Providing Naïve Bayesian Classifier-Based Private Recommendations on Partitioned Data

Cihan Kaleli and Huseyin Polat

Computer Engineering Department, Anadolu University, Eskisehir, 26470 Turkey
{ckaleli,polath}@anadolu.edu.tr

Abstract. Data collected for collaborative filtering (CF) purposes might be split between various parties. Integrating such data is helpful for both e-companies and customers due to mutual advantageous. However, due to privacy reasons, data owners do not want to disclose their data. We hypothesize that if privacy measures are provided, data holders might decide to integrate their data to perform richer CF services. In this paper, we investigate how to achieve naïve Bayesian classifier (NBC)-based CF tasks on partitioned data with privacy. We perform experiments on real data, analyze our outcomes, and provide some suggestions.

1 Introduction

With the evolution of the Internet, the number of users accessing the Internet and the number of products available online are rapidly increasing. To find the most interesting and useful information is imperative. Collaborative filtering (CF) techniques are used for filtering and recommendation purposes. The goal in CF is to predict the preferences of one user (an active user, a), based on a database consisting of a set of votes corresponding to the ratings of users on items [1,6].

To provide more truthful and dependable referrals, data collected for CF purposes should be large enough. It is impossible to produce recommendations from insufficient data. With increasing available data, it is more likely to have enough neighbors and matchings between users. Many online vendors, especially those newly established ones, might not have enough data for CF purposes. If there is a limited number of users, it becomes a challenge to form a large enough neighborhood. Moreover, some vendors might own ratings for a limited number of items; and that makes it harder to compute the similarities between users.

Data might be partitioned horizontally or vertically between various parties, even competing companies. In horizontal partitioning, data owners hold disjoint sets of users' preferences for the same items. In vertical partitioning, they own disjoint sets of items' ratings collected from the same users. Combining horizontally partitioned data (HPD) is helpful when CF systems own a low number of users. Integrating vertically partitioned data (VPD) is advantageous when data holders have ratings for a limited number of items. A referral computed from the joint data is likely more accurate and reliable than the one calculated from one of the disjoint data sets alone. But, due to privacy, legal, and financial reasons, data owners do not want to collaborate and disclose their data to each other.

J.N. Kok et al. (Eds.): PKDD 2007, LNAI 4702, pp. 515–522, 2007.
© Springer-Verlag Berlin Heidelberg 2007

We study how to provide CF services from partitioned data between two parties without greatly exposing their privacy, using the NBC-based CF algorithm proposed by [5]. Our goals are, as follows: First, data holders should not be able to learn the true ratings and rated items in each other's databases. Second, the referrals calculated from partitioned data with privacy concerns should be close to those referrals computed from combined data without privacy concerns. Finally, additional costs introduced due to privacy concerns should be negligible and make it possible to provide referrals to many users in an acceptable time.

2 Related Work

Privacy-preserving data mining on partitioned data has been receiving increasing attention. Vaidya and Clifton present privacy-preserving methods for association rule mining [9], K-means clustering [10], and NBC [11] on VPD. Although such approaches are based on VPD, we study both VPD- and HPD-based CF with privacy using NBC. Privacy-preserving collaborative filtering (PPCF) on VPD problem is discussed in [7]. Unlike their study in which they show how to achieve predictions from numerical ratings, we investigate how to provide CF tasks based on VPD and HPD using binary ratings employing NBC.

Privacy-preserving NBC for HPD is discussed in [4]. They show that using secure summation and logarithm, they can learn distributed NBC securely. Kantarcioglu and Clifton [3] discuss privacy-preserving association rules on HPD. They address secure mining of association rules over HPD while incorporating cryptographic techniques to minimize the shared data. Polat and Du [8] discuss PPCF on HPD using item-based algorithms. Unlike these works, we explore partitioned data-based CF with privacy employing NBC, where users' preferences are represented with binary ratings. Moreover, our schemes can be easily extended to multi-party schemes. We investigate both VPD and HPD-based CF services using NBC, where users' preferences are represented with binary ratings.

In [5], it is proposed to employ NBC for producing recommendations. The users rate items as *like* (1) or *dislike* (0) based on their preferences. a's ratings for items are class labels of the training examples. In the user ratings matrix, other users correspond to features and the matrix entries correspond to feature values. The probability of an item belonging to $class_j$ (c_j), where $j \in \{like, dislike\}$, given its n feature values, can be written, as follows:

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

where both $p(c_j)$ and $p(f_i|c_j)$ can be estimated from training data and f_i corresponds the feature value of the target item (q) for user i. To assign q to a class, the probability of each class is computed, and the example is assigned to the class with the highest probability. Only known features and the data that both users commonly rated are used for predictions.

3 Partitioned Data-Based PPCF Using NBC

Due to privacy concerns, data owners do not want to reveal their data. We propose PPCF schemes to achieve CF tasks using NBC from partitioned data. We assume that the parties communicate through a during providing recommendations online and one of the parties acts as a master site. To derive information, data holders can employ different attacks and the proposed schemes should be secure against such attacks, which can be explained, as follows:

Acting as an active user in multiple scenarios. The party acting as an a employs the same ratings vector during the all recommendation computation processes, manipulating only one rating value each time. Since it gets some conditional probability values computed using its ratings and the users' ratings in the other party's database, the party can easily figure out the differences between such probabilities computed successively. Based on such differences, it is able to find out the ratings of the item for which the rating was manipulated or it can learn whether such item is rated or not.

Bribing. Data holders can offer some incentives or bribery to the users who provided data for filtering services. They then can obtain some data from users and try to derive more information about each other's databases. Since both parties can bribe the same users to derive data or to manipulate each other's data, the required data through such bribed users may not be true or trusted. These users can employ such offers against the other party to get more discounts or coupons. This kind of attack becomes expensive and the derived data through this attack become questionable and doubtful. Therefore, we only consider the "acting as an a in multiple scenarios" attack.

Privacy-Preserving HPD-based Schemes. Two vendors, A and B, hold n_A and n_B users' ratings, respectively, of the same m items, where $n = n_A + n_B$ and n is the number of users. They perform CF tasks using the joint data, which is an $(n_A + n_B) \times m$ matrix, while preserving their privacy. It would be difficult to find out whether two users from different online vendors refer to the same person or not. This can be solved by using some unique identities, which can be exchanged off-line. Since data is partitioned, Eq. (1) can be written, as follows:

$$p(c_j|f_1, \ldots, f_n) \propto p(c_j) \times P_{Aj} \times P_{Bj} = p(c_j) \times \prod_{i=1}^{n_A} p(f_i|c_j) \times \prod_{i=n_A+1}^{n} p(f_i|c_j), \quad (2)$$

where P_{Aj} and P_{Bj} represent the products of conditional probabilities computed from data belonging to A and B, respectively. When B acts as a master site, A computes the required data, P_{Aj} values, and sends it to B through a. HPD-based scheme with privacy can be explained, as follows:

1. a sends her data to both A and B. B computes $p(c_j)$ values.

2. Since both A and B own the feature ratings of q, they can compute the conditional probabilities for classes *like* and *dislike*.

3. A then computes P_{Aj} values and sends them to B through a, while B computes P_{Bj} values.

4. Finally, B can find the probabilities of q belonging to c_j using Eq. (2).

B will not learn the true ratings and the rated items in A's database, because it only gets P_{Aj} values, which are products of n_A values, from A. To further improve privacy, before sending P_{Aj} values to B, A multiplies such values with the same value r_A, where r_A is a random number generated by A. Since both values are multiplied by the same number, the comparison between $p(c_j|f_1, f_2, \ldots f_n)$ values will not be changed for j being *like* or *dislike*.

Privacy-Preserving VPD-based Schemes. A and B hold m_A and m_B items' ratings, respectively, where $m = m_A + m_B$. To make the data sharing possible, the identity of the products should be established across the data holders' databases. This data exchange can be achieved between vendors off-line.

In VPD-based schemes, a sends the corresponding data to A and B. Ratings of q is held by one of the vendors because data is split vertically. Therefore, the party, which does not have q, should conduct the required computations and send the results to the company that owns q; and such party acts as a master site. The party not having q should be able to compute corresponding results required to find the conditional probabilities in such a way to prevent the master site deriving information from its data set. Since class probabilities are known by the master site, it needs to compute the conditional probabilities, as follows:

$$p(f_i|c_j) = \frac{\#(f_i|c_j)}{\#(c_j)}, \tag{3}$$

where $\#(f_i|c_j)$ shows the total number of similarly rated items of c_j as the feature value of q for corresponding user; and $\#(c_j)$ represents the total number of commonly rated items as j, where $j \in \{like, dislike\}$. Since data is partitioned between A and B vertically, the master site gets the results from other party to find the conditional probabilities. Therefore, Eq. (3) can be written, as follows:

$$p(f_i|c_j) = \frac{\#_A(f_i|c_j) + \#_B(f_i|c_j)}{\#_A(c_j) + \#_B(c_j)}, \tag{4}$$

where A and B compute the corresponding parts of $\#(f_i|c_j)$ and $\#(c_j)$ values. Suppose that B owns q. A then should compute $\#_A(f_i|c_j)$ and $\#_A(c_j)$ values for all $i = 1, 2, \ldots, n$ and j being like (1) or dislike (0); and sends them to B. VPD-based scheme with privacy can be explained, as follows:

1. a sends her corresponding data to A and B. a also computes $p(c_j)$ values and sends them to the master site, B.

2. Since A does not know which features of q are known, it computes the corresponding parts of conditional probabilities for all features. Moreover, it computes such values twice, for $f_i = 1$ and $f_i = 0$, because it does not know the feature values of q. However, since $p(f_i = 1|c_j) + p(f_i = 0|c_j) = 1$, it needs to compute $\#_A(f_i|c_j)$ values for each classes for only f_i being 1 or 0. After receiving such values, B selects and/or finds the required data to find the conditional probabilities because it knows the known features of q and their values.

3. Since B gets $p(c_j)$ values from a, it then can figure out how many 1s, 0s, and empty cells are in a's vector. Moreover, B can act as an a in multiple scenarios.

Therefore, A should compute $\#_A(f_i|c_j)$ and $\#_A(c_j)$ values in such a way to prevent B deriving data from its database. To do so, A employs the following steps: It first finds the number of empty cells (m_{ae}) in corresponding part of a's vector. A then uniformly randomly selects a value, R_A, over the range [1, 100]. A then can fill randomly selected R_A percent of these m_{ae} empty cells ($f = m_{ae} \times R_A/100$) with random ratings (1s and 0s). However, with increasing randomness, accuracy diminishes. Instead of filling empty cells with random ratings, A can fill them with default votes (v_ds) of items it holds. Therefore, A finds the v_ds for m_A items it holds. Both parties own the all ratings for items they hold. Therefore, they can compute non-personalized votes for the items they hold without the help of each other. For each item's column, they find the total number of 1s (l) and 0s (d). They then compare l and d values for each item. If $l > d$, then v_d for that item is 1, it is 0 otherwise. Such ratings are computed off-line. A finally fills empty cells with the corresponding v_ds. The number of empty cells to be filled depends on how much privacy and accuracy the parties want. With increasing numbers of filled cells, randomness increases; thus, accuracy diminishes.

4. A then computes the corresponding parts of conditional probability values ($\#_A(f_i|c_j)$ and $\#_A(c_j)$ values) based on a's new or filled ratings vector.

5. Since B does not know how many and which empty cells are selected to be filled, it cannot derive information from the received data. Moreover, since empty cells are filled with non-personalized ratings, which are only known by A, B does not know such values, either.

6. After B gets the required data, it finds the final conditional probabilities, the probabilities for q belonging to c_j, and finally sends the prediction to a.

4 Overhead Costs and Privacy Analysis

The extra storage cost is negligible because A and B need to store v_ds into $1 \times m_A$ and $1 \times m_B$ matrices, respectively. For single predictions, in HPD- and VPD-based schemes, additional number of communications is only 3. Moreover, the amount of data sent also increases. Our HPD-based schemes do not introduce additional computation costs. However, VPD-based schemes introduce extra computation costs due to randomly inserted non-personalized ratings. Computing v_ds is done off-line, which is not critical for overall performance. Our HPD-based schemes are secure. B will not be able to learn the true ratings and the rated items, because it receives two aggregate values, which are products of n_A values. VPD-based schemes are also secure. Even if the master site knows a's ratings, since only commonly rated items between a and other users are used for recommendation computations, it will not be able to derive data from other party's data. Finding v_ds is secure because the parties do not need each other's data to find them. Due to randomly inserted v_ds, B will not be able to derive data from the corresponding parts of conditional probability values.

The master site can guess the randomly selected unrated items. The probabilities of guessing the correct R_A and m_{ae} are 1 out of 100 and 1 out of m_A, respectively. After guessing them, it can compute f. The probability of guessing the f

randomly selected cells among m_{ae} empty cells is 1 out of $C_f^{m_{ae}}$, where C_h^g represents the number of ways of picking h unordered outcomes from g possibilities. Since the master site does not know the v_ds, the probability of guessing the inserted v_d for one item is 1 out of 2. Thus, the probability of guessing the randomly selected empty cells and their ratings is 1 out of $\left(100 \times m_A \times (1/2)^f \times C_f^{m_{ae}}\right)$.

5 Experiments

We performed experiments using Jester and EachMovie (EM) data sets. Jester is a web-based joke recommendation system [2]. It includes continuous ratings ranging from -10 to 10. There are 100 jokes and 17,988 users. The DEC Systems Research Center (www.research.compaq.com/SRC/eachmovie/) collected EM. It contains ratings of 72,916 users for 1,628 movies. User ratings were recorded on a numeric six-point scale, ranging from 0 to 1. We employed classification accuracy (CA) and F-measure (FM) to measure accuracy. We also used coverage as a metrical indicator to show the effectiveness of the NBC-based CF algorithm with combining various amounts of data.

We first transformed the numerical ratings into binary ones as done in [5]. We randomly selected 3,000 and 2,000 users for train and test set, respectively, among those users who have rated at least 50 and 60 items from Jester and EM, respectively. We randomly selected 5 rated items from test users' ratings vectors as test items. The number of users and/or items to be selected varies for various experiment sets. We performed CF tasks using the training sets to provide referrals to test users for test items. We withheld the selected rated items' votes, replaced their entries with null, and tried to predict their values. We compared referrals for them with their withheld votes. We ran the experiments for split sets alone and combined data; and found average CAs and FMs.

We hypothesize that accuracy, privacy, and efficiency depend on various factors. We performed experiments using the disjoint data sets alone and the integrated data. We then compared their outcomes. We varied the number of items (m) and users or features (n) to show how various sizes of disjoint and integrated data sets affect our results. Moreover, since v_ds are inserted randomly selected cells, we performed trials to show how different numbers of randomly selected cells (f) affect accuracy. We also computed computation times. We ran our experiments using MATLAB 7.3.0 on a computer, which is Pentium 4, 3.00 GHz with 1 GB RAM. We performed the following experiments:

It is expected to increase the coverage by integrating split data. Since number of users involving in recommendation process increases, integrating HPD improves coverage. We found coverage values for data owners on data they owned and the combined data. For Jester, when n is 50, the coverage is 99.5% and 100% for split and combined data, respectively. When n is bigger than 100, coverage is 100% for both split and integrated data. For EM, we varied n from 50 to 1,250 to show how coverage changes with combining different sizes of split data. When n is 125, coverage is 63% and 85% for split and combined data, respectively.

We performed experiments with varying n values to show how combining different amounts of HPD affect accuracy and recommendation computation

time (CT) in seconds. We randomly selected training users while varying n from 50 to 1,250 and randomly selected 1,000 test users from train and test sets, respectively. Using our scheme, we found referrals for randomly selected 5 rated items from each test user's ratings vector on disjoint data sets alone and combined data. We compared predictions with true ratings, calculated the average outcomes and displayed them in Table 1, where combined data contains $2 \times n$ users' data. As seen from Table 1, the accuracy of the referrals becomes better both with combined data and increasing n. Although we improve accuracy by combining HPD, time to provide recommendations increases. CTs represent the times to produce 5,000 referrals based on various amounts of data.

Table 1. Overall Performance with Combining Varying Amounts of HPD

		Jester					EM				
n		50	125	300	750	1,250	50	125	300	750	1,250
Split Data	CA (%)	64.86	66.55	67.37	68.07	69.73	70.96	72.95	74.29	74.88	75.14
	FM (%)	63.42	64.77	65.81	66.40	66.64	78.04	79.77	80.85	81.23	81.46
	CT (secs)	15	35	104	345	706	48	127	315	909	1,302
Combined Data	CA (%)	66.14	67.22	69.16	70.15	71.40	73.12	74.62	75.28	75.50	75.86
	FM (%)	64.50	65.76	66.08	67.57	68.12	79.74	81.02	81.56	81.69	81.79
	CT (secs)	21	82	277	926	1,930	83	224	582	1,680	2,986

To show how overall performance changes with integrating varying amounts of VPD, we conducted experiments while varying m. We randomly selected 1,000 training and test users from EM. We computed referrals for randomly selected 5 test items for each test user. We compared the referrals we found with true ratings. We calculated the CAs,the FMs, and the CTs; and showed only the CAs and the CTs in Table 2. As seen from Table 2, accuracy improves with both combining VPD and increasing m. Our results also change with different f values because randomness increases with increasing f. Inserting v_ds into randomly selected cells affects accuracy and the times required to provide referrals. Although accuracy worsens by inserting v_ds, our results are still promising even if all empty cells are filled with non-personalized ratings. With increasing f, accuracy becomes worse and CTs increase. On the other hand, data owners protect their privacy by adding randomness to the private data. Data holders can adjust f to achieve required levels of privacy, accuracy, and efficiency.

Table 2. Overall Performance with Combining Varying Amounts of VPD

	Split Data					Combined Data				
m	200	350	500	650	814	400	700	1,000	1,300	1,628
CA (%)	63.27	65.12	66.16	67.16	67.33	65.96	67.52	68.04	70.94	71.26
CT (secs)	218	452	582	667	811	561	655	896	1,093	1,260

6 Conclusions and Future Work

We have shown that it is still possible to provide accurate recommendations efficiently based on partitioned data between online vendors without greatly jeopardizing their privacy. Our schemes can be easily extended to provide top-N recommendations. We evaluated our schemes in terms of accuracy and computation costs. The experiment results have shown that our outcomes are promising and the proposed schemes allow online vendors to provide accurate referrals efficiently on partitioned data. The proposed schemes for both HPD and VPD can be easily extended to multi-party schemes. We will investigate multi-part schemes in detail. We will explore more attacks and look for solutions to them.

References

1. Breese, J.S., Heckerman, D., Kadie, C.: Empirical analysis of predictive algorithms for collaborative filtering. In: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, pp. 43–52, Madison, WI, USA (July 1998)
2. Gupta, D., Digiovanni, M., Narita, H., Goldberg, K.: Jester 2.0: A new linear-time collaborative ltering algorithm applied to jokes. In: Proceedings of the Workshop on Recommender Systems: Algorithms and Evaluation, 22nd Annual International ACM SIGIR Conference, Berkeley, CA, USA (August 1999)
3. Kantarcioglu, M., Clifton, C.: Privacy-preserving distributed mining of association rules on horizontally partitioned data. Transactions on Knowledge and Data Engineering 16(9), 1026–1037 (2004)
4. Kantarcioglu, M., Vaidya, J.S.: Privacy-preserving naive Bayes classifier for horizontally partitioned data. In: Proceedings of the IEEE ICDM Workshop on PPDM, pp. 3-9, Melbourne, FL, USA (November 19-22, 2003)
5. Miyahara, K., Pazzani, M.J.: Improvement of collaborative filtering with the simple Bayesian classifier. IPSJ Journal 43(11) (2002)
6. Pennock, D.M., Horvitz, E., Lawrence, S., Giles, C.L.: Collaborative filter ing by personality diagnosis: A hybrid memory- and model-based approach. In: Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence, pp. 473–480, Stanford, CA, USA, June 30-July 3 (2000)
7. Polat, H., Du, W.: Privacy-preserving collaborative filtering on vertically partitioned data. In: Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases, Porto, Portugal (October 3-7, 2005)
8. Polat, H., Du, W.: Privacy-preserving top-N recommendation on horizontally partitioned data. In: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, Paris, France (September 19–22, 2005)
9. Vaidya, J.S., Clifton, C.: Privacy-preserving association rule mining in vertically partitioned data. In: Proceedings of the 8th International ACM SIGKDD Conference, pp. 639–644 (July 23-26, 2002)
10. Vaidya, J.S., Clifton, C.: Privacy-preserving k-means clustering over vertically partitioned data. In: Proceedings of the 2004 SIAM Conference on Data Mining, Lake Buena Vista, FL, USA (April 22–24, 2003)
11. Vaidya, J.S., Clifton, C.: Privacy-preserving naÏve Bayes classier for vertically partitioned data. In: Proceedings of the 9th International ACM SIGKDD Confer- ence, pp. 206–215, Washington, DC, USA (August 24-27, 2003)