Gazi University Journal of Science
GU J Sci
26(4): 557-562 (2013)

Effects of Similarity Measures on The Quality of Predictions

Edip SENYUREK[1], Huseyin POLAT[2,♠]

[1,2] *Anadolu University, Faculty of Engineering, Department of Computer Engineering, Eskişehir, TURKEY*

Received:13.02.2013 Accepted:19.10.2013

ABSTRACT

Providing accurate predictions efficiently is vital for the success of recommender systems. There are various factors that might affect the quality of the predictions and online performance. Similarity metric used to determine neighbors is one of such factors. Therefore, given a set of metrics, determining and utilizing the best one is critical for the overall success of collaborative filtering schemes. We scrutinize several binary similarity measures in terms of accuracy and performance. We conduct various real data-based experiments in order to determine the best similarity measure. Our empirical outcomes show that Yule and Kulczynski metrics provide the best results.

Keywords: Accuracy, performance, binary similarity metric, prediction, collaborative filtering

1. INTRODUCTION

Collaborative filtering (CF) is a filtering and recommendation technique, which is widely used by many e-commerce sites in order to overcome the information overload problem. With increasing popularity of the Internet and e-commerce, CF is receiving increasing attention. Due to information overload, various online vendors that collect data from their customers utilize CF techniques to help their customers select appropriate items.

Goldberg et al. [1] define CF as people collaborate to help one another classify their actions as interesting or uninteresting. Customers rate objects such as books, DVDs, movies, and so on based on how much they like them [2]. When an active user (a) wants to purchase an item over the Internet, e-commerce sites recommend the items that could be liked by a while considering the similarity of other users' rates and the active user's previous votes.

Traditional CF process consists of three main steps. They are collecting data, forming neighborhoods based on similarity weights, and recommendation generation. Data collected for filtering purposes might be numeric or binary. While numeric ratings show how much users like or dislike products, binary votes show whether users like or dislike items. Users' preferences gathered from n user for m items are stored in a user-item matrix D. In order to determine the best similar users to a (referred to as the neighbors), similarity weights between a and each user in the database are estimated. According to the data types (numeric or binary), different similarity measures can be used. In the final stage, a recommendation is generated using a recommendation algorithm. CF schemes usually provide two recommendation services. They either offer a sorted list of liked items (called top-N recommendation) or a single prediction for a target item q (called prediction).

Although CF systems face with various challenges, accuracy and online performance are two most important ones. Providing precise recommendations efficiently is very important for the overall success of recommender systems. Similarity measures have effects

♠Corresponding author, e-mail: polath@anadolu.edu.tr

on accuracy and efficiency. Determining the best similarity metric is vital for improving the overall performance of any filtering algorithm. In other words, determining those entities very similar to the active users or target items as neighbors help CF systems improve accuracy. Moreover, time to spend for forming neighborhoods affects online performance. Thus, finding out the best similarity measures and employing them are critical [3].

There are various studies proposed for enhancing accuracy and performance of CF schemes. Papagelis et al. [4] propose a method for improving efficiency based on incremental updates of user-to-user similarities. Robu and La Poutré [5] propose a method for constructing the utility graphs of buyers automatically, which provides high degree of accuracy. Miyahara and Pazzani [6] utilize simple Bayesian classifier, which is one of the most successful supervised machine-learning algorithms, to offer binary ratings-based predictions. Their scheme helps online vendors categorize items as liked or disliked rather than providing how much they will be liked or disliked. They then propose a combined method, user- and item-based CF, which performs better than single collaborative recommendation method [7]. Kaleli and Polat [8] investigate how to improve Bayesian classifier-based CF systems' online performance using clustering. They divide users into clusters so that prediction can be generated on similar, dissimilar, or both similar and dissimilar users. Their scheme aims at improving both performance and accuracy.

In addition to abovementioned studies, researchers investigate similarity metrics. Cha et al. [9] review, categorize, and evaluate various binary vector similarity and dissimilarity measures for character recognition. Zhang and Srihari [10] study seven similarity measures, such as Jaccard-Needham, Correlation, Yule, Russell-Rao, Sokal-Michener, Rogers-Tanimoto and Kulzinsky, for binary feature vectors, which are summarized by Tubbs [11].

In this study, we study the effects of seven most popular similarity measures on the quality of the predictions and efficiency. We focus on binary similarity metrics, which have not been investigated in the context of CF. Since there are too many binary similarity measures, the most popular seven metrics are investigated in terms of both accuracy and performance. Since off-line costs are not critical for the overall performance, the emphasis is given to online costs. We conduct several experiments using real data sets and display our empirical outcomes. We finally provide some suggestions for selecting the best metrics in order to offer accurate predictions efficiently.

2. EXPERIMENTS

In order to investigate binary similarity metrics with respect to preciseness and performance, we performed various experiments using two well-known real data sets. In order to select neighbors for a given *a*, similarity values between *a* and each user in the database are estimated using a binary ratings-based measure. Then, the most similar *k* users can be chosen as neighbors. Therefore, in order to form good neighborhoods, utilizing the best similarity measure becomes imperative. The more accurate the neighborhood is, the better the results are. Moreover, similarity measures might affect online performance.

Although there are several similarity measurements, we scrutinize the most well-known seven binary similarity measurements (SMs), as shown in Table 1. Similarity metrics proposed for binary data are based on four values. First one is the number of ones from two vectors (S_{11}), second one is the number of ones from the first vector and zeros from the second vector (S_{10}), third one is the number of zeros from the first vector and ones from the second vector (S_{01}), and the last one is the number of zeros from two vectors (S_{00}).

Table 1. Binary similarity measurements

SMs	Formula
Anderberg	$\frac{\frac{S_{11}}{S_{11}+S_{10}}+\frac{S_{11}}{S_{11}+S_{01}}+\frac{S_{00}}{S_{01}+S_{00}}+\frac{S_{00}}{S_{10}+S_{00}}}{4}$
Gower2	$\frac{S_{11}S_{00}}{\sqrt{(S_{11}+S_{10})(S_{11}+S_{01})(S_{10}+S_{00})(S_{01}+S_{00})}}$
Jaccard	$\frac{S_{11}}{S_{11}+S_{10}+S_{01}}$
Kulczynski	$\frac{\frac{S_{11}}{S_{11}+S_{10}}+\frac{S_{11}}{S_{11}+S_{01}}}{2}$
Ochiai	$\frac{S_{11}}{\sqrt{(S_{11}+S_{10})(S_{11}+S_{01})}}$
Pearson's Correlation	$\frac{S_{11}S_{00}-S_{10}S_{01}}{\sqrt{(S_{11}+S_{10})(S_{11}+S_{01})(S_{10}+S_{00})(S_{01}+S_{00})}}$
Yule	$\frac{S_{11}S_{00}-S_{10}S_{01}}{S_{11}S_{00}+S_{10}S_{01}}$

2.1. Data Sets

We utilized two data sets collected for CF purposes. We used MovieLens (ML) to represent sparse data set and Jester to represent a dense data set. ML consists of ratings for movies and it was collected by the GroupLens research team at the University of Minnesota (http://www.cs.umn.edu/research/GroupLens). Jester is web-based joke recommendation system (http://eigentaste.berkeley.edu/user/index.php). Table 2 describes both data sets.

Table 2. Data sets

	ML	Jester
Total user	6,041	17,998
Total items	3,900	100
Total ratings	788,063	906,474
Density (%)	3.34	50.37
Rating type	Discrete	Continuous
Rating range	1 to 5	-10 to 10

2.2. Evaluation Criteria

Several types of measures are used for evaluating the success of the recommender systems. There are different evaluation criteria. In this study, F-measure (F1) and classification accuracy (CA) are used to evaluate the similarity measures in terms of accuracy. CA is the ratio of number of correct classifications to number of classifications. F1 is a weighted combination of precision and recall, which are widely used metrics in the informational retrieval, as follows [6, 7]:

$$P = Presicion = \frac{\text{\# of liked items assigned to "Like" class}}{\text{\# of items assigned to "Like" class}} \quad (1)$$

$$R = Recall = \frac{\text{\# of liked items assigned to "Like" class}}{\text{\# of liked items}} \quad (2)$$

F1 can be defined as 2PR/(R+P). In addition to assessing the similarity measures in terms of preciseness, we also evaluate them in terms of online performance. For this purpose, we define T in seconds as the total amount of time required to estimate predictions online.

2.3. Our Methodology

ML and Jester include numeric votes. We first need to transform them into binary ratings. For ML data set, the ratings are transformed into one (*like*) if they are bigger than three; or zero (*dislike*) otherwise. Similarly, for Jester data set, the ratings are converted into one (*like*) if they are bigger than two; or zero (*dislike*) otherwise. Thus, zero (0) represents the disliked items and one (1) represents the liked items.

After data transformation, we uniformly randomly selected 3,000 users who rated at least 50 and 60 items from ML and Jester, respectively. We then uniformly randomly divided these users into two sub sets. One of the sets, referred to as train set, contains 2,000 users. The other set, called test set, includes the remaining 1,000 users. In each set of trials conducted in the followings, two thirds of total numbers of users are used for training and one third of total numbers of users are used for testing.

To provide predictions for single items, naïve Bayesian classifier (NBC)-based algorithm is utilized. A Bayesian classifier is a probabilistic framework for solving classification problems. It is the most successful machine learning algorithms in many classification domains [12]. Given a set of variables, $X = \{x_1, x_2, x_3, \dots, x_d\}$, the posterior probability can be constructed for the event C_j among a set of possible outcomes $C = \{c_1, c_2, c_3, \dots, c_d\}$. X is the predictors and C is the set of discrete levels present in the dependent variable. Using Bayes' rule:

$$p(C_j|x_1, x_2, x_3, \dots, x_d) \propto p(x_1, x_2, x_3, \dots, x_d|C_j)p(C_j), \quad (4)$$

where $p(C_j|x_1, x_2, x_3, \dots, x_d)$ is the posterior probability of class membership, i.e., the probability that X belongs to C_j. Since it is assumed that the conditional probabilities of the independent variables are statistically independent, the likelihood to a product of terms can be decomposed:

$$p(X|C_j) \propto \prod_{k=1}^{d} p(x_k|C_j), \quad (5)$$

and rewrite the posterior as:

$$p(C_j|X) \propto p(C_j) \prod_{k=1}^{d} p(x_k|C_j). \quad (6)$$

Using Bayes' rule above, a new case X with a class level C_j that achieves the highest posterior probability is labeled.

To produce predictions based on binary ratings, NBC-based algorithm can be used. Instead of applying NBC to all available users' data, the most similar users to *a* can be selected as neighbors according to similarity values. Therefore, we first determine the most similar *k* users to *a* using seven similarity measures. Then, we apply NBC algorithm to their data to estimate a prediction for single items. The predictions for single items can be estimated, as follows:

i. Determine similarities between *a* and each user in the train set using a similarity measure.

ii. Sort train users in descending order according to their similarity weights.

iii. Choose the first *k* users as *a*'s neighbors.

iv. Apply NBC-based CF algorithm to *a*'s and her neighbors' data.

v. Estimate predictions for five rated items selected randomly.

vi. Do this for each test user in the test set.

Notice that for each test user, after selecting five rated items randomly, we replace their entries with null and withhold their true votes; and try to predict their ratings using the aforementioned approach. Once we estimate predictions for all test items and for all test users, we then compare the predicted ones with the observed ratings. After computing the overall averages of CA and F1 and T values, we display them.

There are various controlling parameters that might affect the overall performance. Number of users (*n*), number of items (*m*), number of neighbors (*k*), density, and similarity measurements are among such parameters. We try to demonstrate how varying *n*, *m, k* values affect the quality of the predictions for density and sparse data sets with seven similarity measures. Thus, we conduct the following experiments.

2.4. Our Methodology

We first conducted trials using both data sets and seven similarity measures while varying *n* and *k* values. We decreased *n* values from 2,000 to 124, where we varied the corresponding *k* values from total number of users (we assumed that all train users are chosen as neighbors) to 25. Note that we used *n*/2 number of uniformly randomly selected users as test users. We first performed trials for *n* = 2,000. Then, we conducted experiments for *n* = 1,000, 500, 250, or 124. After estimating predictions for all test items, we compared

them with true votes and computed CA, F1 values and T values for both data sets. Since the results show very similar trends with varying n values, we showed the outcomes for n = 2,000 only for both data sets. Likewise, since F1 and CA values show similar trends, we displayed F1 values for Jester and CA values for ML only.

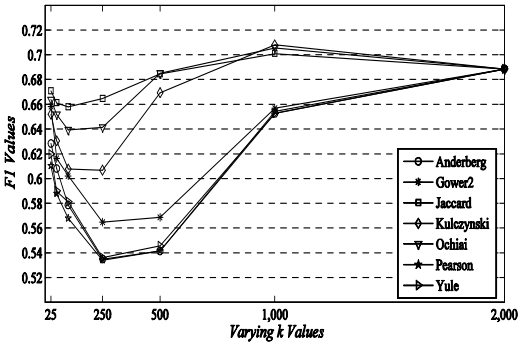

Figure 1. F1 values with varying k values (Jester & n = 2,000)

In Figure 1, we showed F1 values for Jester, where n = 2,000. Note that we varied k values from 2,000 to 25. As seen from the figure, we can see all curves have similar shape for each similarity measurements; however, Kulczynski similarity measurement achieves the highest F1 value when k = 1,000. Ochiai and Jaccard similarity measurements follow Kulczynski similarity measurement. As seen from Figure 1, we can say that Jaccard similarity measurement performs best for all k values except 1,000. On the other hand, Pearson Correlation similarity measurement gives the worst results for all k values.

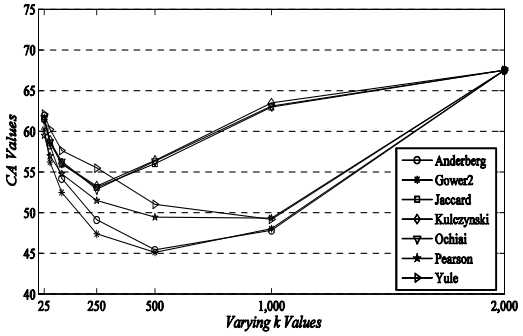

Figure 2. CA values with varying k values (ML & n = 2,000)

In Figure 2, we showed CA values for ML data set, where n = 2,000. Note again that we varied k values from 2,000 to 25. According to the figure, we obtained the highest CA value using Yule similarity measurement with k being 25. For the same k value, Anderberg, Kulczynski, Ochiai, and Jaccard similarity measurements give the best results after Yule. Even we got the highest CA value with Yule similarity measurement for all k values, accuracy decreases with varying k values. Note that when the number of nearest neighbors is equal to the number of train users, the CA value would be the same for each similarity measurement. When k is bigger than 250, outcomes enhance for Kulczynski, Ochiai, and Jaccard. Gower2 similarity measurement gives the worst results when k = 500.

In Figure 3, we showed T (on-line duration) values for ML data set only because we got similar results for Jester. As seen from Figure 3, Gower2 similarity measurement gives the worst results for k values 25, 50, 100, and 250. Then, when k = 500, Ochiai achieves the worst duration result. When k is larger than 500, Anderberg similarity measurement achieves the worst performance. The best results are achieved by Pearson Correlation and Yule similarity measurements.

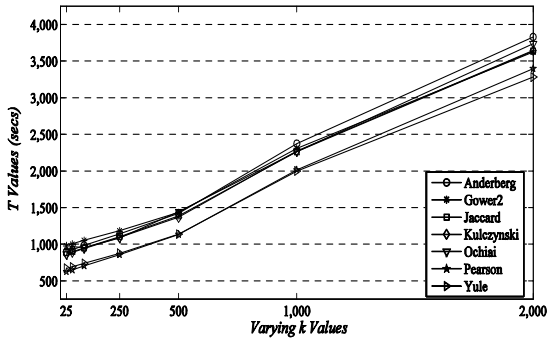

Figure 3. T values with varying k values (ML & n = 2,000)

After scrutinizing similarity metrics with varying n and k values for both data sets, we also studied them while varying m values. In addition to n and k, m is also among the controlling parameters that should be investigated. In order to demonstrate how overall performances of seven similarity metrics change with varying m values while generating predictions, we conducted a set of trials using ML data set only because there are limited numbers of items in Jester. Note that there are 100 jokes only in Jester. Thus, it does not make any sense to perform trials while varying m using it.

We used 900 and 450 train and test users, respectively in which we set k to 100. Due to the low density of new matrices for 500 items, we could use 350 and 175 train and test users, respectively. In these sets of experiments, we varied m from 3,900 to 500. We estimated predictions for five rated items for each active user while varying m (m = 3,900, 2,000, 1,000, or 500) and using different similarity metrics. After computing overall averages of CA, F1, and T values, we demonstrated them.

We first estimated CA values while varying m values and displayed them in Figure 4. Remember that we used 900 train users. However, we only used 350 train users when m is 500 because there are no enough users who provided enough ratings for 500 items. We also set k to 100. We produced predictions for all test items using different similarity metrics. As seen from Figure 4,

Yule similarity measure provides the best predictions in terms of CA values for *m* values of 3,900, 2,000, and 1,000. Gower2 similarity measure, on the other hand, produces the worst results for the same values. When *m* is 500, Jaccard metric achieves the best outcomes, while Anderberg similarity measurement accomplishes the worst results, as seen from Figure 4.

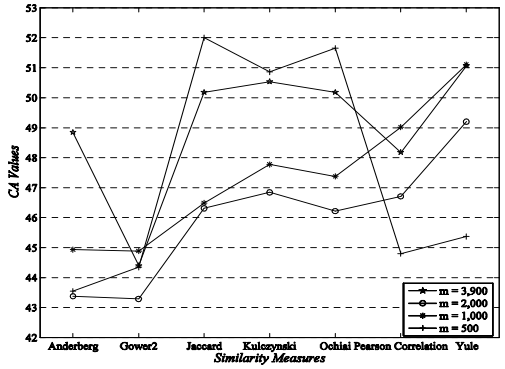

Figure 4. CA values with varying *m* values

We finally computed online duration times for each similarity measures while varying *m* values. We displayed them in Figure 5. As seen from Figure 5, with decreasing number of items, as expected, online time decreases, as well. For smaller *m* values, almost all similarity measures perform similarly. There are no significant differences between measures in terms of online times. With increasing *m* values, on the other hand, Yule and Pearson correlation measures perform better than others do. Anderberg measure on the other hand performs worst with respect to online performance.

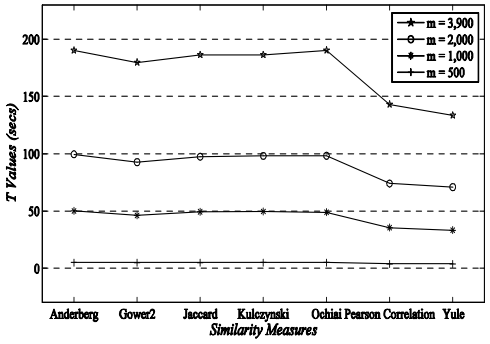

Figure 5. T values with varying *m* values

3. RESULTS AND DISCUSSION

Our empirical outcomes, in general, demonstrate that similarity metrics have effects on accuracy and efficiency. When there are 2,000 train users' ratings collected for CF purposes, in order to get the best outcomes, Yule and Kulczynski similarity measures can be chosen for sparse and dense sets, respectively. They are the most appropriate measures to offer the high quality recommendations on binary ratings. Unlike such measures, Gower2 and Pearson Correlation similarity measurements provide the worst outcomes for sparse and dense sets, respectively. In terms of online computation times for *n* being 2,000, the results are similar for all metrics. However, Anderberg measure is the worst metric in terms of online duration time for both sparse and dense sets. Yule gives very promising results in terms of performance for both data sets for almost all *k* values.

When we varied number of items, accuracy also changes with varying similarity measures. Yule metric achieves the best results. As expected, online performance degrades with increasing *m* values.

REFERENCES

[1] Goldberg, D., Nichols, D. A., Oki, B. M. and Terry, D. B, "Using collaborative filtering to weave an Information Tapestry", *Communications of the ACM*, 35 (12), 61-70, 1992.

[2] Perkowitz, M. and Etzioni, O., "Towards adaptive Web sites: Conceptual framework and case study", *Artificial Intelligence*, 118 (1-2), 245-275, 2000.

[3] Teknomo, K., *Why do we need to measure similarity?*, http://people.revoledu.com/kardi/tutorial/Similarity/Applications.html, Accessed on November 1, 2012.

[4] Papagelis, M., Rousidis, I., Plexousakis, D. and Theoharopoulos, E., "Incremental collaborative filtering for highly-scalable recommendation algorithms", *In Proceedings of the 15th International Conference on Foundations of Intelligent Systems*, Saratoga Springs, NY, USA, 553-561, 2005.

[5] Robu, V. and La Poutré, H., "Learning the structure of utility graphs used in multi-issue negotiation through collaborative filtering", *Lecture Notes in Computer Science*, 4078, 192-206, 2009.

[6] Miyahara, K. and Pazzani, M. J., "Collaborative filtering with the simple Bayesian classifier", *In Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, Melbourne, Australia, 679-689, 2000.

[7] Miyahara, K. and Pazzani, M. J., "Improvement of collaborative filtering with the simple Bayesian classifier", *Information Processing Society of Japan*, 43 (11), 2002.

[8] Kaleli, C. and Polat, H., "Similar or dissimilar users? Or both?", *In Proceedings of the 2009 2nd International Symposium on Electronic Commerce and Security*, Nanchang, China, 184-189, 2009.

[9] Cha, S.-H., Yoon, S. and Tappert, C. C., "On binary similarity measures for handwritten character recognition", *In Proceedings of 8th International Conference on Document Analysis and Recognition*, Seoul, Korea, 4-8, 2005.

[10] Zhang, B. and Srihari, S. N., "Binary vector dissimilarity measures for handwriting identification", *Document Recognition and Retrieval X*, 5010 (1), 28-38, 2003.

[11] Tubbs, J. D., "A note on binary template matching", *Pattern Recognition*, 22 (4), 359-365, 1989.

[12] Friedman, N., Geiger, D. and Goldszmidt, M., "Bayesian network classifiers", *Machine Learning*, 29, 131-163, 1997.