This is the author's version of a work that was submitted/accepted for publication in the following source:

Abdel-Hafez, Ahmad, Xu, Yue, & Josang, Audun (2015)
A normal-distribution based rating aggregation method for generating product reputations.
Web Intelligence, 13(1), pp. 43-51.

This file was downloaded from: http://eprints.qut.edu.au/84694/

© Copyright 2015 IOS Press

Notice: Changes introduced as a result of publishing processes such as copy-editing and formatting may not be reflected in this document. For a definitive version of this work, please refer to the published source:

http://doi.org/10.3233/WEB-150306
A normal-distribution based rating aggregation method for generating product reputations

Ahmad Abdel-Hafez\textsuperscript{a}, Yue Xu\textsuperscript{a}, Audun Jøsang\textsuperscript{b}

\textsuperscript{a}School of Electrical Engineering and Computer Science, Queensland University of Technology, Brisbane, Australia

\textsuperscript{b}Department of Informatics, University of Oslo, Oslo, Norway

E-mails: a.abdelhafez@qut.edu.au, yue.xu@qut.edu.au

\{a.abdelhafez,yue.xu,n.tian\}@qut.edu.au, Brisbane, Australia.

E-mail: josang@mn.uio.no

Abstract. With the extensive use of rating systems in the web, and their significance in decision making process by users, the need for more accurate aggregation methods has emerged. The Naïve aggregation method, using the simple mean, is not adequate anymore in providing accurate reputation scores for items, hence, several researches where conducted in order to provide more accurate alternative aggregation methods. Most of the current reputation models do not consider the distribution of ratings across the different possible ratings values. In this paper, we propose a novel reputation model, which generates more accurate reputation scores for items by deploying the normal distribution over ratings. Experiments show promising results for our proposed model over state-of-the-art ones on sparse and dense datasets.

Keywords: Reputation Model; Ratings Aggregation; Uncertainty; Ratings Prediction.

1. Introduction

People are increasingly dependent on information online in order to decide whether to trust a specific object or not. Therefore, reputation systems are an essential part of any e-commerce or product reviews websites, where they provide methods for collecting and aggregating users’ ratings in order to calculate the overall reputation scores for products, users, or services \cite{12}. The existence of reputation scores in these websites helps people in making decisions about whether to buy a product, or to use a service, etc. Reputation systems play a significant role in users’ decision making process.

Many existing reputation models focused on working with sparse dataset; assuming that the accuracy of reputation scores can be affected with the lack of enough ratings per item. Other models focused on robustness of the reputation score, i.e., the value is not easy to be affected by malicious reviews \cite{6}. In general, the majority of the recently proposed reputation systems involved other factors, besides the ratings, such as the time when the rating was given or the reputation of the user who gave that rating. Usually, this data is incorporated with ratings as weights during the aggregation process, performing the weighted average method. These factors can be easily combined into our proposed methods.

One of the challenges that face any reputation model is its ability to work with different datasets, sparse or dense ones. Within any dataset some items may have rich rating data, while others, especially new ones, have low number of ratings. Sparse datasets are the ones that contain higher percentage of items which do not have many ratings or users who didn’t rate many items. However, with the increased popularity of rating systems on the web particularly, sparse datasets become denser by time as ratings build up on the dataset. Current reputation models focused on providing methods which work well with sparse datasets assuming they are the ones require attention only. However, the accuracy of these models decreases with the increment of dataset density. This address the need for a general
reputation model, which provides more accurate reputation scores with any dataset no matter how sparse or dense it is.

On the other hand, most of the existing reputation models don’t consider the distribution of ratings. People usually have different leniency on rating an item depending on their preferences and expectations. For example, a lenient user would rate an item as 5 stars while he has minor negative opinion about it, while another strict user would rate an item as 4 stars because he is more difficult to satisfy. We believe that the reputation system must acknowledge that both ratings are positive ones. Given the previous example, if we use the rating scale [1 – 5], then the values of {4,5} indicate positive opinions, {1,2} indicate negative opinions and {3} is for the neutral opinions. The distributions of positive and negative ratings for an item should influence its reputation. Let’s look at a simple example. Suppose we have an item with 7 ratings {2,2,2,2,3,5,5}, we can say that we have 4 negative, 1 neutral, and 2 positive opinions. Because of the high frequency of rating 2, rationally, the reputation for this item should be less than 3. However, the average of the ratings is 3, which is considered neutral. In other words, the overall reputation score of a specific product can be skewed towards negative, even the number of positive ratings is higher than the negative ones if the count of ratings is not taken into consideration, and vice versa.

In this paper, we propose to consider the frequency of ratings in the rating aggregation process in order to generate reputation scores. The purpose is to enhance accuracy of reputation scores using any dataset no matter whether it is dense or sparse. The proposed methods are weighted average methods, where the weights are assumed to reflect the distribution of ratings in the overall score. An important contribution of this paper is a method to generate the weights based on the normal distribution of the ratings. We evaluate the accuracy of our results using ratings prediction system, and we compare with state-of-the-art methods. Our methods show promising results dealing with any dataset no matter how dense it is.

In the rest of this paper, we will first introduce a couple of existing product reputation models briefly in Section 2, and then we will explain the proposed methods to calculate reputation scores for products in Sections 3. We will also provide detailed experiments and results evaluation in Section 4 in order to prove the significance of our proposed method. Finally in Section 5 we conclude the paper.

2. Related works

Reputation systems were used with many objects, such as webpages, products, services, users, and also in peer-to-peer networks, where they reflect what is generally said or believed about the target object [9]. Different objects have different factors that may affect their reputation values, while some commonality still stand for all of them; such as the time factor. In specific, an item’s reputation is calculated based on ratings given by many users using a specific aggregation method. Garcia et al [6] analyzed ratings’ aggregators including Arithmetic mean, weighted mean, median and mode, against different factors such as robustness and informativeness. Authors proposed that using median or mode is more robust than using Arithmetic mean and weighted mean.

Many methods used weighted average as an aggregator for the ratings, where the weight can represent user’s reputation, time when the rating was given, or the distance between the current reputation score and the received rating. Shapiro [15] proved that time is important in calculating reputation scores; hence, the time decay factor has been widely used in reputation systems [4,7,11,16]. For example, Leberknight et al. [11] discussed the volatility of online ratings, where the authors aimed to reflect the current trend of users’ ratings. They used weighted average where old ratings have less weight than current ones. On the other hand, Riggs and Wilensky [13] performed collaborative quality filtering, based on the principle of finding the most reliable users. Their proposed method is a weighted average using user’s reliability as its weight, which is defined as the ability of a user to provide a rating for an item that is close to the average of this item’s ratings, given by all users.

One of the baseline methods we use in this paper is proposed by Lauw et al., which is called the Leniency-Aware Quality (LQ) Model [10]. This model is a weighted average model that uses users’ ratings tendency as weights. Rating tendency is a value that reflects how users tend to give higher ratings than others. The authors classified users into lenient or strict users based on the leniency value calculated according to Eq. (1), and the leniency value is used as a weight for the user’s ratings when
they are used to calculate a reputation score of an item using Eq. (2).

\[
\begin{align*}
  l_i &= \text{Avg} \left( \frac{r_{ij} - q_j}{r_{ij}} \right) \\
  q_j &= \text{Avg} \left( r_{ij} \times (1 - \alpha \times l_i) \right)
\end{align*}
\] (1) (2)

where \( q_j \) represents the reputation of the item, which is calculated using arithmetic average method. \( \alpha \in [0,1] \) is a constant, and \( l_i \) is the leniency value for user \( i \). \( l_i < 0 \) indicates that the user is strict, and \( l_i > 0 \) indicates that the user is lenient. This model was claimed to provide more accurate reputations when it is used with sparse dataset, but not with dense datasets.

Another baseline model that we use was introduced by Jøsang and Haller, which is a multinomial Bayesian probability distribution reputation system based on Dirichlet probability distribution [7]. This model is probably the most relevant method to our proposed method because this method also takes into consideration the count of ratings. The model introduced in [7] is a generalization to their previously introduced binomial Beta reputation system [8]. The authors indicated that Bayesian reputation systems provide a statistically sound basis for computing reputation scores. They use a cumulative vector \( \vec{R}_y \) to represent the aggregate ratings for a user or agent \( y \), \( \vec{R}_y = (R_y(i), i = 1, ..., k) \). \( R_y(i) \) is the number of ratings of the level \( i \). They added a decay factor to calculate the aggregate ratings assuming that human agents change their behaviour over time. They proposed 4 different forms to represent the reputation to an agent based on the aggregate ratings. Among them, the most effective and representative way is to calculate a single reputation score based on the multinomial probabilities derived from the aggregate ratings which is defined in Eq. (3). \( S_y(i) \) is the probability of rating \( i \) that other agents give to agent \( y \). The overall reputation is calculated by Eq. (4) which is a weighted sum of the rating probabilities with weights \( v(i) \) evenly distributed in the range [0, 1].

\[
\begin{align*}
  S_y(i) &= \frac{R_y(i) + Ca(i)}{C + \sum_{j=1}^{k} R_y(j)} \quad |i| = 1 \ldots k \\
  \sigma &= \sum_{i=1}^{k} v(i) \times S_y(i), \quad v(i) = \frac{i - 1}{k - 1}
\end{align*}
\] (3) (4)

Where \( \sigma \) represents the overall reputation value, \( \vec{S}_y \) represents the score vector of each rating level, \( C \) is a constant value, and \( a(i) \) is the base rate, which equals to \( 1/k \). This model provides more accurate reputation values when the number of ratings per item is small because the uncertainty in these cases is high. However, when it is used on a dense dataset, it acts similarly to the average method with very minor changes.

Using fuzzy models are also popular in calculating reputation scores because fuzzy logic provides rules for reasoning with fuzzy measures, such as trustworthy, which are usually used to describe reputation. Sabater & Sierra proposed REGRET reputation system [14], which defines a reputation measure (and its reliability) that takes into account the individual dimension, the social dimension and the ontological dimension. Bharadwaj and Al-Shamri [5] proposed a fuzzy computational model for trust and reputation, their model used the beta reputation model proposed by Jøsang [8] in order to calculate the reputation of a user. According to them, the reputation of a user is defined as the accuracy of his prediction to other user’s ratings towards different items. Authors also introduced reliability metric, which represent how reliable is the computed score.

In general, some of the proposed reputation systems compute reputation scores based on the reputation of the user or reviewer, or they normalize the ratings by the behaviour of the reviewer. Other works suggested adding volatility features to ratings. According to our knowledge, most of the currently used aggregating methods in the reputation systems do not reflect the distribution of ratings towards an object, which is actually important in determining the reputation of the object [2]. Besides, there are no general methods that are robust with any dataset and always generate accurate results no matter whether the dataset is dense or sparse, for example, LQ model [10] is good with sparse datasets only and Jøsang and Haller model [7] generates more accurate reputation scores for items with low frequent ratings.

3. Normal distribution based reputation model (NDR)

In this section we will introduce a new aggregation method to generate product reputation scores. Before we start explaining the method in details, we want to present some definitions. First of all, in this paper we
use arithmetic mean method as the Naïve method. Secondly, “rating levels” term is used to represent the number of possible rating values that can be assigned to a specific item by a user. For example, considering the well-known five stars rating system with possible rating values of {1,2,3,4,5}, we say that we have five rating levels; one for each possible rating value.

As mentioned previously, the weighted average is the most currently used method for ratings aggregation, while the weights usually represent the time when the rating was given, or the reviewer reputation. In the simplest case, where we don’t consider other factors such as time and user credibility, the weight for each rating is \( \frac{1}{n} \) if there are \( n \) ratings to an item. No matter for the simplest average method or the weighted average methods that take time or other user related factors into consideration, the frequency of each rating level is not explicitly considered. For example, assume that an item receives a set of ratings \(<2, 2, 2, 3, 5, 5>\), for the simplest average method, the weight for each of the ratings is \( \frac{1}{7} \) even the rating level 2 has higher frequency than the other two rating levels. For other weighted average methods, the weights are only related to time or some user related factors but not rating frequency.

In the following discussion, we will use the Naïve method as an example to explain the strength of our proposed method since the other factors can be easily combined into our methods to make the weights related to other factors such as time or user credibility.

<table>
<thead>
<tr>
<th>Ratings</th>
<th>Rating Weight</th>
<th>Level Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naïve</td>
<td>NDR</td>
<td>Naïve</td>
</tr>
<tr>
<td>2</td>
<td>0.1429</td>
<td>0.0765</td>
</tr>
<tr>
<td>2</td>
<td>0.1429</td>
<td>0.1334</td>
</tr>
<tr>
<td>2</td>
<td>0.1429</td>
<td>0.1861</td>
</tr>
<tr>
<td>2</td>
<td>0.1429</td>
<td>0.208</td>
</tr>
<tr>
<td>3</td>
<td>0.1429</td>
<td>0.1861</td>
</tr>
<tr>
<td>5</td>
<td>0.1429</td>
<td>0.1334</td>
</tr>
<tr>
<td>5</td>
<td>0.1429</td>
<td>0.0765</td>
</tr>
</tbody>
</table>

Our initial intuition is that rating weights should relate to the frequency of rating levels, because the frequency represents the popularity of users’ opinions towards an item. Another important fact that we would like to take into consideration in deriving the rating weights is the distribution of ratings. Not losing generality, like many “natural” phenomena, we can assume that the ratings fall in normal distribution. Usually the middle rating levels such as 3 in a rating scale [1-5] system is the most frequent rating level (we call these rating levels “Popular Rating Levels”) and 1 and 5 are the least frequent levels (we call these levels “Rare Rating Levels”). By taking both the rating frequency and the normal distribution into consideration, we propose to ‘award’ higher frequent rating levels, especially popular rating levels, and ‘punish’ lower frequent rating levels, especially rare rating levels.

Table 1 shows the difference between the Naïve method and the proposed Normal Distribution based Reputation Model (NDR) which will be discussed in Section 3.1. From the second column in Table 1 (i.e., Weight per rating), we can notice that using the Naïve method the weight for each rating is fixed which is \( \frac{1}{7} = 0.1429 \). Different from the Naïve method, the NDR method generates different weights for different ratings, especially, the weights from rare ratings such as 2 and 5 to popular ratings such as 3 are increase and the increment is non-linear. This non-linear increase in weights for repeated ratings of the same level will result in a higher aggregated weight for that rating level. For example, rating level 2 is the most frequent level, in comparison, the aggregated weight generated by the Naïve method for rating level 2 is 0.5714, where the NDR model generates a higher value 0.604 which reflects the contribution from the frequency of rating level 2. On the other hand, rating level 3 gets a higher weight 0.186 in the NDR method than the Naïve method which generates a weight value 0.1429, however, this is not because level 3 is more frequent, but because it is a popular rating level. In contrast, rating Level 5 gets a lower weight in the NDR method because it is a rare rating level and not very frequent in this example.

3.1. Weighting based on the normal distribution

Our method can be described as weighted average where the weights are generated based on both rating distribution and rating frequency. As mentioned before, we use a normal distribution because it represents many “natural” phenomena. In our case, it will provide different weights for ratings, where the
more frequent the rating level is, the higher the weight the level will get. In other words, using this weighting method we can assign higher weights to the highly repeated ratings, which we believe will reflect more accurate reputation tendency.

Suppose that we have \( n \) ratings for a specific product \( P \), represented as a vector \( R_p = \{r_0, r_1, r_2, ..., r_{n-1}\} \) where \( r_0 \) is the smallest rating and \( r_n \) is the largest rating, i.e., \( r_0 \leq r_1 \leq r_2 \leq ... \leq r_{n-1} \). In order to aggregate the ratings, we need to compute the associated weights with each rating, which is also represented as a vector \( W_p = \{w_0, w_1, w_2, ..., w_{n-1}\} \). As we discussed previously, the weights to the ratings will be calculated using the normal distribution density function given in Eq. (5), where \( a_i \) is the weight for the rating at index \( i \), \( i = 0, ..., n - 1 \), \( \mu \) is the mean, \( \sigma \) is the standard deviation, and \( x_i \) is supposed to be the value at index \( i \). The basic idea is to evenly deploy the values between 1 and \( k \) for the rating scale \( [1, k] \) over the indexes from 0 to \( n - 1 \). \( k \) is the number of levels in the rating system, in this paper we use the popular 5-star system, then \( k = 5 \).

\[
a_i = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}} \quad (5)
\]

\[
x_i = \frac{(k - 1) \times i}{n - 1} + 1 \quad (6)
\]

Eq. (6) is used to evenly deploy the values of \( x_i \) between 1 and \( k \), where \( x_0 = 1 \) and \( x_{n-1} = k \). In Eq. (5), the value of the mean is fixed, i.e., \( \mu = (k+1)/2 \). However, the value of \( \sigma \) is the actual standard deviation value extracted from the ratings to this item; hence, each item in the dataset will have different flatness for its normal distribution curve.

The purpose of using such these values for \( x, \mu \) and \( \sigma \) is to produce normally distributed weights associated with the \( k \)-levels rating system. The generated weights in Eq. (5) is then normalised so the summation of all weights is equal to 1, hence, we create the normalised weights vector \( W_p = \{w_0, w_1, w_2, ..., w_{n-1}\} \) using Eq. (7).

\[
w_i = \frac{a_i}{\sum_{j=0}^{n-1} a_j} , \quad \sum_{i=0}^{n-1} w_i = 1 \quad (7)
\]

Figure 1 and Figure 2 show the weights generated for the previous example by the Naïve method and the proposed NDR method, where left-most region represents the overall weight for rating level 2, and the middle region and the right-most region are for rating levels 3 and 5, respectively. We can see that, the weights for all ratings are the same in Figure 1, while in Figure 2, the ratings with index near to the middle will be given higher weights.

In order to calculate the final reputation score, which is affected by the ratings and the weights, we need to sum the weights of each level separately. To this end, we partition all ratings into groups based on levels, \( R^l = \{r_0^l, r_1^l, r_2^l, ..., r_{|l|}^l\} \), \( l = 1, 2, ..., k \), for each rating \( r \in R^l \), \( r = l \). The set of all ratings to item \( p \) is \( R_p = \bigcup_{l=1}^{k} R^l \). The corresponding weights for the ratings in \( R^l \) are represented as \( W^l = \{w_0^l, w_1^l, w_2^l, ..., w_{|l|}^l\} \).

The final reputation score is calculated as weighted average for each rating level using Eq. (8), where \( LW^l \) is called level weight which is calculated in Eq. (9).

\[
NDR_p = \sum_{l=1}^{k} (l \times LW^l) \quad (8)
\]

\[
LW^l = \sum_{j=0}^{|l|} w_j^l \quad (9)
\]

Eq. (9) calculates level weights \( LW^l \) as a summation of the weights of every rating belongs to that level.

![Fig. 1: Average Method Weights for the 7 ratings example](image)
3.2. Enhanced NDR model by adding uncertainty (\textit{NDRU})

In this section we will do a slight modification to our proposed NDR method by combining uncertainty principle, introduced by Jøsang and Haller Dirichlet method [7]. This enhancement is important to deal with sparse dataset, because when the number of ratings is small, the uncertainty is high. The enhanced method is expected to pick up the advantages of both reputation models, i.e., the NDR method and the Dirichlet method. Inspired by the Dirichlet method in [7], the NDRU reputation score is calculated using Eq. (10) which takes uncertainty into consideration:

\[ \text{NDRU}_{p1} = \sum_{l=1}^{k} \left( l \times \frac{n \times LW^l + C \times b}{C + n} \right) \] (10)

\( C \) is a priori constant which is set to 2 in our experiments, and \( b = \frac{1}{k} \) is a base rate for any of the \( k \) rating values.

The NDRU method will reduce the effect of complimenting popular rating levels and depreciating rare rating levels process done by the NDR model. We can say that in all cases if the NDR method provides higher reputation scores than the Naïve method, then the NDRU method will also provide higher reputation scores but marginally less than the NDR ones and vice versa. However, as we have mentioned before, in the case of having a small number of ratings per item, the uncertainty will be higher because the base rate \( b \) is divided by the number of ratings plus a priori constant \( n + C \) in Eq. (10). In this case, the difference between the final reputation scores of the NDR and NDRU methods is noticeable. This advantage of the Dirichlet method to deal with sparse data is adopted by the NDRU method. Yet, when we use dense dataset, the difference between the final reputation scores of the NDR and NDRU methods will be very small, which allow the NDRU to behave similarly to the NDR method.

4. Experiments

In the beginning we want to say that there are no globally acknowledged evaluation methods that appraise the accuracy of reputation models. However, we choose to assess the proposed model in regards to the accuracy of the generated reputation scores, and how the items are ranked. Hence, we conducted two experiments in this research. The first experiment is to predict an item rating using the item reputation score generated by reputation models. The hypothesis is that the more accurate the reputation model the closer the scores it generates to actual users’ ratings. For one item, we will use the same reputation score to predict the item’s rating for different users. The mean absolute error (MAE) metric will be used to measure the prediction accuracy.

The second experiment aims to prove that the proposed method produces different results than the Naïve method in terms of the final ranked list of items based on the item reputations. If the order of the items in the two ranked lists generated by the Naïve and NDR methods is not the same, we say that our method is significant. In this part, we consider a reputation model that generates a list of items with the same order as the list generated by the Naïve method as a useless model, because it doesn’t generate any novel value in it. We will use the Kendall tau coefficient method to measure the association between the two ranked lists.

4.1. Datasets

The dataset used in this experiment is the MovieLens dataset obtained from www.grouplens.org, which is publicly available and widely used in the area of recommender systems. The dataset contains about one million anonymous ratings of approximately 3,706 movies, made by 6,040 MovieLens users. In this dataset each user has evaluated at least 20 movies, and each movie is evaluated by at least 1 user. In our experiment we split the dataset into training and testing datasets with 80% of the users used to build the training dataset and the rest are used for testing.

Three new datasets were extracted from the original dataset in order to test the different
reputation models for different levels of sparsity. In the first dataset, each movie has 4 ratings randomly selected from users’ ratings to this movie. The resulting dataset has 1,361 users and 14,261 ratings. This dataset is the sparsest dataset in the experiment. For the second and the third datasets, each movie has 6 and 8 ratings, respectively. The resulting datasets contain 1,760 users and 21,054 ratings for the second dataset, and 2,098 users and 27,723 ratings for the third dataset. The number of movies included for all the datasets is 3,706 movies. Table 2 summarizes the statistics of the used datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Users</th>
<th>Ratings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only 4 ratings per movie (4RPM)</td>
<td>1361</td>
<td>14261</td>
</tr>
<tr>
<td>Only 6 ratings per movie (6RPM)</td>
<td>1760</td>
<td>21054</td>
</tr>
<tr>
<td>Only 8 ratings per movie (8RPM)</td>
<td>2098</td>
<td>27723</td>
</tr>
<tr>
<td>Complete Data Set – All ratings</td>
<td>6040</td>
<td>999868</td>
</tr>
</tbody>
</table>

### 4.2. Evaluation metrics

In the experiments conducted in this research, we select two well-known metrics to evaluate the proposed methods.

#### 4.2.1. Mean Absolute Error (MAE)

The mean absolute error (MAE) is a statistical accuracy metric used to measure the accuracy of rating prediction. This metric measures the accuracy by comparing the reputation scores with the actual movie ratings. Eq. (11) shows how to calculate the MAE.

\[
MAE = \frac{1}{n} \sum_{i=1}^{n} |p_i - r_i|
\]  

where \(p_i\) is the predicted value (i.e., a reputation score) for a movie \(i\), \(r_i\) is the actual rating given by a user for the movie \(i\), and \(n\) is the number of ratings in the testing dataset. The lower the MAE, the more accurately the reputation model generates scores.

#### 4.2.2. Kendall Tau coefficient

Kendall tau coefficient is a statistic used to measure the association between two ranked lists. In other words, it evaluates the similarity of the orderings of the two lists. Eq. (12) shows how to calculate Kendall Tau coefficient, where it divides the difference between concordant and discordant pairs in the two lists by the total number of pairs \(\frac{1}{2} n(n-1)\). The coefficient must be in the range of \(-1 \leq \tau \leq 1\), where the value of \(\tau = -1\) indicates complete disagreement between two lists, and the value of \(\tau = 1\) indicates complete agreement. In addition, the value of \(\tau = 0\) identify that the two lists are independent.

\[
\tau = \frac{n_c - n_d}{\frac{1}{2} n(n-1)}
\]

where 

\[n_d = \frac{|{(i,j) | A(i) < A(j), NDR(i) > NDR(j)}|}{n(n-1)}\]

\[n_c = \frac{|{(i,j) | A(i) < A(j), NDR(i) < NDR(j)}|}{n(n-1)}\]

\(n_d\) is the number of discordant pairs between the two lists, while \(n_c\) is the number of concordant ones, \(NDR(i)\) is the reputation score for the movie \(i\) generated using the NDR method, while \(A(i)\) is the reputation score generated using the Naïve method, \(n\) is the number of items and \(i\) and \(j\) are items. The aim of using the Kendall tau coefficient method is to compute the ordering difference between the two ranked item lists generated based on the reputations computed using two different reputation models. The higher the value of \(\tau\), the more similar the two ranked lists.

#### 4.3. Ratings prediction

In this experiment we use the training dataset to calculate a reputation score for every movie. Secondly we will use these reputation scores as rating prediction values for all the movies in the testing dataset and will compare these reputation values with users’ actual ratings in the testing dataset. The theory is that a reputation value to an item that is closer to the users’ actual ratings to the item is considered more accurate. The Baseline methods we will compare with include the Naïve method, Dirichlet reputation system proposed by Jøsang and Haller [7], and the Leniency-aware Quality (LQ) model proposed by Lauw et al. [10].

The experiment is done as a five-fold cross-validation, where every time a different 20% of the dataset is used for testing. This method ensures that each user’s data has been used five times; four times in training and one time in testing. We record the MAE in each round for all the implemented methods, and at the end we calculate the average of the five MAE values recorded for each reputation model. We
have tested the ratings prediction accuracy using the four previously described datasets and the results are shown in Table 3.

<table>
<thead>
<tr>
<th>DataSet</th>
<th>Naïve</th>
<th>LQ</th>
<th>Dirichlet</th>
<th>NDR</th>
<th>NDRU</th>
</tr>
</thead>
<tbody>
<tr>
<td>4RPM</td>
<td>0.5560</td>
<td>0.5576</td>
<td>0.5286</td>
<td>0.5614</td>
<td>0.5326</td>
</tr>
<tr>
<td>6RPM</td>
<td>0.5610</td>
<td>0.5628</td>
<td>0.5514</td>
<td>0.5608</td>
<td>0.5498</td>
</tr>
<tr>
<td>8RPM</td>
<td>0.5726</td>
<td>0.5736</td>
<td>0.5705</td>
<td>0.5603</td>
<td>0.5676</td>
</tr>
<tr>
<td>ARPM</td>
<td>0.7924</td>
<td>0.7928</td>
<td>0.7928</td>
<td>0.7851</td>
<td>0.7853</td>
</tr>
</tbody>
</table>

The four datasets we use include three sparse datasets (i.e., 4RPM, 6RPM, and 8RPM) and one dense dataset (i.e., ARPM). The three sparse datasets reflect different levels of sparsity. In Table 3, the MAE results using the sparsest dataset 4RPM shows that the best prediction accuracy was produced by the Dirichlet method. The reason is because the Dirichlet method is the best method among the tested 5 methods to deal with the uncertainty problem which is especially severe for sparse datasets. The proposed enhanced method NDRU achieved the second best result which is close enough to the Dirichlet method result with a small difference, indicating that NDRU is also good at dealing with uncertainty. On the contrary, the proposed NDR method returns the worst result for the sparsest dataset because with small number of ratings there are no enough rating frequencies to feed the distribution weighting system.

However, when we use less sparse datasets 6RPM and 8RPM, the proposed NDRU method achieved the best results. In more details, when we use the 8RPM dataset, the NDR accuracy is the second best result and better than all the baseline methods, but it is still worse than the Dirichlet method using the 6RPM dataset.

Finally, the last row in Table 3 shows the results of ratings prediction accuracy using the whole MovieLens dataset (ARPM) which is considered a dense dataset. We can see that the proposed method NDR has the best accuracy. Moreover, our enhanced method NDRU achieved the second best result with an extremely small difference of 0.0002. In contrast, the other baseline methods do not provide any enhancement in accuracy over the Naïve method on the dense dataset.

From the results we can see that the NDR method produces the best results when we use it with dense datasets, and that the Dirichlet method is the best with sparse datasets. Most importantly, the enhanced NDR method with uncertainty, i.e., the NDRU method, provides good results in any case, and can be used as a general reputation model regardless of the sparsity in datasets. The NDRU method keeps the advantages of both: the advantage from the Dirichlet method when deals with sparse datasets and the advantage from the NDR method when deals with dense datasets.

4.4. Comparisons of item’s ranking

In this experiment, we will compare two lists of items ranked based on their reputation scores generated using the NDR method and the Naïve method. The purpose of this comparison is to show that our method provides relatively different ranking for items from the Naïve method.

The experiment is conducted in 20 rounds, with different percentage of data used every time. In the first round we used a sub-list with only the top 1% of the ranked items in one list to compare with the 1% items in the other list. The number of comparisons is equal to \( \frac{n(n-1)}{2} \), \( n \) is the number of items in the top 1% of each list. For the other 19 rounds we used the top 5%, 10%, 15%, ..., 100%, respectively. The reason for choosing different percentages of top items is to see the difference between different percentages of top items. Usually the top items are more influential or crucial to users.

From Figure 3 we can find that, for all datasets, the more the items taken from the lists, the more similar the order of the items in the lists generated by the two methods. However, usually users are more interested in the top items. Therefore, the order of the top items in the lists is more crucial. If we only look at the top 20% items, we can find that the behaviour of using the whole dataset ARPM (which is much denser than the other three datasets) is different from using other three sparse datasets. For the dense dataset, the similarity reaches its minimal when we only compare the top 1% items and the similarity increases when we compare larger portions of the dataset. This result indicates that for the dense dataset, the proposed method NDR ranks the top items in the item list differently from the item list generated by the Naïve method.
On the other hand, with the sparse datasets, the ranking on the top 1% of the items shows high similarity between the two lists, which indicates that the top 1% items are ranked highly similar for the sparse datasets. This can be explained that due to the sparsity of the dataset, limited items can be selected to be on the top 1% which makes the possibility of choosing the same items very high. When we increase the percentage of the top items, the similarity decreases sharply. In summary, for all the four datasets, the ranking order of the top 20% items in the ranked lists generated by the Naïve method and our proposed method NDR is different.

5. Conclusions and future work

In this work we have proposed a new aggregation method for generating reputation scores for items or products based on customers’ ratings, where the weights are generated using a normal distribution. The method is also enhanced with adding uncertainty part by adopting the idea of the work proposed by Jøsang and Haller [7]. The results of our experiments show that our proposed method outperforms the state-of-the-art methods in ratings prediction over a well-known dataset. Besides, it provides relatively different ranking for items in the ranked list based on the reputation scores. Moreover, our enhanced method proved to generate accurate results with sparse and dense datasets. In future, we plan to use this method in different applications such as recommender systems [1,3]. Besides, this method can be combined with other weighted average reputation models that use time or user reputation in order to improve the accuracy of their results.

References