A Robust Privacy-Preserving Recommendation Algorithm

Alper Bilge   Ihsan Gunes   Huseyin Polat
Computer Engineering Department
Anadolu University, Eskişehir, Turkey
{abilge,ihsang,polath}@anadolu.edu.tr

Abstract
Privacy-preserving collaborative filtering schemes are key recommender system technologies for e-commerce field. They focus on alleviating information overload problem by providing personalized recommendations without deeply jeopardizing customers’ privacy. Like their non-private versions, privacy-preserving recommendation methods might be easily subjected to profile injection attacks for manipulating produced recommendations in favor or disfavor of certain products. Clustering-based prediction schemes have shown to be effective in distinguishing bogus profiles from genuine ones; and they are relatively more resilient than memory-based methods against shilling attacks in non-private environments. Motivating from this fact, we propose bisecting $k$-means clustering-based privacy-preserving recommendation algorithm as a robust recommendation algorithm, which was formerly proposed as a scalable and accurate private recommendation scheme, against previously defined several private shilling attack strategies. We then investigate the algorithm with respect to robustness. Thus, we perform some real data-based experiments on a benchmark data set using six profile injection attacks. We empirically show that the algorithm performs in a robust manner with insignificant alterations in predicted values when fake profiles are injected. The reason for this phenomenon is that the algorithm is able to cluster fake profiles together.

Keywords: Shilling attacks, privacy, robustness analysis, collaborative filtering, clustering, recommendation.

1 Introduction
With increasing amount of information available in everyday life through widespread usage of the Internet services, recommender system technologies have become one of the most practical tools to filter out useful information. Collaborative filtering (CF) is the key recommendation method used in e-commerce facilities to extract products of possible interest such as movies, music CDs, books, and so on. These systems rely on preference information provided explicitly by customers or implicit inferences from web logs. They recommend products by discovering liking patterns of users relying on similarities of past rating trends.

CF algorithms can be mainly classified into three classes: (i) memory-based, (ii) model-based, and (iii) hybrid approaches. Memory-based algorithms are successful in terms of accuracy of predictions and representing word-of-mouth habits [1]; however, they commonly suffer from scalability issues [2]. Model-based solutions, on the other hand, derive a model from the input database off-line and utilize it to produce predictions online [3]. Although their online performance is better, it is computationally expensive and hard to update the model. Hybrid systems like employing content-based filtering along with CF process [4] or employing clustering to perform neighborhood formation prior to prediction estimation [1; 5] combine both approaches’ advantages for an optimal solution.

Since recommender systems rely on obtained preference information of users, they are open to manipulations of malicious users or competing companies. Such interventions are called shilling or profile injection attacks [6], which can be very effective against traditional CF schemes [7; 8]. Shilling attacks can be categorized as push or nuke attacks according to their intent. In those attacks, the attacker creates bogus profiles to favor or disfavor a particular item and inserts them into the database as a genuine user. However, clustering-based approaches are shown to be successful in distinguishing shilling profiles from genuine ones [6; 9] because fake profiles show resemblance among themselves and clustered together.
Submission of preference information causes several privacy risks for individuals such as price discrimination, government surveillance, and so on [10]. Therefore, it has been receiving increasing attention to produce dependable predictions without deeply jeopardizing individual privacy and privacy-preserving collaborative filtering (PPCF) concept has been proposed to estimate sufficiently accurate recommendations without revealing private preferences of users. Cryptographic [11], randomization-based [12], and obfuscation-based [13] methods aid privacy protection. In our previous studies, we demonstrate that memory-based PPCF schemes are also vulnerable to shilling attacks [14; 15]. However, clustering-based PPCF schemes might be robust against profile injection attacks. Thus, we propose bisecting \(k\)-means clustering-based PPCF recommendation algorithm as a robust algorithm. We investigate it with respect to robustness under several attacking strategies.

Remainder of the paper is structured, as follows: Section 2 discusses relevant literature and explains originality of the work. Then, we explain how privacy-preserving bisecting \(k\)-means clustering-based PPCF performs and shilling attacks can be implemented to modify its outputs in Section 3. Section 4 experimentally evaluates the robustness of the algorithm against manipulations under four push and two nuke attack strategies. We finally conclude the paper by a brief discussion and present some future research directions in Section 5.

2 Related Work

CF systems date back to early nineties [16]. They produce personal predictions relying on past preferences of similar users [17]. CF is integrated as a recommender system by online vendors operating on preference data.

Individuals today consider that it might be risky to submit preference data due to privacy concerns and they refrain from submitting genuine ratings [18]. However, dependable referrals can only be produced through authentic data. Hence, researchers propose PPCF schemes to provide referrals with privacy. Initial proposals rely on distributed solutions by Canny [11], where an aggregate data is formed by users of a community using singular value decomposition. Privacy is achieved using cryptographic methods and the proposed scheme can be expanded to a fully peer-to-peer system. Since most CF systems run on central servers, today’s PPCF systems are designed to perform on masked data using randomization and obfuscation methods [13; 19]. In such methods, users control their privacy by masking, perturbing, and substituting their votes. Such disguising is realized by random perturbation techniques, which allows performing aggregate algebraic operations on the collection while not disclosing individual data items.

With increasing popularity of CF systems, several attacking mechanisms arise to manipulate their outputs in favor of particular products. Dellarocas [20] inspires manipulation attacks to recommender systems, where some mechanisms are defined to avoid fraud in online reputation reporting systems. O’Mahony et al. [21] then discuss vulnerabilities of automated prediction estimation process against manipulations. They also describe the amount of information needed about the database to realize effective shilling attacks. Lam and Riedl [22; 23] analyze cost of attacks and propose that there is a relation between privacy and the value of information. Several attacking methods are proposed in the literature like random, average, bandwagon, and segment attacks [24]. Shilling attacks can be designed on either a low or high level of knowledge such as average ratings of products or general tendencies of particular users. Moreover, their effectiveness are investigated against memory- and model-based CF schemes [25]. Recently, Gunes et al. [26] survey recent research on shilling attacks and provide analysis on attack descriptions, detection methodologies, robust algorithms, and evaluation metrics.

Effects of shilling attacks on memory-based PPCF schemes are deeply analyzed and shown that they are vulnerable to manipulations [15]. Novel PPCF schemes should be proposed as robust prediction algorithms. Although bisecting \(k\)-means clustering-based PPCF algorithm is proposed to protect individual privacy while providing predictions with high accuracy efficiently [27], it might be an appropriate proposal as a robust algorithm. Thus, we investigate its robustness against profile injection attacks. Unlike the explained previous studies, we focus on robustness analysis of this algorithm against profile injection attacks and show how clustering-based approaches are resistant to shilling attacks in privacy-preserving environments.
3 Attacking Bisecting $k$-means Clustering-based PPCF Scheme

Due to the reason that recommender systems are open for public usage and therefore vulnerable to manipulations, both non-private and privacy-preserving recommendation algorithms need to have robust mechanisms to estimate predictions. However, state-of-the-art memory-based CF and PPCF schemes are not resistant to such attacks and exposed to significant shifts in predicted values. In this section, we describe bisecting $k$-means clustering-based PPCF algorithm, six attack designs for masked data, and explain how robust the algorithm is.

3.1 Bisecting $k$-means Recommendation Algorithm with Privacy

Privacy risks in online shopping services direct people not willing to provide any preference information. In order to relieve privacy concerns of users, Polat and Du [12] propose the following data masking procedure so that individual preferences are not disclosed to the server; yet, aggregate algebraic operations can still be performed and recommendation can be generated.

- The central server lets each user know global data perturbation parameters, i.e., $\sigma_{\text{max}}$ and $\beta_{\text{max}}$.

- User $u$ calculates her ratings’ z-score values, chooses a distribution (uniform or normal), and creates random disguising and filling values by chosen $\sigma_u$, where $\sigma_u \in (0, \sigma_{\text{max}}]$.

- Each user then perturbs each z-score value by adding a random value onto it and fills $\beta_a \%$ of her empty cells, where $\beta_u \in (0, \beta_{\text{max}}]$. 

- Finally, each user submits her perturbed ratings vector to the server.

The central server collects such disguised user vectors and forms a user-item matrix, $U'_{n \times m}$. At the beginning, the server forms a binary decision tree off-line by utilizing bisecting $k$-means clustering algorithm on $U'$. Given $U'$, and an optimal value of number of neighbors ($N$), $k$-means clustering is applied to divide $U'$ into two clusters at each level (hence, it is called bisecting) and cluster centers are indexed to be used as a forwarding tool for each corresponding level. If number of users in any cluster exceeds $N$, then such clusters are continued to be divided recursively into subsets via the $k$-means clustering. Finally, a binary decision tree is obtained having indexed cluster centers as branch nodes and grouped neighbor users at leaf nodes.

When an active user $a$ arrives and queries for a prediction, instead of calculating similarities with all users, the server only forwards the active user according to her similarity to two cluster centers at each level. While traversing, two similarity calculations are performed at each level to determine next hope (either right or left). Finally, the leaf node that the new user belongs is determined and all users in that corresponding node and its sibling are regarded as neighbors. Then, a prediction is calculated as a weighted average of neighbors on target item is calculated as formulated in Eq. 1 and returned to the active user as a prediction.

$$p_{aq} = \overline{u_a} + \sigma_a \times \frac{\sum_{u \in N} w_{au} \times z_{uq}}{\sum_{u \in N} w_{au}}.$$  \hspace{1cm} (1)  

In Eq. 1, $p_{aq}$ is the prediction for $a$ on target item $q$, $\overline{u_a}$ and $\sigma_a$ are mean value and standard deviation of $a$’s ratings, $N$ is the set of neighbors, $z_{uq}$ is the z-score of user $u$ on $q$, and $w_{au}$ is the similarity between $a$ and neighbor $u$. Afterwards, the tree continues growing in such a way so that if any leaf node population exceeds the stopping criterion, the server immediately bisects that leaf node to grow. Therefore, it is a continual process to update the decision tree.

An example binary decision tree produced by the algorithm is presented in Fig. 1, where initially there are 150 users in the database and the stopping criterion is determined as 20 users. Therefore, the algorithm bisects each internal node starting from the root until there remains at most 20 users in a leaf node. At each internal node, two cluster centers ($C^L$ and $C^R$) are recorded to be used in forwarding purposes.

In order to attack such recommendation scheme, the attacker produces attack profiles and inserts them into the system. Clustering-based CF schemes are successful in detecting fake profiles or bogus profiles [6; 9]. Bhauimik et al. [9] shows that shilling profiles show resemblance to each other; therefore, when they are clustered, they tend to move together into the clusters. Using this intuition, clustering can be used
as a detection tool for shilling attacks and it can be utilized to create robust recommendation algorithms. In this study, we hypothesize that such intuition works best for clustering into two clusters to move shilling profiles together. In addition, applying such clustering repeatedly is supposed to eliminate all shilling profiles after some level of the produced binary decision tree. Therefore, we claim that malicious profiles substantially distinguishes from genuine ones after some level of the tree and it becomes very unlikely for any active user belonging to a leaf node consisting of fake profiles. As a result, the proposed recommendation scheme performs robust against shilling attacks.

### 3.2 Shilling attacks on masked databases

Profile injection attacks are generated by inserting fake or shilling profiles into user-item databases. There are several attacking strategies in the literature, which can be explained in a general format as depicted in Fig. 2 [24]. Note that $I_S$, $I_F$, and $I_\emptyset$ refers to selected, filler, and empty cells in the fake profile, respectively; and a unique item, $i_t$, is targeted. Selected items are chosen for characterizing an attack, filler items are chosen to prevent easy detection of fake profiles, and the target item is assigned either a high or a low rating value for push and nuke attacks, respectively.

As part of their generic form, four push attacks and two nuke attacks covered in this paper can be described, as in the following [15]:

**Random attack (RN).** Selected items set is empty. Arbitrarily chosen filler items are filled with random values and the target item is assigned a high random value.

**Average attack (AV).** Selected items set is empty. Arbitrarily chosen filler items are filled with close random values to the item’s mean and the target item is assigned a high random value.

**Bandwagon attack (BW).** Selected items are chosen among popular items (having high means). Selected and filler items are filled with random values, where selected items are assigned the highest ones and the target item is assigned a high random value.

**Segment attack (SG).** Selected items are chosen from high average items with a certain property (such as horror movies or jazz music). Selected and filler items are filled with random values, where selected items are assigned the highest ones and the target item is assigned a high random value.

**Reverse bandwagon attack (RBW).** Selected items are chosen among unpopular items (having low means). Selected and filler items are
items are filled with random values, where selected items are assigned the lowest ones and the target item is assigned a low random value.

**Love/hate attack (L/H).** Selected items set is empty. Arbitrarily chosen filler items are filled with very high random values and the target item is assigned a low random value.

### 4 Experimental Evaluation

After explaining how shilling attacks can be implemented over privacy-preserving bisecting \( k \)-means recommendation algorithm, we conducted real data-based experiments to scrutinize effects of such attacks on system robustness with respect to two controlling parameters. The controlling parameters, filler size and attack size, are defined in the literature for designing effective shilling attacks. Filler size parameter indicates the percentage of cells to be filled out with fake ratings while creating the attack profiles. Attack size can be described as the pre-attack profile count proportional to the number of users in the database. We conducted two sets of experiments with varying values of the explained parameters against the bisecting \( k \)-means clustering-based PPCF scheme.

#### 4.1 Data set and evaluation criteria

In the experiments, publicly available MovieLens (ML) data set, which is collected by GroupLens (http://movielens.umn.edu/) is utilized. It is the most widely used and well-known real collection for CF purposes. It holds 100K ratings from 943 users on 1,682 movies and the rating range allows 5-star discrete numeric values.

We used prediction shift metric in order to measure the prediction alterations due to effects of shilling attacks. Prediction shift can be described as the average change in the prediction for the attacked item before and after the attack profiles are included.

#### 4.2 Experimental Results and Discussion

During the experiments, we followed all-but-one experimentation methodology, which enables full utilization of the data set. This methodology considers one of the users as the active user \( a \) and the rest of the set as the training users at each of the iterations. The utilized attacks target two separate sets of 50 movies for push and nuke attacks. Those sets for push and nuke attacks are constructed selecting arbitrarily from different rating ranges to represent characteristics of the original data set. Since it is unreasonable to push a popular item with high ratings or similarly nuke an unpopular item, we principally selected items with low mean values to push and high means to nuke. Table 1 shows the statistics of 50 target movies for push and nuke attacks, where each value indicates how many of the movies fall into corresponding group.

<table>
<thead>
<tr>
<th>Ratings</th>
<th>Pushed Items</th>
<th>Nuked Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-50</td>
<td>30</td>
<td>15</td>
</tr>
<tr>
<td>51-150</td>
<td>—</td>
<td>3</td>
</tr>
<tr>
<td>151-250</td>
<td>—</td>
<td>1</td>
</tr>
<tr>
<td>250 and up</td>
<td>—</td>
<td>1</td>
</tr>
</tbody>
</table>

In the experiments, all target items were attacked individually for all users in the system. Binary decision trees were constructed by omitting and including fake shilling profiles. Then, predictions were estimated based on the produced binary decision trees and prediction shift values were observed to demonstrate relative change on predicted values for differing attack models. The stopping criterion for building binary decision trees is set to 30. Although varying stopping criterion might alter obtained prediction shift values especially with varying attack sizes, we fix such parameter due to page limitations and discuss algorithm’s robustness performance relying on a constant stopping condition value. Additionally, all experiments were repeated 100 times due to randomization in perturbation process and average results were calculated. We exclusively presented the obtained empirical results for push and nuke attacks in the following sections.

#### 4.2.1 Effects of push attack models

In order to investigate the effects of the four push attack models on bisecting \( k \)-means clustering-based prediction algorithm with privacy, we performed experiments by varying filler size in the attack profiles and varying attack size inserted into the database. Filler size parameter indicates the number of fake votes for the filler items added to fill out the attack profile; and thus, it is directly related to the success of the attack.
Attack size determines the number of inserted attack profiles; hence, it is also vital in realizing significant manipulations. To clarify the effects of these parameters, we kept each other at 15% while experimenting on one of them. In the experiments, we varied filler size from 3% to 25% and attack size from 1% to 15%. Overall results are shown in Table 2 and Table 3 with varying filler and attack sizes, respectively. The highest prediction shift values are given in bold.

### Table 2. Prediction shift with varying filler size for push attack models

<table>
<thead>
<tr>
<th>Filler size</th>
<th>Attacks</th>
<th>3%</th>
<th>5%</th>
<th>10%</th>
<th>15%</th>
<th>25%</th>
</tr>
</thead>
<tbody>
<tr>
<td>RN</td>
<td>.011</td>
<td>.013</td>
<td>.018</td>
<td>.031</td>
<td>.053</td>
<td></td>
</tr>
<tr>
<td>AV</td>
<td><strong>.006</strong></td>
<td>.004</td>
<td>.003</td>
<td>.002</td>
<td>.002</td>
<td></td>
</tr>
<tr>
<td>BW</td>
<td>.019</td>
<td>.023</td>
<td>.032</td>
<td>.035</td>
<td>.036</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>.195</td>
<td>.200</td>
<td>.209</td>
<td>.201</td>
<td>.195</td>
<td></td>
</tr>
</tbody>
</table>

### Table 3. Prediction shift with varying attack size for push attack models

<table>
<thead>
<tr>
<th>Attack size</th>
<th>Attacks</th>
<th>1%</th>
<th>3%</th>
<th>6%</th>
<th>10%</th>
<th>15%</th>
</tr>
</thead>
<tbody>
<tr>
<td>RN</td>
<td>.015</td>
<td>.039</td>
<td>.069</td>
<td>.078</td>
<td>.052</td>
<td></td>
</tr>
<tr>
<td>AV</td>
<td>.004</td>
<td>.010</td>
<td>.012</td>
<td>.002</td>
<td>.001</td>
<td></td>
</tr>
<tr>
<td>BW</td>
<td>.020</td>
<td>.037</td>
<td>.036</td>
<td>.036</td>
<td>.039</td>
<td></td>
</tr>
<tr>
<td>SG</td>
<td>.037</td>
<td>.166</td>
<td>.168</td>
<td>.180</td>
<td>.196</td>
<td></td>
</tr>
</tbody>
</table>

As can be seen from Table 2 and Table 3, none of the attack models is able to achieve a significant prediction shift for varying filler and attack sizes. With increasing filler size and attack size, the effects get larger; however, increasing these parameters more is not feasible for the sake of detection of attacks. Only segment attack achieves a maximum shift value around 0.2, which can still be ignored because it is only one in twentieth of the prediction range. Also note that privacy-preserving bisecting k-means clustering-based prediction algorithm runs with about 0.72 mean absolute error, as reported in [27], which is about four times larger than the maximum achieved prediction shift. Therefore, we can conclude that bisecting k-means clustering-based prediction algorithm is robust against profile injection push attacks in privacy-preserving environments.

### 4.2.2 Effects of nuke attack models

In addition to push attack models, we also investigated two well-known nuke attack models on privacy-preserving bisecting k-means clustering-based prediction algorithm. Similarly, we performed experiments by varying filler size and attack size; and kept each other at 15% while experimenting on one of them. Overall results are displayed in Table 4 and Table 5 with varying filler size and attack size, respectively. The highest prediction shift values are given in bold.

#### Table 4. Prediction shift with varying filler size for nuke attack models

<table>
<thead>
<tr>
<th>Filler size</th>
<th>Attacks</th>
<th>3%</th>
<th>5%</th>
<th>10%</th>
<th>15%</th>
<th>25%</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBW</td>
<td>-.047</td>
<td>-.050</td>
<td>-.049</td>
<td><strong>-.054</strong></td>
<td>-.037</td>
<td></td>
</tr>
<tr>
<td>L/H</td>
<td>-.001</td>
<td>.002</td>
<td>.003</td>
<td>.003</td>
<td>.003</td>
<td>.007</td>
</tr>
</tbody>
</table>

#### Table 5. Prediction shift with varying attack size for nuke attack models

<table>
<thead>
<tr>
<th>Attack size</th>
<th>Attacks</th>
<th>1%</th>
<th>3%</th>
<th>6%</th>
<th>10%</th>
<th>15%</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBW</td>
<td>-.018</td>
<td>-.040</td>
<td>-.02</td>
<td>-.021</td>
<td><strong>-.041</strong></td>
<td></td>
</tr>
<tr>
<td>L/H</td>
<td>-.001</td>
<td>.003</td>
<td>.004</td>
<td>.006</td>
<td>.007</td>
<td></td>
</tr>
</tbody>
</table>

Table 4 and Table 5 demonstrate that nuke attacks are also not effective and not able to achieve notable prediction shift values for varying filler size and attack size. Also note that love/hate attack is completely ineffective so that it cannot achieve even minus prediction shifts for most of the trials. Reverse bandwagon attack, on the other hand, is more effective than love/hate attack but obtained prediction shifts are negligible compared to the rating range. Therefore, we can again conclude that bisecting k-means clustering-based prediction algorithm is resistant against nuke shilling attacks in privacy-preserving environments.

### 4.2.3 Comparison with a memory-based approach

Robustness against profile injection attacks has not been investigated deeply in terms of privacy-preserving CF algorithms. A former analysis is performed to scrutinize robustness of memory-based k-nn privacy-preserving CF algorithm [15]. Therefore, we compare our experimental findings with such algorithm’s perfor-
mance to provide a clearer ground for proposing bisecting $k$-means clustering-based privacy-preserving recommendation algorithm as a robust approach. Table 6 demonstrates maximum obtained prediction shift values for $k$-nn and bisecting $k$-means clustering-based privacy-preserving recommendation algorithms.

Table 6. Prediction shift values for $k$-nn and bisecting $k$-means clustering (BKM) algorithms

<table>
<thead>
<tr>
<th>Attack type</th>
<th>RN</th>
<th>AV</th>
<th>BW</th>
<th>SG</th>
<th>RBW</th>
<th>LH</th>
</tr>
</thead>
<tbody>
<tr>
<td>$k$-nn</td>
<td>1.34</td>
<td>0.54</td>
<td>1.37</td>
<td>1.50</td>
<td>-1.75</td>
<td>-0.16</td>
</tr>
<tr>
<td>BKM</td>
<td>0.08</td>
<td>0.01</td>
<td>0.04</td>
<td>0.21</td>
<td>-0.05</td>
<td>-0.00</td>
</tr>
</tbody>
</table>

As can be followed in Table 6, bisecting $k$-means clustering-based algorithm performs highly robust and is exposed to insignificant prediction shifts compared to memory-based $k$-nn algorithm for all attack types. This fact originates from bisecting $k$-means clustering-based algorithm’s success in separating most malicious profiles from genuine ones by recursively clustering them. Unlike bisecting $k$-means algorithm, $k$-nn approach is more open to shilling attacks as it directly calculates similarities with all existing profiles and set $k$ nearest of them as neighbors.

5 Conclusions and Future Work

We investigated a formerly proposed private, accurate, and scalable bisecting $k$-means clustering-based prediction algorithm’s robustness against malicious profile injection attacks. We presented modified versions of four well-known push and two nuke attacks to be implemented on privately collected databases. We also explained how such inserted attack profiles can affect the recommendation scheme and why it is expected that the algorithm is robust against them. According to the obtained experimental results, the demonstrated push and nuke attack models are not able to significantly alter final predictions produced by the scheme. Thus, the algorithm is robust against shilling attacks.

We are planning to investigate the robustness of the algorithm in non-private CF environment. It is known that clustering algorithms can be effective as a detection mechanism for shilling attacks. Hence, it warrants future work to utilize this algorithm as a detection tool of fake profiles.

Acknowledgment

This work was supported by the Grant 111E218 from TUBITAK.

References


