2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology

Improving Privacy-Preserving NBC-based Recommendations by Preprocessing

Alper Bilge and Huseyin Polat
Department of Computer Engineering
Anadolu University, 26470 Eskisehir, Turkey
abilge, polath@anadolu.edu.tr

***Abstract*—Providing accurate predictions efficiently with privacy is imperative for both customers and e-commerce vendors. However, privacy, accuracy, and performance are conflicting goals. Although producing referrals with privacy is possible; however, online performance and accuracy degrade due to underlying privacy-preserving measures. We investigate how to improve both efficiency and accuracy of naïve Bayesian classifier-based private recommendations by utilizing preprocessing. We preprocess masked data by selecting the best similar items to each item off-line. Moreover, we fill some of the unrated items' cells to improve density. We perform real data-based experiments to investigate how preprocessing affects online performance and accuracy. Our experiment results show that efficiency and preciseness improve due to preprocessing.**

***Keywords*-Privacy; collaborative filtering; preprocessing; online performance; accuracy; Bayesian classifier.**

I. Introduction

The exponential growth of e-commerce services brings the information overload with itself. Such e-commerce facilities need assistance to deal with incessantly growing data. Collaborative filtering (CF) is designed to present only the most valuable information to the user according to the preferences of them with two basic goals (i) which items to be presented and (ii) when and how they should be presented [4]. CF schemes analyze users' opinions to produce a prediction for a target user, referred to as active user (a) [6].

Miyahara and Pazzani [8] propose a CF approach using naïve Bayesian classifier (NBC) to offer binary ratings-based referrals. With increasing number of users (n) and/or items (m), scalability of NBC-based CF schemes becomes an important issue to deal with. Scalability problem is associated with the performance of CF algorithms run on large data sets [1]. Since NBC-based schemes run on whole user-item matrix (D), as n and/or m increase, the performance of the algorithm decreases. Data sparsity is another well-known problem such schemes face. With increasing sparsity, number of commonly rated items decrease relatively; thus, calculation of neighborhood becomes inaccurate.

Privacy is becoming a major concern with increasing popularity of the Internet and e-commerce. Due to privacy concerns [3], many users do not want to give their data or provide false data. To produce accurate and dependable predictions, collecting sufficient and truthful data is important. Therefore, privacy-preserving CF schemes are becoming popular with increasing privacy concerns. Kaleli and Polat [5] propose a privacy-preserving scheme to offer binary ratings-based predictions while achieving privacy. Although their method achieves privacy and is able to produce referrals with decent accuracy, efficiency significantly degrades with increasing n values and number of groups (M). By selecting the best similar items to each item by preprocessing the disguised data off-line, online performance is more likely to improve because amount of data involved in CF process decreases. Due to privacy concerns, accuracy losses are inevitable. Although such losses are expected and could be said that acceptable, we hypothesize that the quality of referrals might be improved by filling some of the unrated items' cells before applying CF process.

In this paper, we want to enhance online performance of the privacy-preserving NBC-based CF scheme (PPNBC) proposed by [5] without greatly compromising (or sometimes even increasing) accuracy by preprocessing the disguised data so that recommendation processes can be completed in a reasonable amount of time. To achieve this, we take advantage of NBC-based CF algorithm's ability to estimate predictions from a small amount of training data. We reduce the number of items by preprocessing and fill some of the empty cells. This way PPNBC might not suffer from large amount of data (mostly irrelevant and redundant); and can also produce highly accurate referrals.

II. Background

NBC is utilized in CF by classifying ratings as *like* and *dislike* [8]. An active user a for whom the prediction will be produced is having ratings utilized as class labels and D holds ratings from other users corresponding to the feature values, where other users are treated as features. Since it is assumed that features are independent given the class labels, the conditional probability that a given item belongs to $class_j$, where $j \in \{like, dislike\}$ given its n feature values, is given, as follows:

$$p(class_j|f_1, f_2, \ldots, f_n) \propto p(class_j) \prod_{i}^{n} p(f_i|class_j), \quad (1)$$

where f_i is the feature of target item (q) for user i and $p(class_j)$ and $p(f_i|class_j)$ can be determined from training data. The conditional probabilities are calculated and q is assigned to the class with the highest probability.

978-0-7695-4191-4/10 $26.00 © 2010 IEEE
DOI 10.1109/WI-IAT.2010.109

Kaleli and Polat [5] propose PPNBC to offer NBC-based predictions. In their scheme, each user u_i divides her items into M groups and perturbs the ratings of every group individually, where M and θ would be considered as levels of privacy. However, with increasing M values, efficiency of the algorithm becomes worse. Moreover, with decreasing θ values from 1.0 to 0.51, since randomness increases, the quality of the referrals diminishes. Therefore, feasible parameters are proposed as three groups with a perturbation level of 0.7 (M is 3 and θ is 0.7) [5]. Although their scheme preserves individual users' privacy, its performance decreases considerably with privacy concerns. Also, accuracy losses are inevitable due to privacy protection measures.

III. Related Work

Polat and Du [9] propose a method for privacy-preserving collaborative filtering (PPCF), which is a centralized method, where users do not actively participate in prediction production processes once they provide their personal data to a server. They propose to employ randomized perturbation techniques to disguise private data and have shown that accurate recommendations can be produced without greatly jeopardizing users' privacy. Kaleli and Polat [5] investigate how to achieve NBC-based CF services while preserving customers' privacy. Their scheme is able to offer accurate referrals with privacy. However, due to privacy concerns, online performance significantly degrades. The quality of data processing has a significant effect on success or failure of almost all data mining algorithms, which makes the data preprocessing stage extremely important task [7]. Symeonidis et al. [10] propose to combine user and item-based neighborhood formation using nearest bi-clusters with bi-clustering algorithms and data preprocessing tools. Bell and Koren [2] apply data preprocessing to pre-compute certain values on experimental data to enable rapid retrieval of useful information. Unlike their schemes, we propose two preprocessing methods to improve both efficiency and even accuracy of PPNBC proposed by [5]. We select the best similar items to each item to reduce data involved in CF process so that online performance improves; and fill some unrated cells to increase density to improve accuracy.

IV. Enhancement on NBC-based Private Recommendations

Due to privacy-preserving measures, supplementary online computation costs are inevitable. Although PPNBC scheme proposed by Kaleli and Polat [5] provides accurate referrals with privacy, efficiency significantly decreases. Moreover, the quality of the referrals diminishes due to data masking schemes. We hypothesize that online performance and even accuracy of PPNBC can be improved by preprocessing disguised data off-line. If only a subset of items are used in the recommendation generation process, (i) recommendation accuracy would increase with most information providing items and (ii) performance of the algorithm would increase considerably. We apply two preprocessing methods to perturbed ratings: In the first method referred to as *neighborhood formation*, we determine the best similar items to each item using a binary similarity measure. In the second method, which we call *increasing density*, we fill some of the randomly chosen empty cells in the disguised user-item matrix (D') with personalized ratings estimated by PPNBC.

Neighborhood Formation: To improve online performance, amount of data involved in CF process should be decreased. For each item in D, we form the neighborhood by selecting the best similar items. Number of items in a neighborhood directly affects efficiency. Similarly, an item's neighbors significantly influence preciseness. The best N similar items are chosen as neighbors and the optimum value of N can be determined experimentally, where N is a constant and $N < m$. To determine the neighbors, which are expected to give the best results, a binary similarity measure should be selected cautiously. Due to its given importance to commonly rated items, Tanimoto coefficient is preferable.

Let S_{ij} be the number of occurrences of commonly rated items with i in the first pattern and j in the second pattern, where $i, j \in \{0, 1\}$. Given two binary feature vectors X and Y, $T(X, Y)$ denotes modified Tanimoto coefficient between X and Y (similarly, between two items' ratings vectors) and it is calculated, as follows [5]:

$$T(X,Y) = \frac{(S_{11} + S_{00}) - (S_{10} + S_{01})}{S_{11} + S_{00} + S_{10} + S_{01}}, \qquad (2)$$

where S_{11} is the number of users rated "1" both items, S_{10} represents the number of users rated "1" item i and "0" item j, S_{01} is the number of users rated "0" item i and "1" item j, and S_{00} shows the number of users rated "0" both items.

Tanimoto similarity measure computes the similarity between two binary vectors; however, D, representing true users' ratings, does not present. Due to underlying data perturbation methods, it becomes a challenge to estimate the same similarities from perturbed data. D' is obtained after collecting data from many users, as described in [5]. Therefore, actual rates cannot be determined exactly, but according to disguising scheme (number of groups, M and disguising rate, θ), an inference can be made to calculate similarities between features. If we assume that S_{ij} be the occurrence of commonly rated items for two items' ratings vector, $S_{i'j'}$ represents the exact opposite of the ratings, where $i', j' \in \{0, 1\}$. Due to disguising mechanism, we cannot simply increment the related variable, S_{ij}. All the values are correct in D' with a probability of θ. Thus, to estimate S_{ij} values, the followings are considered:

$$\begin{aligned} S_{i,j} &= S_{i,j} + (\theta \times \theta) = S_{i,j} + \theta^2 \\ S_{i',j} &= S_{i',j} + ((1-\theta) \times \theta) = S_{i',j} + \theta - \theta^2 \\ S_{i,j'} &= S_{i,j'} + (\theta \times (1-\theta)) = S_{i,j'} + \theta - \theta^2 \\ S_{i',j'} &= S_{i',j'} + (1-\theta) \times (1-\theta) = S_{i',j'} + (1-\theta)^2. \end{aligned}$$

After estimating S_{ij} values from perturbed data, similarity weights between two items can be estimated using Eq. (2). For each item j, the similarities between j and the remaining m-1 items should be estimated. Since $T(X,Y) = T(Y,X)$, $(m-1) \times (m-2)/2$ number of $T(X,Y)$ similarity values are estimated. For each item j, m-1 similarity weights are sorted decreasingly, and the first N items are chosen as neighbors. Thus, neighborhoods for all items from 1 to m are formed. Since all of these computations are conducted off-line, they are not critical for online efficiency. When an a asks prediction for an item (q), rather than using entire items' ratings, only those N items' rates are used for recommendation generation. The quantity of data used in recommendation processes decreases; thus, online performance enhances. Moreover, accuracy is affected. Since $T(X,Y)$ values are estimated from perturbed data and D' is usually a sparse data set, it might not be possible to improve the quality of the referrals, they might even get worse. To prevent accuracy getting worse or even make it better, we propose to use the second preprocessing method.

Increasing Density: One of the challenges that CF schemes with privacy face is data sparsity. Given an entire set of items, customers usually rate limited number of products leading to very sparse data sets. To find a meaningful correlation between two users or items, there should be enough commonly rated items. For a similarity weight between i and j, sufficient number of users should rate both i and j. Although several measures work when there are at least two commonly rated objects, obtaining dependable and true correlations is only possible when there are enough commonly rated items. To enhance reliability and accuracy of similarity weights, we propose to apply another preprocessing method to sparse masked data matrix, D'.

We hypothesize that we might be able to enhance preciseness if we decrease sparsity by filling some of the randomly chosen unrated cells. Those ratings used to fill empty cells should be generated in such a way so that accuracy improves. There are two ways to find them. One is estimating non-personalized ratings or calculate personalized votes from perturbed data off-line. User, item, or overall mean votes are considered non-personalized ratings. Although it is trivial to estimate such default ratings from D, or even from D', they might not represent users' true preferences. Thus, accuracy might become worse. Personalized ratings are more likely to represent customers' proper preferences. We propose to use personalized votes estimated from D' off-line using PPNBC proposed by [5] to fill some of the blank cells. Since they are estimated off-line, online performance will not be affected. However, online performance is expected to become worse with increasing number of inserted ratings because number of votes engaging in CF process increases. Performance gains by applying the first preprocessing method should not be compromised by applying the second one. Thus, we should determine what percent of the empty cells to be filled cautiously so that accuracy becomes stable or even gets better while online performance still enhances. The optimum value can be determined experimentally. The proposed scheme is described, as follows:

1) The server first determines a value p_a, referred to as performance and accuracy parameter. Such value, which happens to give the optimum results for the data holder, can be determined experimentally. The value of p_a should be associated with the density of D'.
2) The data holder then uniformly randomly chooses R over the range $(0, p_a]$. It then uniformly randomly selects R percent of the unrated items' cells (V).
3) After that, it estimates V number of personalized predictions for those chosen unrated items' cells using PPNBC scheme, as explained in [5].
4) It finally fills such cells with corresponding personalized ratings.

Analysis of the Preprocessing Schemes: All preprocessing steps are performed on D'; therefore, they do not affect privacy. However, they definitely have effects on accuracy and online performance. To scrutinize how they influence accuracy, we carry out various experiments based on well-known real data sets collected for CF purposes. In terms of performance, we consider online communication costs (number of communications and amount of data to be transferred), supplementary storage costs, and finally and the most importantly, online computation costs. Due to preprocessing methods, supplementary storage costs are in the order of $O(m \times N)$ because for each item, the best N similar items are stored in a database. The proposed schemes do not change communication costs in terms of number of communications and amount of data to be transmitted. However, as expected, they significantly affect online computation times. This phenomenon is expected because instead of using m-1 items' ratings, only ratings of N of them are used during online computations. Remember that N is a constant and $N < m$. Furthermore, density of D' increases due to filled cells. Since only the best items' ratings are utilized in recommendation process, online time improves by in the order of $O(N/m)$. Compared to the gains due to neighborhood selection, performance overheads due to filled cells are negligible.

V. Experiments

We performed experiments using MovieLens Public (MLP) data set to better show the performance of the proposed schemes. GroupLens at the University of Minnesota collected MLP (www.cs.umn.edu/research/Grouplens). The data set includes ratings of 943 users about 1,682 movies. The votes are discrete and range from 1 to 5. It is a very sparse set because its density is about 2.1%. We applied two different metrics for accuracy evaluation: Classification accuracy (CA) and F-Measure (F1) [8]. We converted numerical ratings into two labels. We labeled items as 1 if

numerical ratings were bigger than 3, or 0 otherwise [8]. We uniformly randomly chose 243 test users. Number of train users utilized in experiments are determined based on the experiments settings. For each active user, we withheld five rated items' ratings, replaced their entries with null, tried to find predictions for them, and compared the results with true ratings. We ran trials using MATLAB 7.6.0 on a computer, which is Intel Core2Duo, 2.2 GHz with 2 GB RAM.

We first performed experiments to verify the effects of varying N values on accuracy and efficiency; and determined the optimum value of N. We uniformly randomly selected 250 users for training. We utilized randomly chosen 243 test users. For each test user, we generated five referrals for five rated items. We varied N from 10 to 1,682. Note that the results for N being 1,682 represent the outcomes without preprocessing. Since the results for values of N bigger than 250 are not promising in terms of accuracy and especially online performance, we displayed the outcomes up to N being 250. We demonstrated CA and F1 values in Fig. 1. As seen from Fig. 1, optimum value of N is 20. When m is 1,682 or without preprocessing, F1 and CA values are 61.04 and 59.34, respectively. With preprocessing, they are 72.95 and 59.42, respectively. In other words, preprocessing significantly improves F1 while CA slightly enhances. Thus, utilizing the best items' ratings improves accuracy.

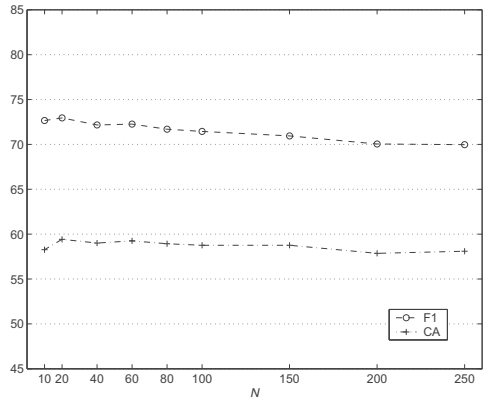

Figure 1. Quality of Recommendations with Varying N Values

We then performed trials to assess the effects of N on online performance. We generated 1,215 predictions using MLP. We estimated online time (T) in seconds required to offer such numbers of referrals. We fixed n at 250. After determining T values with varying N values, we displayed them in Fig. 2 As seen from Fig. 2, gains on efficiency due to preprocessing are significant. We displayed the results up to 250 items. When m is 1,682, T is 5,135 seconds. T values increase with increasing N values. They finally reach their peaks when $N = m$. For optimum N value, T decreases from 5,135 seconds to 104 seconds. Due to preprocessing, online performance enhances by 49.37 times. Thus, preprocessing considerably advances efficiency.

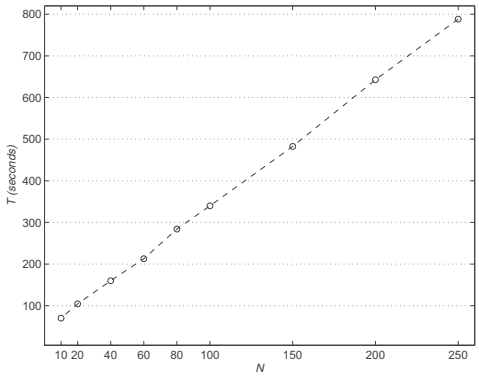

Figure 2. Online Performance with Varying N Values

Table I
F1 WITH VARYING n VALUES

Dataset	n	F1-np	F1-p
MLP	125	60.57	70.54
	250	61.04	72.95
	500	61.57	73.23
	700	60.37	73.52

We conducted trials to show how accuracy and efficiency change with varying n values. We fixed N at its optimum value. We varied n from 125 to 700. To clearly show the effects of the first preprocessing, we also generated referrals without any preprocessing. Since CA and F1 values show similar trends, we demonstrated F1 values only in Table I. Note that F1-p and F1-np represent F1 with preprocessing and without preprocessing, respectively. Similarly, we estimated T values with or without preprocessing, and displayed them in Table II. Again, note that T-p and T-np represent T with preprocessing and without preprocessing, respectively. F1 values notable get better with preprocessing for all n values. F1 enhances by about 11%. As seen from Table I, the relation between accuracy of pure and preprocessed sets do not change as n varies. This shows the applicability and consistency of the preprocessing scheme. T increases with increasing n values for both cases because more users' data are involved in CF process. When we compare T values for both cases, it can be concluded that performance gains are significant due to preprocessing. Due to preprocessing, for all n values, T improves by about 49 times.

Table II
T WITH VARYING n VALUES

Dataset	n	T-np	T-p
MLP	125	2,539	53.62
	250	5,135	104.43
	500	10,246	206.06
	700	14,369	287.43

To show the effects of the second preprocessing, we ran trials while varying p_a from 0 to 12. We associated p_a with the density of MLP. The density (d) of the train set is about

3%. Therefore, we varied p_a from d to $4d$. We fixed N its optimum value, where we set n at 250. We similarly filled some of a's empty cells. After calculating F1 and CA values, we displayed CA values only in Fig. 3. As shown in Fig. 3, increasing p_a from 0 to 3 significantly improves CA. For values of p_a larger than 3, CA enhances slightly. When p_a is 3, CA increases from 59.42 to 62.05. Thus, when we apply both preprocessing methods, CA values even get better.

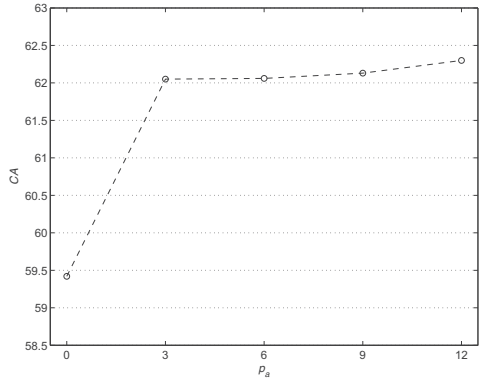

Figure 3. Quality of Recommendations with Varying p_a Values

We hypothesize that increasing density might make efficiency worse because number of ratings involving in the recommendation process increases. To verify this hypothesize and show the effects of the second preprocessing method on efficiency, we estimated T values for 1,215 predictions for different p_a values. We displayed the outcomes in Table III. As seen from Table III, due to our second preprocessing scheme, T faintly becomes worse. When p_a is 3, T gets worse only about 1 second. However, this is for offering 1,215 predictions. Considering one recommendation, performance losses are negligible due to filled cells. Even if we set p_a at 12, T increases less than 3 seconds for 1,215 referrals only. Thus, our proposed preprocessing schemes together considerably enhance efficiency, even accuracy.

Table III
ONLINE PERFORMANCE WITH VARYING p_a VALUES

p_a	0	3	6	9	12
T	104.435	105.122	105.839	106.458	107.096

VI. CONCLUSIONS AND FUTURE WORK

We have presented two preprocessing methods to enhance efficiency and even accuracy of PPNBC scheme. Our first proposed method selects the best similar products to each item so that those items providing the most useful information join in recommendation process. Since the best N items' data are used, efficiency definitely improves, even accuracy becomes better. To increase density so that similarity weights become more reliable and truthful, we used the second preprocessing scheme. We filled some of the empty cells with personalized ratings. This scheme helps CF systems provide more truthful predictions, and makes efficiency slightly worse. We are planning to try other binary similarity measures like Dice, Yule, Jaccard, and so on to determine neighbors. It is also important to choose those cells to be filled in such a way so that accuracy significantly enhances. We are also planning to apply other preprocessing schemes to masked data in order to improve overall performance. In addition to PPNBC scheme, similar privacy-preserving CF systems might suffer with the same problems. There remains work to be done in order to enhance their overall performances by preprocessing.

VII. ACKNOWLEDGEMENTS

This work is supported by Grant 208E211 from TUBITAK.

REFERENCES

[1] A. M. Acilar and A. Arslan. A collaborative filtering method based on artificial immune network. *Expert Systems with Applications*, 36(4):8324–8332, 2009.

[2] R. M. Bell and Y. Koren. Improved neighborhood-based collaborative filtering. In *Proceedings of KDD Cup and Workshop 2007*, pages 7–14, San Jose, CA, USA, August 2007.

[3] L. F. Cranor. 'I didn't buy it for myself' privacy and ecommerce personalization. In *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society*, pages 111–117, Washington, DC, USA, 2003.

[4] F. H. del Olmo and E. Gaudioso. Evaluation of recommender systems: A new approach. *Expert Systems with Applications*, 35(3):790–804, 2008.

[5] C. Kaleli and H. Polat. Providing private recommendations using naïve Bayesian classifier. *Advances in Soft Computing*, 43:168–173, 2007.

[6] B. M. Kim, Q. Li, C. S. Park, S. G. Kim, and J. Y. Kim. A new approach for combining content-based and collaborative filters. *Journal of Intelligent Information Systems*, 27(1):79–91, 2006.

[7] K. Matousek, Z. Kouba, and P. Miksovsky. Data preprocessing support for data mining. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, pages 208–212, Hammamet, Tunisia, 2002.

[8] K. Miyahara and M. J. Pazzani. Improvement of collaborative filtering with the simple Bayesian classifier. *IPSJ Journal*, 43(11), 2002.

[9] H. Polat and W. Du. Privacy-preserving collaborative filtering. *International Journal of Electronic Commerce*, 9(4):9–36, 2005.

[10] P. Symeonidis, A. Nanopoulos, A. N. Papadopoulos, and Y. Manolopoulos. Nearest-biclusters collaborative filtering based on constant and coherent values. *Information Retrieval*, 11(1):51–75, 2008.