Predicting Trust from User Ratings

by

Nikolay Korovaiko
B.Sc., Kazakh State University, 2008

A Thesis Submitted in Partial Fullfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Predicting Trust from User Ratings

by

Nikolay Korovaiko
B.Sc., Kazakh State University, 2008

Supervisory Committee

Dr. Alex Thomo (Department of Computer Science)
Supervisor

Dr. Venkatesh Srinivasan (Department of Computer Science)
Committee Member

Supervisory Committee

Dr. Alex Thomo (Department of Computer Science)
Supervisor

Dr. Venkatesh Srinivasan (Department of Computer Science)
Committee Member

Abstract

Trust relationships between users in various online communities are notoriously hard to model for computer scientists. It can be easily verified that trying to infer trust based on the social network alone is often inefficient. Therefore, the avenue we explore is applying Data Mining algorithms to unearth latent relationships and patterns from background data. In this paper, we focus on a case where the background data is user ratings for online product reviews. We consider as a testing ground a large dataset provided by Epinions.com that contains a trust network as well as user ratings for reviews on products from a wide range of categories. In order to predict trust we define and compute a critical set of features, which we show to be highly effective in providing the basis for trust predictions. Then, we show that state-of-the-art classifiers can do an impressive job in predicting trust based on our extracted features. For this, we employ a variety of measures to evaluate the classification based on these features. We demonstrate that by carefully collecting and synthesizing readily available background information, such as ratings for online reviews, one can accurately predict trust-based social links.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the explosive growth in popularity of social networks and e-commerce systems users are constantly in interaction with each other. The trust factor plays an important role in initiating these interactions and building higher-quality relationships between the users. Even though trust takes many different meanings and highly depends on the context in which users interact with each other, it has been shown that trust can be approximated from other relationships.

Consider a few examples. In Epinions.com, trusting a particular person often means that the reviews written by that person are highly appreciated or that the person has preferences similar to the trustor. E-commerce systems suggest another example. We are more willing to buy an item from a particular seller on E-Bay or Amazon, if either we or our friends had positive experience with that seller in past. On the other hand, we are reluctant to engage in any relationship with strangers. On freelance websites trust means fruitful agreements between a professional and employer. Dating services might try to leverage users' preferences to help their users find a perfect match.

Our attitudes towards trust are often very different and individual. One might believe that a particular seller on E-Bay provides an excellent service, even though this seller sometimes delays shipping by a week. For another person any delay might be unacceptable. Trust-aware systems can help users make the right choices and have relationships that lead to positive outcomes.

1.1 The Trust Prediction Problem

On Epinions, users interact with each other in many ways. For example, the users participate in discussions that grow around various products and write reviews on the products. The rest of the user community comments and rates these reviews. The raters may opt to not show ratings given to reviews.

Additionaly, users can specify whom they trust, thus creating a social network based on trust connections. The problem we study here is how to predict trust. This is an important problem which, when solved effectively, enhances the user online experience by connecting him/her to peers who share the same interests and values. The user can rely on the input from her peers or trustees to form her own opinion about a particular product much faster and easier.

Incorporating background information into trust prediction algorithms allows for more accurate results in general. Furthermore, we might be able infer trust relation-

ships in cases where the other algorithms fail. For instance, by using background information in the form of user ratings for online reviews we might be able to find users who have similar preferences and thus probably trust each other even though, in terms of the current trust graph, they appear to be quite far from each other.

There are various reasons why we decide to trust another person. We might know a person for a long time or we might share many interests in common. The person might be very reliable and trustworthy or just knowledgeable in a particular topic. There are a few key factors affecting the decision of a particular user to trust another one. First, both users might simply have very similar preferences. In other words, they tend to like the same items. Second, the user can trust the other one if she decides that the person is a good reviewer who writes high quality reviews on some products on Epinions. Third, the user might think of the other person as being a good review critic. Finally, both users might be friends. In the latter case, there is typically a mutual trust link between the users, even if they do not have that many things in common.

The rest of this thesis is organized as follows. Chapter 2 discusses previous research on rating/trust prediction and highlights our contributions. In Chapter 3, we derive a set of complex features based on user similarity and rater-reviewer relationships between the users on Epinions. Chapter 4 contains background information relevant

to Random Forests, which is a classification algorithm that we heavily use for building trust prediction models. Chapter 5 expands on the approaches we have mentioned in related works. Chapter 6 includes analysis, evaluation and comparison of our features with the aforementioned approaches. In Chapter 7, we propose an algorithm for inferring trust relationships for a particular user, based on the opinions and input from his/her trustees. Chapter 8 concludes our discussion on trust prediction with several interesting observations and ideas for future research.

Chapter 2

Related Work and Our Contributions

2.1 Rating Prediction

The rating prediction (RP) problem is a counterpart of trust prediction (TP) in many ways. A number of RP algorithms are augmented to employ available trust networks (cf. [5, 1, 19]). On the other hand, there is an intensive on-going research to incorporate additional information including user ratings given to products to enhance TP algorithms. The typical definition for the rating prediction problem is as follows. The model is defined as a tuple $\langle U, I, R \rangle$, where U and I denote the set of users and items (products), respectively. A rating $r_{ui} \in R$ indicates the preference by user u towards item i. Higher values, e.g. 4, 5 (on the scale of 1 to 5), mean stronger preference. Usually, the overwhelming majority of ratings is unknown. For example, in the Netflix data 99% of the possible ratings are missing.

Collaborative Filtering (CF)[1] is one of the most popular classes of algorithms attempting to solve the RP problem. CF systems rely only on past user behavior e.g., their previous transactions or product ratings and does not require explicit profiles.

[1] The term has been coined by the developers of one the first CF systems, Tapestry [9].

Notably, CF techniques require no domain knowledge and avoid the need for extensive data collection. Relying directly on user behavior allows one to uncover complex and unexpected patterns that would be difficult or impossible to profile using known data attributes. Thus, CF algorithms attracted a lot of attention, resulting in remarkable progress and being adopted by large commercial systems, including Amazon, Google, and Netflix (TV shows and movies). The interest in CF systems has been re-ignited in October 2006, when the Netflix Prize competition [4] started. Netflix released a dataset containing 100 million movie ratings and challenged the research community to develop algorithms that could beat the accuracy of its recommendation system, Cinematch. A number of improvements and extensions have been proposed to existing CF algorithms over this period (cf. [3, 2, 23, 14]).

Some of our techniques for trust prediction have been inspired by research on Collaborative Filtering.

2.2 Trust Prediction

Jennifer Golbeck was one of the first pioneers to research the problem of trust prediction from a Computer Science perspective. In [8], Golbeck discusses various properties of trust such as *transitivity*, *composability* and *asymmetry*. Then, she proposes algorithms for inferring binary and continuous trust values from trust networks. The

binary trust inference algorithm infers the trust link between a source (trustor) and sink (trustee) as follows. The "source" polls its trusted neighbors to collect their ratings for the "sink." A neighbor replies with a rating if he/she has a trust connection with the sink. The ratings are averaged and rounded to receive the trust rating for the sink. The neighbors in turn follow the same procedure to infer their trust ratings. rating for this arc is used. The trust ratings can be either rounded for the source only or for each node on the path from source to the sink.

The continuous trust inference algorithm, *TidalTrust*, leverages the path length from the source to sink and various properties of continuous ratings. The simplest version of the algorithm computes the trust rating from the source to the sink as the weighted sum of the neighbors' ratings. The source's trust ratings to the neighbors are used as weights.

Golbeck also suggests another trust inference algorithm called *Sunny* [15]. The algorithm uses a probabilistic sampling technique to estimate our confidence in the trust information from some designated sources. Then, it computes an estimate of trust based on only those information sources with high confidence estimates. The algorithms mentioned above use only trust networks to infer trust relations between users. However, there is often additional information about other user relationships that is available.

In [16], the authors develop a taxonomy of user relationships for the Epinions dataset. This taxonomy is then used to obtain an extensive set of simple features derived from the user relationships in the online community. The features are split into two large groups: user and interaction factors. The first group includes rater-related, writer-related, or commenter-related. The latter captures various interactions between the users. This taxonomy results into a very large number of low-level features. The authors use the Naive Bayes and Support Vector Machine (SVM) classifiers in order to evaluate the discriminative power of the features. However, it is not always feasible to employ this overwhelmingly large number of features. Moreover some of features can be very naturally combined into a single one.

Viet-An Nguyen and *et. al* [20] derive several trust prediction models from a well-studied Trust Antecedent Framework used in management science. The framework captures the three following factors: ability, benevolence and integrity. The authors approximate each factor through a set of quantitative features. For instance, integrity corresponds to the feature called *trustworthiness*. The trustworthiness of a particular user is equal to the number of trust statements the user receives. Ability includes the features that compute the average rating given by a rater to the reviews written by a particular reviewer and the number of reviews rated by the rater. The experiments

conducted show that the proposed models including the ones derived by a SVM classifier have a good performance.

In [17], various features from writer-reviewer interactions are derived and used in personalized and cluster-based classification methods. The former trains one classifier for each user using user-specific training data. The cluster-based method first constructs user clusters before training one classifier for each user cluster. The proposed methods have been evaluated in a series of experiments using two datasets from Epinions.com and have given good results.

2.3 Our Contributions

In this work, we consider both types of relationships: Rater-Rater (User Similarity) and Rater-Reviewer activities. The authors in [20] limit the models to only Rater-Reviewer interactions neglecting important User Similarity information. We experiment with constructing more complex features that allow us to augment various classification algorithms for the trust prediction problem. The experiments show the classifier trained on the complex features often performs much better than the one trained on the simpler features used in [16]. In general, we achieve 5-20% improvements in performance (e.g. precision, recall, f-measure and Roc Area) over the other approaches, which is a substantial gain for this particular problem. In comparison,

the Netflix Prize [4] of $1,000,000 has been awarded for improving the root mean squared error by 10% for rating prediction. We also apply various Data Mining techniques to the features in order to incorporate or remove the biases from both users and evaluate the impact that the biases have on the performance. Also, we introduce an additional Personalized Trust Prediction Model. The model makes predictions for a particular user by weighing in the opinions of the user's trustees.

Chapter 3

Features for Trust Prediction

Users interact, often implicitly, with each other in online communities, such as Epinions. In particular, in the Epinions case, users write reviews about different products as well as rate the reviews of other people. The users can also create a network of trusted users by issuing trust statements. Why do users issue such trust statements? Their main reason is to express their liking of the reviews written by the trusted user. Then, in the future, the users focus first on reading the reviews of the users they trust. However, users often need help in determining whom to trust. In order to help users in this discovery, an intelligent trust recommendation system needs to be build. This system would try to accurately predict trust among users. The predictions could be turned into effective trust recommendations that will greatly enhance the online experience of users.

In this work, we propose the following parameters to explore for predicting uni- or bi- directional trust (or distrust) between two users u and v.

1. u and v give similar ratings to the reviews they read
2. u and v are interested in similar categories of products

3. u and v produce reviews in the same categories that interest them
4. u and v rate the reviews produced by the same reviewers
5. u gives high ratings to reviews produced by v
6. u anonymizes a considerable number of ratings for reviews produced by v
7. v is a reputable reviewer
8. u and v have the same trustees.

Of course, the users might trust each other due to reasons other than the above. For example, they might have been friends for a long time. If this is the case, the users might not have many things in common, even though there are mutual trust links between them. Such trust links should be treated differently from regular ones or even filtered out, so that they do not introduce extra noise into the trust prediction algorithms.

In order to capture the above eight listed parameters in terms of formal features, we first introduce some notation. In the following we will interchangeably use item and review. Let u, v, y be users. We denote by

U the set of users

I_u the set of items rated by u

$I_{u,r}$ the set of items rated r (where $r \in \{1, 2, 3, 4, 5\}$) by u

$I_{u,c}$ the set of items in category c rated by u

$I_{u,y}$ the set of reviews (items) produced by y and rated by u

J_v the set of reviews (items) produced by v

C_u the set or multiset (depending on the feature) of categories of the items in I_u

D_u the set or multiset (depending on the feature) of categories of the reviews (items) produced by u

Y_u the set or multiset (depending on the feature) of reviewers (users) who have produced the reviews (items) in I_u

T_u the set of trustees of user u, *i.e.* those users that u trusts.

For simplicity we will denote by $r_{u,i}$ a rating (u, i, r) that a user u gives for item i. This also reflects the fact that for a given user and a given item there can be not more than one rating.

We treat trust prediction as a classification problem, that is, for each ordered pair (u, v) of users, a new value called class (trust or distrust) has to be assigned. There

is a rich repertoire of classifier algorithms available for the classification problem in the field. We choose to use the Random Forests classifier as this is the one to give the best classification results in test data.

There are several options to convert the eight aforementioned parameters into a set of features.

Parameter 1: *u and v give similar ratings to the reviews they read.*

The first feature we propose is to represent both users as sets of their ratings and then compute the Pearson Correlation (PC) of these sets. Specifically, we define $f_{1,a}$ to be

$$f_{1,a} = \frac{\sum_{i \in I_u \cap I_v} (r_{ui} - \bar{r}_u)(r_{vi} - \bar{r}_v)}{\sqrt{\sum_{i \in I_u \cap I_v} (r_{ui} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_u \cap I_v} (r_{vi} - \bar{r}_v)^2}}.$$

The second, third, fourth, and fifth attribute we propose are based on the number of partisan ratings, 5, 4, and 1, 2. Typically, ratings of 4 or 5 are a strong indicator of user likes, whereas ratings of 1 or 2 are a strong indicator of user dislikes. Intuitively, when u and v have a relatively significant number of compatible partisan ratings, their likes and dislikes are aligned. On the other hand, when u and v have incompatible partisan ratings, e.g. u gives a rating of 1 whereas v gives a rating of 5, their preferences exhibit a conflict.

The observations above can be converted into four features $f_{1,b}$, $f_{1,c}$, $f_{1,d}$, and $f_{1,e}$ we give below. These features are given in terms of Jaccard Similarity and they

measure the relative weight of partisan agreements or disagreements. Let

$$I_{u,\uparrow} = I_{u,5} \cup I_{u,4}$$

$$I_{u,\downarrow} = I_{u,1} \cup I_{u,2}.$$

Also similarly define $I_{v,\uparrow}$ and $I_{v,\downarrow}$. Now we define

$$\begin{aligned} f_{1,b} &= \frac{|I_{u,\uparrow} \cap I_{v,\uparrow}|}{|I_{u,\uparrow} \cup I_{v,\uparrow}|} \\ f_{1,c} &= \frac{|I_{u,\downarrow} \cap I_{v,\downarrow}|}{|I_{u,\downarrow} \cup I_{v,\downarrow}|} \\ f_{1,d} &= \frac{|I_{u,\uparrow} \cap I_{v,\downarrow}|}{|I_{u,\uparrow} \cup I_{v,\downarrow}|} \\ f_{1,e} &= \frac{|I_{u,\downarrow} \cap I_{v,\uparrow}|}{|I_{u,\downarrow} \cup I_{v,\uparrow}|}. \end{aligned}$$

Example 1. Suppose u and v rate i_1, i_2, i_3, i_4, i_5, i_6, i_7, and i_8 as follows.

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u	1	2	1	4	5	5	1	5
v	1	5	4	5	4	1	1	5

We have

$$\begin{aligned} f_{1,b} &= \frac{|\{i_4, i_5, i_8\}|}{|\{i_2, i_3, i_4, i_5, i_6, i_8\}|} = \frac{1}{2} \\ f_{1,c} &= \frac{|\{i_1, i_7\}|}{|\{i_1, i_2, i_3, i_6, i_7\}|} = \frac{2}{5} \\ f_{1,d} &= \frac{|\{i_6\}|}{|\{i_1, i_4, i_5, i_6\}|} = \frac{1}{4} \\ f_{1,e} &= \frac{|\{i_2, i_3\}|}{|\{i_1, i_2, i_3, i_4, i_5\}|} = \frac{2}{5}. \end{aligned}$$

□

Parameter 2: *u and v are interested in similar categories of products.*

Often datasets, such as those from Epinions, do not contain explicit user preferences for different item categories. However latent preferences can be discovered by considering the ratings the users have given to items belonging in given categories.

The first feature we define in this group is

$$f_{2,a} = \frac{|C_u \cap C_v|}{|C_u \cup C_v|}$$

which measures the amount of overlap between the categories of items that u and v have rated. Here C_u and C_v are considered to be multisets. We continue to use Jaccard Similarity in order to take into consideration not only the categories the users prefer in common, but also the users' range of activity.

If a user is interested in a particular category, he typically reads and rates more items in that category. Counting the number of items for the category allows us to estimate the user interest in it. Next, we compute the Pearson Correlation for the two sets (of u and v) of those counts. Formally, we have

$$f_{2,b} = \frac{\sum_{c \in C_u \cap C_v} \left(|I_{u,c}| - \frac{|I_u|}{|C_u|}\right)\left(|I_{v,c}| - \frac{|I_v|}{|C_v|}\right)}{\sqrt{\sum_{c \in C_u \cap C_v} \left(|I_{u,c}| - \frac{|I_u|}{|C_u|}\right)^2} \sqrt{\sum_{c \in C_u \cap C_v} \left(|I_{v,c}| - \frac{|I_v|}{|C_v|}\right)^2}}.$$

In this definition, C_u and C_v are considered as sets.

Another way to receive the estimate of the user's interest in different categories is to compute the user's average ratings for each of the categories. Then we compute the Pearson Correlation for the sets of these averages for u and v.

$$f_{2,c} = \frac{\sum_{c \in C_u \cap C_v} (r_{u,c} - \hat{r}_u)(|r_{v,c}| - \hat{r}_v)}{\sqrt{\sum_{c \in C_u \cap C_v} (r_{u,c} - \hat{r}_u)^2} \sqrt{\sum_{c \in C_u \cap C_v} (r_{v,c} - \hat{r}_v)^2}}$$

where

$$\begin{aligned} r_{u,c} &= \frac{\sum_{i \in I_{u,c}} r_{u,i}}{|I_{u,c}|} \\ \hat{r}_u &= \frac{\sum_{c \in C_u} r_{u,c}}{|C_u|} \\ r_{v,c} &= \frac{\sum_{i \in I_{v,c}} r_{v,i}}{|I_{v,c}|} \\ \hat{r}_v &= \frac{\sum_{c \in C_v} r_{v,c}}{|C_v|}. \end{aligned}$$

For this definition as well, C_u and C_v are considered as sets.

Parameter 3: *u and v produce reviews in the same categories that interest them.*

For this parameter we employ the Jaccard Similarity with respect to the categories of the reviews user u and v have produced. Specifically, we propose the following feature.

$$f_3 = \frac{|D_u \cap D_v|}{|D_u \cup D_v|}.$$

In this definition, D_u and D_v are considered as multisets.

Parameter 4: *u and v rate reviews produced by the same reviewers.*

Another indication of similar user preferences is when both users favor the reviews written by the same reviewers. Typically, if a user likes a reviewer, the user gives higher ratings to reviews from that reviewer. It might seem that this parameter is always correlated with the first one, as the same rating information is used. However, there is often no correlation between two. Consider an example. User u rates the review i written by reviewer r. User v rates a *different* review, j, produced by the same reviewer r. The occurrences of this case are very frequent. We assume that the average rating given by a user to the reviews from a reviewer reflects the user's preferences towards the reviewer. We employ the Pearson correlation to compute the similarity between the two sets of average ratings given by u and v to reviewers. Formally,

$$f_4 = \frac{\sum_{y \in Y_u \cap Y_v} (r_{u,y} - \check{r}_u)\,(|r_{v,y}| - \check{r}_v)}{\sqrt{\sum_{y \in Y_u \cap Y_v} (r_{u,y} - \check{r}_u)^2}\sqrt{\sum_{y \in Y_u \cap Y_v} (r_{v,y} - \check{r}_v)^2}}$$

where

$$\begin{aligned} r_{u,y} &= \frac{\sum_{i \in I_{u,y}} r_{u,i}}{|I_{u,y}|} \\ \check{r}_u &= \frac{\sum_{y \in Y_u} r_{u,y}}{|Y_u|} \\ r_{v,y} &= \frac{\sum_{i \in I_{v,y}} r_{v,i}}{|I_{v,y}|} \\ \check{r}_v &= \frac{\sum_{y \in Y_v} r_{v,y}}{|Y_v|}. \end{aligned}$$

In this definition, Y_u and Y_v are considered as sets.

Parameter 5: *u gives high ratings to reviews produced by v.*

An approach is to compute an average rating that the user gives to the reviews (items) produced by v. However, the baseline predictors technique suggested in [13]

allows us to improve on this crude average by including the linear sum of four components. The first component is the global average of all ratings in our sampled dataset, which we denote by $\bar{r}$. The second, third, and fourth components are the differences from the global average of the following averages:

- the average of all ratings given to the items produced by v, denoted by $\breve{r}_v$
- the average of all ratings u gives, denoted by $\bar{r}_u$
- the average of all ratings that u gave to the items produced by v, denoted–similarly as for Parameter 4–by $r_{u,v}$.

We have

$$f_{5,a} = \bar{r} + (\breve{r}_v - \bar{r}) + (\bar{r}_u - \bar{r}) + (r_{u,v} - \bar{r})\,.$$

Two other features we define are the fraction of high (low) ratings u gives to reviews (items) produced by v. Namely, we have

$$f_{5,b} = \frac{|I_{u,\uparrow} \cap J_v|}{|I_{u,\uparrow}|}$$

and

$$f_{5,c} = \frac{|I_{u,\downarrow} \cap J_v|}{|I_{u,\downarrow}|}.$$

Parameter 6: *u anonymizes a considerable number of ratings for reviews produced by v.*

We start by computing the ratio of anonymized ratings u gives to the v's items, $\frac{|I^-_{u,v}|}{|I_{u,v}|}$, where $I^-_{u,v} \subseteq I_{u,v}$ is the set of v's items rated anonymously by u.

We then we consider the high or low ratings only, and have $\frac{|I^-_{u,v,\uparrow}|}{|I_{u,v,\uparrow}|}$ ($f_{6,d}$) and $\frac{|I^-_{u,v,\downarrow}|}{|I_{u,v,\downarrow}|}$ ($f_{6,e}$), where $I^-_{u,v,\uparrow}$, $I_{u,v,\uparrow}$, $I^-_{u,v,\downarrow}$, and $I_{u,v,\downarrow}$ are defined as their non-arrow counterparts, but considering the high or low ratings only.

The baseline predictors technique can be also applied to these ratios. For this, let

R the set of all ratings

R^- the set of all anonymous ratings ($R^- \subseteq R$)

$R_{\rightarrow v}$ the set of all ratings for v's items

$R^-_{\rightarrow v}$ the set of all anonymous ratings for v's items

$R_{u\rightarrow}$ the set of all u's ratings

$R^-_{u\rightarrow}$ the set of all anonymous u's ratings.

Also let $R_{\uparrow}$, $R^{-}_{\uparrow}$, $R_{\rightarrow v,\uparrow}$, $R^{-}_{\rightarrow v,\uparrow}$, $R_{u\rightarrow,\uparrow}$, $R^{-}_{u\rightarrow,\uparrow}$ be defined similarly as above, but with only the high ratings considered. Likewise for $R_{\uparrow}$, $R^{-}_{\downarrow}$, $R_{\rightarrow v,\downarrow}$, $R^{-}_{\rightarrow v,\downarrow}$, $R_{u\rightarrow,\downarrow}$, $R^{-}_{u\rightarrow,\downarrow}$ but with only the low ratings considered.

We now define

$$f_{6,a} = \frac{|R^{-}|}{|R|} + \frac{|R^{-}_{\rightarrow v}|}{|R_{\rightarrow v}|} + \frac{|R^{-}_{u\rightarrow}|}{|R_{u\rightarrow}|} + \frac{|I^{-}_{u,v}|}{|I_{u,v}|}$$
$$f_{6,b} = \frac{|R^{-}_{\uparrow}|}{|R_{\uparrow}|} + \frac{|R^{-}_{\rightarrow v,\uparrow}|}{|R_{\rightarrow v,\uparrow}|} + \frac{|R^{-}_{u\rightarrow,\uparrow}|}{|R_{u\rightarrow,\uparrow}|} + \frac{|I^{-}_{u,v,\uparrow}|}{|I_{u,v,\uparrow}|}$$
$$f_{6,c} = \frac{|R^{-}_{\downarrow}|}{|R_{\downarrow}|} + \frac{|R^{-}_{\rightarrow v,\downarrow}|}{|R_{\rightarrow v,\downarrow}|} + \frac{|R^{-}_{u\rightarrow,\downarrow}|}{|R_{u\rightarrow,\downarrow}|} + \frac{|I^{-}_{u,v,\downarrow}|}{|I_{u,v,\downarrow}|}.$$

In $f_{6,a}$ the first component is how often, on average, users decide to keep their ratings anonymous, the second is how often, on average, v receives anonymous ratings, and the third is how often, on average, u gives anonymous ratings. Similar comments can also be made for the $\uparrow$ and $\downarrow$ components, but considering the high or low ratings only.

Parameter 7: *v is a reputable reviewer and u is a lenient rater.*

This parameter deals with a reviewer reputation and how the reputation affects the readers' decision to trust the reviewer. Typically, the users express their appreciation to the reviewer by giving higher ratings to his items. The more positive ratings a reviewer receives, the higher his reputation. This observation can be converted into a feature by computing the difference of the average rating given to the items produced

by v from the overall average rating in the dataset.

$$f_{7,a} = \check{r}_v - \bar{r}.$$

On the other hand, the leniency of u also affects the trust decision. u might be very lenient comparing to an overall user leniency giving higher ratings to the reviews and trusting the reviewers more often. The leniency is computed as the difference between the average rating that u gives to the reviews and the overall average rating in the dataset.

$$f_{7,b} = \check{r}_u - \bar{r}.$$

The last feature we include for this parameter is equal to the difference between the average rating given to v and the average rating given by u

$$f_{7,c} = \check{r}_v - \check{r}_u.$$

Parameter 8: *u and v have the same trustees.*

This parameter captures the similarity between the sets of trustees of two users. The intuition behind it is that if the users tend to have the same trustees, they might decide to trust each other as well.

$$f_8 = \frac{|T_u \cap T_v|}{|T_u \cup T_v|}.$$

Chapter 4

Random Forests

4.1 Decision Trees

Machine Learning has a rich repertoire of classification algorithms. Decision trees (or tree classifiers) are one of the most prominent on the Data Mining scene. One might think of a tree classifier as a function that maps a vector $X = (x_1, x_2, \ldots x_n)$ of input parameters called *attributes* to some value Y called the *class*. A pair of $\langle X, Y \rangle$ is usually called an *instance*. A tree can be "learned" by splitting the set of instances into subsets based on an attribute value. This process is repeated on each derived subset in a recursive manner. The process is completed when every instance at a node is of the same class, or when splitting no longer adds value to the predictions. The tree classifier algorithm uses one of *impurity measures* (e.g. *gini*, *entropy*, etc.) to decide on which attribute the data should be split. There are different variants of tree classifiers available. Some decision trees are able to handle both continuous and categorial attributes, whereas some others always split into two branches. In this work we use Random Forest, which is based on a decision tree algorithm called J48

(or C4.5) [10]. C4.5 has the following advantages (see [22]), which are inherited by Random Forest as well.

1. It handles both continuous and discrete attributes. In order to handle continuous attributes, C4.5 creates a threshold and then splits the list into those whose attribute value is above the threshold and those that are less than or equal to it.
2. It handles training data with missing attribute values. Missing attribute values are simply not used in gain and entropy calculations.
3. It handles attributes with differing costs.
4. It prunes trees after creation. C4.5 goes back through the tree once it has been created and attempts to remove branches that do not help by replacing them with leaf nodes.

Table 4.1 and Figure 4.1 contain a very simple dataset, *playTennis*, a decision tree constructed for it using C4.5.

4.2 Random Forests

We mainly use the Random Forest (RF) classifier to construct a classification model on our features. Random Forest is a classification algorithm developed by Leo Breiman and Adele Cutler [24]. It grows an ensemble of C4.5 trees and each tree votes for the

output class. Suppose, the size of the training set is N and the number of features is M. A tree in a forest is constructed as follows:

1. Only m features selected randomly are used to calculate the best split at a particular node. Breiman and Cutler suggested to use m equal to $log_2 M + 1$.
2. The training set is chosen for the tree by choosing n times with replacement from all the N available training instances. The rest is used to estimate the error of the tree, by predicting their classes.
3. The tree is fully grown and not pruned.

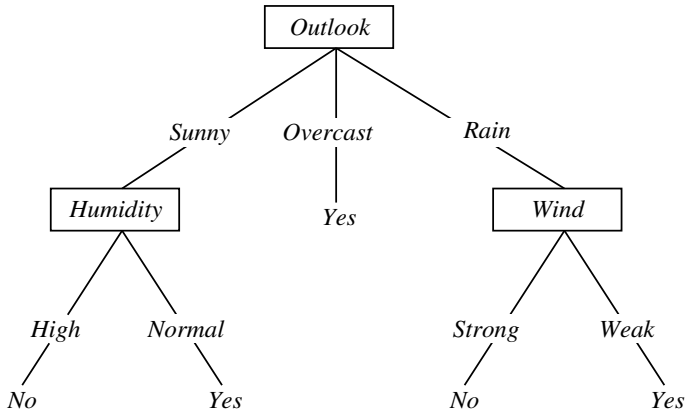

Fig. 4.1. Decision Tree for the *playTennis* dataset

The generalization error depends on two characteristics of the forest:

outlook	temperature	humidity	windy	play
sunny	85	85	FALSE	no
sunny	80	90	TRUE	no
overcast	83	86	FALSE	yes
rainy	70	96	FALSE	yes
rainy	68	80	FALSE	yes
rainy	65	70	TRUE	no
overcast	64	65	TRUE	yes
sunny	72	95	FALSE	no
sunny	69	70	FALSE	yes
rainy	75	80	FALSE	yes
sunny	75	70	TRUE	yes
overcast	72	90	TRUE	yes
overcast	81	75	FALSE	yes
rainy	71	91	TRUE	no

Table 4.1. *playTennis* dataset

1. The correlation between any two trees in the forest. Increasing the correlation increases the forest error rate.
2. The strength of each individual tree in the forest.

The strength and correlation depend on m. Reducing m reduces both the correlation and the strength. Increasing it increases both. One can determine the optimal value of m by using the out-of-bag error (see [24]).

As per Wikipedia article on Random Forest[1], RF has the following advantages:

1. It is one of the most accurate learning algorithms available. For many data sets, it produces a highly accurate classifier.
2. It runs efficiently on large data bases.
3. It can handle thousands of input variables without variable deletion.
4. It gives estimates of what variables are important in the classification.
5. It generates an internal unbiased estimate of the generalization error as the forest building progresses.
6. It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing.
7. It has methods for balancing error in class population unbalanced data sets.
8. Generated forests can be saved for future use on other data.

[1] See `http://en.wikipedia.org/wiki/Random_forest`

9. Prototypes are computed that give information about the relation between the variables and the classification.
10. It computes proximities between pairs of cases that can be used in clustering, locating outliers, or (by scaling) give interesting views of the data.
11. The capabilities of the above can be extended to unlabeled data, leading to unsupervised clustering, data views and outlier detection.
12. It offers an experimental method for detecting variable interactions.

Chapter 5

Trust Prediction Models

In this chapter, we describe two trust prediction models introduced in the literature, against which we compare our approach later. These models are the Trust Antecedent Framework of [20] and the Taxonomy Model of [16].

5.1 Trust Antecedent Framework

Trust Antecedent Framework ([20]) is based on the real-life notions of *ability*, *benelovence*, and *integrity*. In other words, a trustee is evaluated whether he/she

1. is competent enough to deliver a desired outcome (ability),
2. cares for a trustor (benelovence), and
3. adheres to a set of moral principles (integrity).

Ability is approximated with the following set of features.

– Average rating for u_j received from u_i, denoted by s_{ij}. The average rating tells us whether u_i appreciates the reviews written by u_j. To keep the average rating within $[0, 1]$, the authors convert the raw rating scores to $[0, 1]$ by mapping 1 to 5 rating stars to 0.2, 0.4, 0.6, 0.8 and 1.0 respectively.

– Interaction intensity from u_i to u_j, denoted by i_{ij}. This corresponds to the number of reviews of u_j rated by u_i. The authors introduce a transformation function ξ to normalize intensity:

$$i_{ij} = \xi(|R_{ij}|, \alpha, \mu)$$

where $\xi(x, \alpha, \mu)$ is computed as

$$\frac{1}{1 + e^{-\alpha \cdot (x - \mu)}},$$

and μ and α are set to 5 and 0.1, respectively.

Benevolence is often associated with such characteristics as helpfulness, caring, loyalty, receptivity, etc. It is approximated through the *local leniency* (l_{ij}) that accounts for the fact that users have different standards in giving ratings. This is the relative difference between the u_i ratings on the reviews written by u_j and the actual quality of these reviews. Specifically,

$$l_{ij} = \text{Avg}_{r_k \in R_{ij}} \left(\frac{s_{ik} - q_k}{s_{ik}} \right).$$

Quality, q_k, is equal to

$$o_k \cdot \text{Avg}_{r_k \in U_k^R}(s_{ik} \cdot (1 - l_{iw(r_k)} \cdot \beta))$$

where $o_k = \xi(|U_k^R|, \alpha' \mu')$, U_k^R specifies the set of users who rate the review r_k, and $w(r_k)$ denotes the user who wrote the review r_k. μ' and α' are set to 5 and 0.1,

respectively. β is a value in [0,1] to control the maximum amount of score adjustment on s_{ik}. It is set to 0.5.

Lastly, benevelonce (b_{ji}) from candidate trustee u_j to trustor u_i is defined as a mapping of l_{ji} to the range of [0,1]:

$$b_{ji} = \frac{l_{ji} - \text{Min}_{u'_j u'_i} l_{j'i'}}{\text{Max}_{u'_j u'_i} l_{j'i'} - \text{Min}_{u'_j u_i} l_{j'i}}$$

Integrity is measured using the global trustworthiness of the candidate trustee u_j approximated by number of other users who trust him/her. Integrity is computed as

$$\xi(|U^T_{*j}|, \alpha''\mu'')$$

where μ'' and α'' are set to 5 and 0.1, respectively.

The authors use various combinations of these core features to construct Ability-, Benevelonce-, Integrity- only models that consist of ability, benevelonce and integrity features, respectively. The SVM classifier is also employed to build a model from this set of features.

5.2 Epinions Taxonomy Model

The following interactions between users are defined and categorized in [16]:

Write-Rate (WR) Connection. Given two users u_i and u_j, if u_i writes a review r and u_j rates r, then a WR connection is formed between u_i and u_j .

Rate-Rate (RR) Connection. Given two users u_i and u_j, if after u_i rates a review r, u_j rates r as well, then an RR connection is formed between u_i and u_j .

Write-Write (WW) Connection. Given two users u_i and u_j, if after u_i writes a review r_i about a product p, u_j writes another review r_j about p as well, then a WW connection is formed between u_i and u_j .

Write-Comment (WC) Connection. Given two users u_i and u_j, if u_i writes a review r, and u_j comments on r, then an WC connection is formed between u_i and u_j.

Comment-Comment (CC) Connection. Given two users u_i and u_j, if after u_i comments on a review r, u_j comments on r or u_i's comment, then a CC connection is formed between u_i and u_j.

The user takes one more of the following roles: writer, rater and commenter. The authors classify a large number of factors into the three following groups: review, rating, and comment related. The factors in each group are further split into two sub-groups: distribution factors and count-based factors. Distribution factors are statistical metrics, such as *average* and *standard deviation*, while count-based factors are those which are related to counting a specific set of objects. Examples of count-based factors are *number of reviews posted* or *review frequency*.

Due to the overwhelmingly large number of features (the are 1397 features defined in [16]) we decided to implement only the top 7 features of [16] for our comparison. The features selected are major contributors to the perfomance of a classifier and computationally inexpensive. They are

1. The absolute total number of ratings that are given to the reviews written by the writer and rated by the rater.
2. WR feature: The absolute number of ratings with scores higher than 0.8 that are given to the reviews that are written by the writer and rated by the rater.
3. WR feature: The absolute number of first ratings that are given to the writer by the rater.
4. WR feature: The absolute total number of ratings that are given to the writer by the rater.
5. WR feature: The absolute total number of reviews that are written by the writer and rated by the rater
6. WR feature: The absolute number of reviews with product scores higher than 0.8 that are written by the writer and rated by the rater.
7. WR feature: The normalized total number of ratings that are given to the writer by the rater.

Chapter 6

Evaluation

6.1 Extended Epinions Dataset

One of the unique characteristics of Extended Epinions dataset is that the dataset contains ratings on articles or *reviews*(rather than on products) written by users on some products on Epinions. The ratings represent how much a particular user rates a certain article written by an other user. The dataset contains around 132,000 users, who issued 717,667 trust statements (85,000 users received at least one statement). There are 1,560,144 reviews which were given 13,668,319 ratings, in total. The top three most active raters have given ratings to 162,169, 55,791, 46,954 reviews, respectively. 292,793 trust statements out of all trust statements are not built on a direct rater-reviewer relationship i.e. a trustor did not rate a single review written by a trustee. 124,538 are mutual trust links. Moreover, 20,477 out of those mutual trust links are not built on the direct rater-reviewer relationship. This small group mainly consists of users who know each other beyond Epinions.com.

6.2 Experimental Design

We start by ranking our features presented in Chapter 3 with respect to their discriminatory power. Ranking allows us to learn which relationships between users are the most important for trust prediction. We also compare with two other approaches.

The first approach denoted by *ant8* consists of eight features derived from the Antecedent Framework given in [20]. The second one, *top7*, includes top seven features from [16]. In the first comparison with *ant8* all our features are used. This model is denoted by *rf22* (*rf* stands for Random Forests). In the second experiment only our top seven features are included (*rf7*).

The dataset extracted for the first two experiments includes 400,000 (u, v) user pairs with a trust link from u to v ("trusts"), and 1,600,000 randomly selected (u, v) user pairs without a trust statement from u to v ("lack of trusts"). The dataset meets the following criteria:

- There exists a rater-reviewer relationship between the trustor and trustee candidates in the dataset (i.e. the trustor gave a rating to one of the reviews produced by the trustee). This allows the Antecedent Framework model to score the candidate pairs in the data.
- The dataset contains only trusts and lack of trusts. This allows our approach to be directly compared against the other methods.

– The dataset preserves the original distributions for trust and lack of trust. This provides a stratified experimental dataset.

We applied the Random Forest classifier from Weka [10] with the number of trees equal to 30, and the maximum depth of each tree equal to 100 in order to build the models for each set of features. The *J48* algorithm is used to grow a single tree. The models were evaluated using a ten-fold cross-validation.[1] The top seven features of *top7* include features 1,2,4,5,6,8, and 9.

We also compare all three approaches using Random Forest and Support Vector Machines [6] trained on two smaller datasets. The first one contains 1000 trusts and 1000 lack of trusts. This is the only non-stratified dataset used in our experiments. The second dataset is of a moderate size and stratified. It consists of 10,000 trusts and 40,000 lack of trust statements. The models built with the SVM classifier are prefixed with *svm*, whereas *rf* stands for the models constructed with RFs.

6.3 Performance metrics

We use the standard set of performance metrics including *precision*, *recall*, *F-measure*, and *ROC Area*.

[1] The features of *ant8* were computed with $\mu = 5$ and $\alpha = 0.1$. Please see the corresponding section on Antecedent Framework for more information

Precision and recall are the most popular metrics for evaluating information retrieval and recommener systems. In 1968, Cleverdon proposed them as the key metrics for evaluating information-retrieval systems (see [7]). Precision and recall are computed from a *confusion matrix* shown in Figure 6.1.

	Predicted		
Actual		Positive #	Negative #
	Positive #	True Positives # (TP)	False Negatives # (FN)
	Negative #	False Positives # (FP)	True Negatives # (TN)

Fig. 6.1. Confusion Matrix

Precision is the number of correct results divided by the number of all returned results, i.e.

$$precision = \frac{TP}{TP + FP}.$$

Recall is the number of correct results divided by the number of results that should have been returned, i.e.

$$recall = \frac{TP}{TP + FN}.$$

F-measure considers both the precision and recall. It is computed as follows:

$$F\text{-}measure = \frac{2 \cdot precision \cdot recall}{precision + recall}$$

Besides these three, we also use the Area-Under-Curve metrics (ROC Area). The ROC curve-based metric is a theoretically grounded alternative to precision and recall [25, 11]. The ROC model attempts to measure the extent to which a classifier algorithm can successfully distinguish between relevance and noise. The ROC model assumes that the classifier will assign a predicted level of relevance to every potential item. The ROC curve represents a plot of recall versus fallout. A common algorithm for creating an ROC curve is as follows [12]:

- For each predicted item, in decreasing order of predicted relevance (starting the graph at the origin):
- If the predicted item is indeed relevant, draw the curve one step vertically.
- If the predicted item is not relevant, draw the curve one step horizontally to the right.
- If the predicted item has not been rated (i.e., relevance is not known), then the item is simply discarded and does not affect the curve negatively or positively.

An example of an ROC curve constructed in this manner is shown in Figure 6.2 [18, 21]. The area under the ROC curve (*ROC Area*) specifies the probability that a classifier assigns a higher value to the positive than to the negative example drawn at random.

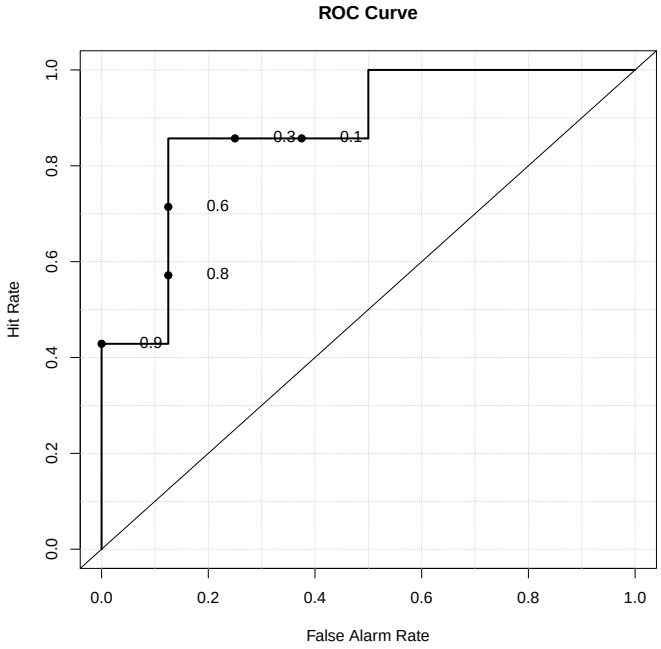

Fig. 6.2. Example of ROC Curve.

6.4 Ranking Features

For each feature we split the computed feature's values into trust and lack of trust sub-populations. The Kolmogorov-Smirnov test tries to determine if these sub-populations differ significantly. The test uses the maximum vertical deviation between the two curves of the Empirical Distribution Functions built from the datasets as the statistic D. For example, Figure 6.3 contrasts the Empirical Distribution Functions for both classes for $f_{5,a}$ and $f_{1,b}$, respectively. The reader might notice that for $f_{1,b}$ the lines overlap and get individually indiscernible, which corresponds to a very small value of D.

Figure 6.4 shows that the Rater-Reviewer features (e.g. $f_{5,a}$, $f_{7,b}$, $f_{5,b}$, $f_{6,e}$, $f_{6,d}$, $f_{6,b}$, $f_{6,c}$, $f_{5,c}$, $f_{6,a}$, $f_{7,c}$, $f_{7,a}$) have much stronger discriminatory power than the User Similarity ones (e.g. $f_{1,e}$, $f_{1,a}$, $f_{1,d}$, f_3, $f_{1,c}$, $f_{2,a}$, $f_{2,b}$, f_4, f_8, $f_{2,c}$, $f_{1,b}$). Eight out of ten (e.g. $f_{5,a}$, $f_{7,b}$, $f_{5,b}$, $f_{6,e}$, $f_{6,d}$, $f_{6,b}$, $f_{6,c}$, $f_{5,c}$) *top* features from Figure 6.4 belong to the Rater-Reviewer class, whereas only two come from the User Similarity group (e.g. $f_{1,e}$, $f_{1,a}$). The feature $f_{5,a}$, corresponding to the user leniency towards the reviewer, has the greatest discriminatory power. It is surprising that the user leniency affects the classifier's accuracy to a much greater extent than the reviewer's reputation. The users in the Epinions community seem to not be influenced by peer pressure!

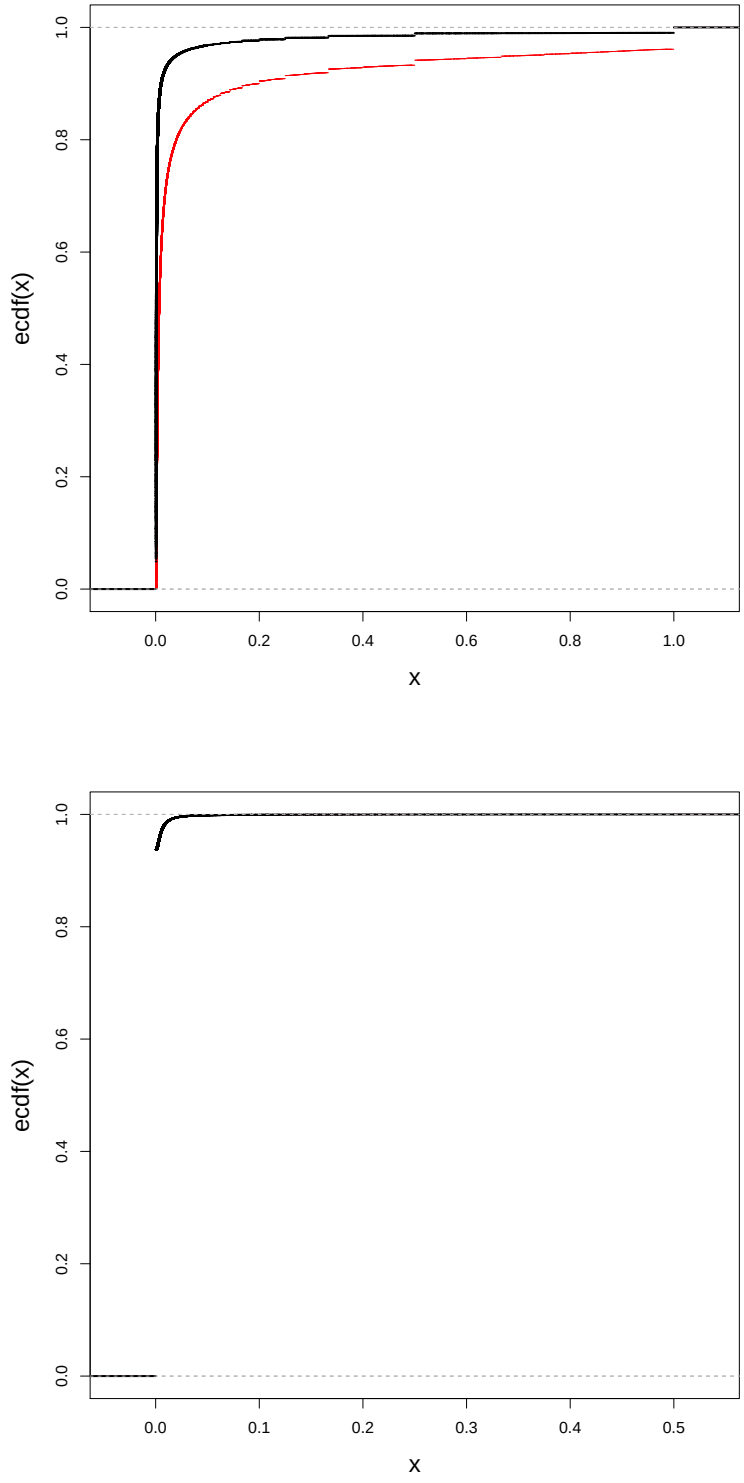

Fig. 6.3. The empirical cumulative distribution functions for trust and lack of trust statements for features $f_{5,a}$ and $f_{1,b}$, top and bottom, respectively.

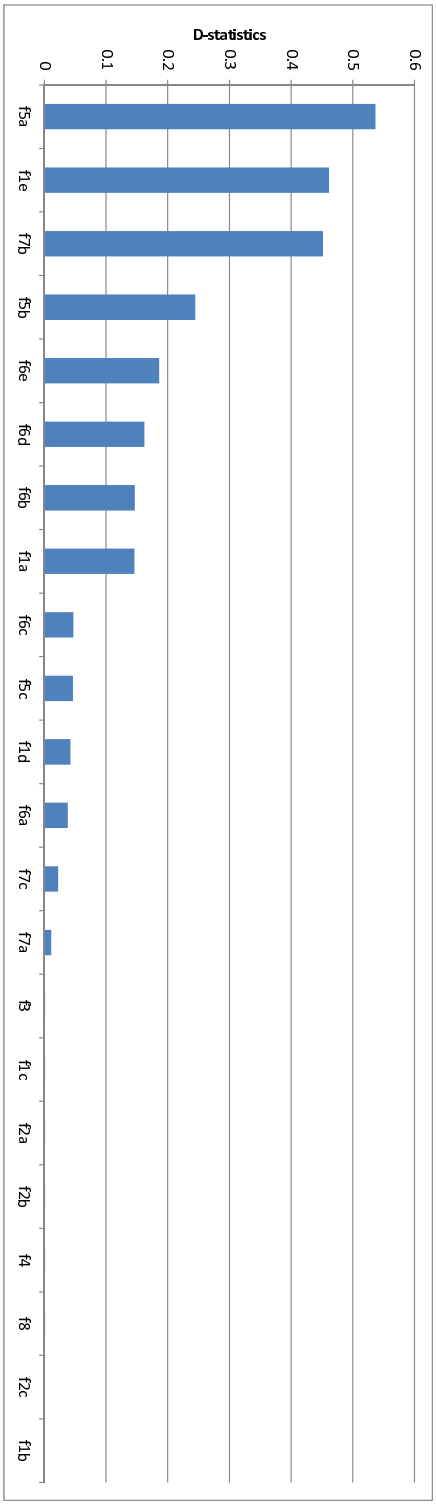

Fig. 6.4. Features in decreasing order of D-statistics

This also shows that using our complex features and applying Data Mining techniques to the features might yield significant gains. The second top feature indicates conflicts in the preferences between the rater and reviewer. The conflicts allow the classifier to discriminate much more accurately than the regular symmetric similarity features derived from the items that both users prefer. The features based on hidden ratings information also tell us a lot about the rater's attitude regarding the reviewer. Another interesting observation is that the features computed from the lower partisan ratings have greater discriminatory power than the features based on higher partisan ratings even though their fraction is significantly smaller than the higher ones'. Users seem to exercise extreme caution and give more thought to their decision of giving a lower rating.

6.5 Random Forest models

The results of using the Random Forest classifier on the two-million instance dataset are shown in Figures 6.5 - 6.11.

– In overall, all metrics show higher scores for *rf22* and *rf7* over the other two models (see Figures 6.5, 6.6).

- Our *rf22* yields the best accuracy of 88.8% followed by *rf7* giving 87.4%. Our best result allows the accuracy to be improved by 5% comparing to *ant8* (83.4%) and by nearly 9% comparing to *top7* (80.1%).
- *rf22* yields the best FP rate of 0.048 followed by *ant8* (0.056) (Figure 6.7).
- *rf22* shows significant improvements of 5% and 20% in precision, over *ant8* (0.64) and *top7* (0.57) , respectively (Figure 6.8).
- The recall metrics for *rf22* and *rf7* is 20% greater than *ant8*. *top7* yields a recall rate of 0.029 (Figure 6.9).
- F-measure reflects the precision and recall scores by showing a 4% improvement for our model (0.7) over *ant8* (0.66) (Figure 6.10).
- Lastly, ROC Area shows that all classifiers perform better than random prediction. *rf22* and *rf7* show a 10% improvement over *ant8* for this metrics (Figure 6.11.

6.6 Random Forest and Support Vector Machine comparison

Figures 6.12 - 6.18 compare the results for the models constructed by Random Forest and Support Vector Machines from the 2000-instance dataset.

- In overall, RFs outperforms SVM on this dataset showing higher scores for precision, recall, F-measure and ROC Area.

- Using SVM reduces the scores for both models. *ant*8 and *top*7 appear to be a bit more stable than our model when using different classifiers. The scores for the two are only slightly worse than the ones received for our model, when using the SVM classifier.
- Our model gives the best results in precision among all models for both classifiers: 0.8 and 0.73 (Figure 6.15).
- The recall scores for both classifiers are somewhat contradictory. Random Forest yields a better recall of 0.8 for our model, whereas SVM improves the recall for *svm_ant*8 up to 0.76 outperforming our model by 3% (Figure 6.16).
- SVM gives much tighter results for f-measure (0.737, 0.715, and 0.126) and ROC Area (0.737, 0.696, and 0.515) between all approaches, with our model performing slightly better than the other two (Figure 6.17).

Figures 6.19 - 6.25 contain the results given by the classifiers for the 50000-instance dataset.

- Again, RFs outperforms SVM. The number of correctly instances is reduced by 5% when using SVM. The scores including precision, recall, f-measure, etc. are higher for RFs than SVM. The SVM classifier trained on the features derived from the models considered appears to handle the skewness of the dataset much worse than RFs.

- Especially, one might notice that the recall scores for SVM are 32-40% lower than the ones RFs gives. There is also a 3% decline in the precision score for our model. F-measure reflects these declines accordingly.
- Using SVM reduces the scores for our model. *svm_ant*8 yields the best results in precision (0.722), recall (0.15) and F-measure (0.249) outperforming our model by 4% on average (see Figures 6.22 - 6.24).
- The ROC Area scores don't exceed 57% showing that the results are only slightly better than random prediction (Figure 6.25).

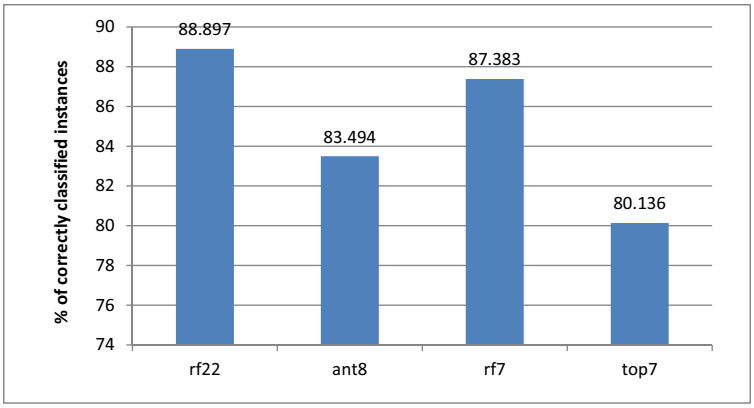

Fig. 6.5. % of correctly classified instances for the models constructed by RFs from the two-million dataset

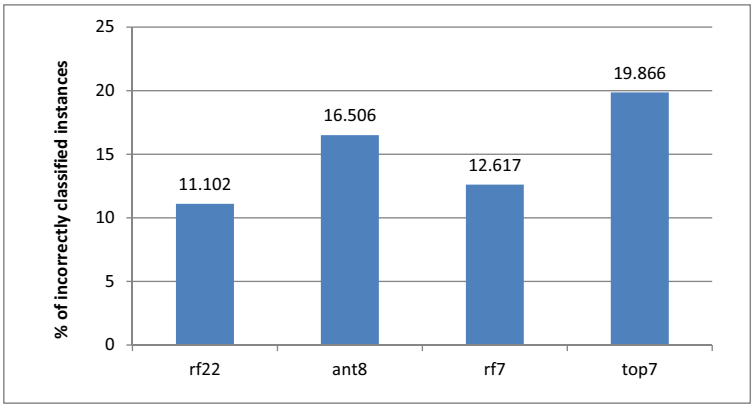

Fig. 6.6. % of incorrectly classified instances for the models constructed by RFs from the two-million dataset

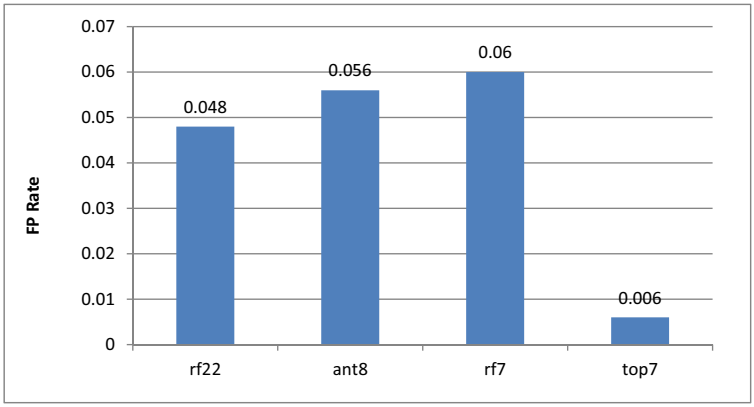

Fig. 6.7. FP rate for the models constructed by RFs from the two-million dataset

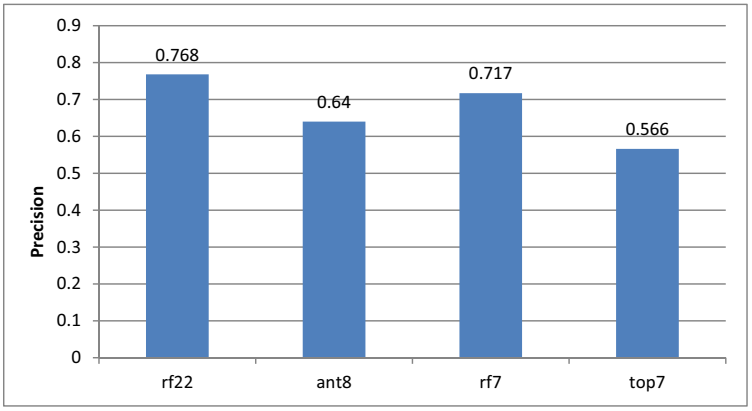

Fig. 6.8. Precision for the models constructed by RFs from the two-million dataset

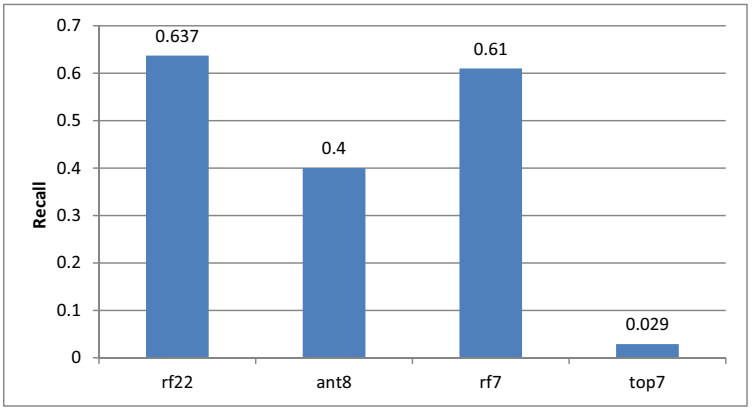

Fig. 6.9. Recall for the models constructed by RFs from the two-million dataset

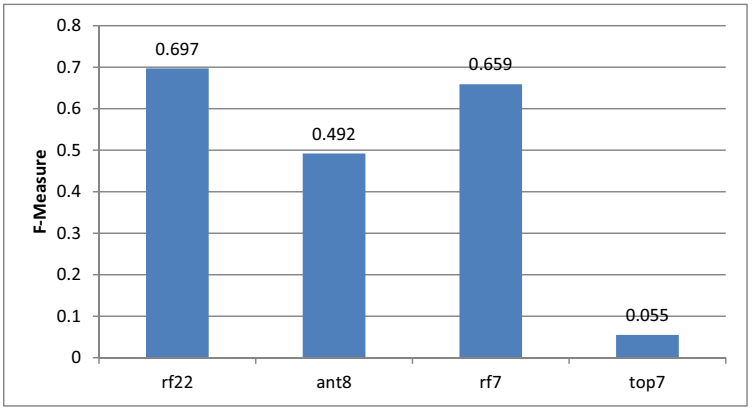

Fig. 6.10. F-Measure for the models constructed by RFs from the two-million dataset

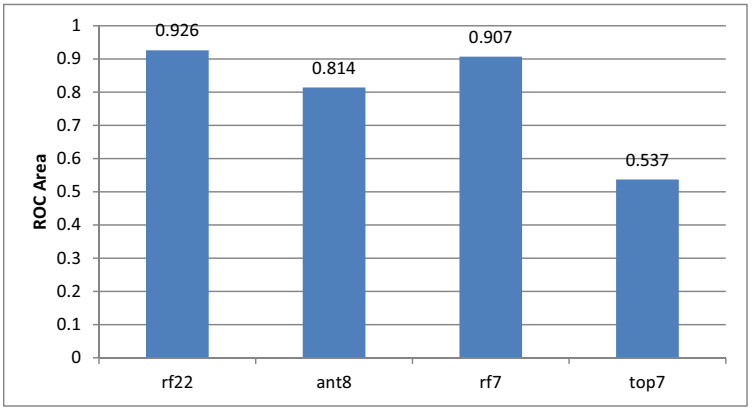

Fig. 6.11. ROC Area for the models constructed by RFs from the two-million dataset

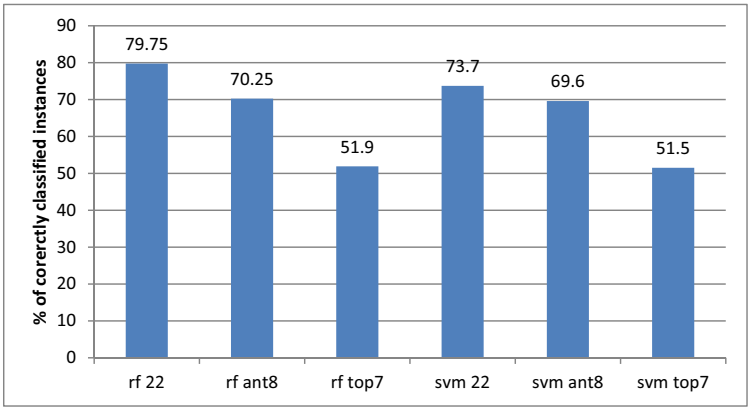

Fig. 6.12. % of correctly classified instances for the models constructed by RFs and SVM from the 2000-instance dataset

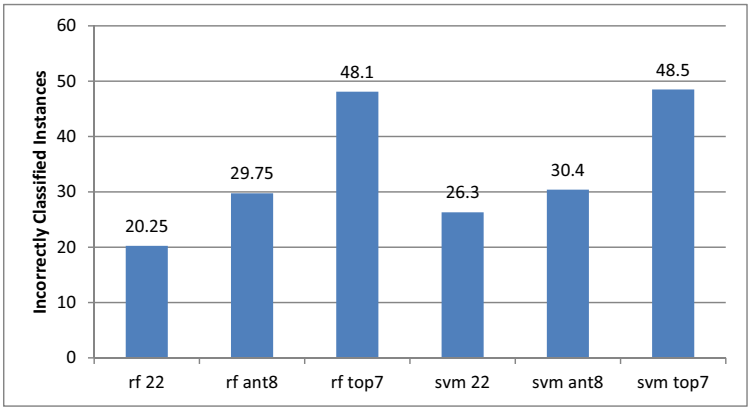

Fig. 6.13. % of incorrectly classified instances for the models constructed by RFs and SVM from the 2000-instance dataset

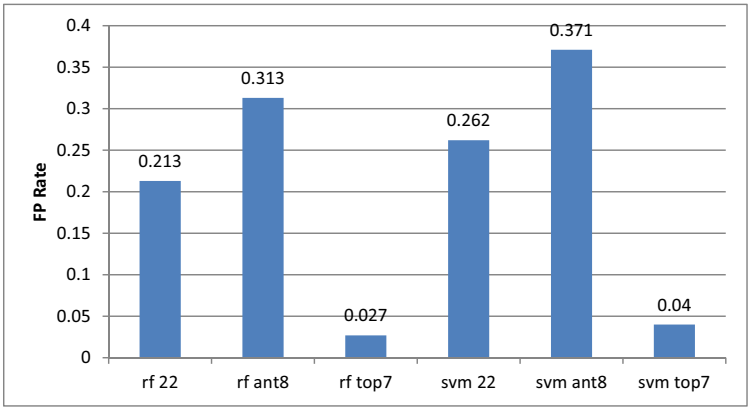

Fig. 6.14. FP rate for the models constructed by RFs and SVM from the 2000-instance dataset

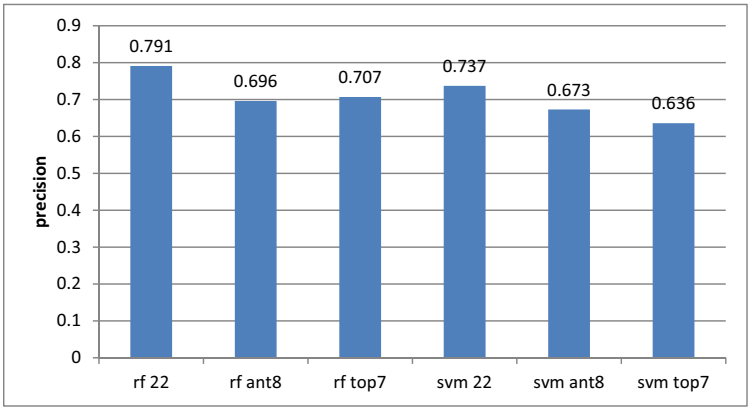

Fig. 6.15. Precision for the models constructed by RFs and SVM from the 2000-instance dataset

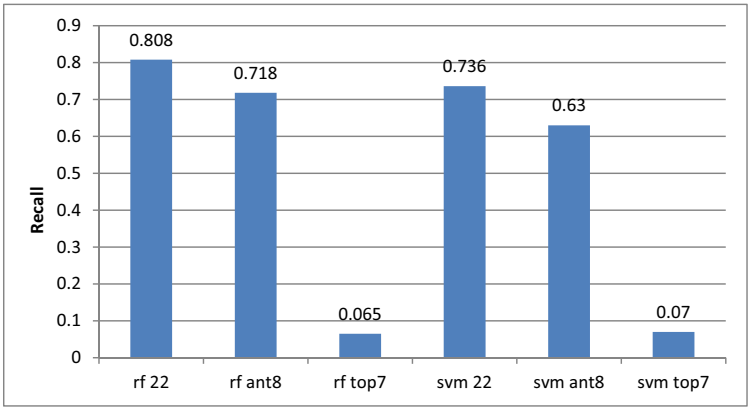

Fig. 6.16. Recall for the models constructed by RFs and SVM from the 2000-instance dataset

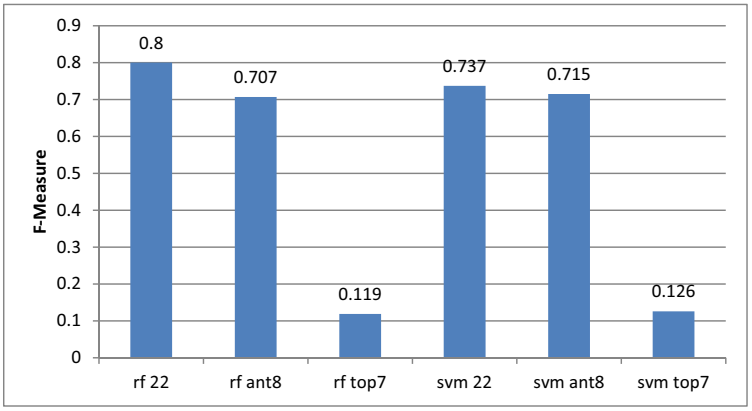

Fig. 6.17. F-Measure for the models constructed by RFs and SVM from the 2000-instance dataset

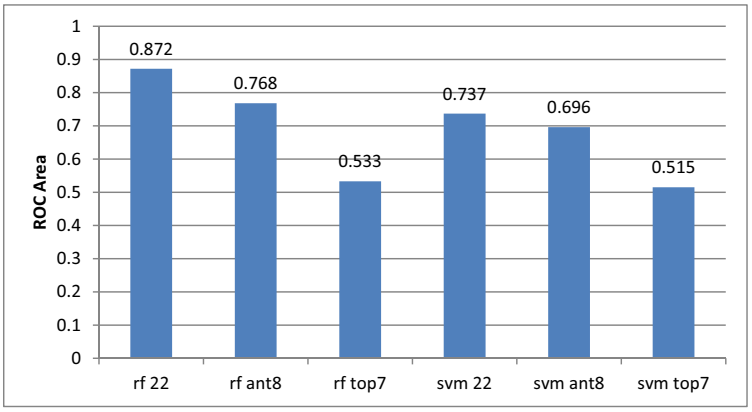

Fig. 6.18. ROC Area for the models constructed by RFs and SVM from the 2000-instance dataset

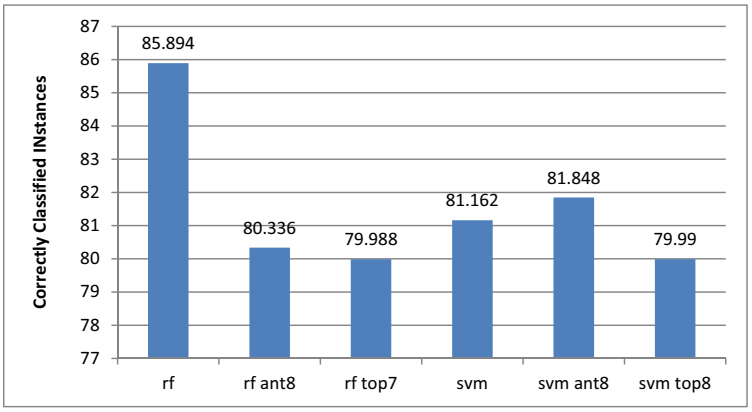

Fig. 6.19. % of correctly classified instances for the models constructed by RFs and SVM from the 50000-instance dataset

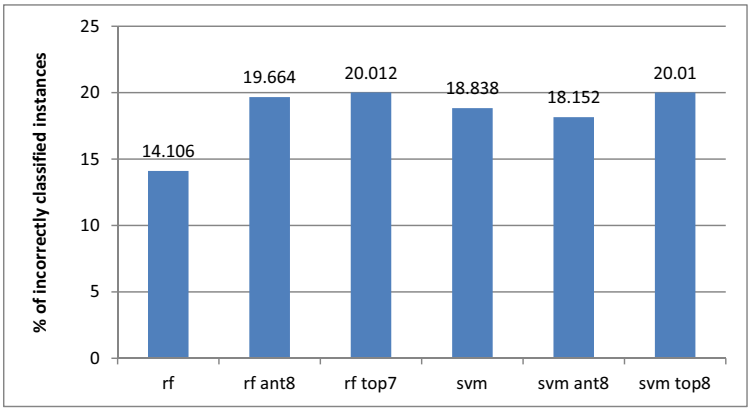

Fig. 6.20. % of incorrectly classified instances for the models constructed by RFs and SVM from the 50000-instance dataset

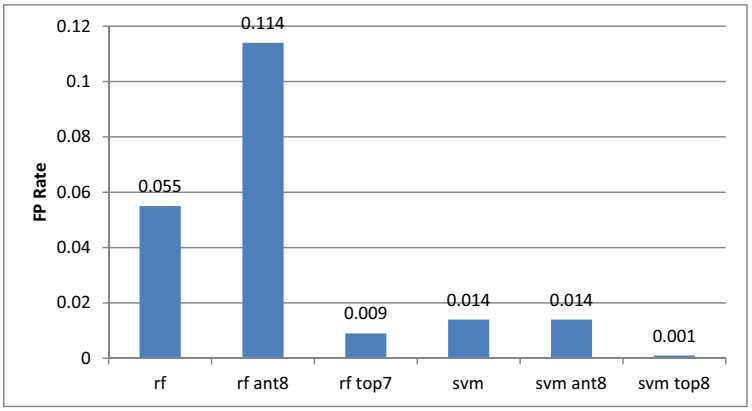

Fig. 6.21. FP rate for the models constructed by RFs and SVM from the 50000-instance dataset

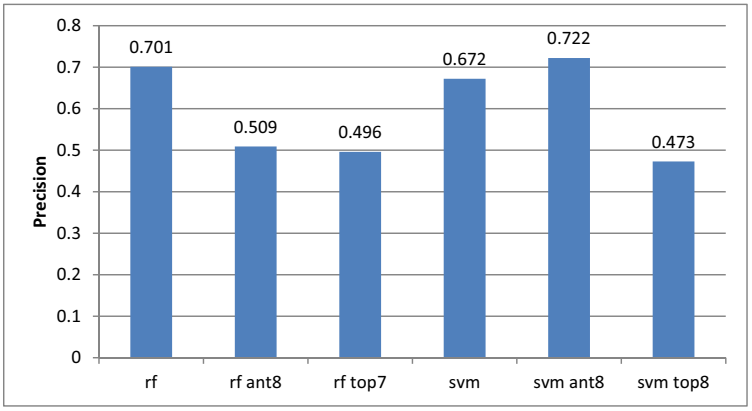

Fig. 6.22. Precision for the models constructed by RFs and SVM from the 50000-instance dataset

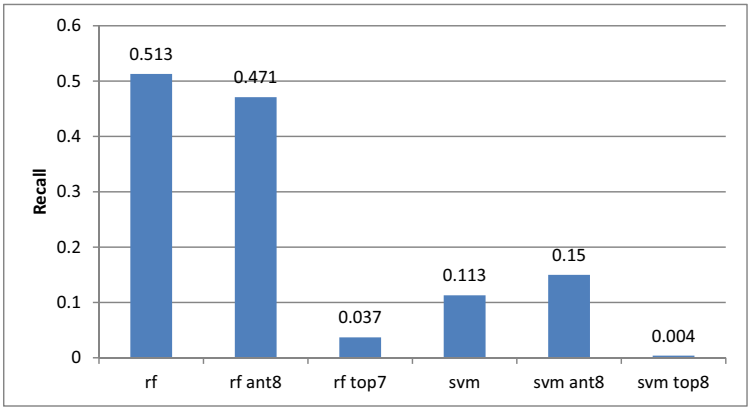

Fig. 6.23. Recall for the models constructed by RFs and SVM from the 50000-instance dataset

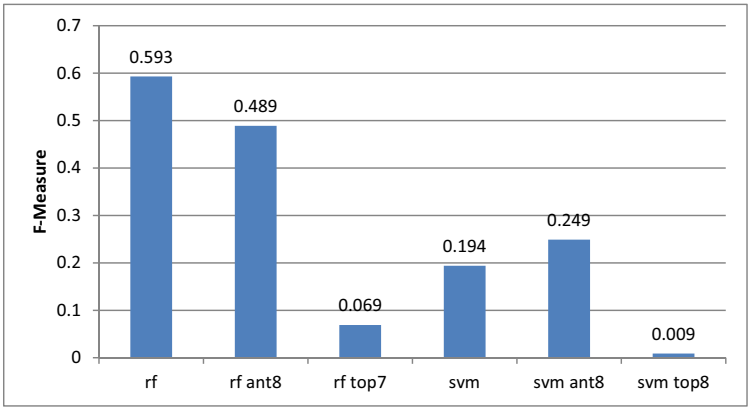

Fig. 6.24. F-Measure for the models constructed by RFs and SVM from the 50000-instance dataset

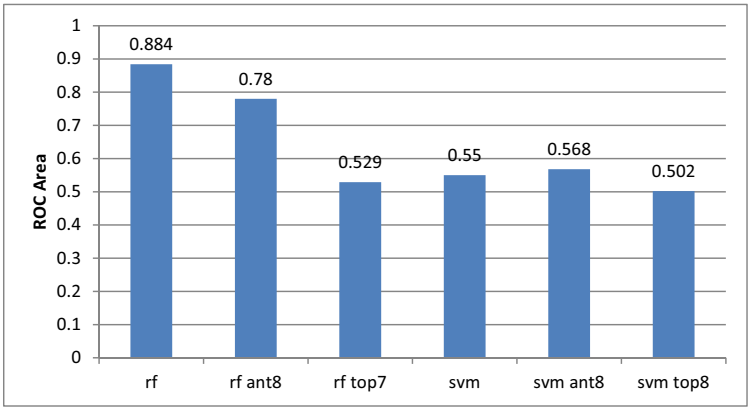

Fig. 6.25. ROC Area for the models constructed by RFs and SVM from the 50000-instance dataset

Chapter 7

Personalized Trust Prediction Model

We suggest a Personalized Trust Prediction Model that assigns a trust score to each pair (u, v). The model infers trust for some user pair (u, v) based on the trust relationships that the trustees of u indicate towards v. This approach has been widely used before. However, our model augments the approach by weighing the opinion of each trustee y of the u's trust network (i.e. y where $y \in T_u$) in order to reflect both Rater-Reviewer and Rater-Rater features between u and y. If u appreciates the reviews written by y, the u's decision on initiating a trust relationship with v might be *influenced* by the y's opinion about v. The more u appreciates y the more the y's opinion influences the u's decision. Various Rater-Reviewer and Rater-Rater features can be used.

We consider the average rating that u gives to the reviews by y in order to express the Rater-Reviewer relationships and feature $f_{1,e}$ for reflecting the Rater-Rater similarity between the users. One can include in more features and/or assign the weights to features for improving the results even further. Three variations (scores) of the Personalized Trust Prediction Model are exercised. $score_1$ weighs the opinion of y by

the average rating that u gives to the reviews by y.

Lastly, if the score is positive, the instance is assigned the trust class. It is assigned the lack of trust label, otherwise. We denote the trust relationship between y and v by t_{yv}. t_{yv} is equal to 1 if $v \in T_y$ or -1, otherwise. $f_{1,e,y,v}$ is the feature $f_{1,e}$ computed for the pair (y, v). We have :

$$
\begin{aligned}
score_1 &= \sum_{y \in T_u} t_{yv} \cdot \bar{r}_{uy}, \\
score_2 &= \sum_{y \in T_u} t_{yv} \cdot f_{1,b,u,y}, \\
score_3 &= \sum_{y \in T_u} t_{yv} \cdot (f_{1,b,u,y} + \bar{r}_{uy}).
\end{aligned}
$$

The results for the PTP model are given in Table 7.1.

- Using only the opinions of the trustees reduces the performance significantly. $score_1$ gives the highest precision of 0.571, whereas $score_3$ yields the best recall of 0.347. The scores are rather low comparing to the results from Figures 6.5 - 6.25.
- The numbers for $score_1$ and $score_2$ again confirm that the Rater-Reviewer interactions have the much greater discriminatory power than the Rater-Rater interactions.

– The ROC Area for all three models is over 50% , thus the model performs better than random prediction.

	Experiments		
	$score_1$	$score_2$	$score_3$
Precision	0.5707	0.4383	0.5660
Recall	0.3439	0.1573	0.3460
F-measure	0.4292	0.2315	0.4295
ROC Area	0.6084	0.5867	0.6090

Table 7.1. Performance scores for the PTP model

Chapter 8

Conclusions

Our experiments show that the Rater-Reviewer relationships drive the trust networks in the Epinions community. One can easily estimate the discriminatory power of various features using Figure 6.4. We suggest a set of complex features based on reviews, reviewers, topics and trustees for incorporating the user similarity information. We show that these complex features have a very good discriminatory power. Using our complex features is a powerful way to improve the performance of classification algorithms for the trust prediction problem.

The series of experiments is conducted to compare our approach with two others using Random Forest and Support Vector Machines for the datasets of various sizes and properties. We also propose the Personalized Trust Prediction Model that allows one to make personalized trust predictions for the user, based on the preferences of the user's trustees. However, our experiments show that it does not perform as well as the models constructed by Random Forest from the larger datasets. The most important lesson learned is that algorithms that incorporate domain knowledge perform much better than generic approaches. However, those algorithms are not easily

generalizable for different domains. Figure 8.1 shows interactions contained in Extended Epinions Dataset. From this figure one easily infers that user similarity can be also computed in terms of products and reviewers. Similarly, reviewer similarity can be computed in terms of products they write reviews on and users that rate these reviews. Even though, these might seem to be strongly correlated, Figure 6.4 shows that the discriminatory power differs by orders of magnitude. Thus, these complement typical user similarity defined in terms of reviews, which is typically the only user similarity measure most generic approaches employ.

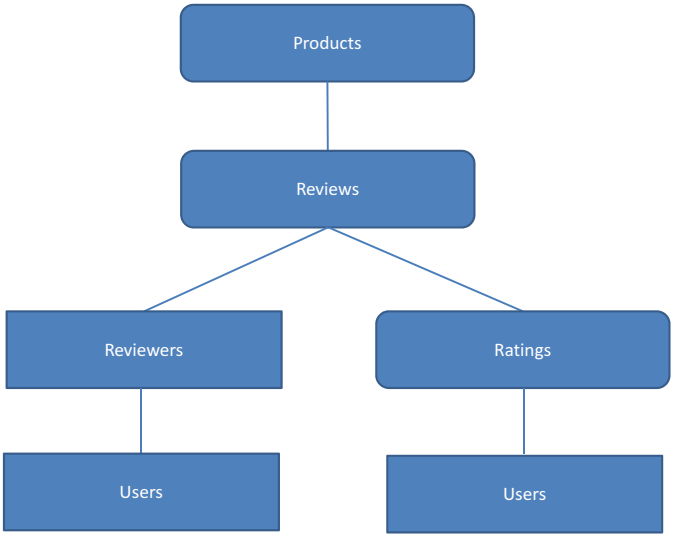

Fig. 8.1. Interactions derived from Extended Opinions Dataset

8.1 Future Work

It has been shown that trust networks consist of a lot of small clusters [8]. It might be interesting to investigate whether these clusters do grow around topics or reviewers and whether Rater-Reviewer or User Similarity information alone form networks similar to trust ones. Augmenting the existing trust propagation algorithms to use weights derived from Rater-Reviewer or User-Similarity information might result in performance gains and also seems to be an interesting area to research. For instance, *Sunny* can be augmented to use the user reputation adjusted by the leniency of a particular user rather than confidence. One might notice that the current graph trust propagation models use only one type of information. For the closest neighbors, one might want to use the approach suggested in Trust Personalized Prediction Model and then switch back to regular connections for further trust propagation. This would be a simple attempt to address obvious deficiencies of the homogenous model and reconcile both types of information. Alternatively, the graph model can be extended to include different types of nodes or arcs (i.e. see [16] for possible relationships between users), which would resolve many fundamental limitations.

Lastly, the following features based on the trustees' opinions can be computed and used for trust prediction:

– the number of the trustees of u (the users u trusts) who trust the reviewer v

- the number of implicit ratings by the trustees of u given to the reviews written by v
- the number of implicit ratings by the trustees of u (who trust v as well) given to the reviews written by v
- the average rating by the trustees of u given to the reviews written by v
- the average rating by the trustees of u (who trust v as well) given to the reviews written by v

Bibliography

[1] Paolo Avesani, Paolo Massa, and Roberto Tiella. A trust-enhanced recommender system application: Moleskiing. In *In SAC 05: Proceedings of the 2005 ACM symposium on Applied computing*, pages 1589–1593. ACM Press, 2004.

[2] Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '07, pages 95–104, New York, NY, USA, 2007. ACM.

[3] Robert M. Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *IEEE International Conference on Data Mining (ICDM)*, 2007.

[4] James Bennett, Stan Lanning, and Netflix Netflix. The netflix prize. In *KDD Cup and Workshop in conjunction with KDD*, 2007.

[5] Maria Chowdhury, Alex Thomo, and Bill Wadge. Trust-based infinitesimals for enhanced collaborative filtering.

[6] Chih chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines, 2001.

[7] C. W. Cleverdon, J. Mills, and M. Keen. Factors determining the performance of indexing systems. 1966.

[8] Jennifer Ann Golbeck. Computing and applying trust in web-based social networks, 2005.

[9] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35:61–70, December 1992.

[10] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, 2009.

[11] J. A. Hanley and B. J. Mcneil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1):29–36, April 1982.

[12] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, John, and T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22:5–53, 2004.

[13] Yehuda Koren. Collaborative filtering with temporal dynamics. *Commun. ACM*, 53:89–97, April 2010.

[14] Yehuda Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data*, 4:1:1–1:24, January 2010.

[15] Ugur Kuter and Jennifer Golbeck. Sunny: A new algorithm for trust inference in social networks using probabilistic confidence models. In *AAAI'07*, pages 1377–1382, 2007.

[16] Haifeng Liu, Ee-Peng Lim, Hady W. Lauw, Minh-Tam Le, Aixin Sun, Jaideep Srivastava, and Young Ae Kim. Predicting trusts among users of online communities: an epinions case study. In *Proceedings of the 9th ACM conference on Electronic commerce*, EC '08, pages 310–319, New York, NY, USA, 2008. ACM.

[17] Nan Ma, Ee peng Lim, and Haifeng Liu. Trust relationship prediction using online product review data.

[18] S. J. Mason and N. E. Graham. Areas beneath the relative operating characteristics (ROC) and relative operating levels (ROL) curves: Statistical significance and interpretation. *Quarterly Journal of the Royal Meteorological Society*, 128(584):2145–2166, July 2002.

[19] Paolo Massa and Paolo Avesani. Trust-aware collaborative filtering for recommender systems. In *In Proc. of Federated Int. Conference On The Move to Meaningful Internet: CoopIS, DOA, ODBASE*, pages 492–508, 2004.

[20] Viet-An Nguyen, Ee-Peng Lim, Jing Jiang, and Aixin Sun. To trust or not to trust? predicting online trusts using trust antecedent framework. In *Data Mining, 2009. ICDM '09. Ninth IEEE International Conference on*, pages 896 –901, dec. 2009.

[21] NCAR Research Application Program. *verification: Forecast verification utilities.*, 2010. R package version 1.31.

[22] J. R. Quinlan. Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research*, 4:77–90, 1996.

[23] Joseph Sill, Gabor Takacs, Lester Mackey, and David Lin. Feature-Weighted Linear Stacking. November 2009.

[24] Leo Breiman Statistics and Leo Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.

[25] John A. Swets. Effectiveness of information retrieval methods. *American Documentation*, 20(1):72–89, 1969.