On the Privacy of Horizontally Partitioned Binary Data-Based Privacy-Preserving Collaborative Filtering

Murat Okkalioglu[1], Mehmet Koc[2], and Huseyin Polat[3](✉)

[1] Department of Computer Engineering, Yalova University, 77200 Yalova, Turkey
murat.okkalioglu@yalova.edu.tr
[2] Department of Electrical and Electronic Engineering,
Bilecik Seyh Edebali University, 11210 Bilecik, Turkey
mehmet.koc@bilecik.edu.tr
[3] Department of Computer Engineering, Anadolu University, 26555 Eskisehir, Turkey
polath@anadolu.edu.tr

Abstract. Collaborative filtering systems provide recommendations for their users. Privacy is not a primary concern in these systems; however, it is an important element for the true user participation. Privacy-preserving collaborative filtering techniques aim to offer privacy measures without neglecting the recommendation accuracy. In general, these systems rely on the data residing on a central server. Studies show that privacy is not protected as much as believed. On the other hand, many e-companies emerge with the advent of the Internet, and these companies might collaborate to offer better recommendations by sharing their data. Thus, partitioned data-based privacy-persevering collaborative filtering schemes have been proposed. In this study, we explore possible attacks on two-party binary privacy-preserving collaborative filtering schemes and evaluate them with respect to privacy performance.

Keywords: Privacy · Collaborative filtering · Binary data · Attack scenarios

1 Introduction

Information technologies have been developing at a great pace and providing comfort for everyone to perform their everyday tasks. The Internet is a great medium to offer countless services due to its ease of access by people from every different taste and class. This trend of moving people's routines to the Internet has made e-commerce very attractive. However, attracting new customers or recommending the right products for the existing ones is a competitive task for an online company to survive in the market. Collaborative filtering (CF) is a technique for providing referrals. It was proposed with the Tapestry project [7]. CF systems aim to produce recommendations for an active user (a) who asks for a prediction by collecting user opinions (ratings) and considering these opinions

J. Garcia-Alfaro et al. (Eds.): DPM and QASA 2015, LNCS 9481, pp. 199–214, 2016.
DOI: 10.1007/978-3-319-29883-2_13

	Apple	Orange	Banana	Strawberry	Peach	Kiwi
Bill	0	-	0	-	0	0
Joe	1	0	1	-	-	0
Alice	1	0	1	0	-	0
Bob	-	-	0	-	0	1
Kevin	0	?	0	1	-	1

Fig. 1. A typical CF system

with similar interest for referrals for a [11,25]. A typical CF scheme is composed of an nxm user-item matrix, where n different users express their opinions on m different items as seen in Fig. 1. User opinions can be expressed using binary ratings showing if the item is *liked* (1) or *disliked* (0).

True participation is critical for CF schemes to produce accurate referrals. However, users might be unwilling to participate in them due to privacy concerns. User data is a valuable asset and might be subjected to different threats such as unsolicited marketing like unwanted e-mails or phone calls, government surveillance, price discrimination, or even subpoena [5,6]. Privacy-preserving collaborative filtering (PPCF) addresses privacy. If a user believes that individual privacy is promised by a CF scheme, she will be eager to be part of it and to provide true ratings. On the other hand, privacy cannot be considered the sole purpose of PPCF. The schemes must also produce accurate referrals. Thus, the objective of PPCF schemes can be considered as balancing equilibrium between privacy and accuracy without neglecting performance [3]. Recently, some scholars argue that PPCF schemes do not protect individual privacy as promised [4,29]. Zhang et al. [29] propose reconstruction methods to recover numerically rated data, which is perturbed by randomization [21]. Calandrino et al. [4] study inference attacks on online CF systems.

Data sparsity is one of the main challenges for CF systems [12,26]. In general, CF systems operate on a large user-item matrix and this matrix is usually extremely sparse because users vote on a relatively small number of items based on their interest. Therefore, companies might lack accurate recommendations if they have insufficient data. The missing ratings of a user-item matrix can be compensated if data is partitioned between parties. When data is partitioned between any two parties, the parties can fill some missing ratings and better recommendation can be achieved. Data can be partitioned horizontally, vertically, or arbitrarily. In horizontally partitioned data (HPD) schemes, two parties having ratings for the same set of items by different users share their data while ratings for different items from the same set of users are shared in vertically partitioned data (VPD). HPD- and VPD-based schemes are displayed in Fig. 2. The users are displayed by $<u_1,\ u_2,\ .....,\ u_7,\ u_8>$ and the items are displayed as $<i_1,\ i_2,\ .....,\ i_{15},\ i_{16}>$. In arbitrarily partitioned data, both parties share arbitrarily ratings for items and users.

A's data set

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}	i_{15}	i_{16}
u_1	0	1	-	-	-	-	1	-	0	-	-	-	1	-	1	-
u_2	-	1	-	1	-	-	-	0	-	-	-	-	0	-	0	0
u_3	-	-	0	1	-	1	-	-	0	-	0	1	-	-	-	-
u_4	0	-	-	-	-	-	-	0	-	1	-	-	1	-	1	-

B's data set

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}	i_{15}	i_{16}
u_5	1	-	-	-	0	-	0	-	-	0	-	1	-	0	0	-
u_6	0	-	-	1	-	-	-	-	1	-	0	1	-	-	-	0
u_7	-	-	1	-	1	-	0	-	-	0	1	-	-	1	0	-
u_8	0	1	-	-	-	0	-	1	-	0	-	0	-	1	-	0

a) Horizontally Partitioned Data - HPD

A's data set

	i_1	i_2	i_3	i_4	i_5	i_6	i_7	i_8
u_1	0	1	-	-	-	-	1	-
u_2	-	1	-	1	-	-	-	0
u_3	-	-	0	1	-	1	-	-
u_4	0	-	-	-	-	-	-	0
u_5	1	-	-	-	0	-	0	-
u_6	0	-	-	1	-	-	-	-
u_7	-	-	1	-	1	-	0	-
u_8	0	1	-	-	-	0	-	1

B's data set

	i_9	i_{10}	i_{11}	i_{12}	i_{13}	i_{14}	i_{15}	i_{16}
u_1	0	-	-	-	1	-	1	-
u_2	-	-	-	-	0	-	0	0
u_3	0	-	0	1	-	-	-	-
u_4	-	1	-	-	1	-	1	-
u_5	-	0	-	1	-	0	0	-
u_6	1	-	0	1	-	-	-	0
u_7	-	0	1	-	-	1	0	-
u_8	-	0	-	0	-	1	-	0

b) Vertically Partitioned Data - VPD

Fig. 2. Partitioned data schemes in CF

This study focuses on the HPD-based PPCF scheme with binary ratings proposed by Polat and Du [22,24]. The authors propose PPCF schemes with *threshold* or *best-N* neighbors based methods. Kaleli and Polat [14] propose a two-party binary PPCF scheme based on naïve Bayes classifier. In these studies [14, 22,24], two-party binary PPCF schemes are only examined in terms of accuracy. However, attacks that can be applied to these schemes are not studied. In this study, we *discuss attack scenarios applicable to HPD schemes in* [22,24] *and perform attacks against HPD-based two-party binary PPCF schemes to examine how much privacy is offered by these schemes.*

The rest of the paper is organized as follows. Section 2 presents related work in the literature. Section 3 discusses the two-party PPCF schemes to inform the reader. In Sects. 4 and 5, we present possible attack scenarios and perform our experiments, respectively. In Sect. 6, final comments and future works are presented.

2 Related Work

Privacy is an important issue for customers. Since customers worry about the privacy, they may not want to share their ratings or they provide fake ratings

to hide their true profiles. Although privacy-preserving methods offer privacy measures to protect individual privacy, researchers argue that they are not successful as promised [16,17]. The authors in [16,17] show that the original data can be closely estimated from the perturbed data using *spectral filtering* (SF). They propose random matrix based SF technique to reveal the data from the perturbed one. Huang et al. [13] propose a data reconstruction technique based on *principal component analysis* (PCA) to derive the correlation structure of the original data. If noise is added to the original data, the method tries to remove the noise using the correlation structure of the original data. This procedure ends with the values which are close to the original ones. Guo et al. [10] give a lower bound and an upper bound for the reconstruction error when the SF techniques are applied to derive the original data. Using these bounds the attackers can decide the quality of their attacks [10]. Agrawal and Aggarwal [1] propose a data reconstruction algorithm based on *expectation maximization* that converges the maximum likelihood estimate of the original distribution, which is an extension of the study presented in [2].

Guo et al. [9] propose an attack based on *interquantile-range*, which is used to measure the amount of the variability of the random variable. Attackers can use this approach to estimate the range of each individual data with a confidence interval. In [29], the authors propose a reconstruction technique targeting a numerically rated PPCF scheme proposed in [21]. Their method use *k-means* clustering and *singular value decomposition* (SVD) to derive original ratings from perturbed data. Calandrino et al. [4] devise different attacks on CF systems. They utilize auxiliary data and make inferences about target live CF systems. They propose attacks including *knn-based* attack, which is also exploited in this study, based on temporal changes of public outputs of the commercial web sites. In [19], the authors propose a reconstruction technique to recover binary rated data masked by using randomized response techniques [27] offered by Polat and Du [23]. The same data disguising method also introduce some fake ratings to enhance privacy and the scholars in [20] analyze this scheme in order to discover fake binary rated items.

In all of the above studies, the attacks proposed by the authors to recover the original ratings are for single party. Unlike these methods, we examine three different attack scenarios to the HDP-based two-party binary PPCF schemes proposed by Polat and Du [22,24] to discover the ratings.

3 Horizontally Partitioned Binary Data-Based Privacy-Preserving Collaborative Filtering

Dense ratings are important factors for PPCF systems in terms of accurate recommendations since the denser the ratings are, the likelier it is to discover the relationships among items or users. Small companies or start up might have sparse ratings due to the limited number of users [3,15,24,28]. Even some companies might be interested in migrating to another business segment and they

might be in need of the related item ratings for the new market. Such companies could find a partner to exchange their ratings without neglecting privacy promises. Essentially, companies can overcome the problem of data sparsity if they share their data with a partner. One party could have ratings for a set of items/users that the collaborating party does not have. Thus, data sharing will be mutually beneficial for both parties to produce better recommendations on the joint data. There are various partitioned data-based PPCF schemes with binary ratings like HPD- and VPD-based ones [14,22,24]. In this study, various attack scenarios have been performed to test the privacy offered by the HPD-based PPCF schemes [22,24]. Before giving the details of these scenarios we would like to introduce these binary data-based PPCF schemes. Assume that one of the parties is A and the other party is B.

3.1 Preliminaries

In general, two-party binary ratings-based PPCF schemes proposed by Polat and Du [22,24] provide *top-N* (TN) results for users. TN schemes return the first N items matching with earlier user preferences as a referral. Their schemes (both HPD- and VPD-based) find a set of neighbors for a based on similarities between a and other users. Remember that a is an active user who is looking for a prediction. Additionally, Polat and Du [22,24] use a protocol to compute similarities without jeopardizing the privacy of the users. In this subsection, the similarity metric and the protocol used to privately compute the similarities are given as a preliminary to the PPCF schemes.

Similarity Metric. This metric is used to calculate how much a user is similar or dissimilar to a. The metric used by Polat and Du [22,24] is based on the difference of similarly rated items from dissimilarly rated items over the commonly rated items. They use this similarity metric to identify the neighbors. The formulation of this metric is given below:

$$W_{au} = \frac{t\,(R_s) - t\,(R_d)}{t\,(R)} \quad . \tag{1}$$

in which u is the related user whose similarity between a is calculated, W_{au} is the similarity between u and a, $t(R_s)$ is the number of similarly rated items, $t(R_d)$ is the number of dissimilar rated items, and $t(R)$ is the number of the commonly rated items ($t(R) = t(R_s)+t(R_d)$). W_{au} can range between 1 and -1. If $W_a u > 0$, then u and a are similar, otherwise they are dissimilar. Upon determining W_{au} values, the neighbors are selected based on the criterion (*best-N* or *threshold*) in the relevant PPCF scheme given in the following subsections. If a dissimilar user is selected as one of the neighbors, her ratings are reversed discussing that dissimilar users have a negative correlation and would vote opposite for the same item. After selecting the neighbors, Polat and Du [22,24] find the number of *likes* (l_j) and *dislikes* (d_j), where j is the item number, among the selected neighbors. Then $ld_j = l_j$ - d_j is calculated. If $ld_j > 0$, then the item will be liked by a. Otherwise, it will be disliked by a.

Private Similarity Computation Protocol PSCP. PSCP is used to compute the similarities without compromising a's privacy [22,24]. Assume that M is the number of rated items by a. This protocol has two cases, where the first case fills some unrated entries while the second case removes some rated entries. The first case is applied if M is initially less than $m/2$; remember that m is the number of items. In this case, unrated entries of a is filled with R_{Aa} number of items, where R_{Aa} is a uniform random number picked from the range (1, m-M). The chosen unrated entries are filled with default votes, where default votes are estimated for each cell based on ld_j values. In the second case, if M initially is greater than $m/2$, then a uniform random number, R_{Ar} is picked from the range (1, M), and R_{Ar} number of ratings are canceled (removed) from a's rating vector.

3.2 HPD-based Privacy-Preserving Schemes

Threshold-Based. This method is proposed by Polat and Du [22,24]. Once a sends a query, which includes the set of unrated items (N_a) that a is looking for TN, both parties start to collaborate to produce TN predictions. Collaborating parties calculate the similarities between its users and a. They select their neighbors on a predefined threshold (τ_n). The scheme is explained as follows:

1. a sends a query to both parties.
2. A calculates similarities between its users and a. Then, the neighbors are selected based on τ_n. A adds a uniform random number to τ_n to prevent B from learning τ_n.
3. A calculates $ld_{Aj} = l_{Aj} - dAj$ values for items $j = 1.N_a$ where N_a is the set of queried items from selected neighbors. Recall that a's query includes N_a unrated items among which she is looking for a prediction. A sends ld_{Aj} values to B through a.
4. B calculates similarities of its users and a; and selects neighbors based on τ_n. Then, B finds $ld_{Bj} = l_{bj} - d_bj$. It computes final ld_j values by adding corresponding ld_{Aj} and ld_{Bj} values. Then, B sorts them. TN of the sorted items are sent to a.

Best N_n-based. This scheme selects the best N_n number of neighbors after the similarities have been calculated [22,24]. PSCP is applied to perform the similarity computations privately. Details of the scheme are as follows:

1. a sends a query to both parties.
2. A finds the similarities (W_{Aua}) between its users, u, and a using PSCP. The similarities (W_{Aua}) are permuted by a function (π_A) only known to A and they are converted to their absolute values ($|W_{Aua}|$) and sent to B through a.
3. B calculates the similarity values W_{Bau} and finds the best N_n neighbors among all neighbors including A's.
4. B calculates ld_{Bj} values for N_a items and sends them to A with the selected neighbors of A.
5. A finds ld_{Aj} values for N_a items and calculates final ld_j values. It then finds TN referrals and sends them to a.

4 Attack Scenarios

Some attack scenarios can be utilized to derive information from two-party binary PPCF schemes; such schemes apply privacy measures arguing that they annihilate possible attacks [22,24]. By privacy measures, we mean the effort being made to prevent the other party from learning unintended information. Privacy has two aspects [23]; (1) to preserve the rating value of users, (2) to disguise if an item is rated or unrated. We will call them as the first and the second aspect of privacy, respectively. In this section, possible attack scenarios applicable to such schemes are discussed. We also explain the reasons of the applicability of each individual attack if no measures are taken in terms of privacy. Moreover, which aspect of privacy can be exploited by each attack will be given. In Sect. 6, we discuss how privacy measures are effective by comparing the outcomes to both cases.

4.1 Acting as an Active User in Multiple Scenarios

Any party who wants to derive data from the other party constructs a rating vector and acts as an active user in multiple scenarios. The active party employs the same rating vector for multiple times by only manipulating one entry at a time. Tracking changes in the similarity values, or interim results between each query, the active party can figure out the rating for the manipulated entry. Note that the change in interim result occurs due to the manipulated entry. Repeating this process for each rating will help the active party learn true ratings for all items in the rating vector. This attack discloses actual ratings of the target users and if an item is rated; therefore, it exploits the first and the second aspect of privacy.

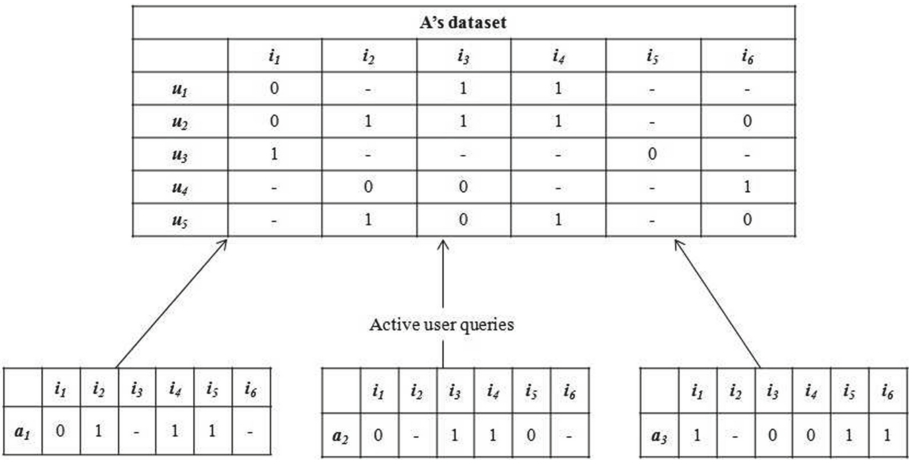
Threshold-based scheme has only one interaction between parties on aggregate values of ld_{Aj} and it does not disclose any individual information about users. However, this attack tracks temporal changes of individual variables on responses for altered queries. Thus, it is not applicable for the *threshold-based* scheme. The *best-*N_n approach includes privacy measures to prevent B from discovering information. Similarities calculated by A are sent to B. If there are no privacy measures, this transaction would be subjected to this attack by tracking changes in the similarities each time a rating is manipulated. First, we eliminate the measures to see the success of this attack without privacy measures. Then, privacy measures are introduced to see how much effective they are. Note that B acts as an active user and sends multiple queries to A to derive data.

4.2 Perfect Match Attack

Polat and Du [22,24] discuss only *acting as an active user* attack. However, their schemes make frequent interactions between parties by exchanging the similarities or the partial similarity values required for the similarity calculation. Exchanging such values might cause information disclosure. In a typical

scenario, where B is the master site and there is no privacy concern, A calculates similarities and sends them to B. If the similarity between any user of A and a is 1 or -1, this means that these users are perfectly matched and such similarities will be called *perfect match*. Every commonly rated entry of a *perfect match* either exactly matches if similarity is 1 or oppositely matches if similarity is -1. By 'opposite', bitwise NOT operation is implied. However, there might be some unrated entries from both sides; those entries are not taken into consideration while calculating similarities by Polat and Du [22,24]. Remember that the similarity metric is based on the similarly and dissimilarly ratings and having a *perfect match* means that the commonly rated entries are rated identically. Thus, B concludes that the corresponding user of a *perfect match* in A's data set has either similarly voted or not voted of a rating in a's vector if the similarity is 1. If the similarity is -1 in a *perfect match*, B concludes that the corresponding user voted opposite or not voted.

Figure 3 displays three different queries from B to A. Pay special attention to u_1, a_1 and a_2. The similarity between u_1 and these two active users are 1, although they have different ratings. Once B finds out that u_1 and a_1 is a *perfect match*, it concludes that rated entries of a_1 are either rated identically or not rated by u_1. u_1 and a_2 is another *perfect match* and the same assumption holds. Remember that a_1 and a_2 have different rating vectors. For example, i_5 is rated 1 and 0 by a_1 and a_2, respectively. Thus, B can figure out that i_5 is unrated by u_1 because each commonly rated item of a *perfect match* needs to be identical. In this case, rating vectors of a_1 and a_2 have different values for i_5. Different queries with a *perfect match* reveal unrated entries of the related user if there are opposite overlapping ratings in the queries.



A's dataset	i_1	i_2	i_3	i_4	i_5	i_6
u_1	0	-	1	1	-	-
u_2	0	1	1	1	-	0
u_3	1	-	-	-	0	-
u_4	-	0	0	-	-	1
u_5	-	1	0	1	-	0

	i_1	i_2	i_3	i_4	i_5	i_6
a_1	0	1	-	1	1	-

	i_1	i_2	i_3	i_4	i_5	i_6
a_2	0	-	1	1	0	-

	i_1	i_2	i_3	i_4	i_5	i_6
a_3	1	-	0	0	1	1

Fig. 3. An example of different queries from B to A

This attack discloses two privacy breaches: (1) even a single *perfect match* reveals that the actual rated value of the target item or it is unrated; (2) unrated entries can be disclosed if multiple *perfect matches* occur. To be more precise for the first vulnerability with our example, when a *perfect match* is captured between u_1 and a_1, B finds that i_1, i_2, i_4, and i_5 are either rated 0, 1, 1, and 1 by u_1, respectively, or they are unrated. When a_2 is queried, the second vulnerability occurs and i_5 is discovered to be unrated. *Perfect match* threatens the first and the second aspect of privacy because it might both disclose the possible rating value and if an item is rated or not.

This attack is applicable for the $best-N_n$ scheme. Note that the similarities of A are sent to B and B can exploit this information. However, the *threshold-based* scheme does not exchange such information; so, this attack is not applicable.

4.3 *knn*-based Scenario

knn-based attacks target the CF schemes that are based on neighboring approach if a history of a user is known [4]. It is claimed that if the history of ratings of a user is known, then the attacker can insert k fake users into the CF system. When a recommendation is queried for any of the fake users, it is highly possible that the neighbors will be selected among the *k-1* fake users and the target user. Thus, any item in the prediction, which is not in the known history of the target user, will be probably coming from the rating of the target user. *knn*-based attack only discloses the information of whether an item belongs to the target user or not, so it poses a threat to the second aspect of privacy.

This attack holds if the related scheme is based on selecting k neighbors. *Threshold-based* and $best - N_n$ schemes utilize the best neighboring approach. But, in the *threshold-based* scheme, we cannot determine how many neighbors will be selected, because neighborhood is determined on τ_n value. There is not a single k value to determine the number of neighbors. Hence, this attack might be applicable for the *threshold-based* scheme if some numbers of fake users are inserted. On the other hand, it is known that how many neighbors will be picked in the $best - N_n$ scheme. k fake users can be inserted into the data set. Being able to insert such fake users makes this attack possible.

5 Experiments

Experiments are performed using MovieLens Million (MLM) and Jester data sets. MLM was collected by GroupLens research group[1]. It contains 1,000,209 ratings from 6,040 users and 3,952 items. The density of the data set is about 4.2 %. Ratings are on a 5-star scale. Jester is a web-based joke recommendation system[2] [8]. It is a fairly dense data set (72.47 %) and ratings are continuous scale between -10 and 10.

[1] www.cs.umn.edu/research/GroupLens.
[2] http://eigentaste.berkeley.edu/dataset/.

Precision and *recall* have been used as evaluation criteria in this study. *Precision* is the percentage of classified items that are relevant while *recall* is the percentage of relevant items. *Precision* and *recall* value are calculated cumulatively.

This study focuses on binary rated PPCF schemes. Ratings are converted to binary scale based on a threshold [18]. MLM ratings greater than 3 are marked as *like* (1) and *dislike* (0) for the rest. Jester is labeled 1 if ratings is greater than 2.0 and labeled 0 if ratings are less than or equal to 2.0. Experiments are repeated 100 times. 500 and 2,000 users are picked randomly from MLM and Jester, respectively. Jester users are picked among the users with at least 60 % rated cells for a denser data set. During our experiments, we permute users and items in HPD, to eliminate density advantages of any party.

5.1 Experiments

Experiment I. This experiment is based on acting as an active user in multiple scenarios. Our aim in this experiment is to show how privacy measures help protect individual privacy when compared to the total disclosure.

An active query is created with the random density between 10 % and 50 % and it is sent to A. As privacy measures, PSCP and the absolute values of similarities are applied. Remember that PSCP fills or cancels (removes) some ratings based on the density of incoming active query. Although the subsequent active queries are different from the first (original) active query with only one cell, some ratings are either inserted into or removed from the active query by PSCP and it invalidates this assumption in practice. In addition, PSCP makes it very difficult to observe empty ratings in A because altering the active query makes every subsequent query almost independent from each other although they are originally different from each other with only one cell. Moreover, taking absolute values for similarities misleads this attack for negative valued similarities. For example, a possible decrease in the similarity for negative values will be reflected as an increase to the absolute values of the similarity and vice versa. We expect that the result will be worse compared to scheme when privacy is not applied due to the reasons mentioned. Table 1 demonstrates the results associated with the experiment. Note that *NP* means no privacy measure is applied while *WP* means scheme is applied with privacy measures and d is the density.

As seen from Table 1, recall values are worse than precision values in NP cases. This is because recall indicates the fraction of correctly guessed items by

Table 1. Acting as an active user attack

	Best-N_n NP		Best-N_n WP	
	MLM d = 4.3 %	Jester d = 85.0 %	MLM d = 4.4 %	Jester d = 4.4 %
Precision	1.000	1.000	0.025	0.448
Recall	0.248	0.341	0.162	0.124

the attack. In the attack scenario, the active query is created with the random density between 10 % and 50 %. This attack does not try to guess the actual ratings for all entries; therefore, we cannot expect full recall results. As one can see in Table 1, precision and recall values are poorer in WP compared to NP. As discussed in the previous paragraph, PSCP and absolute value while calculating similarities help users protect their privacy. Recall results are slightly better for MLM data set compared to Jester in WP case although precision results for MLM is considerably worse. This means that this attack predicts the similar fraction of relevant items from both data sets either dense or sparse. On the other hand, precision value is worse for the sparse data set (MLM) because this attack tries to guess actual values of the items in the active query and those entries are possibly empty in sparse data set. As detailed in the previous paragraph, observing empty ratings are difficult in WP case. Thus, it makes precision to deteriorate when sparsity increases.

In the *best-N_n* approach, the similarities are permuted to prevent B from learning whose similarities it is dealing with. This case has not been considered for this experiment. If this case is applied, we need to consider the probability of guessing correct similarities based on users, which is 1 out of $n_A!$, where n_A is the number of users A holds.

Experiment II. In this experiment, *perfect match* attack has been performed. The density of active query is an important factor in this attack. If the active query is dense, capturing a *perfect match* becomes difficult in comparison with a sparse active query because every common rating between the active query and users' vector must match. Therefore, we first examine the best density rate among 5 %, 10 %, 20 %, and 50 % without privacy measures. We find out that 5 % density rate achieves the best results. The density rate for active queries in this experiment is 5 % for the following tests.

First, we evaluate the performance of this attack without privacy measures. Then, the change in accuracy is compared to schemes when privacy is applied. We randomly fill 5 % of the query vector with 0 s and 1s. This attack has been executed for 100 different subsequent queries for each experiment and averages are listed in the relevant tables.

Remember that *perfect match* attack reveals two kinds of privacy breaches: (1) either the actual value of the relevant item or it is unrated; (2) the unrated entries. Then, the first breach means that if the relevant item is rated, the correct rated value is discovered. We perform a coin toss to determine the actual value of the relevant item or to mark it unrated in the case of the first privacy breach. The second breach reveals that an entry is not rated. Thus, two kinds of precision and recall have been calculated for this experiment. The first is precision and recall values for the actual value of items, $prec_1$ and rec_1 (the first privacy breach); the second is the precision and recall value to determine unrated items, $prec_2$ and rec_2 (the second privacy breach). Table 2 displays the results.

Unlike the previous experiment, precision results are highly correlated with the density in NP case. $Prec_1$ results for MLM, which is very sparse, is about 0.02 while

Table 2. Perfect match attack

	$Best - N_nNP$		$Best - N_nWP$	
	MLM d=4.1 %	*Jester d=84.8 %*	*MLM d=4.2 %*	*Jester d=85.0 %*
$prec_1$	0.023	0.868	0.005	0.371
rec_1	0.106	0.297	0.003	0.003
$prec_2$	0.985	0.238	0.991	0.393
rec_2	0.298	0.521	0.027	0.019

it goes up around 0.87 when data set is fairly dense, Jester. This attack reveals limited ratings with sparse data set on $prec_1$ metric. In a sparse data set, a *perfect match* can be captured with even a single commonly rated item and the remaining items are determined as 1 or 0 based on the active query ratings. However, the most of these ratings are probably unrated due to the sparsity of MLM. On other hand, the dense data set, Jester, performs well and it is more probable that items marked as 0 or 1 by coin toss process is indeed rated.

In terms of discovering unrated items (the second privacy breach); *perfect match* attack achieves better $prec_2$ results with MLM while Jester has better rec_2 results. In this attack, if a cell is marked as unrated, it is guaranteed to be unrated unless privacy measures are applied. However, we perform a coin toss to decide the actual rating of an item and mark some items as unrated based on this coin toss process. This process of determining actual ratings deteriorates the $prec_2$ results because we mark possibly some rated items as unrated. $Prec_2$ is still very high for MLM because it is fairly sparse and marking a rated item as unrated does not affect the metric seriously. On the other hand, marking a rated item as unrated is more expensive for a dense data set for $prec_2$ metric because number of unrated cell is limited. Rec_2 is worse for MLM; this is because items classified as unrated do not form a large number when compared to the number of unrated cells in MLM due to the sparsity. Remember that *d=4.1 %* is for MLM, so MLM is filled with unrated items for about 96 % in this experiment. As mentioned earlier, recall result are not very informative.

As a privacy measure, PSCP and the absolute values of similarities are applied. Only 5 % of the active queries are filled for this experiment, the active queries will always be filled randomly with fake entries from the range (1, *m-m*.05*). In our initial tests, to pick the density of the active query, we see decrease in the result for denser active queries; therefore, decrease in $prec_1$ and rec_1 are expected in WP case. Also, taking absolute values preserves the sign information of *perfect matches* which allows discovering only positive *perfect matches*. Notice WP part in Table 2 that the privacy is almost preserved for MLM and there is a significant decrease for Jester for the first privacy breach ($prec_1$, rec_1). The second privacy breach is mostly affected by coin toss process and taking absolute values. If PSCP is applied alone without these two factors, the second aspect of the privacy would produce full $prec_2$ results. $Prec_2$ results seem slightly better than NP case in both data sets. Since less perfect matches occur with privacy

measures, the coin toss process nominates less unrated items. Practically, 50 % of unrated items determined by coin toss could be rated; thus, $prec_2$ results are better compared to NP case. Conversely, there will be fewer items marked unrated causes poor rec_2. Better recall results could be achieved by more true predictions in number. Note that permutation is not applied in this experiment as in the previous one; the probability of guessing correct similarities would be 1 out of $n_A!$ if permutation is applied.

Experiment III. This experiment performs *knn*-based attack on both NP and WP cases. This attack requires the history of a user. However, HPD schemes do not have such history inherently. To overcome this issue, we assume that the attacker has the half of the history of a targeted user. This attack can be utilized for the $best - N_n$ and *threshold-based* scheme because these schemes use neighborhood for determining the similarities; however, the number of neighbors is not determined for the *threshold-based* scheme beforehand. They are picked based on τ_n. k fake users are required to be inserted in *knn*-based attack, so we insert 200 fake users to accomplish *knn* attack for the *threshold-based* schemes. In this experiment, we are looking for predictions for randomly selected 100 users and 30 items. Final precision and recall values are calculated by taking average of precision and recall values for each user.

In this experiment, we expect the denser data set, Jester, will outperform the sparse data set, MLM. When a data set is sparse, an incoming history query might match some unrelated similarities. The similarity computation only considers the commonly rated entries and even a single match will produce a *perfect match* just like inserted fake users. This means that there will be t users who have similarity of 1 (perfect match), where t $\gg$ k. Notice that our implementation selects the first k users as neighbors. Therefore, inserted fake users could fail to be picked as neighbors. However, a denser data set is more specific while determining the similarities because a denser user vector will probably have more common items with an incoming query so that the resulting similarity is more reliable. More commonly rated items means that it is more probable to have some dissimilar items. As a result, the best k users have more chances to be matched among the inserted k fake users. It is less likely to select unrelated users with denser data sets in comparison with sparse data sets.

Table 3 displays results of this attack. Notice that the results confirm our assumption that denser data sets outperform sparse data sets for all schemes in precision. On one hand, our assumption for denser and sparse data sets holds; on the other hand, it is clear in Table 3 that privacy measures do not have a direct negative effect on the results. Remember that PSCP and absolute values are applied for HPD schemes. It is, prima facie, an interesting result. Although PSCP could affect similarity between a and genuine users, the similarities between k fake users and a are not affected by PSCP because fake users have the exact same vector with the original active query. Items inserted into or removed from the active query does not alter commonly rated items between k fake users and a. Similarities between a and k fake users are still 1. PSCP does not have an

Table 3. *knn*-based attack

	NP				*WP*			
	Best $- N_n$		*Threshold-based*		*Best* $- N_n$		*Threshold-based*	
	MLM	*Jester*	*MLM*	*Jester*	*MLM*	*Jester*	*MLM*	*Jester*
Precision	0.212	0.749	0.278	0.717	0.251	0.731	0.277	0.750
Recall	0.081	0.515	0.134	0.504	0.114	0.506	0.139	0.526

important effect in terms of privacy for this attack, because it does not seriously affect the similarity results. Thus, the results follow similar trends in NP with WP. Consequently, we believe precision and recall results are arbitrarily better for some cases and worse for some others. As a result, PSCP does not prevent privacy breaches for *knn*-based attacks.

6 Conclusions and Future Work

Partitioned data-based collaborative filtering schemes are important because e-commerce companies might prefer to collaborate to offer richer prediction to their customers. To ensure privacy, privacy-preserving collaborative filtering schemes are offered and these schemes have some substantial measure to prevent privacy breaches that might occur during mutual interaction. In this study, we focus on horizontally partitioned data-based privacy-preserving collaborative filtering schemes on binary ratings. We experimentally test their resilience to the known attack types in the literature and propose an attack type, *perfect match*. Acting as an active user attack can be considered serious for dense data set even if privacy measures are introduced. Our proposed attack, *perfect match*, can be considered successful when privacy measures are not applied; however, private similarity computation protocol is an important factor to mitigate its damage in terms of the first aspect of privacy. On the other hand, *perfect match* attack is a serious threat for the second aspect of privacy. *knn*-based attack is not seriously affected by private similarity computation protocol because the protocol does not affect the similarity computation. The similarity metric is calculated by only considering commonly rated items and our initial instinct is that all rated entries of the active user and the related user whose similarity is calculated should be taken into account while considering the similarities. Another inference is that dense data sets are more prone to the attacks.

As a future plan, we plan to offer some measures to enhance privacy levels against the known attack types and try to enlarge our vision in terms of possible privacy risks that may occur beyond the attacks discussed in this study. Furthermore, we wish to apply multi-party version of two-party binary privacy-preserving collaborative filtering schemes attacked in this study; analyze and test it to discover how much privacy they offer against the privacy risks.

Acknowledgements. This work is supported by the Grant 113E262 from TUBITAK.

References

1. Agrawal, D., Aggarwal, C.C.: On the design and quantification of privacy preserving data mining algorithms. In: Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, pp. 247–255, Santa Barbara, CA, USA (2001)
2. Agrawal, R., Srikant, R.: Privacy-preserving data mining. In: Proceedings of the 19th ACM SIGMOD International Conference on Management of Data, pp. 439–450, Dallas, TX, USA. (2000)
3. Bilge, A., Kaleli, C., Yakut, I., Gunes, I., Polat, H.: A survey of privacy-preserving collaborative filtering schemes. Int. J. Softw. Eng. Knowl. Eng. **23**(08), 1085–1108 (2013)
4. Calandrino, J.A., Kilzer, A., Narayanan, A., Felten, E.W., Shmatikov, V.: You might also like: privacy risks of collaborative filtering. In: Proceedings of the IEEE Symposium on Security and Privacy, pp. 231–246, Oakland, CA, USA (2011)
5. Canny, J.: Collaborative filtering with privacy. In: Proceedings of the IEEE Symposium on Security and Privacy, pp. 45–57, Oakland, CA, USA (2002)
6. Cranor, L.F.: I didn't buy it for myself privacy and ecommerce personalization. In: Proceedings of the ACM Workshop on Privacy in the Electronic Society, pp. 111–117, Washington, DC, USA (2003)
7. Goldberg, D., Nichols, D., Oki, B.M., Terry, D.: Using collaborative filtering to weave an information tapestry. Commun. ACM **35**(12), 61–70 (1992). http://doi.acm.org/10.1145/138859.138867
8. Goldberg, K., Roeder, T., Gupta, D., Perkins, C.: Eigentaste: a constant time collaborative filtering algorithm. Inf. Retr. **4**(2), 133–151 (2001). http://dx.org/10.1023/A:1011419012209
9. Guo, S., Wu, X., Li, Y.: Deriving private information from perturbed data using iqr based approach. In: Proceedings 22nd International Conference on Data Engineering Workshops, pp. 92–92 (2006)
10. Guo, S., Wu, X., Li, Y.: Determining error bounds for spectral filtering based reconstruction methods in privacy preserving data mining. Knowl. Inf. Syst. **17**(2), 217–240 (2008)
11. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.T.: An algorithmic framework for performing collaborative filtering. In: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 230–237, Berkeley, CA, USA (1999)
12. Herlocker, J.L., Konstan, J.A., Riedl, J.: Explaining collaborative filtering recommendations. In: Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, CSCW 2000, pp. 241–250. ACM, USA (2000). http://doi.acm.org/10.1145/358916.358995
13. Huang, Z., Du, W., Chen, B.: Deriving private information from randomized data. In: Proceedings of the 24th ACM SIGMOD International Conference on Management of Data, pp. 37–48, Baltimore, MD, USA (2005)
14. Kaleli, C., Polat, H.: Providing Naïve bayesian classifier-based private recommendations on partitioned data. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) PKDD 2007. LNCS (LNAI), vol. 4702, pp. 515–522. Springer, Heidelberg (2007)
15. Kaleli, C., Polat, H.: Privacy–preserving naïve bayesian classifier based recommendations on distributed data. Comput. Intell. **31**(1), 47–68 (2015). http://dx.org/10.1111/coin.12012

16. Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: On the privacy preserving properties of random data perturbation techniques. In: Proceedings of the 3rd IEEE International Conference on Data Mining, pp. 99–106, Melbourne, FL, USA (2003)
17. Kargupta, H., Datta, S., Wang, Q., Sivakumar, K.: Random-data perturbation techniques and privacy-preserving data mining. Knowl. Inf. Syst. **7**(4), 387–414 (2005)
18. Miyahara, K., Pazzani, M.J.: Collaborative filtering with the simple bayesian classifier. In: Mizoguchi, R., Slaney, J. (eds.) PRICAI 2000. LNCS, vol. 1886, pp. 679–689. Springer, Heidelberg (2000)
19. Okkalioglu, M., Koc, M., Polat, H.: Deriving binary ratings from masked data. Submitted for review
20. Okkalioglu, M., Koc, M., Polat, H.: On the discovery of fake binary ratings. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing, SAC 2015, pp. 901–907. ACM, USA (2015). http://doi.acm.org/10.1145/2695664.2695866
21. Polat, H., Du, W.: Privacy-preserving collaborative filtering using randomized perturbation techniques. In: Proceedings of the 3rd IEEE International Conference on Data Mining, pp. 625–628, Melbourne, FL, USA (2003)
22. 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, WI 2005, pp. 725–731. IEEE Computer Society, Washington, DC (2005). http://dx.org/10.1109/WI.2005.117
23. Polat, H., Du, W.: Achieving private recommendations using randomized response techniques. In: Ng, W.-K., Kitsuregawa, M., Li, J., Chang, K. (eds.) PAKDD 2006. LNCS (LNAI), vol. 3918, pp. 637–646. Springer, Heidelberg (2006)
24. Polat, H., Du, W.: Privacy-preserving top-n recommendation on distributed data. J. Am. Soc. Inf. Sci. Technol. **59**(7), 1093–1108 (2008). http://dx.org/10.1002/asi.20831
25. Resnick, P., Varian, H.R.: Recommender systems. Commun. ACM **40**(3), 56–58 (1997). http://doi.acm.org/10.1145/245108.245121
26. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Adv. Artif. Intell, p. 4 (2009). http://dx.org/10.1155/2009/421425
27. Warner, S.L.: Randomized response: a survey technique for eliminating evasive answer bias. J. Am. Stat. Assoc. **60**(309), 63–69 (1965)
28. Yakut, I., Polat, H.: Estimating NBC-based recommendations on arbitrarily partitioned data with privacy. Knowl. Based Syst. **36**(0), 353–362 (2012). http://www.sciencedirect.com/science/article/pii/S0950705112002031
29. Zhang, S., Ford, J., Makedon, F.: Deriving private information from randomly perturbed ratings. In: Proceedings of the 6th SIAM International Conference on Data Mining, pp. 59–69, Bethesda, MD, USA (2006)