Shilling Attacks Against Memory-Based Privacy-Preserving Recommendation Algorithms

Ihsan Gunes[1], Alper Bilge[1] and Huseyin Polat[1]
[1] Computer Engineering Department, Anadolu University, 26555 Eskisehir, Turkey
[E-mail: ihsang@anadolu.edu.tr; abilge@anadolu.edu.tr; polath@anadolu.edu.tr]
*Corresponding author: Huseyin Polat

Received January 29, 2013; revised April 19, 2013; accepted May 6, 2013; published May 31, 2013

Abstract

Privacy-preserving collaborative filtering schemes are becoming increasingly popular because they handle the information overload problem without jeopardizing privacy. However, they may be susceptible to shilling or profile injection attacks, similar to traditional recommender systems without privacy measures. Although researchers have proposed various privacy-preserving recommendation frameworks, it has not been shown that such schemes are resistant to profile injection attacks. In this study, we investigate two memory-based privacy-preserving collaborative filtering algorithms and analyze their robustness against several shilling attack strategies. We first design and apply formerly proposed shilling attack techniques to privately collected databases. We analyze their effectiveness in manipulating predicted recommendations by experimenting on real data-based benchmark data sets. We show that it is still possible to manipulate the predictions significantly on databases consisting of masked preferences even though a few of the attack strategies are not effective in a privacy-preserving environment.

Keywords: Shilling, privacy, robustness, recommendation, profile injection, collaborative filtering

A preliminary version of this paper was presented in the International Conference on Innovation and Information Management (*ICIIM*), Singapore, 2013. While the preliminary version provided the motive and basic idea of our work, this extended version includes another novel idea and more detailed descriptions of related work, analysis and experiments. In addition, all sections are revised to improve presentation and clarity. This extended version contains more than 30 percent additional substantial contributions. This work was partially supported by the Grant 111E218 from TUBITAK.

http://dx.doi.org/10.3837/tiis.2013.05.019

1. Introduction

In the past two decades, recommender systems have become a common technological solution to handle the information overload problem because they can filter out products or services from a large number of candidates. In particular, online vendors employing recommender systems increase their sales and/or profits by offering automated personalized predictions about products. Collaborative filtering (CF) is one of the most popular recommendation approaches. Using CF, items under consideration are chosen using past preferences of like-minded users' tastes. CF can be used to recommend books, movies, CDs, and devices, among other products [1][2].

In a typical CF system, a central server collects preferences (either implicitly or explicitly) from many users about products they previously purchased or services they utilize. Such preferences are expressed as numeric or binary ratings and are recorded into a user-item matrix. An active user typically requests a prediction on a particular product after submitting her ratings to the recommender system. The recommendation estimation process, or CF, is a two-step process [1]: (*i*) a neighborhood is formed using similarities between the active user and other users, and (*ii*) a prediction is estimated as a weighted average based on the neighbors' ratings of the target product. Rather than similarity, consistency, based on information entropy between two users, can be employed as a trust metric for improved CF [3].

Because of the vulnerability that is inherent in the systems' inability to distinguish genuine users from malicious ones, CF systems can be subjected to different types of profile injection, or shilling attacks. Retailers or competing companies might insert fake profiles into user-item matrices to manipulate predictions in favor of their own products or to damage the recommender systems' accuracy. CF systems are known to be susceptible to such attacks, referred to as shilling attacks [4][5]. Using as much information as possible about a CF scheme, the attackers create bogus profiles and insert them into the system, pretending to be authentic users. The intent is to bias estimated predictions in favor of the attackers by either increasing or decreasing the popularity of targeted products, classifying them as *push* or *nuke* attacks, respectively [6].

It is well known that involvement in Internet services may pose a serious risk to individuals' confidential information. As collected preferences might jeopardize personal information and profile users' habits with respect to purchasing, viewing, traveling, and other activities, individual privacy is seriously threatened. Some bankrupted companies have even sold these types of data in the past, because authentic user records are very valuable assets [7]. Thus, as discussed in [8], a significant number of users either submit fabricated ratings or, even worse, abstain from using the services, even if they can benefit in return. As dependable predictions can only be estimated from unpretentious data, the goal is to produce recommendations through privacy-preserving collaborative filtering (PPCF) schemes without violating individual privacy, while still maintaining some accuracy.

Various schemes have been proposed to provide privacy measures within CF applications [9][10][11]. They are mostly based on randomized perturbation approaches, i.e., performing some algebraic operations on the aggregate data while not revealing values of individual data items. Principally, those schemes can be associated with one of three categories, according to the approach used, i.e., they are (*i*) memory-based, (*ii*) model-based, or (*iii*) hybrid methods. Memory-based methods typically run over the entire database during the online prediction

production process. These methods are more successful with respect to the quality of predictions as compared to model-based approaches, which operate on a prototype constructed from the original user-item matrix off-line [12][13]. Hybrid approaches employ both methods' advantages for better performance; however, it is difficult to fine-tune the parameters of hybrid approaches [14][15].

Although PPCF methods can be manipulated through shilling attacks, their robustness against specific attack strategies has not been thoroughly analyzed. Moreover, methods to design shilling attacks against masked data have not been studied. We examine two popular memory-based PPCF schemes based on two variations of the neighborhood-based prediction algorithm [16]. We propose procedures to design shilling profiles in a privacy-preserving environment for six broadly implemented attack strategies, i.e., random, average, bandwagon, and segment push attacks and reverse bandwagon and love/hate nuke attacks. We then investigate the robustness of two memory-based PPCF algorithms under these attacks. Major contributions of the study are as follows:

1. Design methodologies are proposed to revise some principal attacks to be implemented in privacy-preserving environments.
2. Two primary memory-based PPCF algorithms are examined in terms of robustness when exposed to formerly proposed attacks.
3. It is experimentally shown that while the PPCF algorithms are very robust against a couple of attacks, they are still as vulnerable as traditional CF schemes against other types of attacks.

The remainder of the paper is organized as follows. In Section 2, we review the related work in CF, PPCF, and shilling attacks, and we explain the need for and significance of this study. We then briefly present preliminaries in Section 3. Section 4 describes methods to design push and nuke shilling attack strategies based on privately collected data. Real data-based experiments and empirical results are presented in Section 5. Finally, we present the conclusions and discuss some possible future work in Section 6.

2. Related Work

CF and PPCF schemes are deployed primarily by e-commerce companies to increase sales by attracting and influencing customers. However, because of the authentication mechanism involved in such services, they often do not detect the injection of malicious profiles. Using this breach, shilling attacks directed towards the systems can either affect a specific items' popularity or damage the system's quality of predictions.

The concept of CF began in the early 1990s. The term was first coined by the Tapestry system [17], which was designed for e-mail filtering. Briefly, CF operates on collected preferences from many users and estimates predictions relying on similar entities' ratings by employing a weighted average approach [1][18]. A type of widespread implementation of CF is the memory-based application, where similarities are calculated over all pairwise entries (either users or items) using a similarity metric, such as Pearson's correlation coefficient, the cosine similarity, or trust [12][19]. Many successful CF systems with respect to the quality of predictions utilize user-based methods [20]. Those systems marketing over a large number of types of products prefer item-based solutions [21].

To eliminate privacy risks posed by online vendors utilizing CF schemes, such as price discrimination, profiling, and unsolicited marketing, researchers have developed PPCF

techniques [9][22]. Initial approaches to protect individual privacy include the distributed solutions proposed by Canny [7][11], where an aggregate of user data is collected privately to hide confidential information about users by implementing cryptographic techniques. However, it is not feasible to implement this approach. Thus, central server-based systems have been designed in which users submit their preferences after disguising them to ensure protection of their data, yet allowing performance of some algebraic operations to produce accurate predictions. Such masking procedures, called data obfuscation methods, allow filtering processes without violating confidentiality. Primary disguising procedures in the literature include randomized perturbation techniques [9][23][24], randomized response techniques [25][26], and data substitution [10].

The idea of attacking recommender systems originated from the study conducted by Dellarocas [27]. In that study, a set of mechanisms were proposed to alert online reputation reporting systems of fraud. That idea led to the shilling or profile injection attack concept by O'Mahony et al. [4][5], in which they discuss vulnerabilities of recommender systems against manipulations to promote particular products. Since then, researchers have studied defining possible attacking strategies [6][28], detecting attacks [29][30], upgrading the robustness of CF systems or proposing robust algorithms against known attacks [31][32][33]. Zhang [34] describes several attack models and explains some well-known attack detection strategies, such as model-specific attributes. He also evaluates the effects of measuring metrics on attacks and detection schemes. O'Mahony [31] showed that capturing some basic statistical information about the user-item matrix might be enough to create shilling attacks against central server-based CF systems. Lam and Riedl [35] discussed properties that affect shilling attacks, such as the recommendation algorithm, targeted products, and detectability of attacks. The authors investigated privacy with respect to shilling attacks and the value of the information [36]. Mobasher et al. [37] classified attacks according to the knowledge required to create them. This knowledge might sometimes be considered low level knowledge, such as favorite items overall, or high level knowledge, such as standard deviation of ratings for each item in a database. Recently, Gunes et al. [38] presented a comprehensive survey covering up-to-date research on shilling attacks and comprehensively analyzed attack descriptions, detection methods, robust algorithm design, and cost/benefit analysis, including descriptions of evaluation components, such as commonly utilized data sets and metrics.

Although there are extensive studies focusing on responses of recommender systems to promoting attacks in non-private schemes, they fail to protect individual privacy. Additionally, formerly proposed privacy enhancing methods applied to recommender systems have not been studied for their robustness against shilling attacks. Therefore, each group of studies focuses on just one aspect of recommender system technology. We distinctively focus on designing shilling attacks against systems that provide private recommendation facilities, thus extending our previous work [39], in which we studied a limited number of shilling attacks through one memory-based prediction algorithm. We also propose a new attack design mechanism, which is more suitable to randomized perturbation-based PPCF schemes, and evaluate two varieties of well-known neighborhood-based prediction algorithms.

3. Background

3.1 Memory-Based Recommendation Algorithms

CF systems typically collect ratings and form a user-item matrix, $U_{n\times m}$, containing preference information from n users on m items. When an active user (a) interacts with a CF system, she

queries for a prediction on a target item (q), after sending her ratings vector. Then, the central server produces a prediction. It first calculates similarities between a and other users in the system to form a neighborhood. The prediction algorithm proposed by [16], which is also the algorithm used in PPCF frameworks, employs a variance weighting onto neighbors via z-score normalization. In this scheme, users submit their z-score normalized ratings instead of actual ratings. The z-score normalized ratings are calculated as $z_{uj} = (v_{uj} - \overline{v_u})/\sigma_u$ ($j = 1, 2, \ldots, m$) for each available rating v_{uj} where $\overline{v_u}$ and σ_u are the mean and standard deviation of ratings for the user u, respectively. The prediction is estimated as a weighted average of the z-scores, as follows:

$$p_{aq} = \overline{v_a} + \sigma_a \times \frac{\sum_{u \in N} w_{au} \times z_{uq}}{\sum_{u \in N} w_{au}} \tag{1}$$

where N is the set of neighbors and w_{au} is the similarity weight between a and u, which can be calculated as follows:

$$w_{au} = \frac{\sum_{j \in M}(v_{aj} - \overline{v_a}) \times (v_{uj} - \overline{v_u})}{\sigma_a \times \sigma_u} = \frac{z_{aj} \cdot z_{uj}}{\#M} \tag{2}$$

where M is the set of items rated by both a and user u.

There are two methodologies to determine neighbors that enter into the prediction estimation process. In the former approach, the most similar k neighbors are selected. This method is the k-nearest neighbor (k-nn) recommendation algorithm. This algorithm tends to positively consider correlated neighbors only, sorted by their similarity values [1][16]. Unlike this original approach, a modified version of the k-nn algorithm, the correlation-threshold method, also takes negatively correlated users into account that have absolute values of similarity higher than a threshold value (τ) [16]. Because of the negatively correlated neighbors, absolute values of the similarity weights are used in the denominator of Eq. (1). The modified version thus only filters out users with negligible correlation but includes both positively and negatively correlated users in the prediction process. Thus, the exact number of neighbors is not definite in the correlation-threshold algorithm.

3.2 Privacy Protection by Randomization Techniques

Traditional CF schemes pose various privacy risks such as price discrimination, customer profiling, government surveillance, and unsolicited marketing [9][22]. Because of these risks, users tend to give false data or refuse to give data at all. If their privacy is guaranteed, they can provide their preferences about various products as ratings and feel more comfortable providing true data. To achieve confidentiality in PPCF schemes, confidential data are masked using a variety of data disguising methods. Randomized perturbation is one of the more commonly used methods for data masking. Customers' concerns about confidentiality have a variety of causes, such as the value of the data, the type of data, and their feelings about privacy. The phenomena can be described as follows:

Ratings. Customers do not want to share their direct preferences about the products they purchase. Thus, ratings for various items are regarded as private data, and they are disguised prior to submission to the CF system.

Rated/unrated items. Revealing the rated/unrated products might also pose serious privacy risks. Many customers want to hide specific products that they buy or show interest in, such as pornographic magazines, ideological publications, etc. Similarly, customers who do not want unsolicited marketing might want to hide what products they have not rated (and, correspondingly, not purchased). Therefore, ratings and rated/unrated products are considered to be confidential data, and users mask both ratings and rated/unrated items.

Data amount. The amount of private data differs among users because of the varying value

of data and different data types.

Level of privacy. The privacy level each user needs might be different. Some users care deeply about their privacy; for others, privacy may not be that important.

Due to the variation in privacy concerns, users can mask their data as follows:

1. *Data distribution selection*: Each user u decides whether to use a uniform or Gaussian distribution with mean (μ) equal to 0 and standard deviation (σ) generated randomly. Thus, the users uniformly randomly choose random data distribution. Notice that different data distribution might provide different privacy levels.
2. *Determining the value of* σ: Given σ_{max}, each user u uniformly randomly or selectively chooses the value of σ_u from the range $(0, \sigma_{max}]$. With increasing value of σ_{max} (correspondingly the value of σ_u), the level of perturbation increases; thus, confidentiality is increased.
3. *Choosing the amount of data to be masked*: Users decide the number of data items to be disguised. Each user u uniformly randomly or selectively chooses a value (β_u) over the range $(0, \beta_{max}]$, where β_{max} is a parameter whose value is predetermined by the server with respect to data density. They then uniformly randomly or selectively choose β_u% of the data items (m_p) to be disguised.
4. *Random number generation*: Users generate as many random numbers as they need. (The quantity of random numbers to be generated is determined in the third step.) In other words, they create r_j values for $j = 1, 2, \ldots, m_p$.
5. *Masking*: Users add each random number r_j to the corresponding confidential data item v_j ($v_j' = v_j + r_j$ for $j = 1, 2, \ldots, m_p$) that is chosen to be masked. Confidential data items can be rated and/or unrated items.

According to this data perturbation scheme, we assume the following while designing shilling attacks: (*i*) half of the users employ uniform, while the other half employ Gaussian distributions; (*ii*) the value of σ_{max} is set at 2 to provide a reasonable level of privacy [40]; and (*iii*) users mask all of their true ratings and uniformly randomly or selectively choose β_u% of the unrated items, where β_{max} is set to an optimum value of 25 [23].

3.3 Shilling Attack Profiles

To affect the outcomes of the CF schemes to their advantage, attackers inject fake profiles into the database under attack. A generic attack profile is shown in **Fig. 1** [37][41]. A set of items, I_S, is determined by the attacker, with a particular rating function δ, to form the characteristics of the attack. Another set of items, I_F, is selected randomly, with a rating function θ, to obstruct detection of an attack. A unique item is targeted, with a rating function Υ, on which to form a bias. The remaining I_ϕ items remain unrated.

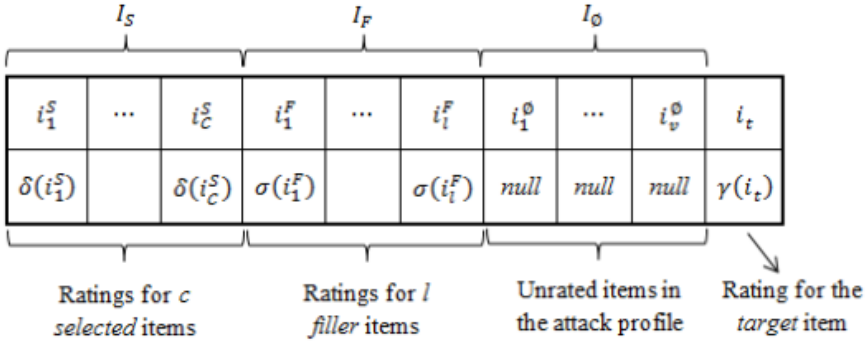

Fig. 1. General form of an attack profile

The basic approach when attacking recommender systems is to insert as many bogus profiles as possible into the neighborhoods of users to obtain strong correlations to manipulate prediction values. The most well-known attack models are random, average, bandwagon, and segment for push attacks; and reverse bandwagon and love/hate for nuke attacks [28][29][42][43]. We present attack profiles for these attack types in **Table 1,** using the general attack profile given in **Fig. 1**.

Table 1. Attack profile summary

Attack type	I_S		I_F		I_ϕ	i_t
	Items	Rating	Items	Rating		
Random	N/A		Randomly chosen	system mean	$I - I_F$	r_{max}
Average	N/A		Randomly chosen	item mean	$I - I_F$	r_{max}
Bandwagon	Popular items	r_{max}	Randomly chosen	system mean	$I - \{I_F \cup I_S\}$	r_{max}
Segment	Segmented items	r_{max}	Randomly chosen	r_{min}	$I - \{I_F \cup I_S\}$	r_{max}
Reverse BW	Unpopular items	r_{min}	Randomly chosen	system mean	$I - \{I_F \cup I_S\}$	r_{min}
Love/hate	N/A		Randomly chosen	r_{max}	$I - \{I_F \cup I_S\}$	r_{min}

4. Designing Attack Models against Databases Including Perturbed Data

PPCF applications collect disguised data from users to protect customers' privacy. Although various attack models are designed against non-private rating collections, they cannot be applied in a straightforward way to databases consisting of masked data. Because users perturb their preferences before submitting them to PPCF servers, an attacker can only gather information about the masked ratings, which requires some modifications to the attack models to be implemented. Effects of various shilling attack strategies on privacy-preserving frameworks have not been extensively studied. In this section, we describe how to design six famous attack models so that they can be applied to perturbed data and then analyze the robustness of two memory-based PPCF algorithms against the revised profile-injection attack models.

Before generating any type of shilling profile, the attackers need to decide the random number distribution as uniform or Gaussian to generate random numbers and select σ_p uniformly randomly from the range $(0, \sigma_{max}]$ for each attack profile p. After determining these parameters, the attacks can be created on disguised databases, as described in the following sections. After discussing how to create push attack models, we analyze the design of nuke attack models.

4.1 Designing Push Attack Models

4.1.1 The Random Attack Model

A random attack is relatively easy to implement and is a baseline model that requires low knowledge compared to the other attack models [28]. Accordingly, a random attack model can be characterized to attack databases including masked data, as follows:

1. The set of selected items is empty ($I_S = \emptyset$).
2. A total of l filler items (I_F) are uniformly randomly selected from all the items except the target item ($I - \{i_t\}$) with respect to a predetermined value of filler size parameter.
3. Using the chosen distribution, l+1 random numbers are generated with μ equal to zero and σ equal to σ_p.

4. The maximum of the generated random numbers is assigned to the target item and the remaining numbers are arbitrarily assigned to the l filler items.
5. All users can be targeted via the resulting shilling profiles.

4.1.2 The Average Attack Model

The average attack is difficult to implement because it requires a high level of knowledge about the system [38][44]. The filler items contain values around each item's mean vote, which requires computation of the averages of each item in the system. Correspondingly, average attack profiles should be modified so that they can be used to attack masked databases, as follows:

1. The set of selected items is empty ($I_S = \emptyset$).
2. A total of l filler items (I_F) are uniformly randomly selected from all items, excluding the target item ($I - \{i_t\}$), using a predetermined value of the filler size parameter.
3. Using the chosen distribution, l random numbers ($r_1, r_2, \ldots, r_l$) are generated in the interval $[-\alpha, \alpha]$, where α is a masking parameter used to avoid disclosure of the attack profile.
4. Each derived random number is added to the corresponding item's mean vote, i.e., $v_i = x_i + r_i$, $i = 1, 2, \ldots, l$, where x_i represents the mean number of votes for the item i and v_i corresponds to the value for the item i in the shilling profiles.
5. Finally, using the chosen distribution, l additional random numbers ($t_1, t_2, \ldots, t_l$) are generated, where μ is equal to zero and σ is equal to σ_p. The maximum of the generated random numbers is assigned to the target item, i.e., $i_t = \max(t_i)$.
6. All users can be targeted via the resulting shilling profiles.

4.1.3 The Bandwagon Attack Model

The bandwagon attack, also known as the popular attack, is a low-knowledge push attack requiring public information about items. Items known to be popular are given high ratings in shilling profiles to abuse users' interest in generally appreciated products [6]. The same strategy can be followed to create the attack model for private collections. Although users disguise their preferences before submission, popular items can still be detected with some accuracy by relying on aggregate data as the number of users increases in the system. The mean votes for items can be estimated as follows:

$$\overline{V_j'} = \frac{\sum_{j \in N} v_j'}{\#N} = \frac{\sum_{j \in N} (v_j + r_j)}{\#N} = \frac{\sum_{j \in N} v_j + \sum_{j \in N} r_j}{\#N} \approx \frac{\sum_{j \in N} v_j}{\#N} \approx \overline{V_j} \tag{3}$$

where $\overline{V_j'}$ is the mean number of votes calculated from the disguised data, $\overline{V_j}$ is the real mean for item j, and N is the set of users who rated item j. As N grows, the expected value of the mean of the random numbers tends to converge to zero because they are all generated from a zero-mean distribution. Thus, the masked bandwagon attack can be implemented as follows:

1. Among the items with the highest averages, a total of c popular products are selected ($I_S = \{p_1, p_2, \ldots, p_c\}$).
2. A total of l filler items (I_F) are uniformly randomly selected among all items except the target item and the selected items ($I - \{i_t \cup I_S\}$) using a predetermined value for the filler size parameter.
3. Using the chosen distribution, $l+c+1$ random numbers ($r_1, r_2, \ldots, r_{l+c+1}$) are created, where the mean is equal to zero and σ is equal to σ_p.
4. The maximum of the generated random numbers is assigned to the target item.

5. Then, the top c of the remaining random numbers are arbitrarily assigned to selected popular items.
6. Finally, the leftover random numbers are arbitrarily placed into l filler items.
7. All users can be targeted via the resulting shilling profiles.

4.1.4 The Segment Attack Model

The segment attack model targets a subset of users that have shown interest in a certain type of product, such as fantastic movies or jazz music [42]. The segment is composed of users who have rated highly most of the selected items. In this way, the attacker tries to abuse the segmented users' positive interest in specific items to push estimated predictions for a target product. Attack profiles targeting segmented users can also be created in private systems, as follows:

1. A total of h products with high average ratings are selected with a certain and common property ($I_S = \{p_1, p_2, \ldots, p_h\}$).
2. A total of l filler items (I_F) are uniformly randomly selected among all items except the target item and selected items ($I - \{i_t \cup I_S\}$) using the predetermined value of the filler size parameter.
3. Using the chosen distribution, l+h+1 random numbers ($r_1, r_2, \ldots, r_{l+h+1}$) are generated, where the mean is equal to zero and σ is equal to σ_p.
4. The maximum of the generated random numbers is assigned to the target item.
5. Then, the top h of the remaining random numbers are assigned arbitrarily to selected segment items.
6. Finally, the remaining random numbers are arbitrarily placed in l filler items.

Users to be attacked, or segmented users, are from private collections and have positively rated at least P% of the selected items.

We design the abovementioned four attack models as push attacks to increase the popularity of some targeted products in the masked database. In addition to these attack models, we design two attack models as nuke attacks to decrease the popularity of targeted items in the perturbed database, as discussed in the following sections.

4.2 Design of the Nuke Attack Models

4.2.1 The Reverse Bandwagon Attack Model

The reverse bandwagon attack is the nuking version of the bandwagon attack. It is particularly effective against item-based algorithms [6]. This attack model does not require any system specific data, but it does require a general knowledge of the product domain, similar to the bandwagon attack, to effectively select low-rated items. Therefore, reverse bandwagon attack profiles for masked data can be created using a method similar to the related bandwagon attack profiles, with slight differences, as follows:

1. From the items with the lowest averages and highest number of ratings, a total of c unpopular products are selected ($I_S = \{p_1, p_2, \ldots, p_c\}$).
2. A total of l filler items (I_F) are uniformly randomly selected from among all items except the target item and selected items ($I - \{i_t \cup I_S\}$) using the predetermined value of the filler size parameter.
3. Using the chosen distribution, l+c+1 random numbers ($r_1, r_2, \ldots, r_{l+c+1}$) are generated, where μ is equal to zero and σ is equal to σ_p.

4. The minimum of the generated random numbers is assigned to the target item.
5. Then, the lowest c of the remaining random numbers are assigned arbitrarily to selected unpopular items.
6. Finally, the remaining random numbers are arbitrarily placed into l filler items.
7. All users can be targeted via the resulting shilling profiles.

4.2.2 The Love/Hate Attack Model

The love/hate attack is one of the most effective models to nuke predictions in user-based CF systems. In addition to its simplicity, this model does not require any knowledge about the system [37]. Using the love/hate attack model to shill the perturbed database can be described as follows:

1. The set of selected items is empty ($I_S = \emptyset$).
2. A total of l filler items (I_F) are uniformly randomly selected among all items but the target item ($I - \{i_t\}$) using a predetermined value of the filler size parameter.
3. Using the chosen distribution, $C \times l$ random numbers are generated, where the mean is equal to 0, σ is equal to σ_p, and C is a constant used to ensure inserting high ratings into the profiles.
4. The maximum l of the generated random numbers are arbitrarily assigned to filler items, and the minimum number is assigned to the target item.
5. All users can be targeted via the resulting shilling profiles.

In order to provide a visual idea of the nature of attacks, we give examples of attack profiles in Fig. 2. We assume that there are 11 items and the last item (m11) corresponds to the target item for each attack profile. Moreover, black, gray, and white cells represent the selected, filler, and empty/unrated items, respectively. The target item is assigned to the maximum and minimum of the generated random numbers for push and nuke attacks, respectively. For all attack profiles, five items (m1, m3, m4, m8, and m10) are selected as filler items. We also assume that m5 and m9 are the items with the highest averages, m2 and m7 are with the lowest averages, and m2 and m6 belong to a particular segment. In the random attack profile, randomly selected five filler items are filled with generated random numbers arbitrarily. For average attack profile, two sets of random numbers are generated. The maximum random number of the second set (tmax) is assigned to the target item. Each filler item's cell is filled with the related item's mean vote and a random number from the first set. In the bandwagon and segment attack profiles, generated random numbers are decreasingly sorted. The largest random numbers are assigned to the selected items and the remaining ones to the filler items arbitrarily. Unlike bandwagon attack, generated random numbers are increasingly sorted in the reverse bandwagon attack. The smallest random numbers are assigned to the selected items and the remaining ones to the filler items in the reverse bandwagon attack. In the love/hate attack, random numbers are decreasingly sorted and the largest ones are arbitrarily assigned to the filler items.

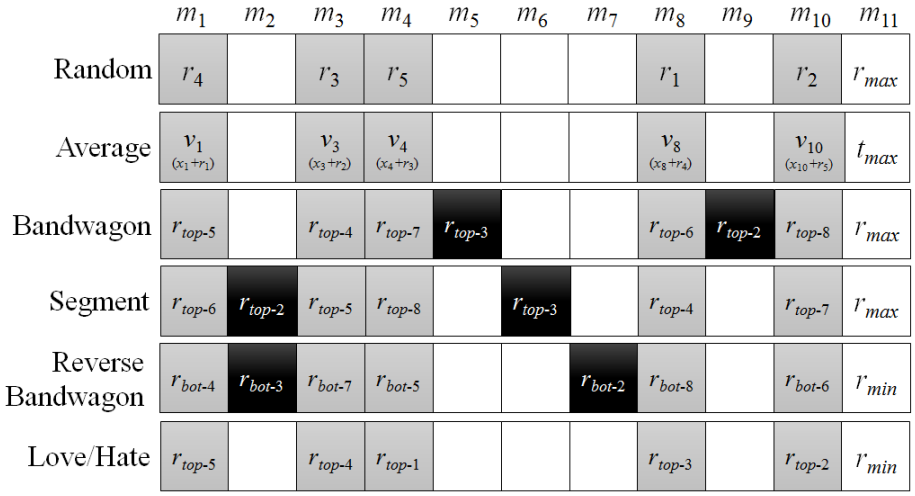

Fig. 2. Example attack profiles

5. Experimental Evaluation

We conducted real data-based experiments to evaluate the effectiveness of our modified shilling attack models on two memory-based PPCF algorithms. We employed two controlling parameters, i.e., *filler size* and *attack size*, in our evaluations, which are defined for performing successful profile injection attacks in the literature [37][41]. *Filler size* is the percentage of empty cells to be filled in bogus profiles, using the rating function, θ, to obstruct detection of the attack, as described in Section 3 [41]. *Attack size* is the number of attack profiles to inject, which is proportional to the number of users in the system [37].

5.1 Data Set and Evaluation Criteria

We performed experiments on a variation of a well-known publicly available data set, MovieLens Public (MLP), which was collected by the GroupLens research team at the University of Minnesota (http://www.grouplens.org). The set contains 100,000 discrete votes on a five-star rating scale for 1,682 movies from 943 users. The performance of profile injections can be measured using various metrics. We utilized the most frequently used metric in assessing shilling attack performance, i.e., *prediction shift*, which is defined as the average alteration in the predicted rating of an attacked item after the attack.

5.2 Experimental Methodology

Experiments were performed using an *all-but-one* method. Throughout all iterations, one of the users is considered to be the active user, and the remaining users form the training set. In addition, two distinct target product sets were constructed, consisting of 50 movies for push and nuke attacks. Selection of the movies was random within different ranges of ratings to sample the original dataset's distribution. It is unreasonable to push the prediction of a popular item or to nuke an unpopular one. Therefore, the set for push attacks consists of items with rating averages from 1 to 3, and the set for nuke attacks ranges from 3 to 5. Statistics of the selected target items are presented in **Table 2**.

Table 2. Statistics of target movies

Ratings Count	**Pushed Items**		**Nuked Items**	
	1 – 2	2 – 3	3 – 4	4 – 5

1 – 50	30	15	12	18
51 – 150	–	3	5	6
151 – 250	–	1	2	3
250 and up	–	1	1	3

Values of the parameters defined in Section 4, required to produce shilling profiles, were selected as follows: (*i*) for the average attack model, α is constant at 0.25, which intuitively provides a sufficient interval to disguise an average of items, where σ_{max} was chosen equal to two, (*ii*) the number of popular and unpopular items (*c*) for bandwagon and reverse bandwagon attacks, respectively was set at 10, (*iii*) the number of segmented items (*h*) for the segment attack was fixed at five and selected from the most-rated horror movies. In addition, users who positively rated at least 60% of five of these movies were included in the segment, and (*iv*) the constant multiplier (*C*) for the love/hate attack was set at four to insert high rating values into the filler items.

5.3 Empirical Results and Discussion

Throughout the experiments, all targeted items were attacked individually for all users in the system. Predictions were estimated before and after injecting the fake profiles. Then, the *prediction shift* values were compared to demonstrate the relative change in predicted values for different attack models. The number of neighbors was set to 30, selected from users who rated the target item for the *k*-nn algorithm. The absolute similarity threshold (τ) was 0.2 for the correlation-threshold algorithm. We exclusively present empirical results for push and nuke attacks designed for attacking perturbed databases in the following subsections.

5.3.1 Evaluating the Effects of Push Attack Models in PPCF Schemes

To demonstrate the effects of the modified push attacks on both memory-based privacy-preserving algorithms, we first conducted experiments with varying *filler size* (from 3% to 25%), which is a parameter directly related to the effect of the attack. During these trials, *attack size* was maintained at 15%, the largest value in the experiment, to magnify the impact of the manipulations. The experiments were repeated 100 times because of the randomization in the perturbation process. The average results are presented in **Fig. 3** and **Fig. 4**.

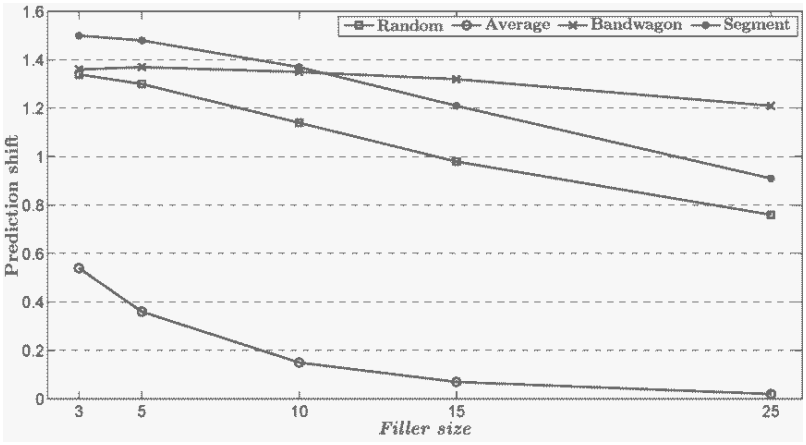

Fig. 3. Prediction shift for varying filler size (*k*-nn algorithm)

As shown in **Fig. 3** and **Fig. 4**, bandwagon and segment attacks are more effective against

privacy-preserving algorithms. The modified bandwagon attack achieved a maximum prediction shift of 1.37 and 0.94 for *k*-nn and correlation-threshold algorithms, respectively. The proposed segment attack is slightly more successful and stable, achieving a prediction shift of approximately 1.45 for both algorithms. On a five-star scale, this prediction shift is considered to be significant. Although a maximum average prediction shift of 1.34 is obtained for the modified random attack against the *k*-nn algorithm, this attack does not perform adequately against the correlation-threshold algorithm. The average attack is less successful but more stable than the random attack model for disguised data and achieves a prediction shift of approximately 0.45 for both algorithms. As *filler size* increases, prediction shift decreases for all attack schemes, which can be interpreted intuitively as the optimum value of *filler size* is interrelated with data density. Hence, the highest prediction shifts are achieved for 3% and 5% filler size, close to the overall density of the data set.

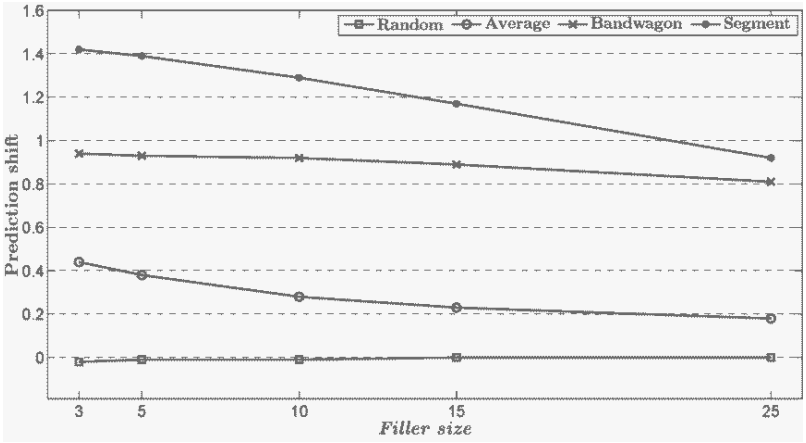

Fig. 4. Prediction shift for varying filler size (correlation-threshold algorithm)

We then performed another set of experiments with varying *attack size* (from 1% to 15%) to investigate the effects on the prediction shift of the number of injected profiles. During this set of experiments, *filler size* is kept constant at 15%, which was expected to maximize the impact. Experiments were repeated 100 times because of the randomization in the perturbation process. The overall averages of the results are presented in **Fig. 5** and **Fig. 6**.

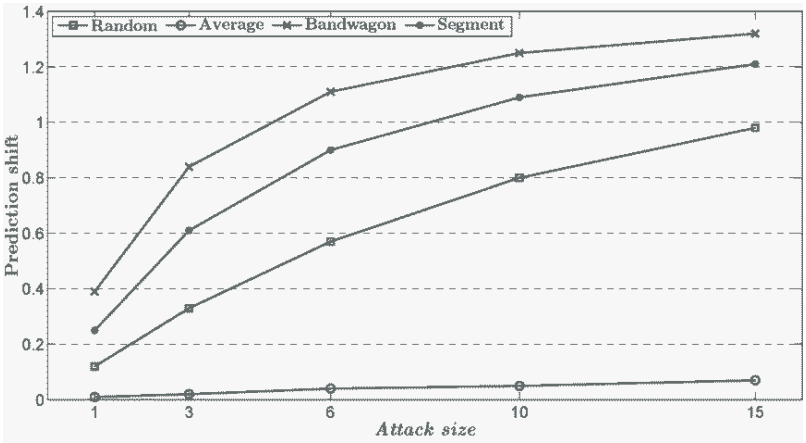

Fig. 5. Prediction shift for varying attack size (*k*-nn algorithm)

Similar to the previous experiments, the random attack performs well against the *k*-nn algorithm, achieving a prediction shift maximum of 0.98; however, it is completely ineffective against the correlation-threshold algorithm. Similarly, the average attack is unsuccessful in all trials for both algorithms because of the *filler size*. In addition, the bandwagon and segment attacks perform similarly as in the previous trials, except that the bandwagon performs slightly better than the segment attack against the *k*-nn algorithm. In general, it can be concluded that the prediction shift grows as the *attack size* increases, as intuitively expected.

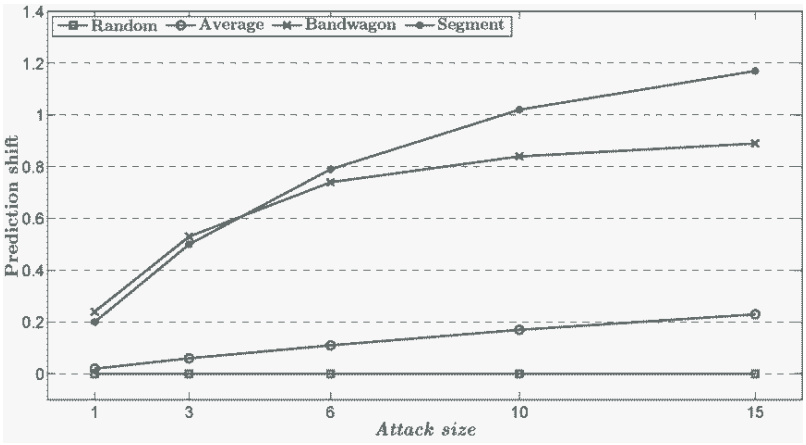

Fig. 6. Prediction shift for varying attack size (correlation-threshold algorithm)

Relying on the overall prediction shifts presented in **Fig. 3** to **Fig. 6**, it can also be concluded that the correlation-threshold algorithm is more robust than the *k*-nn algorithm. This outcome can be explained through the neighbor selection methods of the two algorithms. The correlation-threshold algorithm accepts negatively correlated users as well as positively correlated users in the prediction process, but *k*-nn only includes strongly and positively correlated users. As the very nature of attacks is to establish a positive correlation with users, the *k*-nn algorithm is more vulnerable to the attacks.

5.3.2 Evaluating the Effects of Nuke Attack Models on PPCF Schemes

To demonstrate the effects of the modified nuke attacks on memory-based privacy-preserving algorithms, we performed experiments with varying *filler size* and *attack size*. We first set *attack size* to 15% to test the effects of varying filler sizes for both algorithms. We repeated the experiments 100 times because of the randomization in the perturbation process. Overall averages of the outcomes for varying filler sizes are shown in **Fig. 7**.

As shown in **Fig. 7**, the love/hate attack model is completely impractical against both algorithms and never achieves a significant prediction shift with varying filler sizes. Although the love/hate is an effective attack in non-private environments, it is unlikely to obtain strong correlations relying on shilling profiles from the disguising scheme. Assigning high z-score values does not guarantee high similarity values, because similarity weight is calculated via dot products and profiles also include negative values. Reverse bandwagon, however, performs effectively, especially against the *k*-nn algorithm, reaching a maximum nuke prediction shift of 1.75. Similar to the results of the push attacks, effects of the *filler size* are strongly correlated with the data set density; hence, a lower prediction shift is achieved with greater values of filler size. Moreover, the correlation-threshold algorithm is more robust

compared to the *k*-nn algorithm (maximum 0.56 nuke predictions), because of the reasons previously discussed.

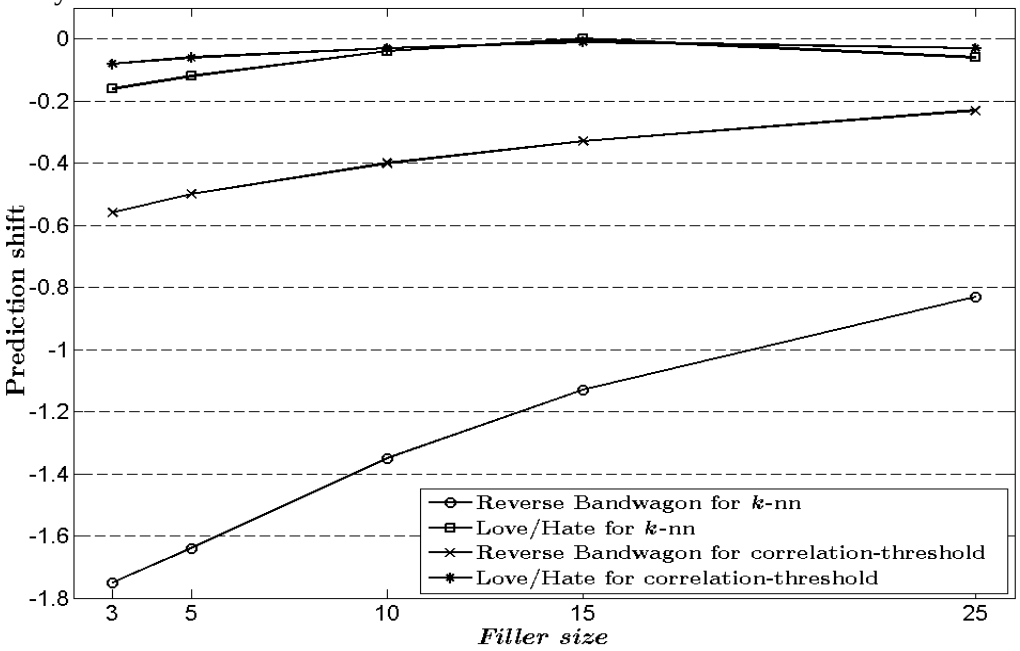

Fig. 7. Prediction shift for varying filler size for both algorithms

We then set *filler size* to 15% to test the effects of varying attack sizes for both algorithms. We again repeated the experiments 100 times because of the randomization in the perturbation process. Our outcomes show that the love/hate attack is not able to achieve any prediction shift (prediction shift values are all 0.0) for varying attack sizes in both algorithms. Unlike the love/hate attack, with increasing attack sizes, the reverse bandwagon attack causes growing prediction shift for both algorithms. However, the correlation-threshold algorithm is more robust than the *k*-nn algorithm. For attack sizes of 1, 3, 6, 10, and 15 percent, the prediction shift values are -0.04, -0.11, -0.19, -0.26, and -0.33 for the correlation-threshold algorithm while they are -0.15, -0.40, -0.66, -0.91, and -1.13, respectively for the *k*-nn algorithm.

6. Conclusions and Future Work

Many studies have examined collaborative filtering schemes without privacy concerns in terms of shilling attacks. Similarly, some researchers have investigated recommendation algorithms with respect to privacy. On the one hand, there are valuable works discussing profile injection attacks that fail to focus on privacy protection. On the other hand, various schemes are proposed to provide recommendations regarding privacy without studying shilling attacks. Privacy-preserving prediction methods can also be subjected to shilling attacks. Such systems have not been evaluated in terms of their robustness against profile injection attacks. Thus, we investigated two well-known memory-based privacy-preserving collaborative filtering algorithms exposed to profile injection attacks. We proposed methods to design shilling profiles to be inserted into disguised databases in privacy-preserving prediction systems. We also extensively evaluated privacy-preserving *k*-nn and correlation-threshold algorithms with respect to their robustness. Empirical results show that these systems are also vulnerable to profile injection attacks, similar to traditional

collaborative filtering schemes. Our proposed modified bandwagon, segment, and reverse bandwagon attacks achieved significant alterations in produced predictions. However, it is shown that the modified love/hate model is ineffective because of the data disguising mechanism. Furthermore, it is experimentally verified that the correlation-threshold algorithm is more robust than the *k*-nn algorithm, because its principle of forming neighborhoods contradicts the logic of shilling attack profile design.

In this study, we investigated the potential for profile injection attacks to be successfully mounted against memory-based privacy-preserving schemes. The importance of empirical results is that they confirm the applicability of some attacks on recommendation schemes with privacy, which leads us to question the robustness of other methods. Therefore, other methods of preserving individual privacy, such as data substitution methods, and other schemes of providing private predictions, such as item- or model-based collaborative filtering schemes with privacy, must be investigated against profile injection attacks. Furthermore, developing robust privacy-preserving methods and performing cost/benefit analyses of applied attacks are other future research directions indicated by the results of this study.

References

[1] J. L. Herlocker, J. A. Konstan, L. G. Terveen and J. T. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5-53, 2004. Article (CrossRef Link)

[2] J.-H Park, "A recommender system for device sharing based on context-aware and personalization," *KSII Transactions on Internet and Information Systems*, vol. 4, no. 2, pp. 174-190, 2010. Article (CrossRef Link)

[3] H. D. Kim, "Applying consistency-based trust definition to collaborative filtering," *KSII Transactions on Internet and Information Systems*, vol. 3, no. 4, pp. 366-375, 2009. Article (CrossRef Link)

[4] M. P. O'Mahony, N. J. Hurley and G. C. M. Silvestre, "Towards robust collaborative filtering," *Lecture Notes in Computer Science*, vol. 2464, pp. 87-94, 2002. Article (CrossRef Link)

[5] M. P. O'Mahony, N. J. Hurley NJ and G. C. M. Silvestre, "Promoting recommendations: An attack on collaborative filtering," in *Proc. of the 13th International Conference on Database and Expert Systems Applications*, pp. 494-503, 2002. Article (CrossRef Link)

[6] M. P. O'Mahony, N. J. Hurley NJ and G. C. M. Silvestre, "Recommender systems: Attack types and strategies," in *Proc. of the 20th National Conference on Artificial Intelligence*, pp. 334-339, 2005. Article (CrossRef Link)

[7] J. Canny, "Collaborative filtering with privacy via factor analysis," in *Proc. of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 238-245, 2002. Article (CrossRef Link)

[8] S. Berkovsky, N. Borisov, Y. Eytani, T. Kuflik and F. Ricci, "Examining users' attitude towards privacy preserving collaborative filtering," in *Proc. of the Workshop on Knowledge Discovery for Ubiquitous User Modeling*, June 25, 2007. Article (CrossRef Link)

[9] H. Polat and W. Du, "Privacy-preserving collaborative filtering," *International Journal of Electronic Commerce*, vol. 9, no. 4, pp. 9-35, 2005. Article (CrossRef Link)

[10] S. Berkovsky, T. Kuflik and F. Ricci, "The impact of data obfuscation on the accuracy of collaborative filtering," *Expert Systems with Applications*, vol. 39, no. 5, pp. 5033-5042, 2012. Article (CrossRef Link)

[11] J. Canny, "Collaborative filtering with privacy," in *Proc. of the IEEE Symposium on Security and Privacy*, pp. 45-57, 2002. Article (CrossRef Link)

[12] B. Sarwar, G. Karypis, J. A. Konstan and J. T. Riedl, "Analysis of recommendation algorithms for e-commerce," in *Proc. of the 2nd ACM Conference on Electronic Commerce*, pp. 158-167, 2000. Article (CrossRef Link)
[13] Vozalis MG, Markos A and Margaritis KG. "Collaborative filtering through SVD-based and hierarchical nonlinear PCA," in *Proc. of the 20th International Conference on Artificial Neural Networks*: Part I. Thessaloniki, Greece, 2010, pp. 395-400. Article (CrossRef Link)
[14] C. Yan-ni and Y. Min, "A hybrid collaborative filtering algorithm based on user-item," in *Proc. of International Conference on Computational and Information Sciences*, pp. 618-621, 2010. Article (CrossRef Link)
[15] S. Russell and V. Yoon, "Applications of wavelet data reduction in a recommender system," *Expert Systems with Applications*, vol. 34, no. 4, pp. 2316-2325, 2008. Article (CrossRef Link)
[16] J. L. Herlocker, J. A. Konstan, A. Borchers and J. T. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 230-237, 1999. Article (CrossRef Link)
[17] D. Goldberg, D. Nichols, B. M. Oki and D. Terry, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, no. 12, pp. 61-70, 1992. Article (CrossRef Link)
[18] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734-749, 2005. Article (CrossRef Link)
[19] N. Dokoohaki, C. Kaleli, H. Polat and M. Matskin, "Achieving optimal privacy in trust-aware social recommender systems," in *Proc. of the 2nd International Conference on Social Informatics*, pp. 62-79, 2010. Article (CrossRef Link)
[20] A. Bilge, S. Gurmeric and H. Polat, "An enhanced collaborative filtering scheme via recursive clustering," in *Proc. of the Workshop on Knowledge Discovery, Data Mining and Machine Learning*, 2012.
[21] G. Linden, B. Smith and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76-80, 2003. Article (CrossRef Link)
[22] S. Berkovsky, Y. Eytani, T. Kuflik and F. Ricci, "Privacy-enhanced collaborative filtering," in *Proc. of the User Modeling Workshop on Privacy-Enhanced Personalization*, pp. 75-84, 2005. Article (CrossRef Link)
[23] A. Bilge and H. Polat, "An improved privacy-preserving DWT-based collaborative filtering scheme," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3841-3854, 2012. Article (CrossRef Link)
[24] A. Bilge and H. Polat, "A comparison of clustering-based privacy-preserving collaborative filtering schemes," *Applied Soft Computing*, 2013, doi: 10.1016/j.asoc.2012.11.046. Article (CrossRef Link)
[25] H. Polat and W. Du, "Achieving private recommendations using randomized response techniques," *Lecture Notes in Computer Science*, vol. 3918, pp. 637-646, 2006. Article (CrossRef Link)
[26] A. Bilge and H. Polat, "Improving privacy-preserving NBC-based recommendations by preprocessing," in *Proc. of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, pp. 143-147, 2010. Article (CrossRef Link)
[27] C. Dellarocas, "Immunizing online reputation reporting systems against unfair ratings and discriminatory behaviour," in *Proc. of the 2nd ACM Conference on Electronic Commerce*, pp. 150-157, 2000. Article (CrossRef Link)
[28] R. D. Burke, B. Mobasher and R. Bhaumik, "Limited knowledge shilling attacks in collaborative filtering systems," in *Proc. of the 3rd Workshop on Intelligent Techniques for Web Personalization*, 2005. Article (CrossRef Link)
[29] R. D. Burke, B. Mobasher, C. Williams and R. Bhaumik, "Classification features for attack detection in collaborative recommender systems," in *Proc. of the 12th ACM SIGKDD International*

Conference on Knowledge Discovery and Data Mining, pp. 542-547, 2006. Article (CrossRef Link)

[30] S. Zhang, A. Chakrabarti, J. Ford and F. Makedon, "Attack detection in time series for recommender systems," in *Proc. of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 809-814, 2006. Article (CrossRef Link)

[31] M. P. O'Mahony, "Towards robust and efficient automated collaborative filtering," *PhD Dissertation*, University College Dublin, 2004.

[32] J. J. Sandvig, B. Mobasher and R. D. Burke, "Robustness of collaborative recommendation based on association rule mining," in *Proc. of the 1st ACM Conference on Recommender Systems*, pp. 105-112, 2007. Article (CrossRef Link)

[33] X. Yan, "Manipulation robustness of collaborative filtering systems," *PhD Dissertation*, Stanford University, 2009.

[34] F. G. Zhang, "A survey of shilling attacks in collaborative filtering recommender systems," in *Proc. of the International Conference on Computational Intelligence and Software Engineering*, pp. 1-4, 2009. Article (CrossRef Link)

[35] S. K. Lam and J. T. Riedl, "Shilling recommender systems for fun and profit," in *Proc. of the 13th International Conference on World Wide Web*, pp. 393-402, 2004. Article (CrossRef Link)

[36] S. K. Lam and J. T. Riedl, "Privacy, shilling, and the value of information in recommender systems," in *Proc. of the User Modeling Workshop on Privacy-Enhanced Personalization*, pp. 85-92, 2005. Article (CrossRef Link)

[37] B. Mobasher, R. Burke, R. Bhaumik and C. Williams, "Towards trustworthy recommender systems: An analysis of attack models and algorithm robustness," *ACM Transactions on Internet Technology*, vol. 7, no. 4, pp. 23-60, 2007. Article (CrossRef Link)

[38] I. Gunes, C. Kaleli, A. Bilge and H. Polat, "Shilling attacks against recommender systems: A comprehensive survey," *Artificial Intelligence Review*, pp. 1-33, 2012. Article (CrossRef Link)

[39] I. Gunes, A. Bilge, C. Kaleli and H. Polat, "Shilling attacks against privacy-preserving collaborative filtering," *Journal of Advanced Management Science*, vol. 1, no. 1, pp. 54-60, 2013. Article (CrossRef Link)

[40] Z. Huang, W. Du and B. Chen, "Deriving private information from randomized data," in *Proc. of the ACM SIGMOD International Conference on Management of Data*, pp. 37-48, 2005. Article (CrossRef Link)

[41] R. Bhaumik, C. A. Williams, B. Mobasher and R. D. Burke, "Securing collaborative filtering against malicious attacks through anomaly detection," in *Proc. of the 4th Workshop on Intelligent Techniques for Web Personalization*, 2006. Article (CrossRef Link)

[42] R. D. Burke, B. Mobasher, R. Bhaumik and C. A. Williams, "Segment-based injection attacks against collaborative filtering recommender systems," in *Proc. of the 5th IEEE International Conference on Data Mining*, pp. 577-580, 2005. Article (CrossRef Link)

[43] B. Mobasher, R. Burke, R. Bhaumik and J. J. Sandvig, "Attacks and remedies in collaborative recommendation," *IEEE Intelligent Systems*, vol. 22, no. 3, pp. 56-63, 2007. Article (CrossRef Link)

[44] F. G. Zhang, "Average shilling attack against trust-based recommender Systems," in *Proc. of the International Conference on Information Management, Innovation Management and Industrial Engineering*, pp. 588-591, 2009. Article (CrossRef Link)

Ihsan Gunes received his BSc and MSc degrees in Computer Engineering Department from Kocaeli University and Anadolu University, respectively. He is currently a PhD candidate in Computer Engineering Department at Anadolu University. His research interest is collaborative filtering in general; and specifically shilling attacks on collaborative filtering.

Alper Bilge received the BSc. degree in Electrical and Electronics Engineering Department, MSc. and PhD degrees in Computer Engineering Department from Anadolu University, Eskisehir, Turkey. His research interests are data mining, privacy-preserving data mining, and collaborative filtering in general.

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 collaborative filtering with privacy, private predictions on selected models, and privacy-preserving data mining.