k)}{\Pr(A_1, A_2, ..., A_m)} \quad (7)$$

$$= \frac{\Pr(R_I = k) \times \prod_{j=1}^{j=m} \Pr(A_j \mid R_I = k)}{\Pr(A_1, A_2, ..., A_m)}, \boldsymbol{A} = \{A_1, A_2, ..., A_m\}$$

in which $Pr(R_I = k)$ is the prior probability that the target item $I$ receives a rating value $k$, and $Pr(A_j / R_I = k)$ is the conditional probability that user attribute $A_j$ of a reviewer has a value of $a_j$ given item $I$ receives a rating $k$ from this reviewer. These two probabilities can be learned by counting the review ratings on the target item $I$ in a manner similar to what we did in learning user preferences. When user attributes are not available, we use $Pr(R_I = k)$, i.e., item $I$'s general acceptance regardless of users, to approximate $Pr(R_I = k / \boldsymbol{A} = \boldsymbol{a}_u)$. In addition, $Pr(A_1, A_2, ..., A_m)$ in Eq. (7) is a normalizing constant.

### 3.1.3 Influence from Immediate Friends

Finally, $Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\})$ is where SNRS utilizes the influences from immediate friends. To estimate this probability, SNRS learns the correlations between the target user $U$ and each of his/her immediate friends $V$ from the items that they both have rated previously, and then assume each pair of friends will behave consistently on reviewing the target item $I$ too. Thus, $U$'s rating can be estimated from $r_{VI}$ according to the correlations. A common practice for learning such correlations is through estimating user similarities or coefficients, either based on user profiles or user ratings. However, user correlations are often so delicate that they cannot be fully captured by a single similarity or coefficient value. It is even worse that most of those measures seem ad hoc. Different measures return different results, and have different conclusions on whether or not a pair of users is really correlated [15]. To another extreme, user correlations can be

also represented in a joint distribution table of $U$s and $V$s ratings on the same items that they have rated; i.e., $Pr(R_{UI}, R_{VI}) \forall I \in \boldsymbol{I}(U) \cap \boldsymbol{I}(V)$. This table fully preserves the correlations between $U$s and $V$s ratings. However, in order to build such a distribution with accurate statistics, it requires a large number of training samples. For example, for ratings ranging from one to five, the joint distribution has 25 degrees of freedom, which is difficult to be estimated robustly with limited training samples. This is especially a problem for recommender systems, because in most of these systems, users only review a few items compared to the large amount of items available in the system, and the co-rated items between users are even less. Therefore, in this study, we use another approach to remedy the problems in both cases.

Friends are similar, and give similar ratings. Our data analysis in Section 4 on a real online social network also shows that immediate friends tend to give more similar ratings than non-friends. Thus, for each pair of immediate friends $U$ and $V$, we consider their ratings on the same item to be close with some error $\varepsilon$. That is,

$$R_{UI} = R_{VI} + \varepsilon, \quad I \in \boldsymbol{I}(U) \cap \boldsymbol{I}(V), V \in \boldsymbol{N}(U) \cap \boldsymbol{U}(I) \tag{8}$$

From Eq. (8), we can see that error $\varepsilon$ can be simulated from the histogram of $U$s and $V$'s rating differences $H(R_{UI} - R_{VI}) \forall I \in \boldsymbol{I}(U) \cap \boldsymbol{I}(\mathrm{V})$. Thus, $H(R_{UI} - R_{VI})$ serves as the correlation measure between $U$ and $V$. For rating ranges from one to five, $H(R_{UI} - R_{VI})$ is a distribution of nine values, i.e. from -4 to 4. Compared to similarity measures, it preserves more details in friends' review ratings. Compared to a joint distribution approach, it has fewer degrees of freedom.

Assuming $U$'s and $V$'s rating difference on the target item $I$ is consistent with $H(R_{UI} - R_{VI})$. Therefore, when $R_{VI}$ has a rating $r_{VI}$ on the target item, the probability that $R_{UI}$ has a value $k$ is proportional to $H(k - r_{VI})$.

$$\Pr(R_{UI} = k \mid R_{VI} = r_{VI}) \infty H(k - r_{VI}). \tag{9}$$

For example, assume that both $U$ and $V$ rated the items as shown in Table 3.1. Given their ratings in the table, we want to predict $U$'s possible ratings on item $I_6$ according to the correlation with $V$. From the previous ratings of $U$ and $V$, we find out that two out of five times $U$'s rating is the same as $V$'s, and three out of five times $U$'s rating is lower than $V$'s by one. According to such a correlation, we predict that there is a 40% chance that $R_{UI6}$ is 4 and 60% chance that $R_{UI6}$ is 3.

| | $U$ | $V$ |
|---|---|---|
| $I_1$ | 5 | 5 |
| $I_2$ | 3 | 4 |
| $I_3$ | 4 | 4 |
| $I_4$ | 2 | 3 |
| $I_5$ | 4 | 5 |
| $I_6$ | ? | 4 |

Table 3.1: An example of predicting user rating from an immediate friend

The previous example illustrates how we utilize the correlation between the target user and one of his/her immediate friends. When the target user has more than one immediate friend who co-rates the target item, the influences from all of those friends can be incorporated in a product of normalized histograms of individual friend pairs.

$$\Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\} = \frac{1}{Z}\prod_{V}\frac{1}{Z_V}H(k - r_{VI}) \tag{10}$$

where $Z_V$ is the normalizing constant for the histogram of each immediate friend pair, and $Z$ is the normalizing constant for the overall product.

Once we obtain $Pr(R_U = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I)$, $Pr(R_I = k \mid \boldsymbol{A} = \boldsymbol{a}_U)$, and $Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\})$, the ultimate rating distribution of $R_{UI}$, under the factors of user preference, item's general acceptance, and the correlations with immediate friends, can be estimated from Eq. (3). $R'_{UI}$, the estimated value of $R_{UI}$, is the expectation of the distribution as shown in Eq. (11).

$$R'_{UI} = \sum_k k \times \Pr(R_{UI} = k \mid \boldsymbol{A}' = \boldsymbol{a}'_I, \boldsymbol{A} = \boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U}(I) \cap \boldsymbol{N}(U)\} \tag{11}$$

## *3.2 Distant Friend Inference*

In the previous section, we introduced the approach to predict the target user's rating on a target item from those of his/her immediate friends on the same item. However, in reality, most immediate friends of the target user may not have reviewed the target item, because there are a large number of items in recommender systems but users may only select a few of them. Therefore, the influences from those friends cannot be utilized in immediate friend inference, and it is even worse that the ratings of many users cannot be predicted because they have no immediate friends who co-rate the target item. To solve this problem, we propose a method to incorporate the influences from distant friends via extending immediate friend inference.

The idea of distant friend inference is intuitive. Even though $V$, an immediate friend of the target user $U$, has no rating on the target item, if $V$ has his/her own immediate friends who rated the target item, we should be able to predict $V$'s rating on the target item via the immediate friend inference, and then to predict $U$'s rating based on the estimated rating of $V$'s. This process conforms to real scenarios, such as Linda's officemate influences Linda who further influences Angela in our previous example. Followed by this intuition, we decide to apply an *iterative classification* method [17, 24, 31] for distant friend inference.

Iterative classification is an approximation technique for classifying relational entities. This method is based on the fact that relational entities are correlated with

each other. Estimating the classification of an entity often depends on the estimations of classification of its neighbors. The improved classification of one entity will help to infer the related neighbors and vice versa. Unlike traditional data mining which assumes that data instances are independent and identically distributed (i.i.d.) samples, and classifies them one by one, iterative classification iteratively classifies all the entities in the testing set simultaneously because the classifications of those entities are correlated. Note that iterative classification is an approximation technique, because exact inference is computationally intractable unless the network structures have certain graph topologies such as sequences, trees or networks with low tree width. Iterative classification has been used to classify company profiles [24], hypertext documents [17], and emails [6] with reasonable success in the previous research.

The algorithm for distant friend inference is shown in Table 3.2. This algorithm predicts the users' ratings on each target item at a time. The original iterative classification method classifies the whole network of users. However, since the number of users in social networks is usually large, we save the computation cost by limiting the inference to a user set $\boldsymbol{N}$ which includes the target users of the target item $I$, and their corresponding immediate friends. In each iteration, we generate a random ordering $O$ of the users in $\boldsymbol{N}$. For each user $U$ in $\boldsymbol{O}$, if $U$ has no immediate friend who belongs to $\boldsymbol{U(I)}$, which is the set of users whose rating (either ground truth or estimated value) is observable, the estimation of $R_{UI}$ will be skipped in this iteration. Otherwise, $Pr(R_{UI} = k \mid \boldsymbol{A}=\boldsymbol{a'_l}, \boldsymbol{A}=\boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N(U)}\})$ will be estimated by immediate friend inference, and $R'_{UI}$ is then obtained from Eq. (11). Because user rating is an integer value, in order to continue the iterative process we round $R'_{UI}$ to a close integer value, and insert into or update $\boldsymbol{U(I)}$ with $R'_{UI}$ if different. This entire process iterates $M$ times or until no update occurs in the current iteration. In our experiment, the process usually converges within $10$ iterations.

It is worth pointing out that after we compute $Pr(R_{UI} = k \mid \boldsymbol{A}'=\boldsymbol{a'_l}, \boldsymbol{A}=\boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N(U)}\})$, there are two other options for updating $R'_{UI}$ besides rounding the expectation in distant friend inference. The first option is to select $R'_{UI}$ with the value $k$ such that it maximizes $Pr(R_{UI} = k \mid \boldsymbol{A}'=\boldsymbol{a'_l}, \boldsymbol{A}=\boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N(U)}\})$. However, by doing so, we are actually throwing out clues of small probabilities at the same time. After many iterations, the errors caused by the greedy selection will be exacerbated. The target users are likely to be classified with the majority class. The other option is to directly use $Pr(R_{UI} = k \mid \boldsymbol{A}'=\boldsymbol{a'_l}, \boldsymbol{A}=\boldsymbol{a}_U, \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N(U)}\})$ as soft evidence to classify other users. However, in our experiments, this approach does not return results as good as those of rounding the expectation.

1. For each item $I$ in the testing set do
2. Select a set of users $\boldsymbol{N}$ for inference. $\boldsymbol{N}$ includes the target users of item $I$ and their corresponding immediate friends.
3. For iteration from $1$ to $M$ do
4. Generate a random ordering, $\boldsymbol{O}$, of users in $\boldsymbol{N}$
5. For each user $U$ in $\boldsymbol{O}$ do
6. If $U$ has no immediate friend who exists in $\boldsymbol{U(I)}$
7. Continue
8. Else
9. Apply immediate friend inference
10. $R'_{UI} = \sum_k k * Pr(R_{UI} = k \mid \boldsymbol{A}=\boldsymbol{a}_U, \boldsymbol{A}'=\boldsymbol{a}'_U, \{R_{VI} = r_{VI}: \forall V \in \boldsymbol{U(I)} \cap \boldsymbol{N(U)}\})$
11. Insert into or Update $\boldsymbol{U(I)}$ with $R'_{UI}$ if different
12. End If
13. End For
14. If no updates in the current iteration
15. Break
16. End If
17. End For
18. Output the final predictions for the target users
19. End For

Table 3.2: Pseudo-code for distant friend inference

# 4 Dataset

In this section, we introduce the dataset that we use for this research, and present some interesting characteristics of this dataset. Our dataset is obtained from a real online social network Yelp.com. As one of the most popular web 2.0 websites, Yelp provides users local search for restaurants, shopping, spas, nightlife, hotels, auto services, and financial services etc. Users that come to this site can either look for information from Yelp or make their own voices by writing reviews for some local commercial entities that they have experienced. Yelp provides a homepage for each local commercial entity. An example of a homepage for a restaurant at Yelp, "Yoshi's Sushi", is shown in Figure 4.1(a). On top of this homepage is a profile of this restaurant, which includes restaurant attributes such as category, location, hours, price range and parking information etc. In addition, this homepage contains a list of reviews of users who have visited this restaurant before. Each review comes with a numerical rating ranging from one to five stars. Five starts means the highest rating to this restaurant, and one star is the lowest rating.

Besides maintaining traditional features of recommender systems, Yelp provides social network features so that it can attract more users. Specifically, Yelp allows users to invite their friends to join Yelp or make new friends existing at Yelp. The friendship at Yelp is mutual relationship, which means that when a user

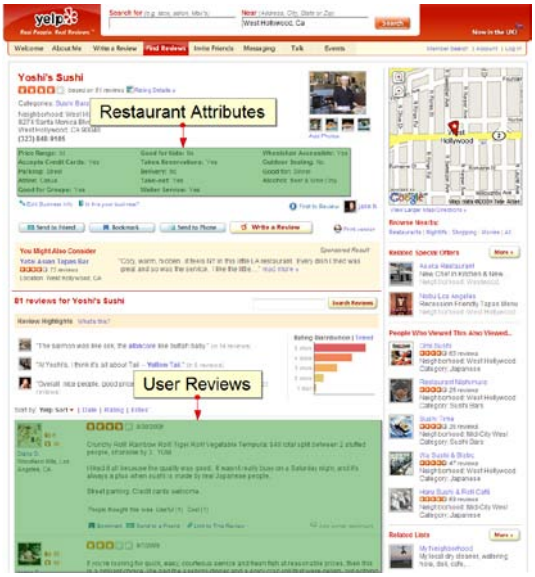
adds another user as a friend, the first user will be automatically added as a friend of the second user. Yelp also provide a homepage for each of its users. Each user homepage contains basic personal information, all the reviews written by this user, and links to the friends that are explicitly identified by this user.

Since restaurant is the most popular category at Yelp, we picked restaurant as the problem domain. We crawled and parsed the homepages of all the Yelp restaurants in the Los Angeles area that registered before November 2007. We ended up with 4152 restaurants. By following the reviewers' links in the Yelp restaurant homepages, we also crawled the homepages of all these reviewers, which resulted in 9414 users. Based on the friend links in each user's homepage, we are able to identify friends from the crawled users, and thus reconstruct a social network. Note that the friends we collected for each user may only be a subset of the actual friends listed on his/her homepage. That is because we require every user in our dataset to have a least one review in the crawled restaurants. In other words, the social network that we crawled focuses on dining.

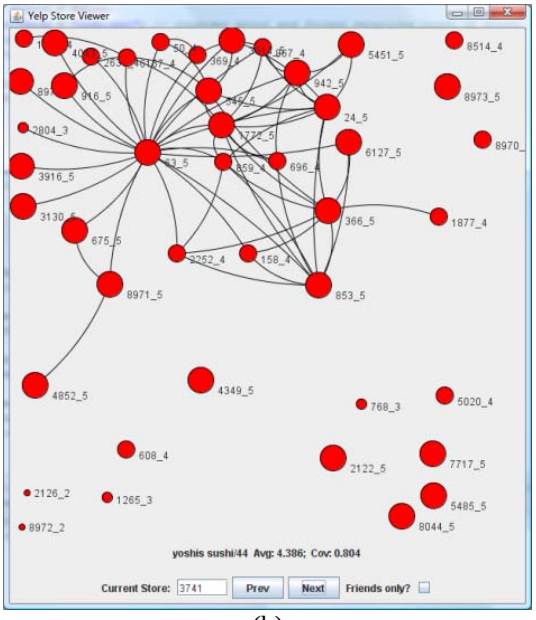
To illustrate users' ratings and their relationships, we built a graphical tool to represent each restaurant in our dataset. Figure 4.1(b) shows the alternative view of "Yoshi's Sushi" in Figure 4.1(a). Each node represents a reviewer of the restaurant, and the size of the node represents the corresponding reviewer's rating on this restaurant. Two nodes are connected if they claim each other as friends. Since friends in Yelp are mutual, the social network structure is an undirected graph. From Figure 4.1(b), we can see that nodes in this graph are highly connected, which means many friends are involved in writing reviews for “Yoshi’s Sushi”.

A preliminary study on this dataset yields the following results. The total number of reviews in this dataset is 55,801. Thus, each Yelp user on average writes 5.93 reviews and each restaurant on average has 13.44 reviews. In terms of friends, the average number of immediate friends of every user is 8.18. If we take a closer look at the relations between the number of users and the number of their immediate friends (as shown in Figure 4.2 (a)), we can see that it actually follows a *power-law distribution*; this means that most users have only a few immediate friends while a few users have a lot of immediate friends. A similar distribution also applies to the relations between the number of users and the number of reviews, as shown in Figure 4.2(b). Because most users on Yelp review only a few restaurants, we expect the dataset to be extremely sparse. In fact, the sparsity of this dataset, i.e., the percentage of user/item pairs whose ratings are unknown, is 99.86%.

Furthermore, we perform the following analysis on this dataset, particularly focusing on immediate friends’ review correlation and rating correlation. Basically, we want to answer two questions: 1) whether or not friends tend to review the same restaurant; and 2) whether or not friends tend to give ratings that are more similar than those from non-friends. Clearly, these two questions are essential to SNRS.

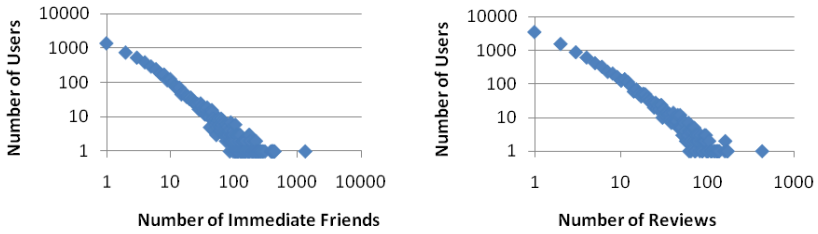



(a)




(b)

Figure 4.1: (a) The homepage of a Yelp restaurant "Yoshi's Sushi" and (b) the corresponding abstract graphical representation of Yoshi's Sushi in which each node represents a reviewer in the restaurant, and nodes are connected by explicit friend relations. The size of each node is proportional to the corresponding reviewer's rating on this restaurant.




(a) (b)

Figure 4.2: (a) The number of users versus the number of immediate friends in the Yelp network, and (b) the number of users versus the number of reviews both follow the power-law distribution.

## *4.1 Review Correlations of Immediate Friends*

Let us first study the correlation of immediate friends in reviewing the same restaurants. Specifically, we want to know if a user reviews a restaurant, what is the chance that at least one of his/her immediate friends has also reviewed the same restaurant? To answer this question, we count, for each user, the percentage of restaurants that has also being reviewed by at least one of his/her immediate friends. The average percentage over all users in the dataset is 18.6%. As a comparison, we calculate the same probability if assuming immediate friends review restaurants uniformly at random and independently. In a social network with $n$ users, for a user with $q$ immediate friends and a restaurant with $m$ reviewers (including the current user), the chance that at least one of $q$ immediate friends appears in $m$ reviewers is $1-\frac{\binom{n-q-1}{m-1}}{\binom{n-1}{m-1}}$. We calculate this value for every user and every restaurant he/she reviewed. The average probability over all users is only 3.7%. Compared to 18.6% as observed in the dataset, it is clear that immediate friends do not review restaurants randomly. There are certain correlations between friends.

We also extend the previous study by considering the probability that at least one of a reviewer's friends within two hops review the same restaurant. Note that this covers the cases where immediate friends have no reviews for the restaurant, but at least one of the second-hop friends does. Such a probability is 45.2%, which is about two and a half times as high as the previous result for immediate friends (18.6%). Since SNRS can make recommendations only when there are friends who have co-rated the same items, if we limit the friends within one hop (immediate friends), then we can only predict ratings for a limited number of users. In other words, this comparison reveals the importance of considering distant friends in SNRS. Meanwhile, if we assume friends review restaurants randomly, the

probability is 34.2% that at least one friend, within two hops, co-reviews the same restaurant as the target user.

Finally, we compare the average number of co-reviewed restaurants between any two immediate friends and any two users on Yelp. The results are 0.85 and 0.03 respectively, which again illustrates the tendency that immediate friends co-review the same restaurants.

### *4.2 Rating Correlations of Immediate Friends*

To show that whether immediate friends tend to give more similar ratings than non-friends, we compare the average rating differences (in absolute values) on the same restaurant between reviewers who are immediate friends and non-friends. We find that, for every restaurant in our dataset, if two reviewers are immediate friends, their ratings on this restaurant differ by 0.88 on average with a deviation of 0.89. If they are not, their rating difference is 1.05 and the standard deviation is 0.98. This result clearly demonstrates that immediate friends, on average, give more similar ratings than non-friends.

In this section we presented some characteristics of our dataset. The results on review correlations as well as rating correlations between immediate friends are critical in validating our assumptions in SNRS. In the next section, we are going to present a set of experiments to demonstrate the advantages of considering social network information in a recommender system.

## 5 Experiments

In the experiments we evaluate the performance of SNRS on the Yelp dataset, focusing on the issues of the prediction accuracy, data sparsity, and cold-start, which are the main issues of current recommender systems. Additionally, we will study the role of distant friends in SNRS.

The following is the setting for our experiments. We used a restaurant's price range as the item attribute. Since there is no useful user attribute on Yelp, we substituted $Pr(R_I = k \mid \mathbf{A} = \mathbf{a}_u)$ with $Pr(R_I = k)$ when estimating item acceptance. Finally, we set a threshold to require every pair of immediate friends to have at least three co-rated restaurants. If they do not, we ignore their friend relationships.

### *5.1 Comparison Methods*

As a comparison, we implemented the following methods along with SNRS.

*Friend Average (FA).* To leverage the ratings of friends for inference, the most straightforward approach is to predict the ratings of the target users on the target items with the average ratings of their immediate friends on the same item. We therefore implemented this method as a baseline.

*Weighted Friends (WVF).* Unlike treating immediate friends equally as in *FA*, *WVF* considers that every immediate friend has a different impact (or weight) on the target user. The more the impact from an immediate friend, the closer the target user's rating is to the rating of that friend. Thus, the probability of the target user's rating is proportional to the accumulated weight in each rating value.

$$\Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{N}(U) \cap \boldsymbol{U}(I)\}) = \frac{1}{Z}\sum_V w(U,V)\delta(k, r_{VI}) \tag{12}$$

in which $z$ is a normalizing constant. $w(U, V)$ is the weight between $U$ and $V$. In this experiment, we use the cosine similarity between $U$s and $V$s ratings as their weight. $\delta(k, r_{VI})$ is the delta function which returns one only when $r_{VI} = k$, and zero otherwise. WVF is essentially same as a relational-neighbor classifier [18] which performs really well on classifying relational datasets such as citations and movies.

*Naive Bayes (NB).* Social networks can be also modeled using Bayesian networks [10]. In this study, we implemented a special form of Bayesian networks, a naive Bayes classifier. Specifically, when predicting the rating of a target user $U$, the *NB* classifier assumes $U$s rating influences the ratings of $U$s immediate friends, and the ratings of $U$s immediate friends are independent of each other. Although with strong assumptions, *NB* classifiers have been widely used for probabilistic modeling and often result in surprisingly good results [16]. Therefore, we also included this method for comparison.

Given the ratings of the immediate friends on the target item $I$, we calculate the conditional probability $Pr(R_{UI} \mid \{R_{VI} : \forall V \in \boldsymbol{N}(U) \cap \boldsymbol{U}(I)\})$ as follows.

$$\begin{aligned} &\Pr(R_{UI} = k \mid \{R_{VI} = r_{VI} : \forall V \in \boldsymbol{N}(U) \cap \boldsymbol{U}(I)\}) \\ &= \frac{1}{Z}\Pr(R_U = k)\prod_V \Pr(R_V = r_{VI} \mid R_U = k) \end{aligned} \tag{13}$$

where $Pr(R_U=k)$ is the prior rating distribution of the target user $U$, which can be estimated by counting the review ratings of $U$. $Pr(R_V=r_{VI} \mid R_U=k)$ is the conditional probability that an immediate friend $V$s rating is equal to $r_{VI}$ given $U$s rating is $k$. Because there are not enough samples to estimate these probabilities for every individual pair of immediate friends, we estimate these probabilities by counting the review ratings for all pairs of immediate friends in the dataset. Moreover, $Z$ is a normalizing constant. The estimated rating of the target user $U$ is the rating value that has the maximum probability.

*Collaborative Filtering (CF).* We implemented the standard collaborative filtering algorithm as we described in Section 2. The $K$ value we used is 20.

## 5.2 Prediction Accuracy And Coverage

We carried out this experiment in a 10-fold cross-validation. The prediction accuracy was measured by the *mean absolute error* (MAE), which is defined as the average absolute deviation of predictions to the ground truth data over all the instances, i.e., target user/item pairs, in the testing set.

$$MAE = \frac{\sum_{U,I} |r_{UI} - r'_{UI}|}{L}, \tag{14}$$

where $L$ is the number of testing instances. The smaller the MAE, the better the inference.

Since SNRS, FA, WVF, and NB rely on friends' ratings on the target item in order to make predictions; thus, there is no prediction when the target user has no friends who have rated the item. Similarly, CF does not make predictions unless it finds similar users for the target user. Therefore, another metric that we study for each method is the *coverage*, which is defined as the percentage of the testing instances for which the method can make predictions.

The experimental results are listed in Table 5.1. From this table, we note that SNRS achieves the best performance in terms of MAE (0.718), while CF is the worst (0.871). SNRS improves the prediction accuracy of CF by 17.8%. The other methods that use the influences from friends also achieve better results than CF. Clearly, considering social influence does improve predictions in recommender systems. In terms of the coverage, the coverage of all these methods is relatively low; e.g., none of these methods have the coverage better than 0.6. This is because the dataset we have is extremely sparse, with a sparsity of 99.86%. However, among these methods, CF is the best. Because most of the time, CF is able to find similar users for the target user from all the other users in the training set. On the other hand, the coverage of the other methods is decided by whether there is a friend who has rated the item, and we pruned many friend relationships by setting a threshold of three co-rated items for each pair of friends. Therefore, the coverage of those methods is lower than CF. The coverage of FA, WVF, and NB is even lower than that of SNRS, because SNRS can still utilize the influence from distant friends even if immediate friends have not rated the restaurant, while the other methods cannot.

| | MAE | COVERAGE |
|---|---|---|
| **SNRS** | ***0.716*** | *0.482* |
| **FA** | *0.814* | *0.228* |
| **WVF** | *0.808* | *0.228* |
| **NB** | *0.756* | *0.237* |
| **CF** | *0.871* | ***0.552*** |

Table 5.1: Comparison of the MAEs of the proposed Social Network-Based Recommender System (SNRS), Collaborative Filtering (CF), Friend Average (FA), Weighted Friends (WVF), and Naive Bayes (NB) in a 10-fold cross-validation.

## *5.3 Data Sparsity*

CF suffers from problems with sparse data. In this study, we want to evaluate the performance of SNRS at various levels of data sparsity. To do so, we randomly divided the whole user/item pairs in our dataset into ten groups, and then randomly selected $n$ sets as the testing set, and the rest as the training set. The value of $n$ controls the sparsity of the dataset. At each value of $n$, we repeated the experiment 100 times. The performance was measured by the average MAEs and the coverage.

Figure 5.1(a) compares the MAEs of SNRS and CF when the percentages of testing sets vary from 10% to 70%. Due to the high sparsity of the underlying Yelp dataset, even when the percentage of testing set is 10%, the actual sparsity is as high as 99.87%. From Figure 5.1(a), we first observe that the MAEs of SNRS are consistently lower than those of the CF, which again shows that SNRS indeed outperforms CF. Second, the prediction accuracy of CF is greatly affected by data sparsity. For example, the MAEs of CF increase by 14.4% from 0.868 and 0.993 when the testing set is increased from 10% to 70% of the whole dataset. Meanwhile, the MAEs of SNRS grow at a much slower pace. For instance, the MAEs of SNRS increase by only 2.8% from 0.716 to 0.736 under the same conditions.

Figure 5.1(b) compares the coverage of both methods. Unfortunately, the coverage of both methods severely drops as the training set becomes sparser. For example, the coverage of CF drops from 0.549 to 0.064 when the size of the testing set increases from 10% to 70%, and the coverage of SNRS decreases from 0.482 to 0.123 at the same time. This decrease in the coverage is expected, as explained earlier, but the trend of these two methods also indicates their differences. CF performs better with a large training set, allowing it to find more similar users. When the training set becomes sparser, CF finds similar users from fewer candidates for each target user. The similarity obtained from each pair of users is less accurate because that there are fewer co-rated items between these users. Thus, both the prediction accuracy and the coverage of CF are adversely affected by the data sparsity. Meanwhile, the coverage of SNRS also decreases because there are fewer friends who have ratings on the target items as the dataset becomes sparser. But the coverage of SNRS decreases with a slower pace compared to that of CF. Initially, CF has a better coverage than SNRS. However, the coverage of SNRS starts to exceed that of CF after the percentage of the testing set is above 30%. Such a change in the trend is because that some users can still be inferred since the influences from distant friends are able to propagate to them even when the dataset is sparse. In Section 5.5, we will study the role of distant friends again. On the other hand, the prediction accuracy of SNRS is consistent at all levels of data sparsity. This is because friends are provided explicitly by social networks, and there is no need for SNRS to find similar users from the training set. Therefore, as long as there are friends who have reviewed the target item, SNRS can make accurate predictions.

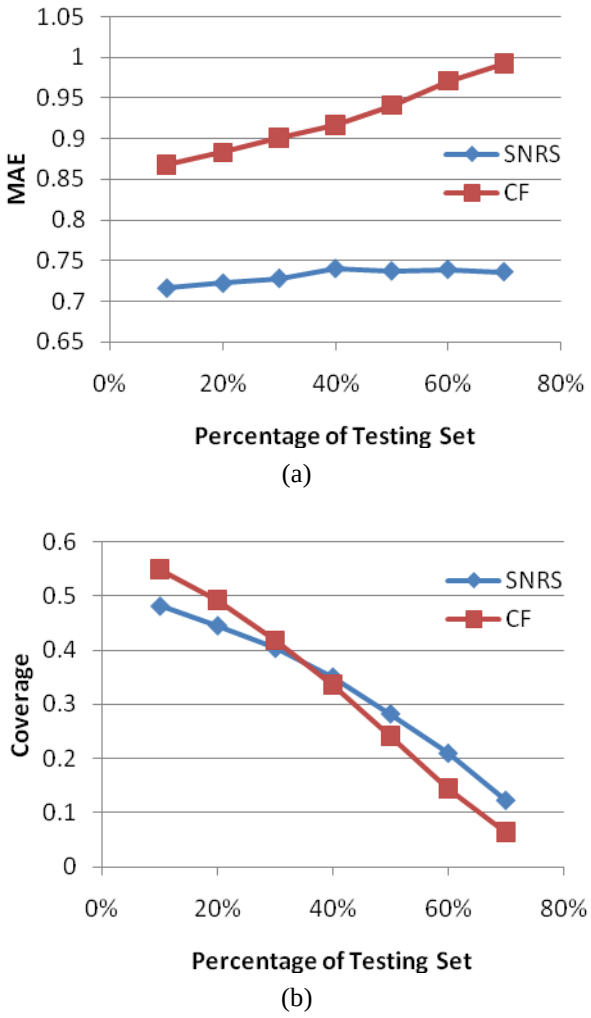



Figure 5.1: Comparison of the (a) MAEs and the (b) coverage of SNRS and CF at different sizes of the testing set.

## *5.4 Cold-Start*

Cold-start is an extreme case of data sparsity where a new user has no reviews. In such a case, CF cannot make a recommendation to this new user since CF is not able to find similar users for him/her. SNRS cannot either if this new user has also no friends. However, in some cases of cold-start when a new user is invited by

some existing users in the system, the initial friend relationships of this new user can still make the inference of SNRS possible. Even though there is no prior knowledge of the new user's own preference, SNRS can make recommendations to this new user based on the preferences of his/her friends. In this study, we simulated the latter case of cold-start by making the following experimental settings: 1) We did not use the target user's prior ratings in the training set; thus, there was no influence from user preference. We simply set the output from $Pr(R_U = k \mid \mathbf{A}' = \mathbf{a}'_k)$ as a uniform distribution. 2) Since we cannot learn the rating correlation between this new user and his/her friends, we directly used friends' rating distribution on the target item, $Pr(\{R_{VI} = r_{VI} : \forall V \in \mathbf{U}(I) \cap \mathbf{N}(U)\})$, as the result from friend inference. 3) Except for the target user, the ratings of all other users were known.

We simulated cold-start for every user in the dataset. The resulting MAE is 0.706 and the coverage is 0.691. This result demonstrates that even in cold-start, SNRS can still perform decently. The coverage of SNRS is high compared to that in the 10-fold cross-validation (0.422) because the ratings of every target user's friends are all observable in the setting of this experiment.

## *5.5 Role of Distant Friends*

In this study we investigate the role of distant friends in SNRS. Specifically, we compared the performance of SNRS with and without distant friend inference in a 10-fold cross-validation. The experimental results are shown in Table 5.2. From these results, we can see that by considering the influences from distant friends, the coverage of SNRS is increased from 0.237 to 0.482, which is equivalent to a 103% improvement. However, the improvement is achieved at the cost of a slight reduction in the prediction accuracy. In our experiments, the MAE increases from 0.682 to 0.716, which is only a 5% difference. This is consistent with our intuition that the impact from distant friends is not as direct as from immediate friends, and certain errors will be inevitably introduced when considering distant friends. On the other hand, compared to the drastic gain in the coverage, the minor loss in precision is still acceptable.

| | MAE | COVERAGE |
|---|---|---|
| With Distant Friend Inference | 0.716 | **0.482** |
| Without Distant Friend Inference | **0.682** | 0.237 |

Table 5.2: Comparison of the performance of SNRS with and without distant friend inference.

# 6 Semantic Filtering of Social Networks

In the previous section we showed that SNRS improves the prediction accuracy of recommender systems by utilizing information such as social influences in social networks. In this section, we shall discuss how to further improve the performance of SNRS by applying semantic filtering of social networks.

Although friends influence each other when selecting items, such influence is sensitive to the types of items. For example, two friends who have similar taste on CDs may not necessarily agree with each other in their choice of favorite restaurants. Therefore, to recommend restaurants, we should not consider friends who have common preferences only in music. In other words, to effectively use the social influence, an appropriate set of friends needs to be selected according to the type of target items, which is what we called semantic filtering of social networks. In fact, we considered this issue when we performed experiments on Yelp. Rather than considering all friends listed in user's profiles, we pruned a set of friends who had reviewed only a small number of common restaurants. For example, even though two real friends may have reviewed many common hotels on Yelp, they are not necessarily friends in SNRS unless they have enough reviews on common restaurants. However, this is still a poor man's version of semantic filtering, because even within the domain of restaurants, friends can be further grouped based on their opinions on different food categories, price range, restaurant environment, etc.

A better selection of relevant friends requires us to know in what aspects two friends influence each other. Unfortunately, such information is not available in most current OSNs. Some OSNs, such as Linkedin, ask how friends know each other, e.g., whether they were/are classmates or colleagues. Information like this definitely helps us understand friend relationships. However, it is still too general to bring a practical usage to recommender systems. Instead, the semantics that we really want to know from friend relationships should be more specific to the domain of interest. For example, in terms of dining, it would be better to know whether two friends are friends because they have a similar taste in food or a similar preference in the price of meals, etc. To obtain such information, the most direct solution is for content providers (e.g., Yelp) to explicitly ask users to rate their friends on those aspects. If that puts too much of a burden on users, an alternative is for content providers to collect finer-grained user ratings rather than overall ratings alone, and then implicitly deduce friend relationships from the semantics in those finer-grained ratings. The problem with overall ratings is that they encapsulate decision-reasoning of users on many factors. For example, when a user gives a rating of 4 to a restaurant, it is not clear if the user really likes the taste, price, service, or environment of this restaurant. If content providers could ask users to rate on those factors, such finer-grained ratings would not only allow us to model user preference and item acceptance more precisely, but also help us to know on which category two friends are in agreement or whether they influence each other. For instance, two friends may not give the same overall rating, but they might still agree on the quality of restaurant service.

In the following text, we describe an experiment that we designed to demonstrate how relevant friends can be selected for inference by obtaining the semantics in friend relationships and user ratings, and then validate its improvement on SNRS.

This experiment was to predict students' ratings for online articles. It was conducted as a class project assignment with 22 students. At the beginning of the experiment, we selected 20 online articles. These articles focus on three topics: the recent economic crisis, controversies in technologies such as stem cell research and file sharing, and controversies in culture like gay marriage. These articles all contain strong opinions expressed by the authors. We collected the demographic information of students, including gender, age, ethnicity, employment, and interests. We also asked them a set of questions related to the articles that we selected. For example, "Has the rise in unemployment affected you or someone in your family?" and "Given the current state of the economy, are you concerned about getting a job after you graduate?". After that we asked the students to review every article by giving four ratings (from 1 to 5) based on each of the following criteria: 1) Interestingness: Is the article interesting? 2) Agreement: How much do you agree with the author? 3) Writing: Is the article well written? and 4) Overall: Overall evaluation. The reason that we included the first three ratings is because they usually play important roles when we give an overall score to an article. Since most students did not know each other before the experiment, it would have been difficult to form a social network from their original relationships. We therefore decided to divide the students into groups and let them get to know each other by discussing the articles within the groups. Specifically, we divided the students into three groups twice. The first grouping was based on students' ethnicities, and the second grouping was based on students' responses to the survey questions. The goal of these groupings was to organize the students in such a way that the students in a group will more likely to be friends after the group discussions. During the discussions, every student needed to explain the reasons why he/she liked or disliked each article. Thus, the other group members were able to know more about the speaker. After the discussions, the students evaluated other group members (using ratings from 1 to 3) according to the following three aspects: 1) Do you have common interests on the articles? 2) Do you agree with his/her opinions on the articles? 3) Do you have common judgments about the author's writing skill? In addition to evaluating group members, the students were allowed to revise their previous ratings to the articles if they had a new understanding of the articles due to the discussion.

Compared to the Yelp dataset, there are mainly two changes in this dataset. First, instead of having just an overall rating, each article now has three fine-grained ratings (interestingness, agreement, and writing) which, as mentioned earlier, provide the semantics of the overall rating. Second, friend relationships have semantics too. Rather than just knowing that two students are friends, we are now able to know whether it is because they have similar interests or similar opinions, etc. In the following experiment, we are going to compare the prediction accuracies of SNRS with and without the consideration of semantic filtering of social networks.

Similar to the experimental setup in Section 5.3, we randomly divided the student/article pairs into ten groups. We randomly selected $n$ groups as the testing set, and the rest as the training set. For each value of $n$, we repeated the experiment 100 times. For each pair in the testing set, we predicted the target student's ratings on the target article by applying and not applying semantic filtering of social networks. When we applied semantic filtering to predict a particular rating, we only considered the ratings of the target user's friends in the corresponding category. For example, to predict the target article's interestingness, we selected the set of students whom the target student had rated as friends (with a rating of 3) in terms of having similar interests, and then used their ratings on interestingness for inference. Thus, the social networks used for predicting each category are different. On the other hand, without semantic filtering, we considered the ratings on interestingness from all the students whom the target user had rated as friends in any of the three aspects. We measured the average MAEs for predicting each rating of the article, and the corresponding MAEs in CF.

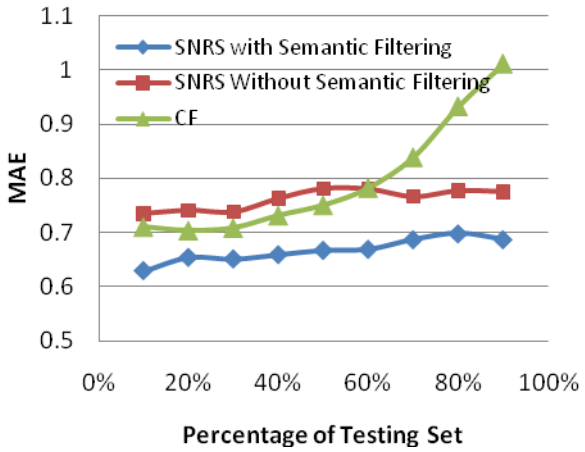
We show the results of predicting student ratings on the interestingness of the articles in Figure 6.1(a). From this figure, we observe two trends. First, regardless of semantic filtering or not, the MAEs of SNRS are persistent for different data sparsity, while the MAEs of CF dramatically increase as the data becomes sparser. This phenomenon is consistent with our findings on the Yelp dataset in Section 5.3. Second, we find that, at any level of data sparsity, the MAEs of SNRS with semantic filtering are consistently lower than those of SNRS without semantic filtering as well as those of CF. This result demonstrates that semantic filtering does indeed improve the prediction accuracy of SNRS. In Figure 6.1(b), (c), and (d), we plot the results of predicting ratings on the agreement and writing of the articles and overall ratings respectively. We observe similar trends in these figures. Note that when predicting the overall rating, we consider the overall ratings of all friends of a target student, which means there is no semantic filtering.

In this paper we assume the reviews that users provide are real. However, in reality, there are always users who purposely provide false reviews to attack the adversaries or praise themselves, and traditional recommender systems have no control on them at all. On the other hand, SNRS is still able to detect and exclude those malicious users through reputation systems [19].

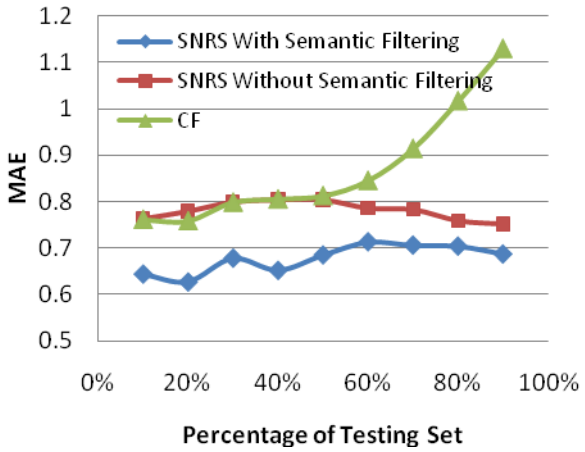
# 7 Related Work

Domings and Richardson proposed to mine customers' network values from a social network [8, 28]. The network value comes from the different potentials of customers to influence their social contacts to purchase the same products. Thus, the more people they can influence, the higher network value they have. Merchandisers can increase the expected lift in profit by sending advertisements only to those users who have high network values. [8] estimates the conditional probability of whether a user will purchase a product given the adoption values of his/her friends, and marketing actions are tailored by using a relaxation labeling

approach. Such a probability is modeled as a weighted sum of each user's internal probability of purchasing a product and an external effect from his/her friends [28]. The authors conduct simulation studies, first on a synthesized social network in [8], and then on Epinon.com, a review website in [28].




(a)




(b)

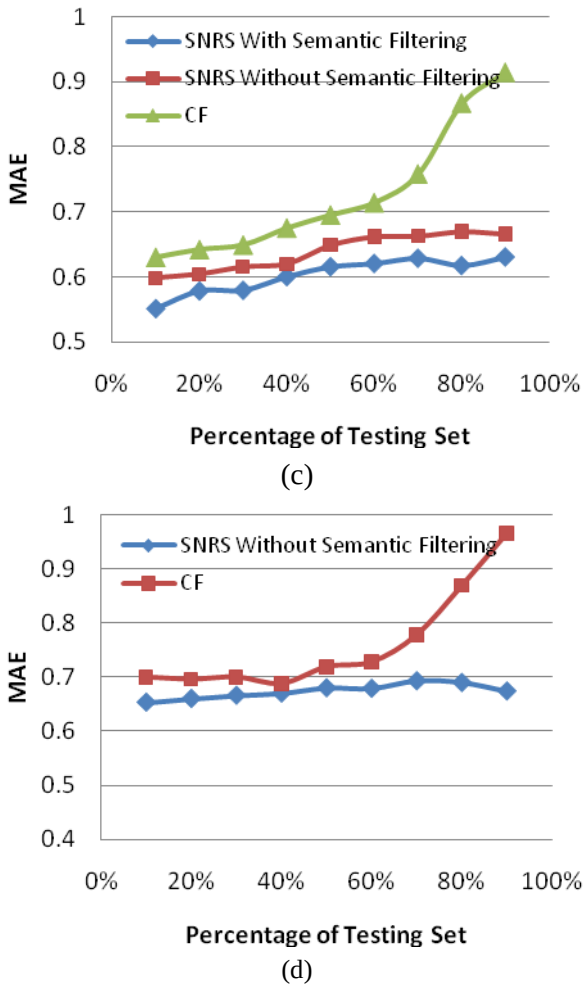



Figure 6.1: Comparisons of the accuracies of CF and SNRS with and without semantic filtering on predicting student ratings on (a) interestingness, (b) agreement, (c) writing, and (d) overall aspects of the articles.

There is also previous work on exploiting explicit user trust in recommender systems. [9] presented a FilmTrust system which used explicit trust values between users as the weights in collaborative filtering. Similarly, [20] proposed a trust-aware recommender system which is also based on explicit trust values between users. They proposed a method for trust propagation in which the trust between distant friends is calculated by a linear decay model. Although these research efforts realized the importance of person-to-person influences in recommender systems, they are limited by the availability of prior knowledge of explicit trust values. These systems need to know not only who is trusting whom,

but also how much they trust each other. Thus, recommender systems that rely on explicit trust values cannot scale. In contrast, our system makes recommendations by using the correlations between friends, which can be viewed as implicit trust. We do not need to acquire trust values since they can be obtained from the rating correlations between friends. In addition to social influences, our system incorporates user's own attributes and the characteristics of items. These two factors are important for making target specific recommendations. Otherwise, recommender systems will simply suggest an item to a user whenever his/her trusted friend likes it.

Interestingly, [4] studied the factors that drive people's decision-making and advice-seeking through empirical studies, and found out that the profile similarity and rating overlap of a recommender have a significant impact on a person's decision. In addition, [4] suggested that recommender systems support the social element of advice seeking through communication and explicit user matching functions. Therefore, advice seekers can judge the validity and appropriateness of a recommendation. In Section 6 we proposed a recommender system design. In this design we think it is more important to consider the semantics in friend relationships when measuring their similarities based on user profiles and rating overlap.

More directly related work is found in [37]. Here, the authors proposed to combine social networks with recommender systems. They estimated the weights in collaborative filtering with an exponential function of the minimal distance of two users in a social network. This is, however, an over-simplified correlation between users. Distance has no semantic meaning of similarity, and two distant friends may still share common opinions. As noted by the authors, this approach does not work well. [37] also proposed another approach to reduce the computational cost in recommender systems by limiting the candidate similar users within a user’s social network neighbors. This approach actually will make the data sparsity problem of a recommender system even worse, because there are far less candidates for similar users than before.

## 8 Conclusions

Social networks provide an important source of information regarding users and their interactions. This is especially valuable to recommender systems. In this paper we presented a social network-based recommender system (SNRS) which makes recommendations by considering a user's own preference, an item's general acceptance and influence from friends. In particular, we proposed to model the correlations between immediate friends with the histogram of friend's rating differences. The influences from distant friends are also considered in an iterative classification. In addition, we have collected data from a real online social network. The analysis on this dataset reveals that friends have a tendency to review the same restaurants and give similar ratings. We compared the performance of SNRS with other methods, such as collaborative filtering (CF), friend average (FA), weighted friends (WVF) and naive Bayes (NB) with the same dataset. In

terms of the prediction accuracy, SNRS achieves the best result. It yields a 17.8% improvement compared to that of CF. In the sparsity test, SNRS returns consistently accurate predictions at different values of data sparsity. The coverage of SNRS decreases when the data is sparse but at a slower speed than CF. In the cold-start test, SNRS still performs well. We also studied the role of distant friends in SNRS, and found that by considering the influences from distant friends, the coverage of SNRS can be significantly improved with only a minor reduction in the prediction accuracy. The performance of SNRS can be further improved by selecting relevant friends for inference, which can be achieved by collecting the semantics of the friend relationships or fine-grained user ratings. Such an approach can be adopted by current content providers.

**Acknowledgements** The authors would like to thank Professor Zhuowen Tu and Mr. Jiayan Jiang of UCLA for their stimulating discussions in statistical modeling of social networks.

## References

1. G. Adomavicius, and A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering.
2. C. Basu, H. Hirsh, and W. Cohen. Recommendation as Classification: Using Social and Content-based Information in Recommendation. In *Recommender System Workshop* '98. pp. 11-15, 1998.
3. D. Billsus and M. Pazzani. Learning Collaborative Information Filters. In *Proceedings of International Conference on Machine Learning*, 1998.
4. P. Bonhard and M. A. Sasse. "Knowing me, knowing you" - Using Profiles and Social Networking to Improve Recommender Systems. *BT Technology Journal*, Vol 24 No 3, July 2006.
5. J. S. Breese, D. Heckerman, and C. Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pp. 43-52, 1998.
6. V. Carvalho, and W. W. Cohen. On the Collective Classification of Email Speech Acts. *Special Interest Group on Information Retrieval*. 2005
7. B. Chandra, M. Gupta, and M. P. Gupta. Robust Approach for Estimating Probabilities in Naive-Bayes Classifier. *In Pattern Recognition and Machine Intelligence*, Volume 4815/2007, pp. 11-16, 2007
8. P. Domings, and M. Richardson. Mining the Network Value of Customers. In *Proceedings of Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 57-66. ACM Press, New York, 2001.
9. J. Golbeck. Generating Predictive Movie Recommendations from Trust in Social Networks. In *Proceedings of the Fourth International Conference on Trust Management*. 2006
10. J. He, W. W. Chu and Z. Liu. Inferring Privacy Information from Social Networks. In Proceedings of *IEEE Intelligence and Security Informatics Conference (ISI 2006),* San Diego, California, May 2006.

11. J. L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl. An Algorithmic Framework for Performing Collaborative Filtering. In *Proceedings of ACM SIGIR*, pp. 230-237. 1999, Berkley, USA.
12. Z. Huang,, H. Chen, D. Zeng (2004). Applying Associative Retrieval Techniques to Alleviate the Sparsity Problem in Collaborative Filtering. *ACM Trans. on Information Systems*, 22(1), 116–142.
13. R. Jin, J. Y. Chai, and L. Si. An Automatic Weighting Scheme for Collaborative Filtering. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval (SIGIR04)*, 2004
14. S. Jurvetson. What Exactly is Viral Marketing? *Red Herring*, 78:110–112, 2000.
15. N. Lathia, S. Hailes, and L. Capra. The Effect of Correlation Coefficients on Communities of Recommenders. In *Proceedings of SAC'08*, March 16-20, 2008.
16. D. Lowd and P. Domingos. Naive Bayes Models for Probability Estimation. In *Proceedings of the Twenty-Second International Conference on Machine Learning (ICML)*. Bonn, Germany: ACM Press, 2005.
17. Q. Lu and L. Getoor. Link-Based Classification. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML)*, pages 496-503, 2003.
18. S. Macskassy and F. Provost. A Simple Relational Classifier. In *Proceedings of the KDD-2003 Workshop on Multirelational Data Mining*, 2003.
19. S. Marti and H. Garcia-Molina. Taxonomy of trust: Categorizing P2P reputation systems. *Computer Networks* 50(4), 472-484 (2006)
20. P. Massa, and P. Avesani. Trust-aware collaborative filtering for recommender systems. In *Proceedings of International Conference on Cooperative*. 2004.
21. P. Melville, R. J. Mooney, and R. Nagarajan. Content-Boosted Collaborative Filtering for Improved Recommendations. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002),* pp. 187-192, Edmonton, Canada, July 2002.
22. R. J. Mooney and L. Roy. Content-Based Book Recommending Using Learning for Text Categorization. In *Proceeding of ACM SIGIR'99 Workshop Recommender Systems: Algorithms and Evaluation*, 1999.
23. M. Morita and Y. Shinoda. Information Filtering based on User Behavior Analysis and Best Match Text Retrieval. In *Proceedings of the Seventh Annual Information ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 272-281, 1994.
24. J. Neville and D. Jensen. Iterative Classification in Relational Data. In *Proceedings of the Workshop on Learning Statistical Models from Relational Data at the Seventeenth National Conference on Artificial Intelligence (AAAI)*, pp. 13-20, 2000.
25. M. Pazzani and D. Billsus. Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning*, vol. 27, pp. 313-331, 1997.
26. M. Pazzani. A Framework for Collaborative, Content-Based, and Demographic Filtering. *Artificial Intelligence Rev.*, pp. 393-408, Dec. 1999.
27. P. Resnick, N. Iakovou, M. Sushak, P. Bergstrom, and J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of 1994 Computer Supported Cooperative Work Conference*, 1994.
28. M. Richardson, and P. Domingos. Mining Knowledge-Sharing Sites for Viral Marketing. In Proceedings of Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 61-70. ACM Press, New York, 2002.
29. Sarwar, G. Karypis, J. Konstain, and J. Riedl. Application of Dimensionality Reduction in Recommender Systems – A Case Study. In Proceedings of ACM WebKDD Workshop, 2000.

30. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-Based Collaborative Filtering Recommendation Algorithms. In Proceedings of 10th International WWW Conference, 2001.
31. P. Sen and L. Getoor. Empirical Comparison of Approximate Inference Algorithms for Networked Data. *ICML Workshop on Open Problems in Statistical Relational Learning*, Pittsburgh, PA, 2006.
32. M. R. Subramani and B. Rajagopalan. Knowledge-Sharing and Influence in Online Social Networks via Viral Marketing. *Communications of the ACM*, 46(12):300–307, 2003.
33. B. Taskar, P. Abbeel, and D. Koller. Discriminative Probabilistic Models for Relational Data. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 485-492, 2002.
34. L. H. Ungar, and D. P. Foster. Clustering Methods for Collaborative Filtering. In *Workshop on Recommender Systems at the 15th National Conference on Artificial Intelligence*, 1998.
35. S. Yang and G. M. Allenby. Modeling interdependent Consumer Preferences. Journal of Marketing Research 40 282-294, 2003.
36. J. Wang, A. P. Vires, M. J.T. Reinders. Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval (SIGIR06)*, August 6-11, 2006.
37. R. Zheng, F. Provost and A. Ghose. Social Network Collaborative Filtering: Preliminary Results. In *Proceedings of the Sixth Workshop on eBusiness (WEB2007)*, December 2007.