Finding Unusual Review Patterns Using Unexpected Rules

Nitin Jindal
Department of Computer Science
University of Illinois at Chicago
851 S. Morgan, Chicago, IL 60607
nitin.jindal@gmail.com

Bing Liu
Department of Computer Science
University of Illinois at Chicago
851 S. Morgan, Chicago, IL 60607
liub@cs.uic.edu

Ee-Peng Lim
School of Information Systems
Singapore Management University
eplim@smu.edu.sg

ABSTRACT
In recent years, opinion mining attracted a great deal of research attention. However, limited work has been done on detecting opinion spam (or fake reviews). The problem is analogous to spam in Web search [1, 9 11]. However, review spam is harder to detect because it is very hard, if not impossible, to recognize fake reviews by manually reading them [2]. This paper deals with a restricted problem, i.e., identifying *unusual review patterns* which can represent suspicious behaviors of reviewers. We formulate the problem as finding unexpected rules. The technique is domain independent. Using the technique, we analyzed an Amazon.com review dataset and found many unexpected rules and rule groups which indicate spam activities.

Categories and Subject Descriptors
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval – *Information filtering*.

General Terms
Algorithms, Experimentation

Keywords
Reviewer behavior, review spam, unexpected patterns

1. INTRODUCTION
With the rapid growth of online reviews, it has become a common practice for people to read reviews for many purposes. This gives good incentives for *review spam*, which refers to writing *fake reviews* to mislead readers or automated systems by giving bogus positive or negative opinions to some target objects to promote them or to damage their reputations. Detecting spam reviews is a critical problem for opinion mining [6] and retrieval [8].

The problem can be seen as a classification problem with two classes, spam and not-spam. However, to obtain training data by manually labeling reviews is very hard as a spammer can easily craft a fake review that is just like any innocent review [2]. In [2], a learning method using duplicate reviews as positive training data is used, but many not duplicated reviews can be spam too. Some researchers also study the helpfulness of reviews [7, 12], but review spam is a different concept. In [4], a user study shows that rating behaviors are good indicators of spam.

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.

In this paper, we study a restricted problem, identifying review patterns representing unusual behaviors of reviewers, which can indicate spam activities [4]. For example, if a reviewer wrote all negative reviews on products of a brand but other reviewers are generally positive about the brand, this reviewer is clearly a spam suspect. To find unusual behaviors, the conventional approach is to write an application specific heuristic program to find such behaviors. However, this is not desirable. It is much better to propose a general framework for solving this class of problems so that the resulting system can also be applied to other domains. This paper proposes such an approach and shows that the problem can be formulated as finding unexpected rules/patterns from data.

The data that we use consists of a set of data records, which are described by a set of normal attributes $A = \{A_1, \ldots, A_n\}$, and a class attribute $C = \{c_1, \ldots, c_m\}$ of m discrete values, called *classes*. The rules are of the form: $X \rightarrow c_i$, where X is a set of conditions from the attributes in A and c_i is a class in C. Such a rule gives the conditional probability of $\Pr(c_i \mid X)$ (called the *confidence*) and the joint probability $\Pr(X, c_i)$ (called the *support*) [5].

For our application, the data can be produced as follows: Each review forms a data record with a set of attributes, e.g., *reviewer-id*, *brand-id*, *product-id*, and a *class*. The class represents the opinion of the reviewer, *positive*, *negative* or *neutral* based on the review rating. In most review sites (e.g., amazon.com), each review has a rating between 1 (lowest) to 5 (highest) assigned by its reviewer. We can assign the rating of 4 and 5 as positive, 3 as neutral, and 1 and 2 as negative. A rule could be that a reviewer gives all positive ratings to a particular brand of products.

The issue is how we know a rule represents an abnormal behavior of a reviewer. To decide that, we need to know what is expected. This paper first defines several types of expectations based on the natural distribution of the data. It then proposes the corresponding unexpectedness measures to rank rules. This method is domain independent as it only depends on the data and the type of rules but not the application. It can thus be applied to other domains. In our experimental study, we report a case study of discovering suspicious behaviors of reviewers based on Amazon reviews, which indicate spam activities or at least biased reviewers. For a case study on a different domain dataset, see [3].

2. UNEXPECTEDNESS DEFINITIONS
Unexpectedness is defined as deviation from expectations. Thus, the definition of expectations is the key. Before the definitions, we give some more notations about the data and the resulting rules.

Given a data set D, let the domain or the set of possible values of attribute A_j be $dom(A_j)$. We use the data to mine *class association rules* (CAR) [5] of the form, $X \rightarrow c_i$. A condition in X is an attribute value pair: $A_j=v_{jk}$ ($v_{jk} \in dom(A_j)$).

CAR mining finds all rules that satisfy the user-given minimum support and minimum confidence constraints. However, we cannot use them because they create holes in the rule space, and remove the context. Here holes mean those rules that do not meet the support and confidence constraints. However, without minimum support and minimum confidence to ensure the feasibility of computation, CAR mining can cause combinatorial explosion [5]. Fortunately, practical applications have shown that users are interested in almost only short rules as it is hard to perform any action on long rules due to low data coverage. Thus, in our system, we focus on mining rules with only 1-3 conditions if their support and confidence is greater than zero.

Approach to defining expectations: Our approach begins by assuming the knowledge of class prior probabilities ($\Pr(c_i)$), which can be easily found from the data automatically. They give us the natural distribution of the data to begin with. Two additional principles govern the definition of expectations:

1. Given no prior knowledge, we expect that the data attributes and classes have no relationships, i.e., they are statistically independent. This is justified as it allows us to find those patterns that show strong relationships.
2. We use shorter rules to compute the expectations of longer rules. This is also logical due to two reasons. First, it enables the user to see interesting short rules first. Second, more importantly, unexpected shorter rules may be the cause of some longer rules being abnormal (see Section 2.2), but not the other way around. Thus knowing such short rules, the longer rules are no longer unexpected.

Based on these two principles, we begin with the discussion of unexpectedness of one-condition rules, and then two-condition rules. For multi-condition rules, see [3].

2.1 Unexpectedness of One-Condition Rules

We define four types of unexpectedness. A one-condition rule is a rule with only one condition (an attribute value pair, $A_j = v_{jk}$).

2.1.1 Confidence Unexpectedness

We want to determine how unexpected the confidence of a rule is. To simplify the notation, we use a single value v_{jk} ($v_{jk} \in dom(A_j)$) to denote the k^{th} value of attribute A_j. A one-condition rule is thus of the following form: $v_{jk} \rightarrow c_i$. The expected confidence of the rule is defined below.

Expectation: Since we consider one-condition rules, we use the information from zero-condition rules to define expectations:

$$\rightarrow c_i,$$

which is the class prior probability of c_i, i.e., $\Pr(c_i)$. Given $\Pr(c_i)$ and no other knowledge, it is reasonable to expect that attribute values and the classes are independent. Thus, the confidence ($\Pr(c_i \mid v_{jk})$) of the above rule ($v_{jk} \rightarrow c_i$) is expected to be $\Pr(c_i)$. We use $E(\Pr(c_i \mid v_{jk}))$ to denote the expected confidence, i.e.,

$$E(\Pr(c_i \mid v_{jk})) = \Pr(c_i). \quad (1)$$

Confidence Unexpectedness (Cu): Confidence unexpectedness of the rule is defined as the ratio of the deviation of the actual confidence to the expected confidence given a support threshold θ. Let the actual confidence of the rule be $\Pr(c_i \mid v_{jk})$. We use $Cu(v_{jk} \rightarrow c_i)$ to denote the unexpectedness of the rule $v_{jk} \rightarrow c_i$.

$$Cu(v_{jk} \rightarrow c_i) = \frac{\Pr(c_i \mid v_{jk}) - E(\Pr(c_i \mid v_{jk}))}{E(\Pr(c_i \mid v_{jk}))} \quad (2)$$

Unexpectedness values can be used to rank rules. One may ask if this is the same as ranking rules based on their confidences. It is not, because of the expectation. First of all, for different classes, the expected confidences are different. When we discuss two-condition rules in Section 2.2, we will see that even in the same class, a high confidence rule may be completely expected.

Significance test: It is important to know whether the actual confidence is significantly different from the expectation, we use the statistical test for proportions.

2.1.2 Support Unexpectedness

The confidence measure does not consider the proportion of data records involved, for which we need support unexpectedness.

Expectation: Given no knowledge, we expect that an attribute value and a class are independent. Thus, we have $\Pr(v_{jk}, c_i) = \Pr(v_{jk})\Pr(c_i)$. $\Pr(c_i)$ is known, but not $\Pr(v_{jk})$. It is reasonable to expect that it is the average probability of all values of A_j. Thus we have ($\Pr(v_{jk})$ is unknown to the user, but is computed),

$$E(\Pr(v_{jk}, c_i)) = \Pr(c_i) \frac{\sum_{a=1}^{|A_j|} \Pr(v_{ja})}{|A_j|} \quad (3)$$

Support Unexpectedness (Su): Support unexpectedness of a rule is defined as follows, given a confidence threshold λ (λ is to ensure that the rule has sufficient predictability):

$$Su(v_{jk} \rightarrow c_i) = \frac{\Pr(v_{jk}, c_i) - E(\Pr(v_{jk}, c_i))}{E(\Pr(v_{jk}, c_i))} \quad (4)$$

This definition of *Su* (Equations (3) and (4)) is reasonable as it ranks those rules with high supports high, which is what we want.

Significance Test: The test for proportions can also be used here.

2.1.3 Attribute Distribution Unexpectedness

Confidence or support unexpectedness considers only a single rule. In many cases, a group of rules together shows an interesting scenario. Here we define an unexpectedness metric based on all values of an attribute and a class, which thus represent multiple rules. This unexpectedness shows how skewed the data records are for the class, i.e., whether the data records of the class concentrate on only a few values of the attribute or they spread evenly to all values, which is expected given no prior knowledge. For example, we may find that most positive reviews for a brand of products are from only one reviewer although there are a large number of reviewers who have reviewed products of the brand. This reviewer is clearly a spam suspect. We use supports (or joint probabilities) to define attribute distribution unexpectedness. Let the attribute be A_j and the class of interest be c_i. The attribute distribution of A_j with respect to class c_i is denoted by:

$$A_j \rightarrow c_i$$

It represents all the rules, $v_{jk} \rightarrow c_i$, $k = 1, 2, \ldots, |A_j|$, where $|A_j|$ is the total number of values in $dom(A_j)$.

Expectation: We can use the expected value of $\Pr(v_{jk}, c_i)$ computed above (Equation (3)) for our purpose here.

Attribute Distribution Unexpectedness (ADu): It is defined as the sum of normalized support deviations of all values of A_j.

$$ADu(A_j \rightarrow c_i) = \sum_{v_{jk}: v_{jk} \in dom(A_j) \wedge Dev > 0} \frac{Dev(v_{jk})}{\Pr(c_i)} \quad (5)$$

where $$Dev(v_{jk}) = \Pr(v_{jk}, c_i) - E(\Pr(v_{jk}, c_i)) \quad (6)$$

We use $\Pr(c_i)$ in Equation (5) because $\sum_{k=1}^{|A_j|} \Pr(v_{jk}, c_i) = \Pr(c_i)$. Note that in this definition negative deviations are not utilized because positive and negative deviations ($Dev(v_{jk})$) are symmetric or equal as $\Pr(c_i)$ is constant and $\sum_{k=1}^{|A_j|} \Pr(v_{jk}, c_i) = \Pr(c_i)$. Thus, considering one side is sufficient.

2.1.4 Attribute Unexpectedness

In this case, we want to discover how the values of an attribute can predict the classes. This is denoted by

$$A_j \rightarrow C,$$

where A_j represents all its values and C indicates all classes. Given no knowledge, our expectation is that A_j and C are independent. In the ideal case (or the most unexpected case), every rule $v_{jk} \rightarrow c_i$ has 100% confidence. Then, the values of A_j can predict the classes in C completely. For example, we may find that a reviewer wrote only positive reviews to one brand, and only negative reviews to another brand, which is clearly suspicious.

Conceptually, the idea is the same as measuring the discriminative power of each attribute in classification learning. The *information gain* measure can be used for the purpose. The expected information is computed based on entropy. Given no knowledge, the entropy of the original data D is (note that $\Pr(c_i)$ is the confidence of the zero-condition rule on class c_i):

$$entropy(D) = -\sum_{i=1}^{m} \Pr(c_i) \log \Pr(c_i) \quad (7)$$

Expectation: The expectation is the entropy of the data D:

$E(A_j \rightarrow C) = entropy(D)$

Attribute Unexpectedness (Au): Attribute unexpectedness is defined as the information gained by adding the attribute A_j. After adding A_j, we obtain the following entropy:

$$entropy_{A_j}(D) = -\sum_{k=1}^{|A_j|} \frac{|D_k|}{|D|} entropy(D_k) \quad (8)$$

Based on the values of A_j, the data set D is partitioned into $|A_j|$ subsets, D_1, D_2, …, $D_{|A_j|}$ (i.e., each subset has a particular value of A_j). The unexpectedness is thus computed with (which is the information gain measure in [10]):

$$Au(A_j \rightarrow C) = entropy(D) - entropy_{A_j}(D) \quad (9)$$

2.2 Unexpectedness of Two-Condition Rules

We now consider two-condition rules. Although we can still assume that the expected confidence of a rule is the class prior probability of its class as for one-condition rules, it is no longer appropriate as a two-condition rule is made up of two one-condition rules, which we already know. As mentioned earlier, it is possible that the unexpectedness of a two-condition rule is caused by a one-condition rule. It is thus not suitable to attribute unexpectedness to the two-condition rule.

For example, let us consider confidence unexpectedness. We have a data set with two classes and each class has 50% of the data, i.e., the class prior probabilities are equal, $\Pr(c_1) = \Pr(c_2) = 0.5$. For a rule $v_1 \rightarrow c_1$ with 100% confidence ($\Pr(c_1 \mid v_1) = 1$), it is highly unexpected based on Equation (2). Now let us look at a two-condition rule, v_1, $v_2 \rightarrow c_1$, which clearly also has 100% confidence ($\Pr(c_1 \mid v_1, v_2) = 1$). If we assume no knowledge, its expected confidence should be 50%. Then, we say that this rule is highly unexpected. However, if we know $v_1 \rightarrow c_1$, 100% confidence for rule v_1, $v_2 \rightarrow c_1$ is completely expected. The 100% confidence of rule $v_1 \rightarrow c_1$ is the cause for rule, v_1, $v_2 \rightarrow c_1$, to have the 100% confidence. More importantly, this example shows that ranking rules according to confidence unexpectedness is not equivalent to ranking rules purely based to their confidences.

With the knowledge of one-condition rules, we define different types of unexpectedness of two-condition rules of the form:

v_{jk}, $v_{gh} \rightarrow c_i$.

2.2.1 Confidence Unexpectedness

We first compute the expected confidence of the two-condition rule based on two one-condition rules:

$v_{jk} \rightarrow c_i$ and $v_{gh} \rightarrow c_i$

Expectation: Given the confidences of the two rules, $\Pr(c_i \mid v_{jk})$ and $\Pr(c_i \mid v_{gh})$, we compute the expected probability of $\Pr(c_i \mid v_{jk}, v_{gh})$ using the Bayes' rule and obtain:

$$\Pr(c_i \mid v_{jk}, v_{gh}) = \frac{\Pr(v_{jk}, v_{gh} \mid c_i) \Pr(c_i)}{\sum_{r=1}^{m} \Pr(v_{jk}, v_{gh} \mid c_r) \Pr(c_r)} \quad (10)$$

The first term of the numerator can be further written as

$$\Pr(v_{jk}, v_{gh} \mid c_i) = \Pr(v_{jk} \mid v_{gh}, c_i) \Pr(v_{gh} \mid c_i) \quad (11)$$

Conditional independence assumption: With no prior knowledge, it is reasonable to expect that all attributes are conditionally independent given class c_i. Formally, we expect that

$$\Pr(v_k \mid v_{gh}, c_i) = \Pr(v_{jk} \mid c_i) \quad (12)$$

Based on Equation (10), the expected value of $\Pr(c_i \mid v_{jk}, v_{gh})$ is:

$$E(\Pr(c_i \mid v_{jk}, v_{gh})) = \frac{\Pr(v_{jk} \mid c_i) \Pr(v_{gh} \mid c_i) \Pr(c_i)}{\sum_{r=1}^{m} \Pr(v_{jk} \mid c_r) \Pr(v_{gh} \mid c_r) \Pr(c_r)} \quad (13)$$

Since we know $\Pr(c_i \mid v_{jk})$ and $\Pr(c_i \mid v_{gh})$, we finally have:

$$E(\Pr(c_i \mid v_{jk}, v_{gh})) = \frac{\Pr(c_i \mid v_{jk}) \Pr(c_i \mid v_{gh})}{\Pr(c_i) \sum_{r=1}^{m} \frac{\Pr(c_r \mid v_{jk}) \Pr(c_r \mid v_{gh})}{\Pr(c_r)}} \quad (14)$$

Confidence Unexpectedness (Cu):

$$Cu(v_{jk}, v_{gh} \rightarrow c_i) = \frac{\Pr(c_i \mid v_{jk}, v_{gh}) - E(\Pr(c_i \mid v_{jk}, v_{gh}))}{E(\Pr(c_i \mid v_{jk}, v_{gh}))} \quad (15)$$

2.2.2 Support Unexpectedness

As above, we first compute the expected support of v_{jk}, $v_{gh} \rightarrow c_i$.

Expectation: The expected support $\Pr(v_{jk}, v_{gh}, c_i)$ is computed based on the following:

$$\Pr(v_{jk}, v_{gh}, c_i) = \Pr(c_i \mid v_{jk}, v_{gh}) \Pr(v_{jk}, v_{gh}) \quad (16)$$

Using the conditional independence assumption above, we know the value for $\Pr(c_i \mid v_{jk}, v_{gh})$. Let us compute the value for $\Pr(v_{jk}, v_{gh})$ based on the same assumption:

$$\Pr(v_{jk}, v_{gh}) = \Pr(v_{jk}) \Pr(v_{gh}) \sum_{r=1}^{m} \frac{\Pr(c_r \mid v_{jk}) \Pr(c_r \mid v_{gh})}{\Pr(c_r)} \quad (17)$$

By combining Equations (10) and (17), we obtain,

$$E(\Pr(v_{jk}, v_{gh}, c_i)) = \frac{\Pr(v_{jk}, c_i) \Pr(v_{gh}, c_i)}{\Pr(c_i)} \quad (18)$$

Support Unexpectedness (Su):

$$Su(v_{jk}, v_{gh} \rightarrow c_i) = \frac{\Pr(v_{jk}, v_{gh}, c_i) - E(\Pr(v_{jk}, v_{gh}, c_i))}{E(\Pr(v_{jk}, v_{gh}, c_i))} \quad (19)$$

2.2.3 Attribute Distribution Unexpectedness

Since for two-condition rules, two attributes are involved. To compute attribute distribution unexpectedness, we need to fix an attribute. Without loss of generality, we assume v_{jk} is fixed, and include (or vary) all the values of attribute A_g. We thus compute the unexpectedness of:

$$v_{jk}, A_g \rightarrow c_i$$

This attribute distribution represents all rules, v_{jk}, $v_{gh} \rightarrow c_i$, h = 1, 2, …, $|A_g|$, where $|A_g|$ is the number of values of attribute A_g.

Expectation: We can make use of the expected value of Pr(v_{jk}, v_{gh}, c_i) computed above in Equation (18).

Attribute Distribution Unexpectedness (ADu):

$$ADu(v_{jk}, A_g \rightarrow c_i) = \sum_{v_{gh}: v_{gh} \in dom(A_g) \wedge Dev > 0} \frac{Dev(v_{gh})}{\Pr(v_{jk}, c_i)} \quad (20)$$

$$Dev(v_{gh}) = \Pr(v_{jk}, v_{gh}, c_i) - E(\Pr(v_{jk}, v_{gh}, c_i))$$

2.2.4 Attribute Unexpectedness

In this case, we compute the unexpectedness of an attribute given a constraint, which is of the form:

$$v_{jk}, A_g \rightarrow C.$$

Attribute unexpectedness can be defined easily (see [3]).

3. A CASE STUDTY

We present a case study to show the effectiveness of the proposed system. We used reviews of manufactured products from Amazon crawled in 2006 [2]. The class attribute is the review rating. Three classes are made from the ratings. Table 1 shows the classes, ratings and class prior probabilities. Note that we only assigns the rating of 5 to positive, and assigns the rating of 4 to neutral as we want to study extreme behaviors of reviewers.

Table 1. Classes, ratings and class prior probabilities

Class (c_i)	**Ratings**	**Pr(c_i)**
Positive	Rating = 5	0.47
Neutral	Rating = 3 or 4	0.29
Negative	Rating = 1 or 2	0.24

Table 2: The review data set

# of Reviews	415179
# of Products	32714
# of Reviewers	311428
# of Product Brands	3757

In our data, each review forms a data record with three attributes and a class, i.e., *reviewer-id*, *product-id*, *brand-id* and *class*. We have a total of 475K reviews, out of which 50K were written by anonymous reviewers, which were removed in our analysis. Then, we have about 415K data records. A brief description of the data set is given below in Table 2. Due to space limitations, we only show some findings of unexpected confidences and unexpected supports. For other results, see [3].

One-Condition Rules

Confidence Unexpectedness: The top ranked rules show that out of 17863 reviewers with at least 3 reviews (support threshold), 4340 of them have the confidence of 1, meaning they always gave only one class of rating. Those reviewers who wrote only positive (2602 reviewers) and only negative (807 reviewers) reviews are somewhat suspicious or unexpected. Some may be involved in spam activities, writing fake reviews. Since the negative class had the lowest expectation (Pr(*negative*) = 0.24), the reviewers who wrote many negative reviews had the highest unexpectedness. For example, the top ranked reviewer wrote 16 all negative reviews (for rules with the same unexpectedness values, we also sort them using their supports). This reviewer is quite abnormal.

Support Unexpectedness: In this case, people who write more reviews will have higher support unexpectedness. The top ranked rule shows a particular reviewer wrote 626 reviews and all of them have positive ratings, which is highly unusual or suspicious.

Two-Condition Rules

Confidence Unexpectedness: Here we also found many unexpected/interesting rules. Although in one-condition rules, we know that many people write a combination of positive, neutral and negative reviews, here we found many such reviewers actually wrote only positive or only negative reviews on some specific brands. This is suspicious. For example, the top ranked reviewer wrote 27 positive reviews on products of a particular brand (confidence is 1 for positive class), while the expected confidence is only 0.45 as this reviewer wrote many other reviews with varied ratings (the average rating for this brand from all reviewers is only 3.6). There are hundreds of such reviewers.

Support Unexpectedness: Since the data is sparse in the brands and the reviewers, the expected support of a reviewer writing on a brand is low. So, the support unexpectedness is generally high. Using 80% as the confidence cutoff, the top ranked rule shows that a reviewer wrote 30 positive reviews for a particular brand.

4. CONCLUSIONS

This paper studied the problem of identifying atypical behaviors of reviewers. The problem was formulated as finding unexpected rules and rule groups. A set of expectations was defined, and their corresponding unexpectedness measures were proposed. Unexpected rules and groups represent abnormal or unusual behaviors of reviewers, which indicate spam activities. In our experiment, we reported a case study using reviews from Amazon.com, where we found many suspicious reviewers.

5. REFERENCES

[1] Gyongyi, Z. and Garcia-Molina, H. *Web Spam Taxonomy*. Technical Report, Stanford University, 2004.

[2] Jindal, N, Liu, B, Opinion spam and analysis. *WSDM*, 2008.

[3] Jindal, N., Liu, B. and Lim, E-P. *Finding atypical review patterns* for detecting opinion spammers. UIC Tech. Rep., 2010.

[4] Lim, E-P., Nguyen, V-A., Jindal, N., Liu, B. and Lauw, H. W. Detecting product review spammers using rating behaviors. *CIKM*, 2010.

[5] Liu, B., Hsu W., and Ma Y. Integrating classification and association rule mining. *KDD*, 1998.

[6] Liu, B. Sentiment Analysis and Subjectivity. Chapter in the 2nd Edition, *Natural Language Processing Handbook*, 2010.

[7] Liu, J. Cao, Y. Lin, C. Huang, Y. Zhou, M. Low-quality product review detection in opinion summarization. *EMNLP*, 2007.

[8] MacDonald, C. Ounis, I, and Soboroff, I. Overview of the TREC2007 Blog Track. 2007.

[9] Ntoulas, A., Najork, M., Manasse M., Fetterly, D. Detecting Spam Web Pages through Content Analysis. *WWW*, 2006.

[10] Quinlan J.R. *C4.5: Programs for Machine Learning*. 1993.

[11] Wu, B., Goel V. & Davison, B. D. Topical TrustRank: using topicality to combat Web spam. *WWW*, 2006.

[12] Zhang, Z. and Varadarajan, B. Utility scoring of product reviews, *CIKM*, 2006.