Neurocomputing 207 (2016) 374–386

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Reconstructing rated items from perturbed data

Burcu Demirelli Okkalioglu [a], Mehmet Koc [b], Huseyin Polat [c,*]

[a] Computer Engineering Department, Yalova University, 77100 Yalova, Turkey
[b] Electrical & Electronics Engineering Department, Bilecik Seyh Edebali University, 11210 Bilecik, Turkey
[c] Computer Engineering Department, Anadolu University, 26470 Eskisehir, Turkey

ARTICLE INFO

Article history:
Received 13 February 2015
Received in revised form
4 May 2016
Accepted 5 May 2016
Communicated by Peng Cui
Available online 12 May 2016

Keywords:
Data reconstruction
Noise reduction
Auxiliary information
Privacy
Randomized perturbation
Collaborative filtering

ABSTRACT

The basic idea behind privacy-preserving collaborative filtering schemes is to prevent data collectors from deriving the actual rating values and the rated items. Different data perturbation methods have been proposed to protect individual privacy. Due to different privacy concerns, users might disguise their data variably to meet their own privacy concerns. In addition to reconstructing the true rating values, data collectors might try to reconstruct the rated items.

In this paper, our goal is to reconstruct the rated items with the help of auxiliary information when users mask their confidential data inconsistently in privacy-preserving prediction systems. We first need to estimate the number of the rated items. Then we have to predict the rated items. To do so, we first use existing methods to eliminate noise from the disguised data. We improve our predictions by utilizing the auxiliary information. Our real data-based empirical outcomes show that our proposed approaches are able to reconstruct the rated items with decent accuracy in spite of variable data masking.

1. Introduction

With the growth of the Internet, the number of e-commerce Web sites has gradually increased every passing year. E-commerce companies provide a fast way for people to purchase what they search or desire without spending too much time. People do not have to visit any physical store anymore to buy things. They prefer purchasing products online. For example, an item located at different cities or countries can be easily bought online. However, as the number of the products gets higher, e-commerce companies have begun to handle too much information to provide quality services for customers. Thus, a new problem called *information overload* has emerged. To deal with this problem, collaborative filtering (CF) has been proposed [1,2]. In the past, people shared their opinions with like-minded friends and relatives about books, movies, places, and restaurants; and they gave advices to each other about things they like. Now, CF systems take the place of the previous idea of exchanging recommendation between people. A CF system makes recommendations by matching users based on their past preferences or tastes [3,4].

Cranor et al. [5] conduct a survey to figure out online users' privacy concerns. According to the survey's results, people have different levels of privacy concerns. Most of the people do not want to disclose personally identifiable information. The other important result is that people think that some information such as social security number, credit card number, or phone number is more sensitive than other information such as e-mail address, age, or favorite TV. People are much less willing to share data with the third parties because they want to be sure that their information will not be shared with other companies for different purposes. Unsolicited communication is another problem for people. On the other hand, some people do not worry about privacy. They are willing to share information to provide advantages such as getting free coupons or pamphlets. All results show that each person has different privacy concerns and some people have more privacy concerns than others. Likewise, CF systems have similar privacy risks for users. Since CF systems fail to protect users' privacy, researchers propose privacy-preserving collaborative filtering (PPCF) schemes. PPCF methods allow users to share their data while protecting privacy as well as providing accurate predictions. To preserve privacy, sensitive/private data has to be disguised before sending it to the CF system. Several data disguising methods have been proposed to preserve privacy including but not limited to homomorphic encryption [6,7], anonymization [8,9], randomized response techniques [10,11], secure multi-party computation [12–14], and randomization [15,16].

Randomization is a very common data disguising method in PPCF. Polat and Du [16] propose randomization for PPCF to protect individual privacy. According to their data disguising method, each user first transforms the original ratings into z-score values and then adds random numbers to them using randomization. The disguised data is sent to the server instead of the original data.

* Corresponding author. Tel.: +90 222 321 3550; fax: +90 222 323 9501.
E-mail address: polath@anadolu.edu.tr (H. Polat).

http://dx.doi.org/10.1016/j.neucom.2016.05.014
0925-2312/

However, some researchers show that randomization may not protect individual privacy [17–19]. Private data can be reconstructed from the disguised data. Zhang et al. [19] claim that a great number of data is derived from the disguised one when the approach proposed by Polat and Du [16] is used. Thus, Polat and Du [20] propose to use inconsistent data disguising method to increase privacy instead of the naïve solution [16]. The naïve solution may not sufficiently protect user's privacy [19]. The server or the CF system knows the rated items. Users might think that which items they purchase or rate are also confidential. Hence, this kind of information may decrease privacy. To deal with varying levels of privacy concerns, Polat and Du [20] suggest different data disguising methods and allow users to disguise their sensitive data inconsistently. According to one of their scenarios, each user disguises some of the uniformly randomly selected empty items with random numbers besides the rated items. There are different scenarios how empty items are handled based on users' concerns. Some users want to disguise all empty items, although others disguise a predefined percentage of empty items only. There are some recent studies based on inconsistent data disguising methods to hide the ratings and the rated items with decent accuracy [21–24].

The aim of this paper is to reconstruct the rated items from disguised data, which masked inconsistently. The disguised data contain the rated items and the filled items with fake ratings. There are two cases to define the number of the filled cells with fake ratings. In the first case, each user randomly selects a predefined percentage of empty items in the same way. The number of empty cells to be filled can differ for each user depending on her concerns in the second case. Inconsistent data disguising method increases privacy compared with the naïve method. Our purpose is to predict the real rated items when inconsistent data disguising method is preferred by users to mask their sensitive data. Different from the previous works, we present several approaches to estimate the rated items in several scenarios. There are two important tasks in the paper to be solved. The first task is to estimate the number of the real ratings from the disguised data and the second one is to identify the rated items. We also claim that auxiliary information can help us reconstruct the rated items in addition to existing data reconstruction methods.

The main contributions of the paper can be listed as follows:

(1) We study how to reconstruct the rated items from inconsistently perturbed numeric data. To the best of our knowledge, this study is the first one focusing on deriving the rated items in numeric ratings-based PPCF systems.
(2) Two approaches are proposed to estimate the number of the real ratings from the disguised data.
(3) In PPCF, matrix factorization methods are widely used for prediction. However, we use the matrix factorization methods to decrease the effects of the noise data.
(4) We finally scrutinize the joint effect of auxiliary public information and characteristic properties of CF systems in addition to existing matrix factorization methods.

The rest of the paper is structured as follows. In Section 2, the related work is explained. Section 3 gives the detailed information about inconsistent data disguising scenarios. Privacy and problem definition are explained in Section 4. Then we show that how the number of real ratings from the disguised ones are estimated. We continue with describing different noise elimination methods and different approaches utilizing auxiliary information in Section 5. In Section 6 and Section 7, several experiments are performed to evaluate our approaches and Section 8 demonstrates our conclusions and future research directions.

2. Related work

Protecting individual privacy is the main concern in PPCF. Polat and Du [16] propose randomization as a data disguising method in PPCF to alleviate users' concerns. According to their data disguising scheme, each user transforms the original ratings into z-score values and then disguises z-score values using randomization. Users send the disguised data rather than the original data to the CF system or the server. Randomization is first proposed in the privacy-preserving data mining (PPDM) by Agrawal and Srikant [15]. The idea behind randomization is to add a random number (r_i) to the original data (x_i) so that unauthorized people cannot learn the original data from masked data (x_i+r_i). Polat and Du [16,25] show that the CF systems can perform CF computations with decent accuracy on perturbed data although they do not know the original data.

Randomization is a common and simple method to protect individual privacy in PPCF. However, some studies show that randomization may not keep private information as much as believed. Agrawal and Srikant [15] show that reconstructing the distribution of the original data from the disguised data is possible. Their work is extended by Agrawal and Aggarwal [26] using expectation maximization (EM) algorithm. They show that EM algorithms work well to estimate the original distribution when the data is large enough. Kargupta et al. [17,27] claim that randomization may not hide the private information. They propose a random matrix-based spectral filtering method to prove their hypothesis. They show how the original data can be obtained from the disguised data using random matrices properties. Huang et al. [18] conduct a similar study with Kargupta et al. [17]. They propose two data reconstruction methods, which are principal component analysis (PCA) and Bayes estimation. They show that when the correlations between attributes are increased, the reconstruction results improve. Guo and Wu [28] and Guo et al. [29,30] also use the spectral filtering method to show how an attacker can compare their results with the original data. Guo et al. [29,30] propose singular value decomposition (SVD)-based data reconstruction method. By using this method, the data owners can decide the least amount of random data that should be added to the original data to protect individual privacy. Zhang et al. [19] show the derivation of the private data from the disguised one by using k-means clustering and SVD-based data reconstruction method. Their study differs from the previous studies because previous studies conduct experiments with complete matrices. Used data sets are usually artificial and do not contain empty cells. As the nature of recommender systems, CF systems contain many empty cells; the study of Zhang et al. [19] is prominent.

There are different existing data reconstruction methods in the literature. Apart from data reconstruction methods, characteristic properties of CF system and auxiliary information can help to derive the original data from the disguised one. With the emergence of social networks, there are different sources of obtaining auxiliary information about users. Calandrino et al. [31] use auxiliary information to infer users' transaction, which is performed on a CF system. They claim that public information can be obtained from some popular services and can be used to derive user related data from CF systems. In another study, auxiliary information is used to alleviate data sparsity and cold-start problems in CF [32]. The authors claim that social network data is a huge potential to overcome the data sparsity and the cold-start problems. They employ a star-structure graph and utilize useful auxiliary information from cross-domains. Their experiments provide better results with the integration of auxiliary cross-domain information. Learning insights about users from different social domains could be an important factor for business intelligence [33]. Liu et al. [33,34] aim to link users among different

social network platforms by heterogeneous behavior modeling claiming that such linkage would be beneficial for user profiling due to completeness, consistency, and continuity of users' behavior. Hydra framework that links users across different platforms is proposed. The framework benefits from unique features of social data such as social behavior over a period of time shows consistency and users' core structures are similar between different platforms and demonstrate highly distinct quality. Their analysis shows that the proposed solution beats current algorithms in identifying user linkage in cross-domains.

Do all users have the same privacy concerns? Is it enough to disguise only the original data to protect individual privacy? Polat and Du [20] claim that each user has a different level of privacy concerns. Although some users do not want to reveal their ratings, some users do not desire to divulge their ratings and their rated items. They propose that each user fills some empty cells with random numbers so that the CF system does not learn the rated items. The purpose of this paper is to predict which cells are actually filled and/or which cells are empty from the disguised data by utilizing different matrix factorization techniques widely used in CF community for prediction [35–37] as well. The authors in [35] use probabilistic matrix factorization for imbalanced and sparse data sets as well as scaling on very large data sets. The authors in [36] discuss that matrix factorization is better compared to neighborhood-based approaches for recommendation purposes and could be integrated with additional information. A probabilistic matrix factorization method is proposed to incorporate social contextual factors, individual preference, and interpersonal influence [37]. Integration of social contextual factors greatly improves recommendation results for social data compared to earlier approaches. The authors in [38] give an analysis of different CF algorithms in detail. However, instead of offering predictions, the main concentration of our study is to determine the genuinely rated items in the original CF data before PPCF methods are applied to disguise the genuinely rated items. Therefore, this paper aims to reconstruct the genuinely rated items while the aforementioned studies are devised for recommendation purposes for CF-type systems. On the other hand, Okkalioglu et al. [39] also investigate if originally rated items could be differentiated from randomly appended items for a binary PPCF scheme [40]. The authors utilize auxiliary public information to distinguish genuinely rated from randomly appended fake items. Unlike [39], this study discovers the rated items from a numerically rated PPCF algorithm, which is perturbed by randomization. It also reconstructs the unrated items, which are disguised by inconsistent data perturbation. Moreover, the work in [39] utilizes auxiliary information only to determine the originally rated items; however, this study employs noise elimination techniques as well as auxiliary information to enhance the results.

3. Preliminaries

3.1. Data disguising by randomization

It is essential to protect privacy while performing CF calculations. There are a couple of methods proposed to alleviate users' privacy concern. Randomization is one of the most common method in PPCF, which allows users to mask their sensitive data. The idea behind randomization is simple, so it can be easily adapted to CF systems. With randomization, a random number is added to the original value. Rather than obtaining the original value, the third parties have the masked one only.

Polat and Du [16] first use randomization as a data disguising method in PPCF. They propose different scenarios for data disguising based on users' privacy concerns [20]. Polat and Du [20] define two main scenarios: all users either use the same parameters values or select different parameters values depending on privacy expectations for a random number distribution. According to their proposed disguising method, users can perturb their rated items only; as a result, the users do not need to send their real values to the third parties. The basic and first scenario in data disguising method to generate random numbers is that all users use the same parameters values. The server first determines the type of random number distribution (uniform or Gaussian). It then determines the value of the standard deviation of the random numbers (σ). It finally let each user know the type of the distribution and the σ [16]. In that case, distribution type (uniform or Gaussian) should be known by everyone. According to the second scenario, the server only determines the upper bound of the σ value (σ_{max}) and let each user know it. Then, each user selects σ from the range [0, σ_{max}] and decides the distribution type by coin tosses. After deciding σ, each user computes her mean and standard deviation; and then calculates her z-score values for each rated items. Each user creates a random data using σ and type of the distribution. The number of the random data should be equal to the number of items the user has voted. Before sending z-score values to the CF system, each user adds a random number to her z-score value to generate a disguised z-score value. Users send the disguised z-score values instead of the original ones to the system.

In addition to hiding the true values, the rated and the unrated items are also prominent and are considered confidential in PPCF. Thus, data disguising method allows individuals to fill some unrated item cells with random numbers. Polat and Du [20] propose inconsistent data perturbation for individuals who have different privacy expectations. Like selecting σ, there are two cases to pick which unrated items should be filled with random numbers. In the first case, the server determines β and let each user knows. In the second case, the server defines β_{max} and each user uniformly randomly selects β_u between 0 and β_{max}. Then, users calculate the number of unrated items in their rating vectors by using β_u. The users uniformly randomly choose β_u percent of their unrated item cells to be filled with random data. Using the selected distribution type and σ, which is picked by the server or each user selects own σ, each user generates random numbers for her rated and selected unrated items. Finally, each user disguises her z-score values and the selected unrated items with the corresponding random numbers.

Data disguising methods' scenarios can be summarized based on which parameters are used as follows:

- All users disguise their rated and some of unrated items with the fixed β and use the fixed σ: The server determines the distribution type (uniform or Gaussian), β, σ.
- All users disguise their rated and some of unrated items with the fixed β and use the random σ: The server determines β and σ_{max}, then each user selects own σ_u and the distribution type randomly.
- All users disguise their rated and some of unrated items with the random β and use the fixed σ: The server determines the distribution type, σ and β_{max}, then each user selects own β_u.
- All users disguise their rated and some of unrated items with the random β and use the random σ: The server determines σ_{max} and β_{max}, then each user selects randomly own σ_u, distribution type, and β_u.

4. Privacy and problem definition

There are two main privacy concerns in PPCF. Hiding the original ratings from the CF system is the first prominent goal. To achieve this goal, randomization is proposed and random numbers

are added to the original ratings [16]. With the randomization, the system cannot learn the real ratings. Masking the original ratings only is not enough for some users because they may not want to disclose their rated items either. Therefore, some unrated items are filled with random numbers to disguise which items are genuinely rated [20]. In this study, we focus on the second privacy concern.

The problem we try to solve is to determine the rated items from the disguised data. Each user has fake ratings besides the original ratings in her rating vector. Assume that there are m items in the rating vector and the user u rates k items. In addition, a user wants to fill β percent of unrated items so that $((m-k)\times\beta)/100$ number of fakes items will be appended to the original vector with random numbers before sending the rating vector back to the server, where $(m-k)$ is the number of unrated items. The server receives $u_{TotalRated}$ items which adds up to $(k+((m-k)\times\beta)/100)$ rated items including the fake ones. The server does not know the value of k and which items are fake as well as, in some cases; the server does not know the value of β. The aim of the study is to predict which items are rated among $u_{TotalRated}$ items. Our solution is divided into three main parts:

- Eliminate the noise. The effect of fake items needs to be decreased.
- Estimate the number of originally rated items (k') from $u_{TotalRated}$ based on β selection.
- Select k' items from $u_{TotalRated}$ items to determine the real ratings.

5. Determining real rated items

Since noise is added to the original z-scores, it should be eliminated to see data as close as to its original form. The first task to determine the real rated cells is to eliminate the noise associated with the original data. The value in the unrated cells is the noise introduced during the randomization process. Thus, the estimated value for unrated cells approaches to 0 after noise elimination. On the other hand, the rated cells have prior z-scores before the randomization process and the estimated cells for the rated items are expected to have relatively greater absolute values than the unrated ones after noise elimination because unrated cells lack prior ratings. As a result, greater absolute values will help us determine the rated cells. However, it is still unknown how many cells need to be picked as rated cells. At this juncture, the number of real rated cells can be estimated to make a better prediction to determine real rated cells. The second task in determining the real ratings is to estimate the number of real rated cells and find out rated items based on this estimation. This number can be estimated based on β value. We also hypothesize that some auxiliary information could help while finding rated cells. In this section, the method of determining the real rated cells is discussed.

5.1. Noise elimination methods

We explain four different noise elimination methods briefly in this section. The aim of the selected methods is to eliminate noise from the disguised data. Note that random numbers are added to the rated items and some unrated items. If the noise is removed, the rest data can be used to predict the real rated items.

5.1.1. Singular value decomposition

SVD is widely used in signal processing, pattern recognition, statistics, atmospheric sciences, and recommender systems for compression, recommendation, and noise reduction. The purpose of the SVD is to decompose a matrix A into three matrices called U, S, and V: U is an orthogonal matrix called the left singular vectors, S is a diagonal matrix with all singular values, and V is the transpose of an orthogonal matrix, which is called the right singular vectors. SVD is defined as follows:

$$A_{n\times m} = U_{n\times n} \times S_{n\times m} \times V^T_{m\times m} \tag{1}$$

SVD theorem has an interesting and useful property. SVD allows users to transform the data into low-rank (or reduced) matrix. The dimensionality reduction approach in SVD is important for noise reduction. When all singular values are sorted in descending order, small singular values represent noise. If the small singular values are discarded, the top singular values are used to reconstruct the original data. SVD is a common method in PPCF. Also, there are studies to reconstruct the original data from the disguised one [19,29,30].

5.1.2. Principal component analysis

PCA has many applications in neuro-science, pattern recognition, image compression, finance, economies, genetics, geology, meteorology, etc. The basic approach is to transform a large number of correlated attributes into a small number of uncorrelated attributes so that predictions or operations in data can be easily done [18,41]. To apply PCA, the covariance matrix has to be calculated. Assume that the original data set is A, the covariance matrix is calculated as A^TA. After finding the covariance matrix, the eigenvalues (Λ) and eigenvectors (U and V^T) are calculated from the covariance matrix. PCA is defined as follows:

$$A^TA = U \times \Lambda \times V^T \tag{2}$$

The main approach in PCA is dimensionality reduction as in SVD. While top singular values are chosen in SVD, top uncorrelated principal components are selected in PCA. To show the relationship between two methods, Eq. (1) is substituted for A in Eq. (2) as follows [42]:

$$\begin{aligned} A^TA &= (USV^T)^T(USV^T) \\ A^TA &= VSU^TUSV^T \\ A^TA &= VS^2V^T \end{aligned} \tag{3}$$

Eq. (3) is similar to the one shown in Eq. (2). Since the singular values can be calculated by taking the square roots of the eigenvalues of the matrix A^TA, the eigenvalues can be represented as the squares of the singular values of A in Eq. (3).

5.1.3. Discrete cosine transform

Discrete cosine transformation (DCT) is usually used for pattern recognition, Wiener filtering, image, and video compression. The purpose of DCT is to transform data to frequency domain so that the redundancy can be removed [43,44]. DCT provides a transformation like SVD and PCA. The basic approach behind DCT is to transform correlated data into discrete (uncorrelated) cosine transform coefficients. In other words, the purpose is to define low and high frequency coefficients because low frequency coefficients are more significant than high frequency ones. In our experiment, DCT is used for noise reduction from the disguised data. High frequency coefficients' values are accepted as noise and ignored when the data is reconstructed using the inverse DCT function. Since the data is two-dimensional, there are $m \times n$ coefficients, so 2D-DCT is applied in the experiment.

5.1.4. QR factorization

QR factorization (or QR decomposition) is one of the most important matrix decomposition methods. QR decomposes a matrix A into two matrices called Q and R. $A = QR$, where Q is an orthogonal matrix and R is an upper triangular matrix. The main idea of QR factorization is the calculation of the eigenvalues. There are different methods to compute QR factorization. Two of the most popular methods are Graham-Schmidt process and

Householder transformation. The purpose of QR is similar to SVD or PCA. Briefly, all used data reconstruction methods in the paper transform the data into a reduced matrix to eliminate noise.

5.2. Filling the missing entries

Data reconstruction methods utilized in the paper can only perform on complete data. Therefore, we first need to fill all unrated empty cells with appropriate default values to obtain a denser data. There are some methods to fill empty cells, which are *zero rating*, *user average*, and *item average*. The first method for filling all empty cells is to assign a zero rating. We know that the used data set is mostly empty and filling all missing values with zero is an appropriate way to reconstruct the data from the filled data. The second approach is to fill empty cells with corresponding user averages. According to the data disguising method, users convert their ratings into z-scores and add random numbers to the rated and some unrated cells. Hence, the server first estimates the user averages from the disguised data. Bilge and Polat [24] show how to guess such values from the disguised data to obtain a denser data. The average z-scores for each user can be estimated from the disguised data as follows:

$$\overline{z}'_u = \frac{\sum_{j=1}^{R_u} z'_{uj}}{R_u} = \frac{\sum_{j=1}^{R_u}(z_{uj}+r_{uj})}{R_u} = \frac{\sum_{j=1}^{R_u}(z_{uj})}{R_u} + \frac{\sum_{j=1}^{R_u}(r_{uj})}{R_u} \approx \frac{\sum_{j=1}^{R_u}(z_{uj})}{R_u} \tag{4}$$

As the data set contains the disguised z-scores instead of the real ratings, the average of z-scores for each user can be calculated roughly using Eq. (4). The average of items is also preferred to fill all empty cells. In that case, the server tries to guess the item averages from the disguised data. Due to the random data distribution with zero mean, we expect that the average of the random numbers approximates zero as in the case of the user average estimations. In that case, Eq. (5) can be used to estimate the averages of each item's z-scores as follows:

$$\overline{z}'_i = \frac{\sum_{j=1}^{R_i} z'_{ij}}{R_i} = \frac{\sum_{j=1}^{R_i}(z_{ij}+r_{ij})}{R_i} = \frac{\sum_{j=1}^{R_i}(z_{ij})}{R_i} + \frac{\sum_{j=1}^{R_i}(r_{ij})}{R_i} \approx \frac{\sum_{j=1}^{R_i}(z_{ij})}{R_i} \tag{5}$$

5.3. Estimating number of real ratings from disguised ones

Before marking the real rated items, it is important to estimate how many real rated items are rated by users. We know that each user disguises her ratings and empty ratings with random numbers. The server does not know which ratings are actually rated by users or which ratings are filled with random numbers. The server needs to predict the number of the real ratings from the disguised data before estimating which one is actually rated.

We define two formulas to estimate the number of the real ratings based on the selection of β. When each user uses the same β to fill empty cells with the random numbers, the server can estimate the number of rated items very close to the original number of rated items. The server knows β and the total rated items from the disguised data. We can easily estimate the number of rated items for each user using the following equation. In Eq. (6), *nItems* depicts the number of total items. The information is known publicly. *nRated* denotes the total number of the rated items, which are calculated from the disguised data. β is known by the server. $nRated_{items}$ depicts the number of true ratings in the original data. The purpose is to estimate $nRated_{items}$ for each user.

$$nRated_{items} + (nItems - nRated_{items}) * \beta = nRated \tag{6}$$

After rearranging given formula, Eq. (7) is obtained. We round the result to find an integer number. Eq. (7) shows that the server can estimate the number of rated items for each user although the user disguises her ratings and some empty ratings.

$$nRated_{items} \cong round\left(\frac{nRated - (nItems * \beta)}{1-\beta}\right) \tag{7}$$

When users use different β values (from 0 to β_{max}) to fill empty ratings with random numbers, the estimation has to be changed. The server does not know the value of β chosen by each user. The server knows β_{max} only. In that case, as we do not know the selected β value by each user, we propose to use the expected value of β (β_{exp}). For each item, we try to estimate how many users should rate it using Eq. (8). *nUsers* depicts total number of users in the data set and is known publicly. *nRated* is calculated from the disguised data and shows how many users rate the item. *nRated* contains real users and fake users who rate the item. The value of B_{exp} is calculated as $\beta/2$. The purpose is to estimate how many users truly rate the item, $nRated_{users}$.

$$nRated_{users} + (nUsers - nRated_{users}) * \beta_{\exp} = nRated \tag{8}$$

After rearranging Eq. (8) we obtain the following Eq. (9):

$$nRated_{users} \cong round\left(\frac{nRated - (nUsers * \beta_{\exp})}{1-\beta_{\exp}}\right) \tag{9}$$

5.4. Utilizing auxiliary information

While our purpose is to reconstruct the rated items from the disguised data, some auxiliary information can help us estimate the real rated items as well as improving accuracy results. The basic approach is to take the advantage of the distribution types' feature. Based on the distribution type, some data lie out of the range and those help us reconstruct the rated items. Since our data set is a movie-related, another auxiliary information from Internet Movie Database (www.imdb.com) called *total votes* guides us to identify the real rated items from the disguised data. *User demographic information* is available and used as auxiliary information. Most of the CF systems contain user-related information in addition to ratings. The relationship between user demographic information and items lead us to reconstruct the rated items.

5.4.1. Features of random data distributions

Uniform and Gaussian distributions have unique characteristics. When Gaussian distribution is used, 99.7% of data fall within the range of $[\mu - 3\sigma, \mu + 3\sigma]$. On the other hand, when the distribution type is chosen as uniform, all data lie within the range of $\left[\mu - \sqrt{3}\sigma, \mu + \sqrt{3}\sigma\right]$. According to the data disguising method, each user fills selected unrated items with her mean. After transforming the original values to z-score values, each selected unrated item's z-score value is calculated as 0. Therefore, if the random numbers are eliminated based on the distribution type, points, which lie out of the range, are considered as rated items. Zhang et al. [19] used this idea to calculate which items are rated when all items filled with random numbers. However, using only this approach is not good enough to identify the rated items, because many rated items are ignored as they point out in their paper. Besides, when σ gets larger, the advantage of this approach decreases. This approach is used as auxiliary information with the main data reconstruction method in our experiments. If values lie out of the predefined range, those are chosen as rated items.

5.4.2. Total votes

Since users mask their rated and unrated items, which items might be rated are required to be decided. Besides data reconstruction methods to mark the rated items, auxiliary information can be used to improve reconstruction accuracy. The auxiliary information is taken from Internet Movie Database (IMDb), which

is a famous movie website and has millions registered users all around the world. The information in IMDb is consistent and reliable. IMDb offers different properties of movies based on registered users' opinions. As well as movies' average rating, awards, popular and unpopular movies, IMDb provides how many registered users rate each movie. Total votes about movies are considered as a prominent auxiliary information to decide which item might be rated when users disguise rated and some unrated items. Without needing to check a movie is popular or unpopular or what the movie rating is, total votes for each movie is enough as auxiliary information to identify which items might be rated. Even if a movie is rated with a lower rating, if a predefined number of registered users rate the movie, the movie can be chosen as a rated item. The lower rating only shows that users do not like the movie. The values of ratings are not important in this case. The important thing is to identify correctly rated items among rated and fake items. We define a threshold value in our experiments to choose the lower bound of total votes for each movie. If total votes for each movie is greater than the threshold, we accept this movie is rated by the users in our experiments. The purpose of the approach is to make a decision about items when the rated items are not known. The main idea is that the more users vote the movie in IMDb, the likelier the same movie is rated by users in our data set no matter what its rating is.

5.4.3. User demographic information

A CF system usually keeps user demographic information such as age, gender, occupation, and address besides items' properties and ratings. This kind of information is sometimes related to ratings. For example, gender is important demographic information to estimate which items might be enjoyed by users. Female users might be interested in female-specific items although male users might be fascinated by items female users do not like. Likewise, age range is also prominent demographic information for CF systems. To give an example, we use movie-related data set in our experiments. Generally, younger users like comedy or action movies while older users enjoy drama movies. We think that user demographic information especially age range helps us reconstruct the rated items from rated and fake items in our experiments. We divide ages into five groups, which are 0–24, 25–34, 35–44, 45–54, and 55+ and define a movie genre to each group. Our hypothesis is that different age groups are more likely to enjoy the specific movie genre. We combine our hypothesis with the main method to test whether the approach is useful to guess the rated items.

5.5. Complexity analysis

Complexity analysis of the proposed solution is given by laying out the main steps of the reconstruction algorithm. The algorithm includes four main steps: (1) It starts with eliminating noise by widely used methods (SVD/PCA/QR/2D-DCT). For a matrix of n rows (users) and m columns (items), the computational complexity of a full SVD is $O(m^2n)$. However, when we select *top k singular values* instead of all to eliminate the noise, SVD matrix can be computed in $O(nmk)$. Likewise, the computational complexity of QR is approximately same as SVD. On the other hand, PCA requires more time than SVD because PCA first needs to compute the covariance matrix taking $O(n^2m)$ and then apply the dimension reduction taking $O(nmk)$. The total computational complexity of PCA is $O(n^2m+nmk)$. The computational complexity of 2D-DCT and inverse 2D-DCT is the same and is $O(n^3+m^3)$. (2) In the second step, the number of the real ratings is estimated from the disguised data for each user or item based on β selection ($nRated_{items}/nRated_{users}$). Estimating the number of real ratings from the disguised data is explained in detail in Section 5.3. (3) Sorting has been accomplished because we claim that larger values demonstrate the real rated items. (4) In the last step, top $nRated_{items}/nRated_{users}$ items/users are selected from the sorted data. There is a small difference between total computational complexities of the proposed method based on β selection. When each user selects β in the same way, estimating real rated number calculation is performed for each user. All items for each user should be sorted, which takes $O(mlogm)$. When β is selected randomly by each user, estimating real rated number is performed for each item and the algorithm iterates among items. This time, the ratings of each item are sorted taking $O(nlogn)$. There are m items in the matrix, so the total time is $O(mnlogn)$. The complexity analysis is summarized in Table 1.

Table 1
The computational complexity of the proposed solution.

Steps	*Computational Complexity*
Eliminating noise (SVD)	$O(nmk)$
Number of the estimated real ratings on β	$O(nm)$
Sorting the items/users	$O(nmlogm)/O(mnlogn)$
Selecting top t items/users from the sorted data	$O(nt)/O(mt)$
Total	$O(nmk+n(m+mlogm+t))/O(nmk+m(n+nlogn+t))$

Table 2
The computational complexity of total votes approach.

Steps	*Computational Complexity*
Number of the estimated real ratings	$O(nm)$
Taking IMDb data (m) based on rated items (assume p)	$O(nmp)$
Sorting the items (p)	$O(nplogp)$
Selecting top t items from the sorted data	$O(nt)$
Total	$O(n(m+pm+plogp+t))$

Complexity analysis of our proposed method with auxiliary information should also be considered briefly. The computation complexity of random data distribution is $O(nm)$ because for each user we check all items whether an item lies out of the predefined range or not. Total votes approach is similar to our main method. In that case, after estimating the number of the real rated items for each user, we use total votes approach instead of SVD decomposition. The summary of the computational complexity is given in Table 2.

The computational complexity of the main method given in Table 1 and using total votes approach only given in Table 2 are similar. When the main method and auxiliary information are used together, their computational time is taken into consideration separately to compute total computation time.

6. Experiments

We have tested how many real rated items are marked correctly from the disguised data based on data disguising method's scenarios. To test our experiments, we use data reconstruction methods as well as auxiliary information. Auxiliary information is a very strong tool to estimate rated items. In some cases, the contribution of only auxiliary information is equal to the main data reconstruction method (such as SVD-based method).

6.1. Data set and evaluation metric

MovieLens, a well-known movie data set, is used in the experiments. The standard 100K MovieLens data set contains 943 users and 1862 items. Five-star scale is used to vote items. We

measure precision and recall results of our experiments. Note that precision is the ratio of the number of the correctly reconstructed rated items to the number of marked rated items. Recall is the ratio of the correctly reconstructed rated items to the number of rated items.

6.2. Methodology

We run each experiment 100 times to obtain average precision and recall results. To generate random numbers, different σ values are used. We choose 0.33, 0.67, 1, 2, and 3 as σ values. The aim of selecting different σ values is to show how results change when the σ values get larger. Also, we test our experiments based on different β values. 6% of data in MovieLens is filled and the rest is empty. When we choose how many empty cells are filled, we use a priori knowledge about MovieLens. If we choose larger values, our data set turns into a random data set. In that case, predicting the real ratings from large random data might be illogical. Therefore, we limit the value of β in terms of the density of the data set. We know that the ratio of the rated items in MovieLens is 6%. Assume that the density, 6%, is depicted d. We select different values both less and greater than d. The β values are set $d/4$, $d/2$, d, and $2d$ to test how precision and recall results change based on different density values.

7. Experimental results

- **Experiment 1:** Since MovieLens is a sparse data set, we first need to select a suitable default value to fill all empty cells. We conduct several tests for each different method based on various σ and β. As mentioned before, zero ratings, user average, and item average can be commonly used to obtain a denser data set. A small part of our results for each reconstruction method is shown in Fig. 1 when Gaussian distribution is selected. The recall results differ from various σ and β; however, the figure shows similar trends for different σ and β. According to our results as seen from Fig. 1, zero ratings are the best compared with the other two methods. We expect that assigning zero ratings to all empty cells is the best way to estimate the rated items from the disguised data because our hypothesis is bigger absolute ratings are considered as rated items. Therefore, filling all empty cells with zero helps us ignore small absolute ratings, which refer unrated and random ratings. Consequently, the approach is selected as our default value to fill all empty cells.

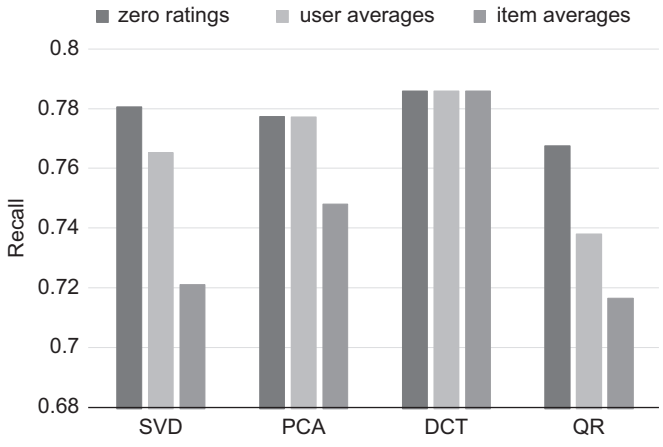

Fig. 1. Recall values for selecting default values when β=6% and σ=0.33.

After selecting our default value to construct a denser data set, we evaluate the best method from four different methods to eliminate the noise from the disguised data. Our hypothesis is that the fake rated items might be considered as the noise. If the noise is eliminated, the rest can help us discover the real rated items. We also expect that the noisy data cannot be eliminated completely; hence, the data, which is close to zero, is considered as the noise. To test our approach, we set 6% as β value. It means that all users randomly select 6% of their unrated items to fill random numbers. To generate random numbers based on Gaussian distribution in Table 3, different σ values are selected. The number of dimensions/principals is set to 10 for SVD, PCA, and QR. We define a threshold value for 2D-DCT to eliminate high frequency coefficients, which indicate the noise in our experiment and set values less than the threshold to zero. Our threshold value is 0.01. Then, we reconstruct the data using the inverse 2D-DCT function. After reconstructing data using one of the four methods, the estimation of the real rated items process has begun. Since the number of the real ratings is predicted for each user, we mark the greater ratings as rated from the reconstructed data. The purpose is to discard the smaller data because the smaller data is considered as the noise. Table 3 shows recall and precision for each method. As seen from Table 3, SVD gives better results compared with others. Only when σ is 0.33, DCT beats SVD; however, when σ gets larger, its advantage goes down. PCA gives almost the same results with SVD. On the other hand, QR is the worst in comparison with SVD and PCA. The more the noise is eliminated, the more the results are improved. The results show that SVD is better than the others to eliminate the noise. After testing different methods with different σ values, we decide to continue with SVD. For following experiments, we choose SVD as our main method to predict the real rated items within the rated and fake items.

- **Experiment 2:** As mentioned earlier, different data disguising methods are proposed to protect individual privacy. Thus, we first evaluate that all users select the β values in the same way. According to the results of Experiment 1, SVD is chosen to estimate the rated items from the disguised data. In this experiment, we test various σ values with different β values using SVD. Table 4 shows that when the σ is 1, how different β values affect the estimation of rated items as well as random items. As seen from Table 4, there are two different cases based on selecting the σ.

(1) The first case is that the server defines σ and the distribution type and let the users know them. Each user uses the same σ and the distribution type as in Table 3. Table 4 first shows the results when uniform distribution is used and then illustrates the results while Gaussian distribution is chosen as the distribution type with the same σ. Their results are almost the same. As the value of β increases, accuracy for estimating real rated items decreases. The simple reason is that when we fill more unrated items with the random numbers, accuracy decreases. On the other hand, predicting which items are random items increases. The more randomness we add, the more accurate results for predicting the fake items. It is obvious that *recallR/precR*, which denote recall and precision results for the estimation of the rated items, and β are inversely correlated. On the contrary, *recallU/precU*, which depict recall and precision results for the estimation of unrated/fake items, and β are correlated. When β increases, accuracy for the fake items goes up.

(2) In the second case, the server only defines σ_{max}, and each user randomly selects σ. The distribution type is identified randomly. Table 4 shows the best results in that case. For example, when β is 6%, the results (recallR) for uniform,

Table 3
Recall and precision results for different reconstruction methods.

			SVD		PCA		DCT		QR	
	σ	β	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**
Gaussian Dist. (σ is fixed)	0.33	6	0. 78095	0.78507	0.77776	0.78187	**0.78649**	**0.79065**	0.71347	0.71723
	0.67	6	**0.76435**	**0.76838**	0.76037	0.76439	0.70394	0.70765	0.69299	0.69665
	1	6	**0.74776**	**0.75171**	0.74437	0.74830	0.66762	0.67115	0.68158	0.68518
	2	6	**0.72278**	**0.72660**	0.72021	0.72401	0.63353	0.63688	0.66275	0.66625
	3	6	**0.71603**	**0.71981**	0.71483	0.71861	0.62478	0.62808	0.65814	0.66162

Table 4
Recall and precision results for SVD when σ is 1.

	σ	β	recallR	precR	recallU	precU
Uniform Dist. (σ is fixed)	1	0.015	0.88686	0.8922	0.5072	0.49361
	1	0.03	0.82689	0.8314	0.6196	0.61204
	1	0.06	0.74791	0.7519	0.7217	0.71746
	1	0.12	0.64908	0.6524	0.8056	0.80328
Gaussian Dist. (σ is fixed)	1	0.015	0.88726	0.8926	0.5090	0.49537
	1	0.03	0.82729	0.8318	0.6205	0.61294
	1	0.06	0.74825	0.7522	0.7221	0.71783
	1	0.12	0.64916	0.6524	0.8056	0.80333
Mix (σ is random)	1	0.015	**0.89416**	**0.8996**	**0.5408**	**0.52625**
	1	0.03	**0.83955**	**0.8442**	**0.6483**	**0.64042**
	1	0.06	**0.76751**	**0.7716**	**0.7438**	**0.73942**
	1	0.12	**0.67688**	**0.6803**	**0.8212**	**0.81886**

Gaussian, and mix distribution (the second case) are 0.74791, 0.74825, and 0.76751, respectively. According to the first case results, the distribution type may not affect the results because their results are very close to each other. Therefore, the server does not need to know the distribution type. For example, recallR with β=1.5% is 0.88686 and 0.88726 for uniform and Gaussian, respectively. The trials show that their results are almost the same. However, σ values affect accuracy as seen from mix distribution in Table 4.

The σ value is chosen as 1 at the beginning of the trials; however, users randomly select σ values in the second case (mix). As a result, for mix distribution (seen from Table 4), not all users select σ with 1 as in the first case (uniform or Gaussian dist.). Some users might select smaller σ values. Thus, we test how the σ affects the results (as seen from Table 5) after obtaining the results from Table 4. The results show that when the σ increases, accuracy decreases. Since users choose different σ values and they can select σ_{max} as a largest σ value, accuracy is better than using σ_{max} as a σ value for each user. For example, recallR for σ=1 is 0.64908 for uniform distribution with β=12% while recallR with the same β for mix distribution is 0.67688 using randomly selected σ. Table 5 shows the detailed results with different σ values when the β is fixed.

- **Experiment 3:** This experiment is conducted to test the approach about total votes for each item in MovieLens. We obtain the public information from IMDb, which is about how many users rate each item. In other words, we retrieve the public information from IMDb for each item in MovieLens. The aim is to predict rated items from the disguised data set. We claim that the more users rate the movie in IMDb, the more likely the same movie might be voted by the MovieLens's users ignoring its rating. Even if a movie gets a lower rating by users, the only important thing is how many users rate it. In this experiment, we sort MovieLens's items in descending order based on public information. For each user, the number of rated items, $nRated_{items}$, from the disguised data is estimated before we take top $nRated_{items}$ items from sorted data. As in Experiment 2, we first show the results in Table 6 when the σ is 1 with different β values. Table 6 achieves better results than Table 4 for each β values. Table 4 gives the best result for mix distribution with β=1.5%. The *recallR* is 0.89416 in Table 4 while the result for the same parameters in Table 6 is 0.90496. The other remarkable thing is that the results are not relevant with the distribution type and different σ values. While the distribution type is changed in Table 6, all results are almost the same. Likewise, as seen from Fig. 2, different σ values do not affect the results. On the other hand, when β values increase, the results decrease because of randomness. When β value increases, we add more random numbers to the unrated items; we increase the number of fake rated items as seen from Table 6, while recallR for β=3% is 0.90481, recallR for β=12% is 0.69066 for uniform distribution. When the same parameters are chosen in

Table 5
Recall and precision results for SVD when β is 6%.

	σ	β	recallR	precR	recallU	precU
Uniform Dist. (σ is fixed)	0.33	0.06	0.7807	0.7848	0.75861	0.75415
	0.67	0.06	0.7645	0.7686	0.74044	0.73608
	1	0.06	0.7479	0.7519	0.72171	0.71746
	2	0.06	0.7223	0.7261	0.69280	0.68873
	3	0.06	0.7153	0.7191	0.68496	0.68093
Gaussian Dist. (σ is fixed)	0.33	0.06	0.7806	0.7847	0.75854	0.75408
	0.67	0.06	0.7647	0.7687	0.74062	0.73626
	1	0.06	0.7483	0.7522	0.72208	0.71783
	2	0.06	0.7223	0.7261	0.69279	0.68872
	3	0.06	0.7156	0.7194	0.68528	0.68125
Mix (σ is random)	0.33	0.06	**0.7850**	**0.7892**	**0.76354**	**0.75905**
	0.67	0.06	**0.7776**	**0.7817**	**0.75512**	**0.75068**
	1	0.06	**0.7675**	**0.7716**	**0.74380**	**0.73942**
	2	0.06	**0.7358**	**0.7397**	**0.70806**	**0.70390**
	3	0.06	**0.7234**	**0.7273**	**0.69411**	**0.69002**

Table 6
Recall and precision results based on IMDb data when σ is 1.

	σ	β	recallR	precR	recallU	precU
Uniform Dist.(σ is fixed)	1	0.015	0.90481	0.91027	0.58979	0.57395
	1	0.03	0.85284	0.85750	0.67846	0.67019
	1	0.06	0.78079	0.78491	0.75877	0.75431
	1	0.12	0.69066	0.69415	0.82892	0.82659
Gaussian Dist. (σ is fixed)	1	0.015	0.90501	0.91047	0.59069	0.57483
	1	0.03	0.85278	0.85744	0.67833	0.67006
	1	0.06	0.78081	0.78493	0.75879	0.75433
	1	0.12	0.69082	0.69431	0.82901	0.82668
Mix (σ is random)	1	0.015	0.90496	0.91042	0.59045	0.57459
	1	0.03	0.85276	0.85742	0.67829	0.67002
	1	0.06	0.78083	0.78495	0.75881	0.75435
	1	0.12	0.69058	0.69407	0.82888	0.82655

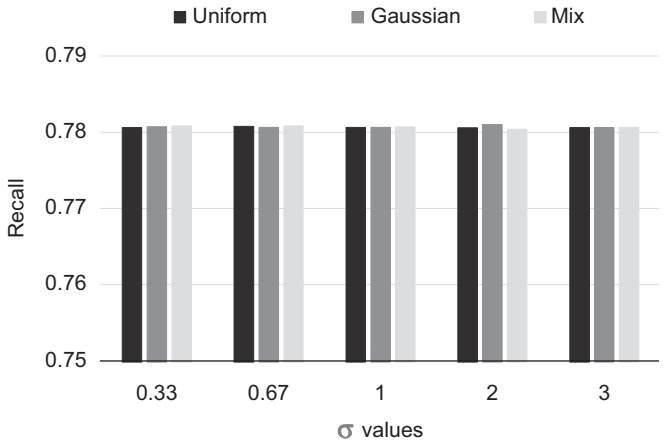

Fig. 2. Recall values for IMDb data when β is 6%.

Table 7
Recall and precision results based on distribution type's feature when σ is 1.

	σ	β	recallR	precR
Uniform Dist. (σ is fixed)	1	0.015	**0.23365**	**1**
	1	0.03	**0.23316**	**1**
	1	0.06	**0.23272**	**1**
	1	0.12	**0.23279**	**1**
Gaussian Dist. (σ is fixed)	1	0.015	0.03257	0.98301
	1	0.03	0.03263	0.96432
	1	0.06	0.03261	0.93102
	1	0.12	0.03253	0.87217
Mix (σ is random)	1	0.015	0.12831	0.98834
	1	0.03	0.12852	0.97667
	1	0.06	0.12772	0.95293
	1	0.12	0.12868	0.91373

Table 4, recallR for β=3% and β=12% is 0.82689 and 0.64908, respectively. The results show that auxiliary information is very strong assumption to estimate the real rated items.

- **Experiment 4:** The other auxiliary information is to utilize the distribution type's feature. Items, which fall out of the range, are expected to be rated. To evaluate how much this approach improves the result, we measure recall and precision for each distribution type. As we expected, precision is 1 for the uniform distribution for each β value. On the other hand, precision value is changed for the Gaussian distribution when β value goes up. With increasing β, we also increase the randomness and as a result, some unrated items might be selected as a rated item. Therefore, our precision value decreases. Precision is really important for us because we actually want to calculate how many real rated items are predicted among the items we marked rated. However, only precision results should not be enough to test our approach because when we compare our recall results with Experiment 2 and Experiment 3, recall results of Experiment 4 are very low. The remarkable thing in the experiment for Gaussian distribution and mix is that precision and β are inversely correlated. However, changing β does not affect recall. Recall is almost the same for each different β with the same σ. For example, Table 7 shows that recall results in mix distribution are 0.12852 and 0.12868 for β=1.5% and 12%, respectively.
  Although different β values may not affect recall results with the same σ, changing σ values decrease recall results. As σ goes up, the range of random numbers added to the original data increases. As a result, the real rated items, which fall out of the range, become less. As seen from Table 8, in uniform distribution, recall values are 0.60348 for σ=0.33 and 0.07791 for σ=3, respectively. As mentioned at the beginning of this experiment, precision results for uniform distribution are always 1 regardless of σ and β values.
  On the other hand, precision results decrease when σ values increase for Gaussian and mix distribution. The simple reason is that when σ increases, so does the randomness. The chance of the rated items falling out of the range decreases due to the randomness. Thus, precision results are better for small σ values compared the greater one. For example, while the precision (precR) is 0.99346 with σ=0.33, the precision is 0.65177 with σ=3 for Gaussian distribution.

Table 8
Recall and precision results based on distribution type's feature when β is 6%.

	σ	β	recallR	precR
Uniform Dist. (σ is fixed)	0.33	0.06	**0.60348**	**1**
	0.67	0.06	**0.34506**	**1**
	1	0.06	**0.23272**	**1**
	2	0.06	**0.1164**	**1**
	3	0.06	**0.07791**	**1**
Gaussian Dist. (σ is fixed)	0.33	0.06	0.35704	0.99346
	0.67	0.06	0.09169	0.97427
	1	0.06	0.03261	0.93102
	2	0.06	0.00726	0.75565
	3	0.06	0.00443	0.65177
Mix (σ is random)	0.33	0.06	0.59425	0.9898
	0.67	0.06	0.28241	0.97841
	1	0.06	0.12772	0.95293
	2	0.06	0.02771	0.81255
	3	0.06	0.01502	0.70946

- **Experiment 5:** In this experiment, we want to test our approach, which is related to using auxiliary information and SVD together. According to Experiment 4, precision results are very good; however, recall results are worse compared with Experiment 2 and Experiment 3. Therefore, using distribution type's feature alone is not enough to estimate the rated items, but it is helpful as auxiliary information when it is used with other methods. In this experiment, we first decide to use the distribution type's feature as in Experiment 4. We accept that ratings fall out of the range are voted by users. After marking those ratings, we use the approach in Experiment 3. Experiment 3 demonstrates good results although only auxiliary information is used to estimate the rated items. In this experiment, we make some small changes. We define some constraints about auxiliary information used in Experiment 3. If a movie is rated by at least 100,000 users in IMDb, we accept that the movie is likely rated by MovieLens's users. If a movie is voted by less than 5000 users in IMDb, in that case, we ignore the movie because we think that the movie is unlikely rated by MovieLens's users. The discarded movie is never selected as a rated item. After applying two different auxiliary information approaches, if the number of selected rated items is smaller than the number of estimated rated items, SVD is applied as in Experiment 2. As seen from Table 9, uniform distribution results are better than the others. Compared with the previous experiments, this experiment gives better results for greater β values. The results for small β values are similar. For example, recall and precision results of uniform distribution with β=12% are 0.72563 and 0.73120 while recall and precision results for the same parameters in Experiment 3 are 0.69066 and 0.69415, respectively. Recall and precision results for the same parameters in Experiment 2 are 0.64908 and 0.65236. Results show that using auxiliary information with SVD provides better results than using only auxiliary information or SVD. Our approach using auxiliary information with SVD also gives better

Table 9
Recall and precision results with σ being 1 when auxiliary information and SVD is used together.

	σ	β	recallR	precR	recallU	precU
Uniform Dist. (σ is fixed)	1	0.015	**0.89845**	**0.9169**	**0.62669**	**0.57296**
	1	0.03	**0.85797**	**0.8695**	**0.70811**	**0.68726**
	1	0.06	**0.79949**	**0.8077**	**0.78537**	**0.77649**
	1	0.12	**0.72563**	**0.7312**	**0.85006**	**0.84642**
Gaussian Dist. (σ is fixed)	1	0.015	0.88518	0.9099	0.59882	0.53139
	1	0.03	0.84196	0.8565	0.68008	0.65479
	1	0.06	0.77552	0.7850	0.76051	0.7503
	1	0.12	0.68866	0.6948	0.82992	0.82584
Mix (σ is random)	1	0.015	0.89336	0.9150	0.61971	0.55821
	1	0.03	0.85315	0.8662	0.70128	0.67793
	1	0.06	0.79126	0.8002	0.77722	0.76758
	1	0.12	0.71051	0.7163	0.84178	0.83799

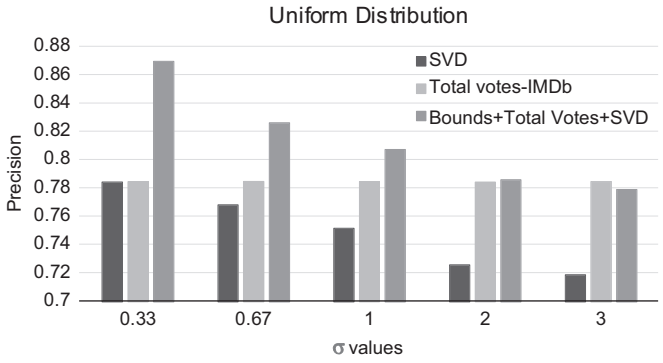

Fig. 3. Precision values for SVD, auxiliary information (total votes) and experiment 5 when uniform distribution with β=6% is used.

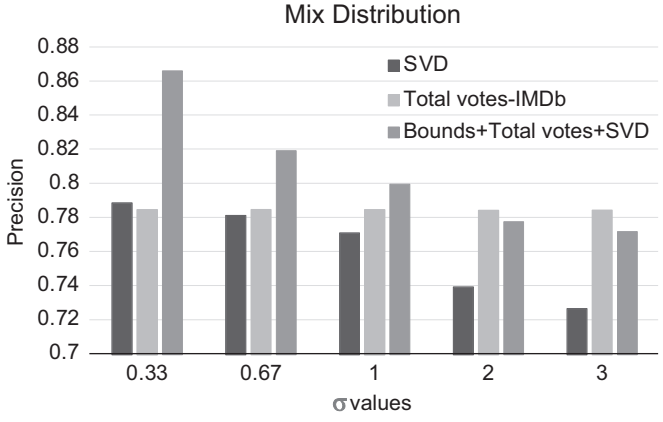

Fig. 4. Precision values for SVD, auxiliary information (total votes) and experiment 5 when mix distribution with β=6% is used.

results for estimating the fake items. For example, *RecallU* for uniform distribution when β=12% in Tables 9, 6, and 4 are 0.85006, 0.82892, and 0.80560, respectively.

Fig. 3 shows results of different methods when uniform distribution is used. We compare Experiment 2, called "SVD", Experiment 3, depicted "Total votes−IMDb", and Experiment 5, defined in the figure as "Bounds+Total votes+SVD", respectively. We set the β value to 6%. In this case, we change σ values while fixing β value. The results show that auxiliary information makes a great contribution than using only SVD while increasing σ. For small σ values, the contribution is much more than greater σ values. For greater σ values such as 2 or 3, only using the approach in used

Table 10
Recall and precision results based on SVD with σ=1 when β is random.

	σ	B	recallR	precR
Uniform Dist. (σ is fixed)	1	0.015	0.93653	0.93995
	1	0.03	0.89798	0.89941
	1	0.06	0.84127	0.84062
	1	0.12	0.76292	0.76239
Gaussian Dist. (σ is fixed)	1	0.015	0.93640	0.94063
	1	0.03	0.89800	0.90037
	1	0.06	0.84303	0.83932
	1	0.12	0.76398	0.75806
Mix (σ is random)	1	0.015	**0.93856**	**0.94191**
	1	0.03	**0.90200**	**0.90312**
	1	0.06	**0.84831**	**0.84967**
	1	0.12	**0.77899**	**0.77173**

Experiment 3 is enough rather than using all approaches together. The only reason is that when σ gets greater, the effect of distribution type (as seen from Experiment 4) lessens.

Fig. 4 displays similar results with Fig. 3. The results for mix are slightly worse than uniform distribution. However, the behavior of figures is the same. When the σ is larger, the gain decreases. Two figures demonstrate that either only auxiliary information or auxiliary information with SVD used to estimate the real rated items is much better than only using SVD.

- **Experiment 6:** We test how many real rated items are guessed within the fake and real rated items when each user selects the β value in a different way. In the previous experiments, each user has to use the same β value in the same way. This experiment allows users to select different β values to disguise their unrated items with random numbers. The maximum β value, β_{max}, is defined at the beginning of the experiment. Then, each user selects randomly β from the range. Predicting which items should be rated is more difficult than previous experiments because each user fills randomly unrated items with random numbers based on selected β. In the previous experiments, each user fills in unrated items with the same predefined β. The experiment is similar to Experiment 2. There are two different cases selecting σ values as it is explained in Experiment 2. Table 10 shows the results when σ is 1 and different β values are used. All results are very close for each distribution. For small β values, the results are very good. As β increases, the results gradually decrease. As seen from Table 10, the distribution type does not affect the results. Mix distribution is slightly better than the others. This also proves that the distribution type is inert. The only reason why mix distribution is better than the others is that σ is randomly selected by each user. The probability of selecting σ=1 for every user is smaller than the case in uniform or Gaussian distribution. We know that σ and accuracy are inversely correlated as in Experiment 2.

  When we compare Experiment 6 with Experiment 2, the best precision for mix distribution in Experiment 6 when β is 1.5% is 0.94191 while the best precision for the same case in Experiment 2 is 0.89960. The worst precision for mix distribution is obtained when β is 12%, which are 0.77173 and 0.68030, respectively. The results show that SVD gives better results with the approach used in Experiment 6.

  Fig. 5 compares precision results for both Experiment 6 and Experiment 2 when β is set 6% and σ gets different values. As explained at the beginning of Experiment 6, β is selected by each user randomly. As seen from Fig. 5, SVD gives better results in Experiment 6. The other important thing is that when σ gets larger, precisions are slightly worse although the drop of results in Experiment 2 is much more.

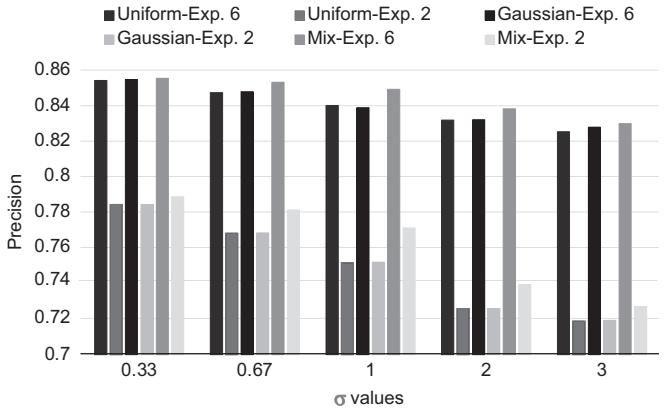

Fig. 5. Precision values for SVD whether β is fixed or random.

- **Experiment 7:** We perform experiments using auxiliary information with SVD to improve the results of Experiment 6. When β is random, we estimate the number of real ratings using β_{exp} value. In that case, we try to guess which users likely rate each item. On the other hand, when β is fixed as in Experiment 3 and Experiment 5, we can easily estimate how many real rated items should be selected using β value. In those experiments, we try to select which items should be rated by users and so we use total votes for each item. We cannot apply the same approach in this experiment. The question is which users might vote the specific item. We think that user demographic information can help guess items. MovieLens data set contains users' age. Also, each item has movie's types. Our hypothesis is that different age range like different movie types. Assume that the ages up to 25 like comedy movies, ages from 25 to 34 like action and adventure movies, ages 35–44 enjoy Sci-fiction, ages 45–54 like thriller, and ages 55 and older enjoy drama movies. In our experiment, we first apply SVD as in Experiment 6 and then we apply auxiliary information to improve our prediction. For example, if a movie's type is comedy, then we check whether that movie is rated by the ages up to 25. If there is a user in that range and the user is not selected, we mark the user and add to the results because we assume the user could rate the item. After adding a new user, we remove the last user from the SVD results. As seen from Fig. 6, our assumption does not hold, as we believed. Some movies have more than one movie's type.

We also test how results vary with different β values. As seen from Fig. 7, for small β values, our approaches are close to SVD results. However, as β gets larger, the gap between our approach and SVD expands.

8. Conclusions

We claim that each user has a different level of privacy concern. Various data disguising methods are proposed to protect user privacy. In this paper, we try to show that data disguising methods may not protect the privacy as much as thought. Even though users disguise their private data inconsistently, the system can guess the rated items. Experiments show that it is possible to predict which items could be voted by users. We first try to predict items using existing data reconstruction methods. A common property of the selected data reconstruction methods is to eliminate the noise. Since all data reconstruction methods need a complete data to perform their job, we evaluate different approaches to fill in all empty cells. Zero ratings approach wins among three approaches shown by empirical results. Besides, different data reconstruction

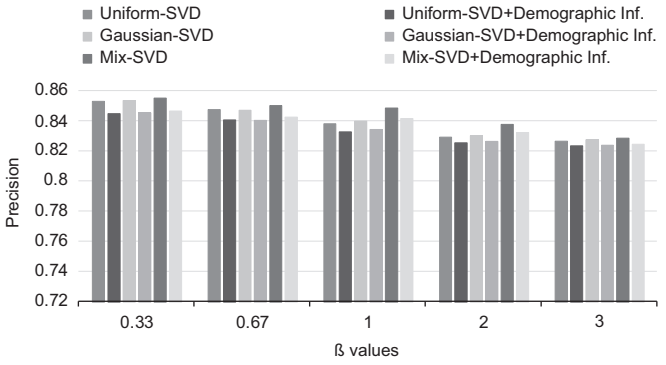

Fig. 6. Precision values for SVD and demographic information with SVD when β is 6% with various σ values.

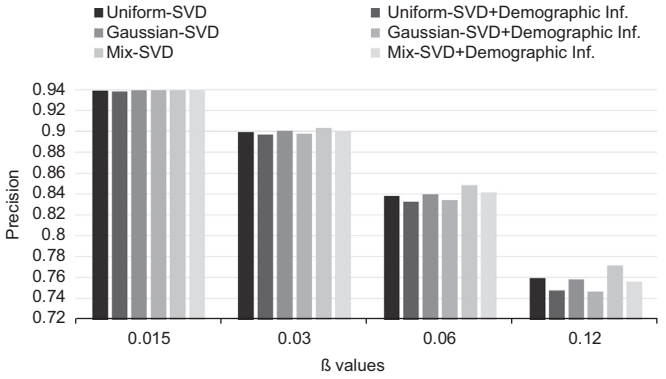

Fig. 7. Precision values for SVD and demographic information with SVD when σ is 1 with various β values.

methods are evaluated and SVD is chosen as a main method in our experiments due to its success. For $\sigma=1$, SVD has a 65-88% recall results in correctly predicting the real cells when the same β is used by each user in the same way although when each user picks varying β in the different way, SVD has a 76–94% recall results for choosing the real cells. Our experiments demonstrate that small σ values provide better results. As β values increase (the randomness increases), predicting the real items decrease. On the other hand, predicting the ratio of fake items increase. The other important outcome from the results of the experiments is that precision results are very close to recall results.

We also indicate that auxiliary information can help us improve the estimation results. The experiments show that while some powerful auxiliary information makes great contribution to the results, some auxiliary information does not. For example, the contribution of IMDb data (total votes for each item) is very high for prediction, while the user demographic information approach used in the experiment cannot make a contribution to our results. The distribution type's feature also provides advantages. Although randomization is used to disguise the original data, values, which fall out the range based on the distribution type, are accepted as the rated items. We show that precision and recall results are improved by using auxiliary information with SVD. For small σ values, the contribution is much more because the effect of the distribution type's feature decreases with the increasing σ values. Our experiments show that identifying correct and useful auxiliary information is an important issue because all auxiliary information may not help us as much as we think.

In this paper, we assume that a central server holds entire data set. However, data might be horizontally or vertically partitioned between two more parties. We are planning to investigate how to reconstruct private data in such cases. We are also planning to

modify existing data reconstruction methods to improve prediction accuracy.

Acknowledgment

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

References

[1] J.B. Schafer, J.A. Konstan, J.T. Riedl, E-commerce recommendation applications, Data Min. Knowl. Disc 5 (2001) 115–153, http://dx.doi.org/10.1023/A:1009804230409.
[2] J.L. Herlocker, J.A. Konstan, L.G. Terveen, J.T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Inform. Syst. 22 (2004) 5–53, http://dx.doi.org/10.1145/963770.963772.
[3] K. Ji, H. Shen, Making recommendations from top-N user-item subgroups, Neurocomputing 165 (2015) 228–237, http://dx.doi.org/10.1016/j.neucom.2015.03.013.
[4] Y.-C. Hu, Nonadditive similarity-based single-layer perceptron for multicriteria collaborative filtering, Neurocomputing 129 (2014) 306–314, http://dx.doi.org/10.1016/j.neucom.2013.09.027.
[5] L.F. Cranor, J. Reagle, M.S. Ackerman, Beyond Concern: Understanding Net Users' Attitudes about Online Privacy. AT&T Labs-Research Technical Report TR 99.4.3, 1999.
[6] J. Canny, Collaborative filtering with privacy, in: Proceedings of 2002 Symposium on Security Privacy, Oakland, CA, USA, 2002, pp. 45–57. doi:10.1109/SECPRI.2002.1004361.
[7] J. Canny, Collaborative filtering with privacy via factor Analysis, in: Proceedings of the 25th Annual International ACM SIGIR Conference on Research Development Information Retrieval, Tampere, Finland, 2002, pp. 238–245. doi:10.1145/564376.564419.
[8] A. Pfitzmann, M. Hansen, A Terminology for Talking about Privacy by Data Minimization: Anonymity, Unlinkability, Undetectability, Unobservability, Pseudonymity, and Identity Management, 2010.
[9] W. Yang, S. Qiao, A. Novel Anonymization, Algorithm: privacy protection and knowledge preservation, Expert Syst. Appl. 37 (2010) 756–766, http://dx.doi.org/10.1016/j.eswa.2009.05.097.
[10] S.L. Warner, Randomized response: a survey technique for eliminating evasive answer bias, J. Am. Stat. Assoc. 60 (1965) 63–69, http://dx.doi.org/10.2307/2283137.
[11] C. Kaleli, H. Polat, Providing private recommendations using Naïve Bayesian classifier, in: K. Wegrzyn-Wolska, P. Szczepaniak (Eds.), Advance Intellectual Web Mastering, Springer Berlin Heidelberg, 2007, pp. 168–173, http://dx.doi.org/10.1007/978-3-540-72575-6_27.
[12] P. Bogetoft, D.L. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J.D. Nielsen, J.B. Nielsen, K. Nielsen, J. Pagter, M. Schwartzbach, T. Toft, Secure multiparty computation goes live, in: Financial Cryptography and Data Security, Springer, 2009, pp. 325–343.
[13] Y. Lindell, B. Pinkas, Secure multiparty computation for privacy-preserving data mining, J. Priv. Confidentiality 1 (2009) 5.
[14] R. Wright, Z. Yang, Privacy-preserving Bayesian network structure computation on distributed heterogeneous data, in: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery Data Mining, Seattle, WA, USA, 2004, pp. 713–718. doi:10.1145/1014052.1014145.
[15] R. Agrawal, R. Srikant, Privacy-preserving data mining, SIGMOD Rec. 29 (2000) 439–450.
[16] H. Polat, W. Du, Privacy-preserving collaborative filtering using randomized perturbation techniques, in: Proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, FL, USA, 2003, pp. 625–628.
[17] H. Kargupta, S. Datta, Q. Wang, K. Sivakumar, On the privacy preserving properties of random data perturbation techniques, in: Proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, FL, USA, 2003, pp. 99–106.
[18] Z. Huang, W. Du, B. Chen, Deriving private information from randomized data, in: Proceedings of the 2005 ACM SIGMOD International Conference on Managing Data, Baltimore, MD, USA, 2005, pp. 37–48.
[19] S. Zhang, J. Ford, F. Makedon, Deriving private information from randomly perturbed ratings, in: Proceedings of the 6th SIAM International Conference on Data Mining, Bethesda, MD, USA, 2006, p. 59.
[20] H. Polat, W. Du, Effects of Inconsistently masked data using RPT on CF with privacy, in: Proceedings of the 2007 ACM Symposium on Applications Computing, Seoul, Korea, 2007, pp. 649–653.
[21] I. Yakut, H. Polat, Achieving pPrivate SVD-based recommendations on inconsistently masked data, in: Proceedings of the 1st International Conference on Security of Information and Networks, Gazimagusa, North Cyprus, 2007, pp. 172–176.
[22] A. Bilge, H. Polat, Improving privacy-preserving NBC-based recommendations by preprocessing, in: Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence Agent Technology, Toronto, Canada, 2010, pp. 143–147. doi:10.1109/WI-IAT.2010.109.
[23] A. Bilge, H. Polat, An improved privacy-preserving DWT-based collaborative filtering scheme, Expert Syst. Appl. 39 (2012) 3841–3854, http://dx.doi.org/10.1016/j.eswa.2011.09.094.
[24] A. Bilge, H. Polat, A comparison of clustering-based privacy-preserving collaborative filtering schemes, Appl. Soft Comput. 13 (2013) 2478–2489, http://dx.doi.org/10.1016/j.asoc.2012.11.046.
[25] H. Polat, W. Du, SVD-based collaborative filtering with privacy, in: Proceedings of the 2005 ACM Symposium on Applied Computing, Santa Fe, NM, USA, 2005, pp. 791–795.
[26] D. Agrawal, C.C. Aggarwal, On the design and quantification of privacy preserving data mining algorithms, in: Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principle Database System, Santa Barbara, CA, USA, 2001, pp. 247–255.
[27] H. Kargupta, S. Datta, Q. Wang, K. Sivakumar, Random-data perturbation techniques and privacy-preserving data mining, Knowl. Inf. Syst. 7 (2005) 387–414, http://dx.doi.org/10.1007/s10115-004-0173-6.
[28] S. Guo, X. Wu, On the use of spectral filtering for privacy preserving data mining, in: Proceedings of the 2006 ACM Symposium on Applied Computing, Dijon, France, 2006, pp. 622–626.
[29] S. Guo, X. Wu, Y. Li, On the lower bound of reconstruction error for spectral filtering based privacy preserving data mining, Lect. Notes Artif. Intell. 4213 (2006) 520–527, http://dx.doi.org/10.1007/11871637_51.
[30] S. Guo, X. Wu, Y. Li, Determining error bounds for spectral filtering based reconstruction methods in privacy preserving data mining, Knowl. Inf. Syst. 17 (2008) 217–240, http://dx.doi.org/10.1007/s10115-008-0123-9.
[31] J.A. Calandrino, A. Kilzer, A. Narayanan, E.W. Felten, V. Shmatikov, "You Might Also Like:" privacy risks of collaborative filtering, in: Proceedings of the 2011 IEEE Symposium on Security Privacy, Berkeley, CA, USA, 2011, pp. 231–246. doi:10.1109/SP.2011.40.
[32] M. Jiang, P. Cui, X. Chen, F. Wang, W. Zhu, S. Yang, Social recommendation with cross-domain transferable knowledge, IEEE Trans. Knowl. Data Eng. 27 (2015) 3084–3097, http://dx.doi.org/10.1109/TKDE.2015.2432811.
[33] S. Liu, S. Wang, F. Zhu, J. Zhang, R. Krishnan, HYDRA: large-scale social identity linkage via heterogeneous behavior modeling, in: Proceedings of the 2014 ACM SIGMOD International Conference on Management Data, Snowbird, UT, USA, 2014, pp. 51–62. doi:10.1145/2588555.2588559.
[34] S. Liu, S. Wang, F. Zhu, Structured learning from heterogeneous behavior for social identity linkage, IEEE Trans. Knowl. Data Eng. 27 (2015) 2005–2019, http://dx.doi.org/10.1109/TKDE.2015.2397434.
[35] A. Mnih, R. Salakhutdinov, Probabilistic matrix factorization, Proceedings of the Advance Neural Information Processing System 20 (2007) 1257–1264.
[36] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37, http://dx.doi.org/10.1109/MC.2009.263.
[37] M. Jiang, P. Cui, F. Wang, W. Zhu, S. Yang, Scalable recommendation with social contextual information, IEEE Trans. Knowl. Data Eng. 26 (2014) 2789–2802, http://dx.doi.org/10.1109/TKDE.2014.2300487.
[38] J.S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, in: Proceedings of the 14th Conference on Uncertainty and Artificial Intelligence, San Francisco, CA, USA, 1998, pp. 43–52. http://dl.acm.org/citation.cfm?id=2074094.2074100.
[39] M. Okkalioglu, M. Koc, H. Polat, On the discovery of fake binary ratings, in: Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, 2015, pp. 901–907. doi:10.1145/2695664.2695866.
[40] H. Polat, W. Du, Achieving, Private recommendations using randomized response, Tech. Lect. Notes Comp. Sci. 3918 (2006) 637–646, http://dx.doi.org/10.1007/11731139_73.
[41] H. Abdi, L.J. Williams, Principal component analysis, Wiley Interdiscip. Rev. Comput. Stat. 2 (2010) 433–459, http://dx.doi.org/10.1002/wics.101.
[42] M.L. Johnson, Essential Numerical Computer Methods, Academic Press, 2010.
[43] N. Ahmed, T. Natarajan, K.R. Rao, Discrete cosine transform, IEEE Trans. Comput. C-23 (1974) 90–93, http://dx.doi.org/10.1109/T-C.1974.223784.
[44] K.R. Rao, P. Yip, Discrete Cosine Transform: Algorithms, Advantages, Applications, Academic Press, 2014.

Burcu D. Okkalioglu received her BSc degree from Computer Engineering Department at Pamukkale University, Denizli, Turkey in 2007. She received her MSc degree from Computer Science Department at the University of Texas at San Antonio in 2012. She is currently a PhD student in the Computer Engineering Department at Anadolu University. Her research interests include data mining, privacy-preserving data mining, and deriving private numeric data from masked data.

Mehmet Koc received his two BSc degrees from Mathematics and Electrical–Electronics Engineering Departments, MSc degree from Electrical and Electronics Engineering Department at Eskisehir Osmangazi University, Eskisehir, Turkey; and PhD degree from Anadolu University, Eskisehir, Turkey in 2003, 2004, 2006, and 2012, respectively. Currently he is an Assistant Professor in Electrical–Electronics Engineering department at Bilecik Seyh Edebali University, Bilecik, Turkey. His research interests include pattern recognition, computer vision, and data mining.

Huseyin Polat is an Associate Professor in Computer Engineering Department at Anadolu University, Turkey. He got his Master's degree and PhD from Computer Science Department at Syracuse University in 2001 and 2006, respectively. His research interests are primarily collaborative filtering with privacy, private predictions on selected models, and privacy-preserving data mining in general.