ON THE ROBUSTNESS OF PRIVACY-PRESERVING COLLABORATIVE FILTERING SCHEMES

İhsan GÜNEŞ

Ph.D. Dissertation

Graduate School of Sciences
Computer Engineering Program
April, 2015

This dissertation is supported by the Grant 111E218 from TÜBİTAK.

JÜRİ VE ENSTİTÜ ONAYI

İhsan Güneş'in "**On The Robustness Of Privacy-Preserving Collaborative Filtering Schemes**" başlıklı **Bilgisayar Mühendisliği** Anabilim Dalındaki, Doktora Tezi 20.03.2015 tarihinde, aşağıdaki jüri tarafından Anadolu Üniversitesi Lisansüstü Eğitim-Öğretim ve Sınav Yönetmeliğinin ilgili maddeleri uyarınca değerlendirilerek kabul edilmiştir.

	Adı-Soyadı	İmza
Üye (Tez Danışmanı):	**Doç. Dr. HÜSEYİN POLAT**	**……**
Üye	**: Prof. Dr. YAŞAR HOŞCAN**	**……**
Üye	**: Doç. Dr. EYYÜP GÜLBANDILAR**	**……**
Üye	**: Yard. Doç. Dr. GÜRKAN ÖZTÜRK**	**……**
Üye	**: Yard. Doç. Dr. MEHMET KOÇ**	**……**

Anadolu Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun …… tarih ve …… sayılı kararıyla onaylanmıştır.

Enstitü Müdürü

ABSTRACT

Ph.D. Dissertation

ON THE ROBUSTNESS OF PRIVACY-PRESERVING COLLABORATIVE FILTERING SCHEMES

İhsan GÜNEŞ

Anadolu University

Graduate School of Sciences

Computer Engineering Program

Supervisor: Assoc. Prof. Dr. Hüseyin POLAT

2015, 120 pages

Privacy-preserving collaborative filtering has been receiving increasing attention. There are various algorithms providing accurate recommendations while preserving privacy. Like collaborative filtering algorithms, privacy-preserving collaborative filtering methods might be subjected to shilling attacks. Such attacks are employed by malicious users to increase/decrease the popularity of some target items. They might affect the overall performance of recommendation systems. Therefore, it is imperative to design such attacks with privacy concerns, determine how robust the privacy-preserving collaborative filtering schemes are, how to find out fake profiles, and analyze them.

In this dissertation, designing shilling attacks with privacy concerns is studied. Also, robustness analysis of various privacy-preserving collaborative filtering schemes (memory-based, model-based, and hybrid methods) is performed. Determining fake or shilling profiles from perturbed databases is scrutinized. Besides employing the modified existing detection methods, a new shilling attack detection algorithm is proposed. Real data-based experiments are conducted for assessing the overall performance. Empirical outcomes show that designing effective shilling attacks with privacy concerns is possible. Also, existing detection methods can be effectively used to determine fake profiles from masked data. In addition, the novel detection method is successful on filtering out shilling profiles. Compared to memory-based and hybrid schemes, privacy-preserving model-based recommendation algorithms are very robust against shilling attacks.

Keywords: Privacy, Shilling, Robustness, Collaborative Filtering, Performance, Recommendation.

ÖZET

Doktora Tezi

GİZLİLİK-TABANLI ORTAK FİLTRELEME METOTLARININ GÜRBÜZLÜĞÜ ÜZERİNE

İhsan GÜNEŞ

Anadolu Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Hüseyin POLAT

2015, 120 sayfa

Gizlilik-tabanlı ortak filtreleme artan ilgi görmektedir. Gizliliği ihlal etmeden doğru öneriler üreten değişik algoritmalar vardır. Ortak filtreleme algoritmalarında olduğu gibi gizlilik-tabanlı ortak filtreleme algoritmaları da şilin ataklarına maruz kalabilir. Bu atakların amacı belli ürünlerin popüleritesini artırmak veya azaltmaktır. Bunlar sistemin genel performansını etkileyebilir. Bu nedenle, bu tür atakların gizliliği koruyarak nasıl tasarlanacağı, gizlilik-tabanlı ortak filtreleme algoritmalarının ne kadar gürbüz oldukları, şilin profillerin nasıl tespit edileceği ve bunların analizlerinin yapılması önemlidir.

Bu tezde öncelikle gizlilik endişeleri olduğunda şilin atakların nasıl tasarlanacağı çalışılmıştır. Ayrıca gizliliği koruyan hafıza-tabanlı, model-tabanlı ve hibrit ortak filtreleme algoritmalarının gürbüzlük analizleri yapılmıştır. Şilin atakların maskelenmiş profiller içeren veri tabanlarında nasıl tespit edilebilecekleri araştırılmıştır. Varolan şilin profil tespit etme metotlarına ek olarak, yeni bir şilin atak tespit algoritması önerilmiştir. Genel performansın analizi için gerçek verilerle deneyler yapılmıştır. Bu deney sonuçları gizliliği koruyarak etkili şilin ataklarının tasarlanabileceğini göstermiştir. Ayrıca mevcut şilin profil tespit metotlarının maskelenmiş veri tabanlarında şilin ataklarını etkili şekilde tespit edebildiklerini göstermiştir. Bunlara ek olarak, yeni metodun şilin profilleri başarılı şekilde tespit ettiği gözlenmiştir. Son olarak, hafıza-tabanlı ve hibrit algoritmalara göre model-tabanlı gizliliği koruyan ortak filtreleme algoritmalarının şilin ataklarına karşı daha gürbüz oldukları görülmüştür.

Anahtar Kelimeler: Gizlilik, Şilin, Gürbüzlük, Ortak Filtreleme, Performans, Öneri.

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Assoc. Prof. Dr. Hüseyin Polat for his great guidance, support, and patience during my dissertation. I feel lucky because I had the opportunity of studying with him. He improved my academic vision.

I would also like to thank my committee members, Prof. Dr. Yaşar Hoşcan, Assoc. Prof. Dr. Eyyüp Gülbandılar, Assist. Prof. Dr. Gürkan Öztürk, and Assist. Prof. Dr. Mehmet Koç for their valuable contributions.

Special thanks to my colleagues, Cihan Kaleli and Alper Bilge, for their scientific support.

Finally, I would like to thank my family, my wife, and all of my friends for their support.

İhsan Güneş

March, 2015

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

a	: Active user
CF	: Collaborative filtering
DegSim	: Degree of similarity with neighbors
DWT	: Discrete wavelet transform
FMD	: Filler mean difference
FMTD	: Filler mean target difference
FMV	: Filler mean variance
i_t	: Target item in shilling profile
I_F	: Filler items
I_S	: Selected items
k-nn	: k-nearest neighbors
lengthVar	: Length variance
MLP	: MovieLens public
p_{aq}	: Prediction on item q for user a
p	: Attack profile
PCA	: Principal component analysis
PCC	: Pearson's correlation coefficient
PLSA	: Probabilistic latent semantic analysis
PPCF	: Privacy-preserving collaborative filtering
PPDM	: Privacy-preserving data mining
q	: Target item
RDMA	: Rating deviation from mean agreement
RMF	: Robust matrix factorization
RPT	: Randomized perturbation techniques
RRT	: Randomized response techniques
SVD	: Singular value decomposition
TFS	: Number of prediction-differences
TMF	: Target model focus
WDA	: Weighted degree of agreement

WDMA	: Weighted deviation from mean agreement
μ	: Mean
σ	: Standard deviation
τ	: Threshold value

1. INTRODUCTION

With the fast improvements in the Internet technologies, e-commerce has attracted growing attention. Nowadays, many people favor shopping via the Internet. Customers can purchase different items such as books, music CDs, foods, and so on via the Internet thorough e-commerce companies. There are millions of items marketed in the e-commerce sites and consumers may have to select from millions of products. With the rise in the number of options; however, amount of information that consumers must take into account has also increased before the customers are able to choose which items meet their needs. Hence, e-commerce sites utilize collaborative filtering (CF) systems to help customers choose the right products. Online sellers receive help from these sites so that they can increase their sales and/or profits by giving suggestions to their customers (Schafer et al., 2001).

The basic functions of CF systems cover recommending items to the consumers, giving personalized item information, reviewing community opinion, and offering community critiques. These systems are modelled for permitting users to locate the preferable items quickly and for preventing the system from the possible excess information. They employ data mining methods to control the similarity among thousands or even millions of data. There are three main processes in these systems: data collections and representations, similarity in determinations, and suggestion computations.

In this chapter, brief information about CF systems is given. Then, how CF systems work and classes of CF systems are explained. There are different challenges that various CF systems face with. Thus, the challenges that CF systems expose and some developed methods to overcome them are explained. Next, the methods developed for disguising data in CF systems are considered. Such methods are used to preserve privacy while still allowing CF scheems to produce recommendations with decent accuracy. Shilling attack problem is dealt with and the methods to overcome this problem are examined. Finally, contributions and organization of the thesis are given.

1.1. Collaborative Filtering

CF schemes are primarily organized by e-commerce companies in order to increase sales by attracting and affecting customers. The term CF was first invented by the Tapestry system (Goldberg et al., 1992), which was originally sketched for e-mail filtering in the early 1990s. CF operates on collected preferences from many users and evaluates predictions depending on similar entities' ratings by using a weighted average approach (Herlocker et al., 2004; Adomavicius and Tuzhilin, 2005). Similarities are computed over all pairwise entries (either users or items) utilizing a similarity metric, such as Pearson's correlation coefficient (PCC), cosine similarity, or trust (Sarwar et al., 2000a; Dokoohaki et al., 2010). Many successful CF systems with respect to the quality of predictions utilize user-based techniques (Bilge et al., 2012; Ortega et al., 2013; Wu et al., 2013). Those systems marketing over a large number of different sorts of items favor item-based solutions (Linden et al., 2003; Li et al., 2014; Zhang et al., 2014).

CF methods have been successful in allowing the prediction of user choices in the recommendation systems (Hill et al., 1995; Ekstrand et al., 2011; Bobadilla et al., 2013). The goal of CF depicting the relationship between the individual and the available data is to further decide the similarity and deliver recommendations. How to term the similarity is an important problem. CF utilizes various similarity decision methods. One assumption is that similar users have similar choices in CF (Desrosiers and Karypis, 2011). Predictions are often in two types: offering prediction for single items and top-N list items that would be preferred by an active user (a).

The users are evaluated according to their choices by CF. For this reason, a database of users' preferences needs to exist. The preferences can be gathered either explicitly or implicitly. In the first case, the user's participation is needed. The user explicitly presents her rating of the given item. Such rating can, for example, be rated with a scale from 1 to 5. Implicit ratings are obtained from observing the user's attitude. In the context of the Web, access logs can be controlled for deciding such implicit selections. For instance, if the user accesses the document, she implicitly rates it 1. If not, the document is supposed to be rated as 0 by the user (Grčar, 2004).

In a traditional CF process, provided a user-item matrix, a similarity metric is used for valuing the similarities between *a* and each user in the matrix. Then, the best similar *k* users are chosen as *a*'s neighbors. Finally, using a CF algorithm and the neighbors' data, a recommendation for a target item (*q*) is approximated. The prediction (p_{aq}) is sent back to *a*.

There are basically three classes of CF algorithms: memory-based, model-based, and hybrid. Memory-based CF algorithms utilize either the whole or a sample of the user-item database to produce a prediction. The neighborhood-based CF algorithm, which is known as a prevalent memory-based CF algorithm, employs the following steps: compute the similarity or weight, w_{ij} reflecting distance, correlation, or weight between any two entities (users or items), *i* and *j*; establishes a neighborhood, and generates a prediction for the active user *a* by getting the weighted average of all the ratings of the user or item on a certain item or user, or using a simple weighted average (Herlocker et al., 2004). When the goal is to produce a top-*N* recommendation, *k* most similar users or items (the closest neighbors) after calculating the similarities need to be found. Upon that the neighbors are accumulated in order to get the top-*N* most frequently purchased items as the recommendation (Su and Khoshgoftaar, 2009).

Model-based CF algorithms develop a model from the system data (user ratings), and this model is utilized for giving suggestions. There are many sorts of model-based algorithms including cluster models, probabilistic models, Bayesian network, rule-based methods, and dimensionality reduction methods. The clustering technique, for instance, first trials to separate the data set into groups of users (Sarwar et al., 2000a). The clustering method employed is the bisecting *k*-means algorithm (Steinbach et al., 2000), a variant of the *k*-means clustering algorithm. The Bayesian network model expresses a probabilistic model for the CF problem (Breese et al., 1998). The rule-based technique applies association regulation discovery algorithms to find association between co-purchased products and then produces item suggestion based on the intensity of the association between items (Sarwar et al., 2000a). Billsus and Pazzani (1998) present a learning algorithm reporting the limitations of CF methods. Their suggested technique is based on dimensionality decrease through singular value decomposition (SVD) of an initial

matrix of user ratings. SVD is used for dimensionality reduction for advancing the functioning of the CF algorithm (Sarwar et al., 2000b). Russell and Yoon (2008) propose employing discrete wavelet transform (DWT) on recommender systems. For making the duration long enough prior to a prediction, data are transformed to these systems and reduced significantly. A new algorithm based on incremental SVD and generalized Hebbian algorithm is proposed (Polezhaeva, 2011). The user/item profiles are revised by the new algorithm effectively when a new user or a new item emerges. It is not necessary to save the initial data matrix.

Hybrid methods unite the advantages of memory- and model-based techniques for solving issues related to the limitations of pure CF. Recommendation functioning of these algorithms is most of the time better than some pure memory- or model-based CF algorithms. Probabilistic memory-based CF unites memory- and model-based approaches (Yu et al., 2004). This method utilizes a combined model built on the basis of a set of saved user profiles and benefits the posterior delivery of user ratings to make prediction. Personality diagnosis is an illustrative hybrid CF method uniting memory and model-based CF algorithms and retaining some benefits of both algorithms (Pennock et al., 2000).

1.2. Challenges of Collaborative Filtering Schemes

CF systems yield quite successful results. Yet, their common use has revealed some real challenges (Schafer et al., 2007; Bobadilla et al., 2013; García et al., 2013). The most important challenges are keeping privacy and being subject to shilling attacks. In addition to these challenges, other challenges can be recorded as accuracy, scalability, sparsity, synonymy, and so on.

Privacy: If the users' privacy is not sheltered by the CF system, the users may reject giving data at all or provide false information. Customers want to assure that their private and personal information are kept properly. Therefore, it is difficult to gather quality user data for CF goals. Having not enough quality user data causes poor recommendation and not accurate prediction for users. When privacy measures are in the system, it will be easy to collect trustable data (Canny, 2002a; Polat and Du, 2005a).

Shilling attacks: If the CF system is susceptible to outline injection attacks, in order to bias the suggestions and prediction, people may provide tons of positive or negative rates. Hateful users or service providers acting as a user may affect the popularity of some target items in terms of either increase or decrease. For achieving that goal, such users or sites want to add false user profiles into data sets. Profile injection attacks are extensively utilized against CF algorithms (O'Mahony et al., 2004; Burke et al., 2005b).

Accuracy: System accuracy is a fundamental matter for recommender systems. Users can easily estimate the accuracy of any system by searching and controlling recommendations for items. For such items the users may already have a choice. By providing lower quality suggestions, any system may collapse in meeting users' expectations. As a result, much research has been performed in terms of evaluating and upgrading the accuracy of recommender systems (O'Mahony, 2004; Herlocker et al., 2004; Choi and Suh, 2013).

Scalability: If the system has high number of users and/or items data, traditional CF algorithms will have serious scalability problems, with computational resources reaching values above the practical or acceptable levels. Many systems have to react immediately to online requirements and make suggestions for all users independent of their purchases and ratings history. This history normally speaking demands a high scalability of a CF system (Schafer et al., 2007; Su and Khoshgoftaar, 2009).

Data sparsity: Practically speaking, many recommender systems are utilized to estimate very large item sets. In this way, the user-item matrix used for filtering will be extremely sparse and the functioning of the prediction or recommendations of the CF systems are questioned. The data sparsity challenge emerges in various states. In particular, the cold start problem happens when a new user or item has just entered the system. It turns to be a challenge to find similar ones since there is scarce information. The cold start problem is also defined as either the new user problem or new item issue. New items cannot be suggested until some users rate it (Su and Khoshgoftaar, 2009).

Synonymy: Synonymy is defined as the tendency of a number of the same or very similar items to have different names or entries. Most recommender systems

cannot find this latent association and in this way, handle these items differently. For instance, the seemingly different items “children movie” and “children film” are in fact the same item. Yet, memory-based CF systems would not able to find match between them to calculate similarity. Indeed, the extent of variability in descriptive term usage is greater than commonly questioned. The prevalence of synonyms lowers down the recommendation performance of CF systems (Su and Khoshgoftaar, 2009).

In the literature, there are several reports proposed to focus on the abovementioned challenges (Sarwar et al., 2000a; Sarwar et al.,2002; Su and Khoshgoftaar, 2009). Since privacy and shilling attacks are essential for the overall success of CF schemes, both privacy and shilling attacks have been seriously investigated. There are many techniques proposed to accomplish confidentiality while achieving truthful recommendations (Agrawal and Srikant, 2000; Polat and Du, 2006; Troiano and Díaz, 2014). In parallel to this, there are many proposed algorithms to detect shilling attacks and enhance the robustness of CF schemes without privacy issues (Chirita et al., 2005; Burke et al., 2006a; Williams et al., 2007; Sandvig et al., 2008; Cheng and Hurley, 2010b). However, as expected, privacy-preserving collaborative filtering (PPCF) might be questioned by shilling or profile injection attacks. On one hand, there are various PPCF schemes. On the other hand, CF schemes might be subjected to shilling attacks. Likewise, PPCF schemes might be subjected to such shilling attack. However, there is no report on PPCF schemes in terms of shilling attacks. Whether the proposed techniques for detecting and preventing shilling attacks in CF systems can be employed to PPCF schemes as well or not has not been studied. If they cannot be applied to PPCF algorithms, new methods should be developed for detecting and preventing profile injection attacks in PPCF techniques. In this dissertation, the abovementioned reports will be addressed. Recent reports on shilling attacks will be investigated. PPCF schemes will be evaluated in terms of profile injection attacks. New techniques will be explored to find and prevent shilling attacks in PPCF systems.

1.3. Privacy-Preserving Collaborative Filtering

Personal privacy is one of the serious threats CF systems face with. For this reason, researchers apply data mining methods whose major concern is privacy. These methods are defined as PPCF schemes. PPCF is one of the privacy-preserving data mining (PPDM) studies. After the research by Agrawal and Srikant (2000) and Lindell and Pinkas (2002), reports on PPDM have begun to increase. PPDM mediates privacy by several ways such as using techniques establishing anonymous results, randomization, cryptographic schemes, and so on. Aggarwal and Yu (2008) made surveys on PPDM models and algorithms.

Privacy concerns in CF services were first mentioned by Canny (Canny, 2002a; Canny, 2002b). The author suggests two different schemes for PPCF. In the first one, he explains a new method for CF that may help with protecting the privacy of individual data. This first technique deals with a probabilistic factor analysis model. Privacy protection is provided by a peer-to-peer protocol. In the second schema, he describes an alternative model. In the alternative model, users control all of their log data. He proposes an algorithm whereby a community of users can calculate a public "aggregate" of their data that does not expose individual users' data. The aggregate permits personalized recommendations to be computed by either members of the community or outsiders.

Polat and Du (2005a) utilized randomized perturbation techniques (RPTs) for establishing privacy in CF systems. In their technique, users perturb their data by adding random numbers to actual ratings. Such random numbers are chosen from a predefined distribution. It is based on the assumption that the value that will be hidden is x, then, to perturb x with RPT, a random number r is added to it. Ultimately, $x + r$ takes place in the database rather than x. In another work, the same authors examined achieving referrals using item-based algorithms on binary ratings with providing users' privacy (Polat and Du, 2005a). For disturbing users' data, their suggestion was employing randomized response techniques (RRTs). In another study by Polat and Du (2005c), how to offer SVD-based prediction with privacy was studied. RPT for achieving confidentiality was applied. The authors in (Polat and Du, 2005b) presented a scheme for binary ratings-based top-N

recommendation on horizontally partitioned data in which two parties own disjoint sets of users' ratings for the same items while preserving data owners' privacy. A privacy-preserving protocol for CF grounded on vertically partitioned data was proposed as well (Polat and Du, 2007). The users might disturb their private data differently. This causes inconsistently masked data. Polat and Du (2007) examine how inconsistent data perturbing affect accuracy and privacy. In their survey, Bilge et al. (2013) mainly concentrated on studying various privacy-preserving recommendation methods according to the data partitioning cases and the utilized methods for preserving confidentiality. The suggested schemes were discussed in terms of their limitations and practical implementation challenges.

1.4. Shilling Attacks

Intensely used by e-commerce web sites to increase sales, CF and PPCF schemes can be susceptible to shilling or profile injection attacks. Although shilling attack concept is first initiated by O'Mahony et al. (2002a, b), Dellarocas (2000) addressed fraudulent attitudes against reputation reporting systems. The goal in that study was to form more robust online reputation systems by identifying frauds. O'Mahony et al. (2002a, b) claimed susceptibilities of recommender systems against attacks to encourage specific recommendations. There are several studies in order to define such possible attacks, detect them, enhance robustness of recommender systems or develop robust algorithms against known attacks, and perform cost/benefit analysis. Moreover, there are a number of studies compiling up-to-date progresses in this area. Some researchers emphasized surveying on shilling attacks and their effects on recommendation systems.

Mehta and Hofmann (2008) surveyed about robust CF methods only. Some robust CF techniques via intelligent adjacent selection, association rules, probabilistic latent semantic analysis (PLSA), SVD, and robust matrix factorization (RMF) were evaluated. These methods fail in guaranteeing to produce robust recommendations under shilling attack scenarios. A relatively recent model-based approach, VarSelect SVD, was also evaluated to give robustness to recommender systems and its stability to shilling was shown. In another survey report, Sandvig et

al. (2008) tested robustness of several model-based CF methods such as clustering, feature reduction, and association rules. In particular, they applied principal component analysis (PCA) to compute similarities and Apriori algorithm to generate recommendations. According to the presented results, model-based approaches are considered to be more resistive to shilling attacks than conventional nearest neighbor-based algorithms. Zhang (2009c) presented a survey of research on shilling attacks, attack detection, and attack evaluation metrics. Zhang (2009c) explains some attack models like random, average, bandwagon, segment, and reverse bandwagon attack in addition to describing well-known attack detection approaches such as generic and model-specific assignments and addressing prediction shift, hit ratio, and *ExptopN* as evaluation metrics.

In addition to the above mentioned survey papers, Mobasher et al. (2007a, b) classified attack forms by taking their dimensions into account, i.e., required knowledge to recognize the attack, intent of attacking, and volume of attack. In the conclusion of the paper, particular attacks with samples were described. In addition to analyzing attack types, the authors also covered detection methods and evaluation metrics in identifying shilling attacks. They also investigated responses of model-based, hybrid, and trust-based recommender systems against shilling attacks. Burke et al. (2005d) outlined some of the important problems for continuing research in robust CF systems such as attack models, algorithms, profiling methods, detection, and evaluation. Burke et al. (2011) discussed attack profiles and concentrated on in particular some of the attack detection methods and presented some of the robust algorithms.

The research performed by Sandvig et al. (2008) concentrated on robust model-based algorithms only. Zhang (2009c) reviewed limited number of attack types, attack detection strategies, and evaluation metrics. The studies presented in (Burke et al., 2005d; Burke et al., 2011; Mobasher et al., 2007b; Mobasher et al., 2007a) cover different aspects of shilling attacks. In a survey paper, Gunes et al. (2014) gave a comprehensive survey including research that has been carried out on the issue of shilling attacks so far as well as analyzed attack descriptions with details, detection methods, robust algorithm design, and cost/benefit analysis and metrics.

1.5. Contributions

There are several studies suggesting to deliver recommendations while maintaining data confidentiality. Likewise, there are different studies targeting possible increases in the robustness of CF systems. Yet, as in CF schemes without privacy concerns, PPCF schemes can also be exposed to shilling attacks. Malicious users and/or sites might try to add fake profiles to obtain nuke and push attacks. This will make the robustness of such schemes worse. The main objective of the dissertation is studying PPCF schemes in terms of profile injection attacks. Main contributions of the dissertation can be summarized as follows.

PPCF schemes mentioned in the literature are analyzed in details, and in this way survey study was conducted (Bilge et al., 2013). Considering various partitioning cases, distributed systems are surveyed for their computational and application level drawbacks. Bilge et al. (2013) examine privacy-preserving measures used in PPCF methods like randomization, cryptography, anonymization, and so on. In this research, a brief explanation was given regarding the evaluation of the overall performance of PPCF schemes.

Shilling attacks against various CF algorithms are surveyed in details (Gunes et al., 2014). Several works with respect to shilling attacks were examined. Arrangements are explained briefly for categorizing shilling attacks and introducing new ones. Major research directions (attack types, attack detection, robust algorithms, and cost/benefit analysis) are studied. Since evaluation takes attention of most of the researchers, detailed explanation of evaluation methodology, benchmark data sets, evaluation measures, and briefly discuss cost in terms of shilling attacks are given in details.

Although PPCF methods can be manipulated through shilling attacks, their robustness against specific attack strategies has not been evaluated explicitly. Furthermore, techniques to establish shilling attacks against masked data have not been studied. Two widespread memory-based PPCF schemes based on two variations of the neighborhood-based prediction algorithm are tested (Gunes et al., 2013a, b). Design methodologies are proposed to modify some principal attacks to be applied in privacy-preserving environments. Six attack strategies, i.e., random,

average, bandwagon, and segment push attacks and reverse bandwagon and love/hate nuke attacks are formed for PPCF schemes. Two primary memory-based PPCF algorithms are tested in terms of robustness when exposed to formerly suggested attacks. It is experimentally depicted that while the PPCF algorithms are very robust against a couple of attacks, they are still as susceptible as well established CF schemes against other types of attacks.

In these previous studies, two memory-based algorithms were studied to show robustness of them against these attacks. The question of whether or not model-based PPCF schemes are robust against shilling attacks is inspected (Bilge et al., 2014). Robustness of four state-of-the-art model-based PPCF schemes is checked against six attack models. The six attack models are constructed for manipulate private preference collections. The model-based schemes that are investigated are *k*-means-, SVD-, item-, and DWT-based PPCF schemes. Revised forms of random, average, bandwagon, and segment push attacks along with reverse bandwagon and love/hate nuke attack models are employed against such PPCF schemes.

In addition to memory- and model-based PPCF schemes, there are hybrid PPCF schemes. Thus, robustness analysis of a hybrid PPCF scheme is conducted (Gunes and Polat, 2015a). The analysis shows that the hybrid scheme is also vulnerable against shilling attacks.

Detecting these types of attacks and lowering their effects for recommendation systems to function correctly are significantly essential. Various detection methods developed and applied to CF algorithms are cited in the literature. However, any work related to detecting the shilling profiles in PPCF algorithms was not performed so far. In this contribution, the most commonly used detection methods applied to CF algorithms are applied for PPCF schemes. In order to test this purpose, the current detection approaches are rendered in such a way that they are applicable to PPCF techniques and experiments might be performed with real data. In practice, six of the modified attacking models formed previously for attacking PPCF algorithms are utilized. A new detection method for PPCF schemes is proposed (Gunes and Polat, 2015b). The novel scheme is based on hierarchical clustering. In order to improve the detection performance, analysis of the ratings of the target items is also proposed.

1.6. Organization of the Dissertation

The rest of the dissertation is organized as follows: In Chapter 2, general background and preliminaries are explained. In Chapter 3, shilling attack models design against PPCF schemes are described in details. Chapter 4 presents shilling attacks against memory-based PPCF and two memory-based PPCF schemes' robustness. Chapter 5 analyzes four model-based PPCF schemes if they are robust against several shilling attack models. In Chapter 6, a hybrid-based PPCF scheme is analyzed in terms of robustness. Chapter 7 scrutinizes how to detect shilling profiles inserted into PPCF systems' databases. Finally, in Chapter 8, concluding remarks and recommendations for further research are discussed.

2. PRELIMINARIES

In this chapter, general background and preliminaries on CF, PPCF, and shilling attacks are explained. Upon describing recommendation systems, CF prediction estimation algorithm is defined. Following that, randomization-based individual privacy protection mechanisms are explained. Then, shilling attack models mentioned in the literature are covered. The properties and the way they are formed are discussed. Finally, real data sets and evaluation metrics utilized in the tests are described.

2.1. Prediction Estimation

In a typical CF system, ratings are saved and a user-item matrix is established, $U_{n\times m}$, having preference information from n users on m items. During an online interaction with a CF system, an active user (a), who has the goal of getting a prediction for a target item q, directs her available ratings to the system. CF prediction approximation is a process with two-steps: (1) locating adjacent ones by calculating similarities between a and all other users in the system and (2) approximating a prediction as a weighted average based on favorites of the adjacent ones on q. Such similarities between a and any user u are computed by different techniques. PCC is one of the best similarity measures. PCC is shown in Eq. 2.1. This formulation was first mentioned in the GroupLens project (Resnick et al., 1994). In that study, PCC was described as the basis for the weight calculation. The correlation between active user a and user u is computed as follows (Breese et al., 1998):

$$w_{au} = \frac{\sum_{j\in M}\left(v_{aj} - \overline{v_a}\right)\left(v_{uj} - \overline{v_u}\right)}{\sqrt{\sum_{j\in M}\left(v_{aj} - \overline{v_a}\right)^2}\sqrt{\sum_{j\in M}\left(v_{uj} - \overline{v_u}\right)^2}} \tag{2.1}$$

in which v_{aj} and v_{uj} are the votes for item j by users a and u, respectively. Likewise, $\overline{v_a}$ and $\overline{v_u}$ are the average votes of users a and u, respectively. M is the number of co-rated items by both a and u. Upon computing the similarities, the most similar k users are marked as neighbors (Herlocker et al., 2004).

GroupLens presented an automated CF system utilizing a neighborhood-based algorithm (Resnick et al., 1994; Konstan et al., 1997). GroupLens gave personalized prediction for Usenet news articles. PCC was utilized by the original GroupLens system to weight user similarity. A prediction for a on q, given as p_{aq}, is generated as a weighted average of scores of the ones adjacent to each other on q by the formula given in Eq. 2.2.

$$p_{aq} = \overline{v_a} + \frac{\sum_{u=1}^{N}(v_{uq} - \overline{v_u})w_{au}}{\sum_{u=1}^{N} w_{au}} \tag{2.2}$$

in which w_{au} is the similarity weight between a and u.

An extension of the GroupLens algorithm, which is used in the current study was proposed by Herlocker et al. (1999). The authors compared the performance of various normalization methods including the bias-from-mean, the z-scores, and the non-normalized ratings. The performance of z-scores was significantly better than the non-normalized rating method. The mean and the standard deviation of the z-scores are 0 and 1, respectively. If the v_{uj} is user u's vote on item j, $\overline{v_u}$ is the mean vote of the user u, and σ_u is the standard deviation for the user u, then the z-scores (z_{uj}) can be given as follows:

$$z_{uj} = (v_{uj} - \overline{v_u})/\sigma_u \tag{2.3}$$

The differences in spread between users' rating distributions were explained by Herlocker et al. (1999) converting ratings to z-scores. A weighted average of the z-scores is computed in the following way:

$$p_{aq} = \overline{v_a} + \sigma_a \times \frac{\sum_{u \in N} w_{au} \times z_{uq}}{\sum_{u \in N} w_{au}} \tag{2.4}$$

$$w_{au} = \frac{\sum_{j \in M}(v_{aj} - \overline{v_a}) \times (v_{uj} - \overline{v_u})}{\sigma_a \times \sigma_u} \tag{2.5}$$

in which M is the item set rated by both the active user a and the user u. σ_a and σ_u are standard deviations of the active user a's ratings and the user u's ratings, respectively. The similarities are calculated and the neighbors are selected based on such similarity weights.

2.2. Privacy Protection by Randomization

Utilizing RPTs make privacy applications possible. A random value r by these methods is added to private data item x for covering that value. The purpose of a random number is to get a predetermined distribution in addition to data values saved in the database in the form $x + r$. Since recommendation systems are applied on accumulated data instead of individual data, the systems can successfully generate recommendation on the grouped and perturbed data. In PPCF schemes, there are two purposes of the privacy protection process in general. These are preventing server to learn true ratings and rated items. Forming random values and accumulating them to their rates help with getting masked data. Users can also generate random values to add some of the randomly chosen unrated items. Gaussian or uniform distribution with zero mean (μ) and standard deviation (σ) are employed by the users for producing random values (Polat and Du, 2005a). In the PPCF scheme, z-score is applied by the users to normalize the ratings. The users define σ_{max} (maximum standard deviation to produce random numbers) and β_{max} designating the maximum percentage of filling unrated items to be filled with noise. They choose σ_u and β_u from the ranges (0, σ_{max}) and (0, β_{max}), respectively. Data masking can be summarized as follows:

1. Each user u computes their z-score values of their ratings.
2. The users decide the values of σ_{max} and β_{max}.
3. β_u and β_u percent of their unrated items are selected by each user u randomly to be filled with random numbers.
4. Then, standard deviation σ_u of random numbers is selected by each user u prior to performing random number distribution. The distribution of random numbers (either uniform or Gaussian) by coin tosses is determined by users.
5. Random numbers (r_{uj} values) for real ratings and unrated items are formed by users. Each user masks their z-score values through random value addition ($z'_{uj} = z_{uj} + r_{uj}$). Each user ultimately fills the selected unrated items by the corresponding random numbers.
6. Finally, the masked vectors are sent by users to the defined server.

2.3. Shilling Attack Models

For increasing the robustness of a CF system against any possible attack, first for which aims attacks are conducted and how generally they are recognized need to be clarified. There are two possible common motivations behind almost all shilling attacks: to either push or nuke a specific item's reputation to get economical advantage over competitors. Usually, a push attack is established to enhance the reputation of a target item so that the recommender system brings it back as a strong suggestion to their customers. Whereas, a nuke attack is planned to lower down the reputation of a target item. Thus, the probability of the target item being recommended will be low (Mobasher et al., 2007b).

Prior to conducting any shilling attack, the attackers must be informed about the recommender system that they try to attack. Such information might cover but not limited to the average rating and standard deviation for each item and/or user in the user-item matrix, ratings distribution, and so on. Low-knowledge attacks must have system independent knowledge that might be received through public sources. However, very detailed knowledge about the recommender system and ratings distribution are required for high-knowledge attacks (Mobasher et al., 2007b). Compared to low- or high-knowledge attacks, the most information is necessary for informed attacks for the target CF system. It is essential to get the high degree of domain knowledge, which is required to choose proper items and ratings used to produce attack shapes (Burke et al., 2011).

Attackers generally recognize shilling attacks by injecting an attack profile as shown in Fig. 2.1, which is first addressed by (Bhaumik et al., 2006; Mobasher et al., 2007b; Mobasher et al., 2007a) to mislead the CF system. Such profiles can be separated into four set of items. First, a set of items, I_S, is determined by the attacker together with a particular rating function δ to establish the properties of the attack. Moreover, another set of items, I_F, is chosen arbitrarily with a rating function θ to hinder detection of an attack. Ultimately, a sole item i_t is targeted with a rating function, Υ, to establish a bias on. Residual items are left unrated shown as I_ϕ in Fig. 2.1. An intelligent strategy for selecting filler items to realize more effective shilling attacks was suggested by (Ray and Mahanti, 2009a). Malicious user

functions as authentic users and forms false profiles (note that user choices about different items characterized in a vector is defined as the profile of that user). Then, these are directed by the user to the recommender system to attack (she injects such false profiles into the attacked system's database).

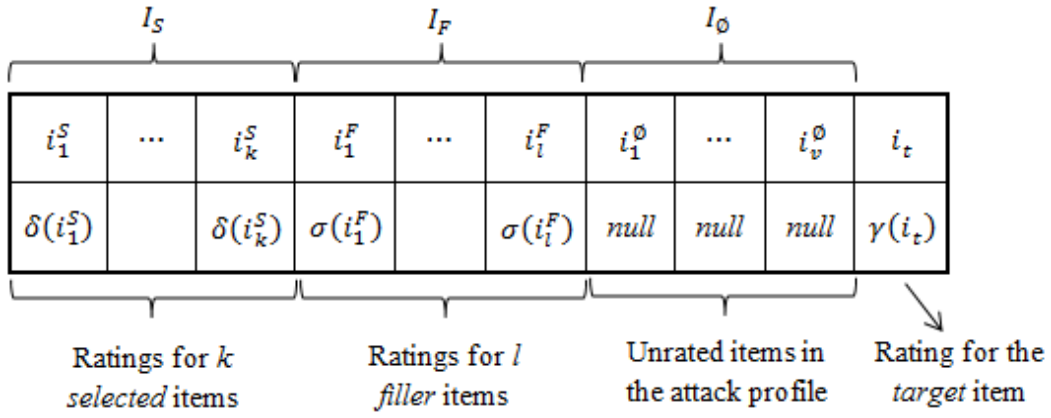

Figure 2.1. General form of an attack profile

Typically, an attack is recognized by placing different attack profiles into a recommender system database to generate bias on chosen target items. Attacks might be employed for different aims and they can be separated into various dimensions like intent of attack and required knowledge (Lam and Riedl, 2004). Table 2.1 lists the most well-known attack types like random, average, bandwagon, segment, love/hate, and reverse bandwagon described by Mobasher et al. (2007b). Table 2.2 shows the attack profiles of popular attack types based on general attack profile given in Fig. 2.1.

Table 2.1. Attack types according to intent and required knowledge

Attack Type	*Intent*		*Required Knowledge*	
	Push	*Nuke*	*Low*	*High*
Random	✓	✓	✓	
Average	✓	✓		✓
Bandwagon	✓		✓	
Segment	✓		✓	
Reverse Bandwagon		✓	✓	
Love/Hate		✓	✓	

Table 2.2. Attack profile summary

Attack Type	I_S		I_F		$I_\emptyset$	i_t
	Items	*Rating*	*Items*	*Rating*		
Random	Not used		Randomly chosen	System mean	$I - I_F$	r_{max}/r_{min}
Average	Not used		Randomly chosen	Item mean	$I - I_F$	r_{max}/r_{min}
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 Bandwagon	Unpopular items	r_{max}	Randomly chosen	System mean	$I - \{I_F \cup I_S\}$	r_{min}
Love/Hate	Not used		Randomly chosen	r_{max}	$I - \{I_F \cup I_S\}$	r_{min}

As shown in Table 2.1, shilling attacks can be defined as either push or nuke according to their intent. Similarly, they are classified as low, high, or informed attacks relating to needed knowledge. Although some attacks can only be utilized for either to push or nuke an item, some can be employed for both intents. As could be seen in Table 2.1, attacks usually need low knowledge. Whereas, average attack requires high knowledge. The most well-known attack kinds can be shortly explained in the following.

Random attack functions through attack profiles with ratings to randomly selected unfilled cells around system overall average and r_{max} or r_{min} to target item for push and nuke attacks, respectively (Burke et al., 2005a; Burke et al., 2005d; Burke et al., 2006a; Mobasher et al., 2007a; Mobasher et al., 2007b; Ray and Mahanti, 2009b). The alternative term for this type of attack is RandomBot attack (Lam and Riedl, 2004; Chirita et al., 2005). *Average attack* functions through attack profiles with ratings to arbitrarily selected unfilled cells around each item's average and r_{max} or r_{min} to target item for push and nuke attacks, respectively. This attack needs high level knowledge. As a result, it is difficult to apply (Burke et al., 2006b; Mehta et al., 2007a; Mehta et al., 2007b; Mobasher et al., 2007b; Mehta and Nejdl, 2008; Ray and Mahanti, 2009b). Alternatively, it is defined as AverageBot attack (Lam and Riedl, 2004; Hurley et al., 2007). In *bandwagon or popular attack*, an attacker produces profiles with elevated ratings to well-known products and the highest possible rating to the target item. In this way, injected profiles can easily be

related to other users in the system in terms of similarity and push the prediction to the target item. It is not difficult to apply this attack since it needs public knowledge rather than domain specific knowledge and as effective as the average attack (O'Mahony et al., 2005; O'Mahony et al., 2006; Cheng and Hurley, 2010a). *Segment or segmented attack* is probed to target a specific group of users who have high intention to purchase a specific item. In attack profiles, attacker injects high ratings for the items the users in the segment probably will prefer to buy, and low ratings for others. Thus, similarity between users in the segment and injected profiles emerges as high probability, and the probability that the target item will be suggested will be high (Burke et al., 2005c; Burke et al., 2005b; Sandvig et al., 2007).

Reverse bandwagon attack is an alternative to bandwagon attack to nuke particular products. In this attack, profiles are produced according to low ratings to products with lower reputation and target item. Likewise, reverse bandwagon attack is comparatively easy to employ (Mobasher et al., 2007a; Zhang, 2009b). *Love*/*hate attack* is an exceptionally effective nuke attack. In this attack, randomly selected filler items are rated with the highest possible rating while the target item takes the lowest one in attack profiles (Mobasher et al., 2007a; Zhang, 2009b).

CF algorithms are often grouped into three major classes as mentioned before: memory-based, model-based, and hybrid CF algorithms. For approximating the predictions, memory-based ones function over the entire user-item matrix (Breese et al., 1998). Thus, their online performance is not good. Whereas, model-based algorithms first form a model off-line from user-item matrix; they then use that model to generate prediction online (Breese et al., 1998). Due to off-line model generation, their online performance is much more promising compared to memory-based schemes. Although model-based CF schemes are faster than memory-based ones, their accuracy is slightly worse than memory-based ones' accuracy. The advantages of both memory- and model-based CF algorithms are united by hybrid approaches (Pennock et al., 2000). Since each of the three algorithms has different properties, upon intending to attack them, different shilling attack strategies should be built.

2.4. Data Sets and Evaluation Criteria

This dissertation 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. It is 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 (Burke et al., 2005b). For measuring the performance of detection methods, the standard measurements of *precision* and *recall* are used. The basic definition of such metrics is given as follows (Han et al., 2011):

Precision = Number of true positives / (Number of true positives + Number of false positives)

Recall = Number of true positives / (Number of true positives + Number of false negatives)

Since we are primarily interested in how successful the algorithms are in detecting the possible attacks, each of these metrics with respect to attack identification is controlled. Thus, *number of true positives* is the number of correctly classified attack profiles, while *number of false positives* is the number of authentic profiles misclassified as attack profiles, and *number of false negatives* is the number of attack profiles misclassified as authentic profiles.

2.5. Experimental Methodology

All-but-one experimentation methodology was used in the experiments. In this methodology, each user is defined as a test or an active user once and the remaining users are assigned to the training set. Moreover, two distinct target item sets were formed, each consisting of 50 movies for push and nuke attacks. Items were randomly chosen using stratified sampling. Intuitively, trying to push a

popular item or nuke an unpopular one is deemed to be unreasonable. Thus, push and nuke attack sets consist of items with averages within range 1-3 and 3-5, respectively. Table 2.3 shows the statistics of the selected target items. During the experiments, all target items were attacked for all test users in the system and predictions were estimated pre- and post-injection of attack profiles. Then, prediction shift values were calculated to indicate relative changes on estimated recommendations for each different attack model. Obtained empirical results for masked push and nuke attack models are presented.

Table 2.3. 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

3. SHILLING ATTACK DESIGN IN PIVACY-PRESERVING COLLABORATIVE FILTERING

In PPCF applications, disguised data from users are gathered for protecting customers' privacy. Although different attack models are planned against non-private rating groups, they cannot be employed directly to databases with covered data. Since the preferences of the users are perturbed prior to submitting them to PPCF servers, an attacker can only collect information about the disguised ratings. This action needs some alterations to the attack models to be applied. Effects of various shilling attack strategies on privacy-preserving frameworks have not been investigated in details. In this section, how to design six famous attack models so that they can be applied to perturbed data is described.

Prior to the generation of different kinds of shilling profiles, the attackers must determine the random number distribution as uniform or Gaussian. Thus, random numbers might be generated and it is possible to choose σ_p uniformly randomly from the range $(0, \sigma_{max}]$ for each attack profile p. Upon determination of these parameters, the attacks can be formed on masked databases, as explained in the following subsections. After discussing how to establish push attack models, the design of nuke attack models will be analyzed in details.

Values of the parameters defined in previous sections, 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. The attacks were then generated based on these values.

3.1. Designing Push Attack Models

3.1.1. Random attack model

Random attack is comparatively easy to employ and a baseline model requiring low knowledge compared to the other attack models (Burke et al., 2005a). Accordingly, a random attack model can be characterized to attack databases including disguised data as follows:

1. The set of chosen items is unfilled ($I_S = \emptyset$).
2. A total of l filler items (I_F) are uniformly randomly chosen from the items except the target item ($I - \{i_t\}$) with respect to a predetermined value of filler size parameter.
3. Utilizing the selected distribution, l+1 arbitrary numbers are produced with μ equal to zero and σ equal to σ_p.
4. The highest value of the produced random numbers is allocated to the target item and the residual numbers are randomly allocated to the l filler items.
5. All users can be targeted via the resulting shilling profiles.

3.1.2. Average attack model

It is difficult to apply average attack since it needs a high level of knowledge about the system (Zhang, 2009a; Gunes et al., 2014). The filler items have values around each item's mean vote. This situation requires calculation of the average value of each item in the system. Correspondingly, average attack profiles should be altered. They can be utilized for attacking the disguised databases as follows:

1. The set of chosen items is unfilled ($I_S = \emptyset$).
2. A total of l filler items (I_F) are uniformly arbitrarily chosen from all items, excluding the target item ($I - \{i_t\}$), employing a predetermined value of the filler size parameter.

3. Utilizing the selected distribution, l random numbers (r_1, r_2, …, r_l) are produced in the interval [$-\alpha$, α]. In this case, α is a disguising parameter utilized to evade discovery of the attack profile.
4. Each derived random number is inserted to the resultant item's mean vote, i.e., $v_i = x_i + r_i$, $i = 1, 2, \ldots l$, where x_i is the mean number of votes for the item i and v_i resembles the value for the item i in the shilling profiles.
5. Ultimately, utilizing the selected distribution, l additional random numbers (t_1, t_2, …, t_l) are produced. In this equation, μ is equal to zero and σ is equal to σ_p. The highest value of the produced arbitrary numbers is appointed to the target item, i.e., $i_t = \max(t_i)$.
6. All users can be targeted via the resulting shilling profiles.

3.1.3. Bandwagon attack model

Bandwagon attack is also known as popular attack. The bandwagon attack is a low-knowledge push attack that needs public information about items. Items known to be popular are given high ratings in shilling profiles to misuse users' interest in generally valued items (O'Mahony et al., 2005). The same strategy can be employed for forming the attack model for privileged collections. Although users mask their choices prior to submission, popular items can still be identified with some accuracy by depending on amassed data as the number of users increase in the system. Average votes for items can be approximated as follows:

$$\overline{V_j}' = \frac{\sum_{j \in N} v_j'}{\#N} = \frac{\sum_{j \in N}(v_j + r_j)}{\#N} = \frac{\sum_{j \in N} v_j + \sum_{j \in N} r_j}{\#N} \approx \frac{\sum_{j \in N} v_j}{\#N} \approx \overline{V_j} \qquad (2.6)$$

in which $\overline{V_j}'$ is the average number of votes computed from the masked data, $\overline{V_j}$ is the real average for item j, and N is the set of users who rated item j. As N enlarges, the expected value of the average of the random numbers tends to converge to zero since they are all produced from a zero-mean distribution. In this way, the disguised bandwagon attack can be applied as follows:

1. Among the items with the highest means, a total of c items with high reputation are chosen ($I_S = \{p_1, p_2, \ldots, p_c\}$).

2. A total of l filler items (I_F) are uniformly arbitrarily chosen among all products except the target item and the chosen items ($I - \{i_t \cup I_S\}$) utilizing a predetermined value for the filler size parameter.
3. Utilizing the selected distribution, $l+c+1$ random numbers ($r_1, r_2, \ldots, r_{l+c+1}$) are formed, where the mean is equal to zero and σ is equal to σ_p.
4. The highest value of the produced arbitrary numbers is appointed to the target item.
5. Then, the top c of the residual arbitrary numbers are randomly appointed to items with higher reputation.
6. Ultimately, the residual arbitrary numbers are randomly placed into l filler items.
7. All users can be targeted via the resulting shilling profiles.

3.1.4. Segment attack model

Segment attack model targets a subset of users with interest in a specific kind of item, such as fantastic movies or jazz music (Burke et al., 2005c). The segment consists of users who have rated highly most of the chosen items. Thus, the attacker attempts to misuse the segmented users' positive interest in the specific items to push approximated prediction for a target item. Attack profiles targeting segmented users can also be formed in reserved systems as follows:

1. A total of h items with high average ratings are chosen 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 arbitrarily chosen among all items except the target item and chosen items ($I - \{i_t \cup I_S\}$) utilizing the predetermined value of the filler size parameter.
3. Utilizing the selected distribution, $l+h+1$ random numbers ($r_1, r_2, \ldots, r_{l+h+1}$) are produced, where the average is equal to zero and σ is equal to σ_p.
4. The highest value of the produced arbitrary numbers is appointed to the target item.
5. Then, the top h of the residual arbitrary numbers are appointed randomly to chosen segment items.

6. Ultimately, the residual arbitrary numbers are randomly inserted into l filler items.

Users to be attacked, or segmented users, are from reserved groups and have positively rated at least P% of the chosen items.

The abovementioned four attack models are designed as push attacks to enhance the reputation of some targeted items in the disguised database. Besides these attack models, two more attack models are designed as nuke attacks to lower down the reputation of targeted items in the perturbed database. This will be discussed in the following sections.

3.2. Designing Nuke Attack Models

3.2.1. Reverse bandwagon attack model

Reverse bandwagon attack is the nuking form of the bandwagon attack. It is intensely operational against item-based algorithms (O'Mahony et al., 2005). This attack model does not need any system particular data, but it needs an accustomed knowledge of the product domain. This will be similar to the bandwagon attack effectively choosing the products with low reputation. Consequently, reverse bandwagon attack profiles for disguised data can be formed utilizing a technique similar to the related bandwagon attack profiles with small differences as follows:

1. From the products with the lowest means and ratings with the highest values, a total of c items with low reputation are chosen ($I_S = \{p_1, p_2, \ldots, p_c\}$).
2. A total of l filler items (I_F) are consistently arbitrarily chosen from among all products except the target item and chosen items ($I - \{i_t \cup I_S\}$) utilizing the predetermined value of the filler size parameter.
3. Utilizing the chosen distribution, $l+c+1$ arbitrary numbers ($r_1, r_2, \ldots, r_{l+c+1}$) are produced, where μ is equal to zero and σ is equal to σ_p.
4. The minimum of the produced arbitrary numbers is appointed to the target item.

5. Then, the lowest c of the residual arbitrary numbers are appointed arbitrarily to chosen unpopular items.
6. Ultimately, the residual arbitrary numbers are randomly placed into l filler items.
7. All users can be targeted via the resulting shilling profiles.

3.2.2. Love/hate attack model

Love/hate attack is one of the most effective models to nuke prediction in user-based CF systems. Besides being simple, it is not necessary to have any knowledge about the system in this model (Mobasher et al., 2007b). Utilizing the love/hate attack model to shill the disturbed database can be explained as follows:

1. The set of chosen items is unfilled ($I_S = \emptyset$).
2. A total of l filler items (I_F) are consistently arbitrarily chosen among all the products but the target item ($I - \{i_t\}$) utilizing a predetermined value of the filler size parameter.
3. Employing the selected distribution, $C \times l$ arbitrary numbers are produced, where the average is equal to 0, σ is equal to σ_p, and C is a constant utilized to assure placing high ratings into the profiles.
4. The highest l of the produced arbitrary numbers are randomly appointed to the filler items, while the minimum number is allocated to the target item.
5. Any user can be directed via the resulting shilling profiles.

4. ROBUSTNESS OF MEMORY-BASED PIVACY-PRESERVING COLLABORATIVE FILTERING SCHEMES

In this chapter, two memory-based PPCF algorithms are investigated, their robustness against several shilling attack strategies are analyzed. The effectiveness of the PPCF filtering algorithms in manipulating predicted recommendations are examined by experimenting on real data-based benchmark data set. It is shown that it is still possible to manipulate the predictions significantly on databases having disguised favorites even though a few of the attack strategies are not operational in a privacy-preserving environment.

4.1. Introduction

There are widespread reports concentrating on responses of recommender systems to promoting attacks in non-private schemes. Yet, they are not successful in protecting individual privacy. Further, previously suggested privacy increasing approaches employed to recommender systems have not been studied for their robustness against shilling attacks. Each group addresses only one side of recommender system technology. There are two techniques to determine neighbors entering into the prediction approximation procedure. In the previous method, among the neighbors, the most similar k neighbors are chosen. This approach is defined as the k-nearest neighbor (k-nn) recommendation algorithm. This algorithm tends to positively consider correlated neighbors only, organized by their similarity values (Herlocker et al., 1999; Herlocker et al., 2004). Contrary to this original method, a revised version of the k-nn algorithm, the correlation-threshold method, also takes negatively correlated users into account with absolute values of similarity higher than a threshold value (τ) (Herlocker et al., 1999). Due to the negatively correlated neighbors, absolute values of the similarity weights are used in the denominator of Eq. 2.5, which is defined previously. In this way, the revised version only filters out users with negligible correlation. However, it covers both positively and negatively correlated users in the prediction procedure. Thus, the exact number of neighbors is not obvious in the correlation-threshold algorithm.

In this part, two primary memory-based PPCF algorithms are tested in terms of robustness when exposed to previously suggested attacks. It is experimentally shown that while the PPCF algorithms are very robust against a couple of attacks, they are still as susceptible as usual CF schemes against other sorts of attacks.

4.2. Experimental Evaluation

Real data-based experiments are performed to estimate the effectiveness of our revised shilling attack models on two memory-based PPCF algorithms. Two control parameters, i.e., *filler size* and *attack size*, were employed in the current evaluations. These are considered for implementing successful shilling attacks in the literature (Bhaumik et al., 2006; Mobasher et al., 2007b). *Filler size* is the percentage of unfilled cells to be filled in bogus profiles, utilizing the rating function, θ, to hinder recognition of the attack, as explained in Section 3 (Bhaumik et al., 2006). *Attack size* is the number of attack profiles to insert, and this is directly proportional to the number of clients in the system (Mobasher et al., 2007b). For instance, five percent attack size is to have 50 attack profiles against a system holding initially 1,000 users. Privacy-preserving parameters are kept constant, β_{max} = 25% and σ_{max} = 2. Such values are enough to give a decent level of individual privacy (Bilge and Polat, 2012).

4.2.1. Empirical results

Throughout the experiments, all focused products were attacked individually for all users in the system. Each prediction was approximated prior to and after inserting the false profiles. Then, the *prediction shift* values were evaluated to show the relative change in forecasted values for various attack models. The number of neighbor was set to 30, chosen 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 present experimental results for push and nuke attacks objected for attacking disturbed databases in the following.

4.2.1.1. Evaluating the effects of push attack models

To exhibit the effects of the revised push attacks on both of the memory-based PPCF algorithms, first experiments with varying filler size were conducted (from 3% to 25%), which is a parameter directly related to the effect of the attack. During these tests, attack size was kept at 15%, which is the highest value in the trial, to enlarge the influence of the operations. The trials were repeated 100 times due to the randomization. The average results are shown in Fig. 4.1 and Fig. 4.2 for *k*-nn and correlation-threshold algorithms, respectively.

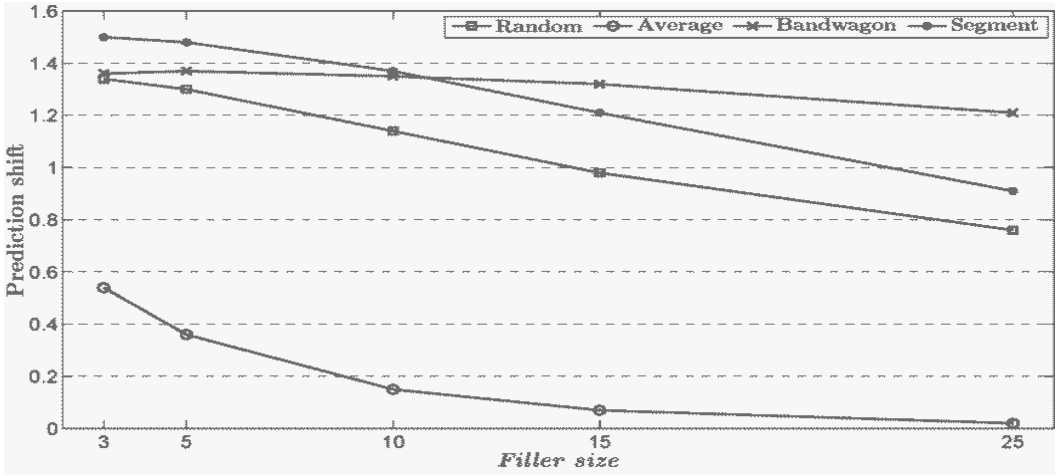

Figure 4.1. Prediction shifts for varying filler size (*k*-nn algorithm)

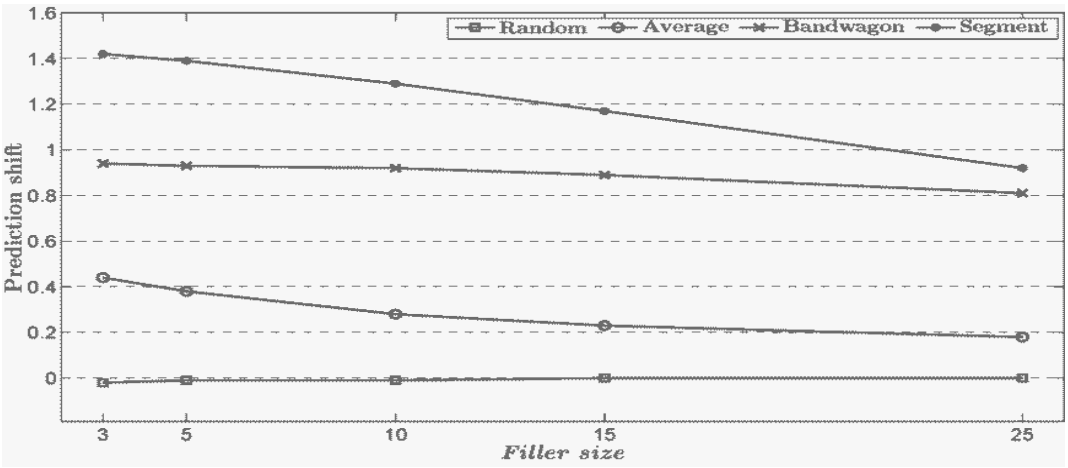

Figure 4.2. Prediction shifts for varying filler size (Correlation-threshold algorithm)

As indicated in Fig. 4.1 and Fig. 4.2, bandwagon and segment attacks are more efficient against PPCF algorithms. The revised bandwagon attack obtained a largest prediction shift of 1.37 and 0.94 for *k*-nn and correlation-threshold algorithms, respectively. The suggested segment attack is slightly more fruitful and steady, obtaining a prediction shift of nearly 1.45 for each algorithm. On a five-star scale, this prediction shift has importance. A highest average prediction shift of 1.34 is calculated for the revised random attack against the *k*-nn algorithm. Yet, this attack does not function sufficiently against the correlation-threshold algorithm. The mean attack is less fruitful but more steady than the random attack model for masked data and reaches a prediction shift of approximately 0.45 for both algorithms. There is inverse proportionality between filler size and prediction shift for all attack schemes. This is explained instinctively as the optimum value of filler size is interrelated with data density. In this way, the maximum prediction shifts are reached for 3% and 5% filler size, approximately close to the general density of the data set.

Then another set of experiments was performed with differing attack size (from 1% to 15%) to inspect the effects on the prediction shift of the number of inserted profiles. During this set of trials, filler size is maintained constant at 15%. This was anticipated to maximize the influence. Experiments were reiterated 100 times due to the randomization in the disturbance procedure. The general means of the results are shown in Fig. 4.3 and Fig. 4.4 for *k*-nn and correlation-threshold algorithms, respectively.

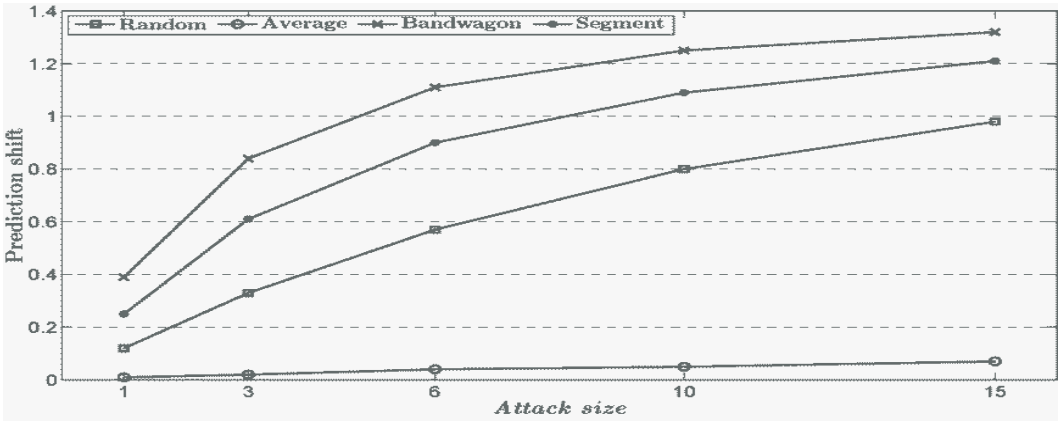

Figure 4.3. Prediction shifts for varying attack size (*k*-nn algorithm)

Similar to the former experiments, the random attack functions well against the *k*-nn algorithm, obtaining a prediction shift maximum of 0.98. However, it is entirely ineffective against the correlation-threshold algorithm. Likewise, the average attack is not successful in all experiments for both of the algorithms due to the filler size. Moreover, both the bandwagon and segment attacks function similarly as in the case of the former experiments, except that the performance of the bandwagon is slightly better than the segment attack against the *k*-nn algorithm. Overall, it can be concluded that the prediction shift grows as the attack size increases, as instinctively anticipated.

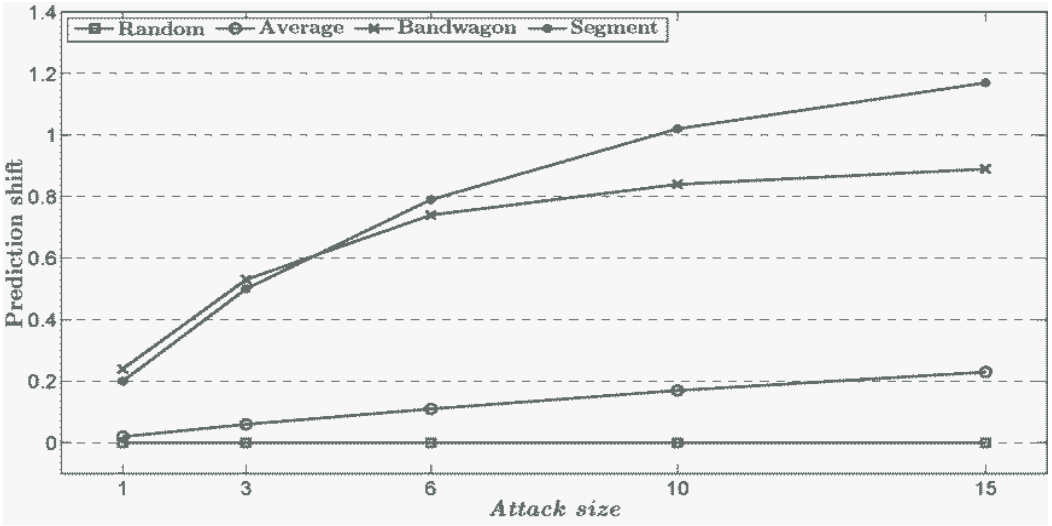

Figure 4.4. Prediction shifts for varying attack size (Correlation-threshold algorithm)

Depending on the general prediction shifts indicated in Fig. 4.1 to Fig. 4.4, it is also concluded that the correlation-threshold algorithm is more robust than the *k*-nn algorithm. This result is clarified through the neighbor choosing techniques of the two algorithms. The correlation-threshold algorithm accepts both negatively and positively correlated users. Yet, *k*-nn only covers strongly and positively correlated users. As the very nature of the attacks is to form a positive relationship with users, the *k*-nn algorithm is more susceptible to push attacks like random, average, bandwagon, and segment attacks.

4.2.1.2. Evaluating the effects of nuke attack models

To show the effects of the revised nuke attacks on memory-based privacy-protecting algorithms, new experiments with varying *filler size* and *attack size* were conducted. First, attack size is set to 15% to try the effects of differing filler sizes for both of the algorithms. The experiments were reiterated 100 times to generate randomness in the perturbation process. General means of the results for varying filler sizes are indicated in Fig. 4.5.

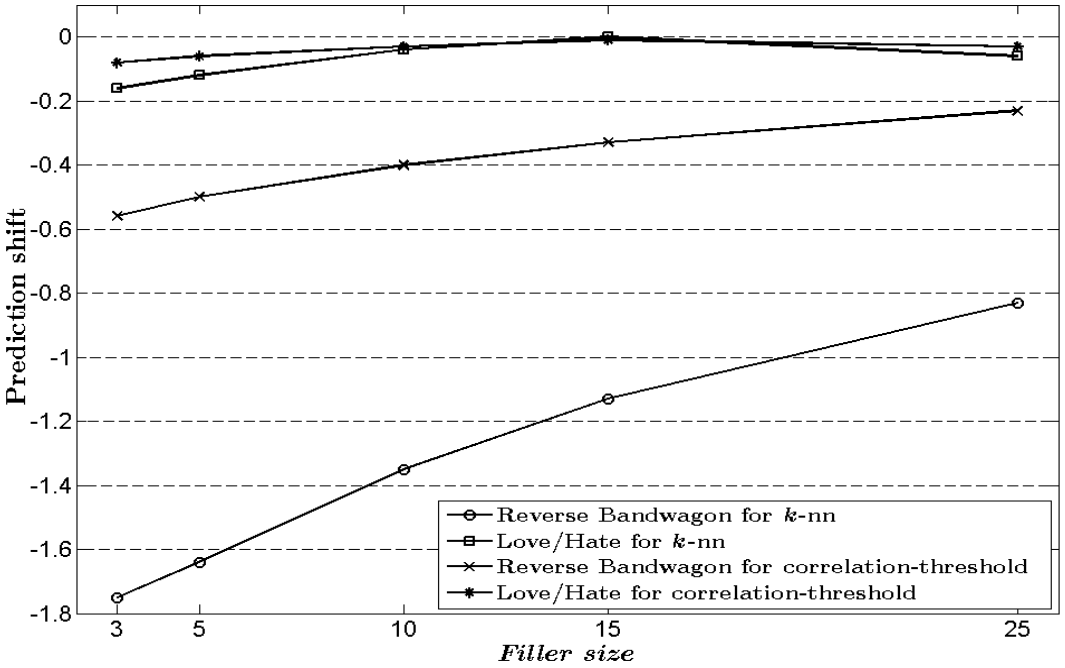

Figure 4.5. Prediction shifts for varying filler size

As shown in Fig. 4.5, the love/hate attack model is entirely impractical against both algorithms and never obtains a significant prediction shift with differing filler magnitudes. Even though the love/hate is an effective attack in non-private environments, it is not likely to achieve strong relationships depending on shilling profiles from the masking scheme. Appointing high z-score values does not assure high similarity values, since similarity weight is approximated via dot products and profiles also cover negative values. Whereas, reverse bandwagon functions effectively, in particular against the *k*-nn algorithm, reaching a maximum nuke

prediction shift of 1.75. Likewise to the outcomes of the push attacks, effects of the filler size are intensely related to the data set density. In this way, a lower prediction shift is obtained with greater values of filler size. Furthermore, correlation-threshold algorithm is more robust compared to the *k*-nn algorithm (maximum 0.56 nuke predictions), due to the reasons covered above.

The next step of the experiment is to set the filler size to 15% to examine the effects of differing attack sizes for both of the algorithms. Once more the experiments are reiterated 100 times to produce randomness in the disturbance procedure. The results indicate that the love/hate attack cannot obtain any prediction shift (prediction shift values are all 0.0) for differing attack sizes in both of the algorithms. Contrary to the love/hate attack, in the case of bandwagon attack, increasing attack sizes results in growing prediction shift for both of the algorithms. However, the correlation-threshold algorithm is healthier than the *k*-nn algorithm. For the 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.

4.3. Conclusions

Many studies have tested CF schemes without privacy concerns in terms of shilling attacks. Likewise, some researchers also explored recommendation algorithms with respect to privacy. On the one hand, there are helpful studies describing shilling attacks that fail to focus on privacy protection. On the other hand, various schemes are suggested to give recommendations on privacy without studying shilling attacks. Privacy-preserving prediction techniques can also be exposed to shilling attacks. These systems have not been assessed in terms of their robustness against profile injection attacks. Hence, two well-known memory-based PPCF algorithms subjected to shilling attacks were studied. New methods to form shilling profiles to be injected into masked databases in privacy-preserving prediction systems are proposed in the current work. Privacy-preserving *k*-nn and correlation-threshold algorithms with respect to their robustness are also intuitively assessed. Experimental outcomes indicate that these systems are also susceptible to

profile injection attacks, similar to classical CF schemes. Our suggested revised bandwagon, segment, and reverse bandwagon attacks obtained significant changes in generated prediction. However, it is exhibited that the revised love/hate model is not effective, which is attributed to the data masking mechanism. Moreover, it is empirically confirmed that the correlation-threshold algorithm is more robust than the *k*-nn algorithm, since its principle of establishing neighborhoods disproves the logic of shilling attack profile design.

In the present section, the capacity for profile injection attacks to be successfully attached against memory-based privacy-preserving schemes is explored. The significance of experimental outcomes is that the outcomes verify the applicability of some attacks on recommendation schemes with privacy. This leads us to question the robustness of other techniques. Therefore, other ways of preserving individual privacy, such as data substitution techniques, and other schemes of giving private prediction, such as item- or model-based CF schemes with privacy, need to be explored against profile injection attacks.

5. ROBUSTNESS OF MODEL-BASED PIVACY-PRESERVING COLLABORATIVE FILTERING SCHEMES

In this chapter, robustness of four well-known privacy-preserving model-based recommendation methods against six-shilling attacks is investigated. First, disguised data-based profile injection attacks are employed to privacy-preserving *k*-means-, DWT-, and SVD-, and item-based prediction algorithms. Then complete experiments are conducted using real data to assess their robustness against profile injection attacks. Next, non-private model-based methods are compared with their privacy-preserving correspondences in terms of robustness. Furthermore, well-known privacy-preserving memory- and model-based prediction techniques are compared with respect to robustness against shilling attacks. Experimental analysis indicates that couple of model-based schemes with privacy is very robust.

5.1. Introduction

PPCF schemes are generally classified as either memory- or model-based schemes. Memory-based methods with privacy are the simplest heuristic techniques. It is not difficult to implement such techniques in the process of generating predictions. Because memory-based algorithms work online, inserting a new user or item into the collection is facile. It is not required to assess the content of the suggested products. The mechanism scales well with co-rated items. Whereas, in scaling these systems the data size might be an obstacle. When a new user enters into the system, suggestion for that user might not be possible due to data sparseness. Privacy-preserving model-based CF algorithms generate a model relying on user ratings as well as giving prediction. Even though they function better in terms of scalability and sparsity issues, employing them is harder compared to memory-based ones. Using the model, they find either item or user similarities off-line. When a new item or user is inserted, a fresh model should be established. Yet, this procedure is computationally costly. Further, useful data can be wasted during a particular model generation. This may decrease accuracy.

CF schemes without privacy concerns are explored in terms of profile injection attacks. Various methods are suggested to solve the shilling issue. Similarly, several PPCF schemes are recommended to overcome the privacy problem. In addition to protecting confidentiality, preventive techniques for PPCF schemes against shilling attacks are also claimed. However, there are not sufficient studies to inspect PPCF schemes with respect to shilling attacks. There are not many researches on PPCF's robustness against shilling attacks. In the previous chapter, couple of memory-based PPCF schemes are investigated with respect to shilling attacks. In this chapter, the issue of whether or not model-based PPCF schemes are robust against shilling attacks is tested. Robustness of four state-of-the-art model-based PPCF schemes is controlled against six attack models. These models are planned to manipulate private preference collections. Investigated model-based schemes are *k*-means-, SVD-, item-, and DWT-based PPCF schemes. Revised versions of random, average, bandwagon, and segment push attacks along with reverse bandwagon and love/hate nuke attack models are employed against such PPCF schemes.

5.2. Model-based Collaborative Filtering Schemes

In this section, four state-of-the-art model-based CF algorithms covered in this study are described to give the reader a background on recommendation mechanisms of algorithms.

5.2.1. *k*-means clustering-based collaborative filtering

In the *k*-means clustering algorithm, initial objects for cluster centers are arbitrarily selected. Each article is then appointed to the closest cluster based on similarity scale. In each repetition, cluster centers are re-approximated as the mean of the articles. This algorithm is completed when there is no change observed in the cluster members. *k*-means clustering is also utilized for solving the scalability issue in CF (Kim et al., 2011). *k*-means clustering-based CF algorithm places user

profiles into *k* clusters off-line. When an active user *a* wants a prediction for item *q*, the server decides *a*'s similarity to each cluster center using PCC as follows:

$$w_{ac} = \frac{\sum_{j=1}^{m}\left(v_{aj} - \overline{v_a}\right)\left(v_{cj} - \overline{v_c}\right)}{\sigma_a \sigma_c} \tag{5.1}$$

in which *c* is cluster center, $\overline{v_a}$ and $\overline{v_c}$ are average ratings of *a* and the cluster center *c*, respectively. Similarly, σ_a and σ_a are standard deviations of the ratings of *a* and the cluster center *c*, respectively. Further, v_{ij}, usually, is the rating of user *i* on item *j*. The cluster with the largest similarity with *a* is decided. After that the similarities between *a* and her cluster members are computed. In this way, clustering in CF reduces down similarity procedure by computing similarities between *a* and her cluster members only rather than all users within the system. Prediction for *a* on item *q* is computed as weighted mean of the neighbors' z-scores as follows:

$$p_{aq} = \overline{v_a} + \sigma_a \times \frac{\sum_{u=1}^{N} z_{uq} w_{au}}{\sum_{u=1}^{N} w_{au}} \tag{5.2}$$

in which *N* is the number of users in the corresponding cluster, w_{au} is the similarity between the active user *a* and the adjacent user *u* and z_{uq} is the z-score of the user *u* on the item *q*.

5.2.2. SVD-based collaborative filtering

By increasing the number of users and/or items, CF systems might face with the scalability issue. For overcoming such an issue, it is possible to employ SVD for CF algorithms. SVD lowers down dimensionality of database containing user/item rates and increases the functioning of the CF algorithm. SVD is known as a matrix factorization method factoring an $n \times m$ matrix *A* into three matrices as $A = USV^T$. Note that *U* and *V* represent two orthogonal matrices of size $n \times r$ and $m \times r$, respectively, while *r* is the rank of the matrix *A*, and *S* is a diagonal matrix of size $r \times r$ having all singular values of matrix *A* on its diagonal entries. SVD-based CF algorithm is employed as a scalable technique (Sarwar et al., 2000b, Polezhaeva, 2011). First, the empty user-item matrix *A* is filled through employing the mean item for items. Adding z-scores normalizes the filled matrix and A_{norm} is found.

Then, A_{norm} matrix is factored into three matrices as *U*, *S*, and *V* by using SVD. To get matrix S_k, $r \times r$ matrix *S* is reduced by choosing only *k* largest diagonal values, where $k << r$. $U_k\sqrt{S_k}$ and $\sqrt{S_k}V_k^T$ are then computed. The scalar product of a^{th} row of $U_k\sqrt{S_k}$ and the q^{th} column of $\sqrt{S_k}V_k^T$ is calculated, the outcome is de-normalized, and the prediction for user *a* on item *q* is approximated as follows:

$$p_{aq} = \overline{v_a} + \left[U_k\sqrt{S_k}(a)\sqrt{S_k}V_k^T(q)\right]. \quad (5.3)$$

Note that p_{aq} is the prediction for the active user *a* on the aimed item *q* and $\overline{v_a}$ is the *a*'s average rating.

5.2.3. Item-based collaborative filtering

While user- or memory-based CF methods indicate challenges as scalability and sparsity, item-based CF approaches are developed to solve these challenges (Desrosiers and Karypis, 2011; Sarwar et al., 2001; Wen and Zhou, 2012). Several commercial recommender systems are selected for assessing large items sets. Users can buy or rate a small amount of items on these systems. Likewise, a new user or item can be just entered into the system. In these circumstances, detecting similar objects can be difficult because of the insufficient information. Therefore, recommender systems based on neighbor algorithms may not produce a recommendation for a particular user.

Item-based CF depends on calculating item-item similarities off-line. A set of items that are rated by *a* are explored. The algorithm then computes how similar they are to the item *q*. The most similar *k* items are decided as neighbors. After detecting the most similar item, the prediction is computed online through weighted mean of *a*'s ratings on these similar items. To compute item-item similarities, adjusted cosine item-item similarity metric can be used as follows:

$$sim_{ij} = \frac{\sum_{u\in U}(v_{ui} - \overline{v_u})\left(v_{uj} - \overline{v_u}\right)}{\sqrt{\sum_{u\in U}(v_{ui} - \overline{v_u})}\sqrt{\sum_{u\in U}\left(v_{uj} - \overline{v_u}\right)}} \quad (5.4)$$

in which *U* represents set of users who rated items *i* and *j*, v_{ui} is the rating for user *u* on item *i*, $\overline{v_u}$ is the mean rating for user *u*, and sim_{ij} is the resemblance weight

between i and j. Prediction for user a on item q is ultimately computed by calculating weighted sum of a's ratings for similar item as follows:

$$p_{aq} = \frac{\sum_{j \in N} v_{aj} sim_{jq}}{\sum_{j \in N} sim_{jq}} \tag{5.5}$$

in which N is q's neighbors and v_{aj} is the rating for user a on item j.

5.2.4. DWT-based collaborative filtering

One of the methods employed for data reduction is called DWT. DWT-based CF schemes are planned to solve the scalability issue of memory-based recommendation techniques. DWT was first utilized by Russell and Yoon (2008) to reduce the amount of items for obtaining scalability in recommendation procedure. The scheme divides the original user-item matrix into two components for each pair. These constituents are called approximation and detail coefficients, which are shown as follows:

$$C_{appx} = \frac{v_{aj} + v_{a(j+1)}}{\sqrt{2}} \;\&\; C_{dtl} = \frac{v_{aj} - v_{a(j+1)}}{\sqrt{2}} \tag{5.6}$$

in which v_{aj} is the rating for user a on item j. Even though each coefficient composed of half of the items, approximation coefficient has a large fraction of the information. DWT functions with successful transformations on approximation coefficient, which stores a large fraction of the original data. However, a small portion of information is lost in the procedure. Given an $n \times m$ matrix, DWT lowers down the size of the matrix to $n \times (m/2^k)$ after k transformations. Russell and Yoon (2008) employed PCC to compute similarities among users by reduced data. Since the number of items is lowered down, computing similarities becomes faster. After the completion of similarity computing, the best N similar users are chosen as adjacent for a particular user a. The prediction for a is predicted by employing adjusted weighted sum CF technique.

5.3. Shilling Attacks against Model-based Prediction Schemes with Privacy

On the one hand, model-based recommendation schemes are explored in terms of shilling attacks and their strength against profile injection attacks is assessed without thinking privacy safety. On the other hand, various schemes are offered to propose predictions using such model-based techniques while maintaining confidentiality without thinking shilling attacks. Yet, privacy-preserving model-based CF algorithms should be evaluated with respect to strength against different shilling attacks because they might be exposed to such attacks. In this way, four well-known model-based CF schemes are scrutinized in terms of strength against six shilling attacks.

Privacy-preserving *k*-means clustering-based CF scheme is suggested by Bilge and Polat (2013). The authors study how to propose *k*-means clustering-based predictions fundamentally while maintaining individual user's privacy. Polat and Du (2005c) offered SVD-based CF having privacy protection. They employed RPTs for obtaining privacy in addition to proposing accurate suggestions. Due to z-score normalization, predictions are approximated as follows:

$$p_{aq} = \overline{v_a} + \sigma_a \times \left[U_k\sqrt{S_k}(a)\sqrt{S_k}V_k^T(q)\right]. \quad (5.7)$$

Recall that $\overline{v_a}$ and σ_a symbolize the active user a's mean rating and standard deviation of her ratings, respectively; and $U_k\sqrt{S_k}(a)\sqrt{S_k}V_k^T(q)$ is the scalar product of a^{th} row of $U_k\sqrt{S_k}$ and the q^{th} column of $\sqrt{S_k}V_k^T$. Polat (2006) extends item-based prediction algorithm by changing ratings into z-scores. Predictions are approximated with the equation shown below:

$$p_{aq} = \overline{v_a} + \sigma_a \times \frac{\sum_{j\in N} z_{aj} sim_{jq}}{\sum_{j\in N} sim_{jq}}. \quad (5.8)$$

Notice that because z-scores (z_{aj} values) are utilized, the weighted mean is de-normalized by multiplying it with the a's standard deviation (σ_a) and adding the a's mean rating ($\overline{v_a}$). Also note that sim_{jq} symbolizes the similarity weight between items j and q. Russell and Yoon (2008) do not count privacy preservation on DWT-based CF scheme. Bilge and Polat (2012) suggest privacy-maintaining DWT-based

CF without putting users' privacy at risk. Due to z-score normalization, Bilge and Polat (2012) calculate p_{aq} as follows:

$$p_{aq} = \overline{v_a} + \sigma_a \times \frac{\sum_{u=1}^{N} z_{uq} w_{au}}{\sum_{u=1}^{N} w_{au}}. \tag{5.9}$$

in which w_{au} is the similarity weight between users a and u; and z_{uq} is the z-score of user u on item q.

In this thesis, four privacy-preserving model-based CF schemes are scrutinized with respect to strength against six well-known shilling attacks. Four of these attacks are push attacks (random, average, segment, and bandwagon attacks). These push attacks target to enhance the reputation of target items. Two of them are known as nuke attacks (reverse bandwagon and love/hate attacks). These two attacks are utilized to lower down the popularity of target items.

In PPCF schemes, users mask their personal and private data before uploading them on CF systems. As a result, it becomes difficult to employ traditional shilling attack models against PPCF systems. Because of the disguised ratings in PPCF schemes, the attackers need some revision on conventional attack models. Gunes et al. (2013b) redesign traditional attack models against disguised databases. Then the robustness of memory-based CF scheme against six modified shilling attack models is explored. As stated by Gunes et al. (2013b), attackers have to determine on random number distribution as either uniform or Gaussian to produce arbitrary numbers. Furthermore, σ is chosen uniformly arbitrarily from the range (0, σ_{max}] for each attack profile prior to producing shilling profiles.

5.4. Experimental Evaluation

To indicate the effects of the six shilling attack models on four model-based PPCF algorithms, real data-based experiments were performed. Effects of shilling attacks were assessed in terms of the two control parameters, *filler size* and *attack size*. Empirical outcomes show that model-based PPCF schemes are more robust than memory-based ones.

5.4.1. Empirical results

5.4.1.1. Effects of filler size parameter

Experiments were conducted with the aim of showing the effects of the disguised push and nuke attack models with changing filler size values on four privacy-preserving model-based recommendation algorithms. Filler size is directly correlated to the success of a conducted attack since filler items establish the base for leaking into neighborhoods of genuine users in the recommendation procedure. Since β_{max} is set to 25% initially, during the tests, filler size is changed from 3% to 25%. Further, attack size is kept constant at 15%, which is the highest value of attack size value tested. The number of predefined clusters is considered as three for *k*-means clustering-based PPCF and a three-level alteration is conducted for DWT-based PPCF scheme, as it is mentioned to be optimal by Russell and Yoon (2008) and Bilge and Polat (2012). Experiments were repeated 100 times due to the necessity of establishing randomization in the disturbance procedure and mean outcomes are given. Prediction shifts for DWT- and *k*-means clustering-based PPCF schemes are shown in Fig. 5.1 and Fig. 5.2, respectively due to relatively high shifts. Yet, prediction shifts for SVD- and item-based PPCF algorithms are given in Table 5.1 due to smaller shifts being closer to zero.

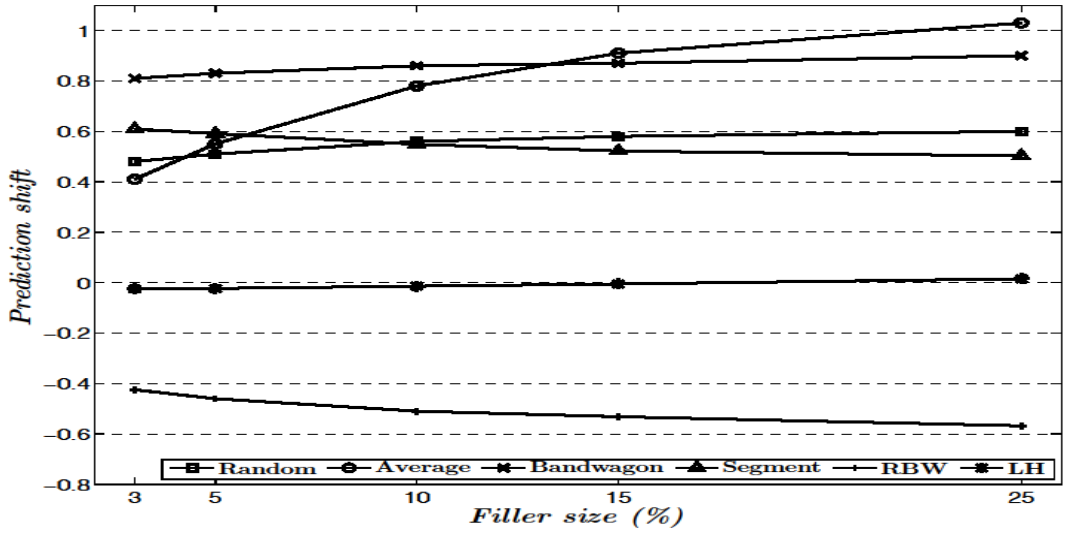

Figure 5.1. Prediction shifts for varying filler size (DWT-based scheme)

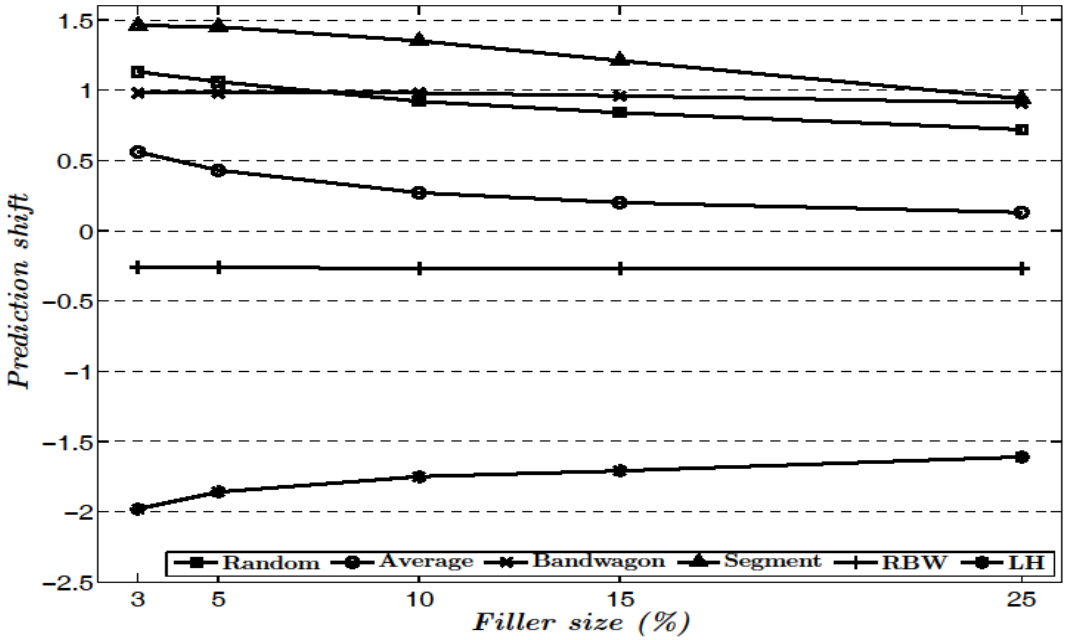

Figure 5.2. Prediction shifts for varying filler size (*k*-means-based scheme)

As indicated in Fig. 5.1 and Fig. 5.2, DWT- and *k*-means clustering-based PPCF algorithms can be susceptible to shilling attack manipulations. In terms of DWT-based scheme, all attack models excluding love/hate attack obtain prediction shift values worth noting. A positive shift between 0.7 and 1.0 is possible for average and bandwagon push attack models. Segment and random attack models also have a prediction shift around 0.5. Moreover, as filler size expands, already achieved shifts significantly enhance for average push attack and slightly lower down for segment attack. This happens due to the alteration of the successive items together. As filler size expands, such alteration can be conducted between more items increasing success of average attack, which does not rely on particularly chosen items but using all filler items with their mean votes. However, segment attack heavily depends on a chosen category of items. As a result, the more the filler items are added into profiles, the less it is likely to keep such chosen items' manipulation effects during alteration. The same case is also valid for bandwagon attack. However, it is not deeply affected from increases in filler size. Moreover, reverse bandwagon attack is also successful in decreasing the reputation of items. Love/hate attack is entirely ineffective. This again arises from the DWT-based algorithm's alteration procedure. This diminishes the effects of "love" part of the

attack, i.e. all high values given to the filler items are normalized through the transformation. Therefore, it might be concluded that the DWT-based scheme is not strong against shilling revision. Yet, it is more resistant to attacks, which focus on chosen item strategy and profiles having all extreme values due to its normalization-based approach. Similarly, *k*-means clustering-based recommendation scheme is not strong enough against random, bandwagon, segment, and love/hate attack models. This situation happens since such scheme does not change the profiles anyhow during the recommendation process. Therefore, proposed attacks can be more easily recognized as long as the attack profiles meet with genuine users in clusters. Thus, it obtains positive prediction shifts close to 1.5 and negative shifts as -2.0. This might be thought of highly significant in a five-star rating scale. Because the number of clusters is limited, such a scheme becomes susceptible to attacks such as memory-based algorithms. Moreover, when filler size expands the effects shrink, because more filler items aid the scheme discriminate attack profiles and instinctively cluster them together. However, discrimination mechanism over clusters is prone to group exceptionally similar attack profiles together as in average attack. As a result, as filler size expands average attack becomes less successful.

Table 5.1. Prediction shifts for varying filler size

Attack Type	*Filler Size (%)*				
	3	5	10	15	25
SVD-based PPCF					
Random	0.0000	0.0000	0.0000	0.0001	0.0003
Average	0.0001	0.0001	0.0002	0.0003	0.0003
Bandwagon	0.0001	0.0001	0.0001	0.0001	0.0002
Segment	0.0001	0.0002	0.0002	0.0002	0.0002
Reverse BW	-0.0014	-0.0015	-0.0016	-0.0016	-0.0016
Love/Hate	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000
Item-based PPCF					
Random	0.0172	0.0174	0.0178	0.0180	0.0183
Average	0.0184	0.0188	0.0194	0.0199	0.0209
Bandwagon	0.0169	0.0173	0.0179	0.0180	0.0180
Segment	0.0710	0.0727	0.0754	0.0786	0.0805
Reverse BW	-0.0159	-0.0159	-0.0164	-0.0168	-0.0169
Love/Hate	-0.0181	-0.0181	-0.0182	-0.0182	-0.0186

SVD- and item-based PPCF have strength against the employed attack models as shown in Table 5.1. SVD-based scheme is exceptionally strong against

manipulations as in the case of being in non-private schemes (Mehta and Hofmann, 2008). Achieved prediction shifts are not significant for both of the push and nuke attacks. SVD is utilized in noise removal procedure usually. As a result, it is successful in removing noisy effects of attack profiles. The item-based PPCF scheme is also resistant to such attacks. However, the origin of its resistance does not arise from its strong recommendation technique, but a natural defense mechanism due to production of the predictions based on item-item similarities. Since it is not sensible to add an item profile into a PPCF database, all attack models concentrate on inserting user profiles. This in turn gives item similarities-based recommendation schemes a strong mechanism. Compared to SVD-based PPCF scheme, item-based is slightly more susceptible to attacks. Yet, only two-digit fractions are achieved at most. This concludes that SVD- and item-based PPCF schemes function in a robust manner against shilling attacks.

5.4.1.2. Effects of attack size parameter

Another set of experiments was conducted to inspect the effects of the attack models with changing attack size values on model-based PPCF algorithms. Attack size is the second parameter directly affecting overall success of a profile injection attack. While the filler size parameter handles utility perspective of an attack, attack size concentrates on influence of such usefulness by deciding the number of bogus profiles to be added into a database. It is clear that the more attack profiles inserted into the system, the larger the obtained shifts are. However, it establishes a trade-off between the detectability and the influence of the employed attack model. Therefore, for defining different effects of the attack size parameter, it is varied from 1% to 15% while the filler size is kept constant at 15%. As in the previous set of tests, initial number of clusters is set to three for *k*-means clustering-based PPCF and a three-level alteration is conducted with the DWT-based scheme. Likewise, the experiments were repeated 100 times in order to generate randomization. Average prediction shifts values for DWT- and *k*-means clustering-based PPCF schemes are shown in Fig. 5.3 and Fig. 5.4, respectively. Experimental results for SVD- and item-based PPCF schemes are listed in Table 5.2.

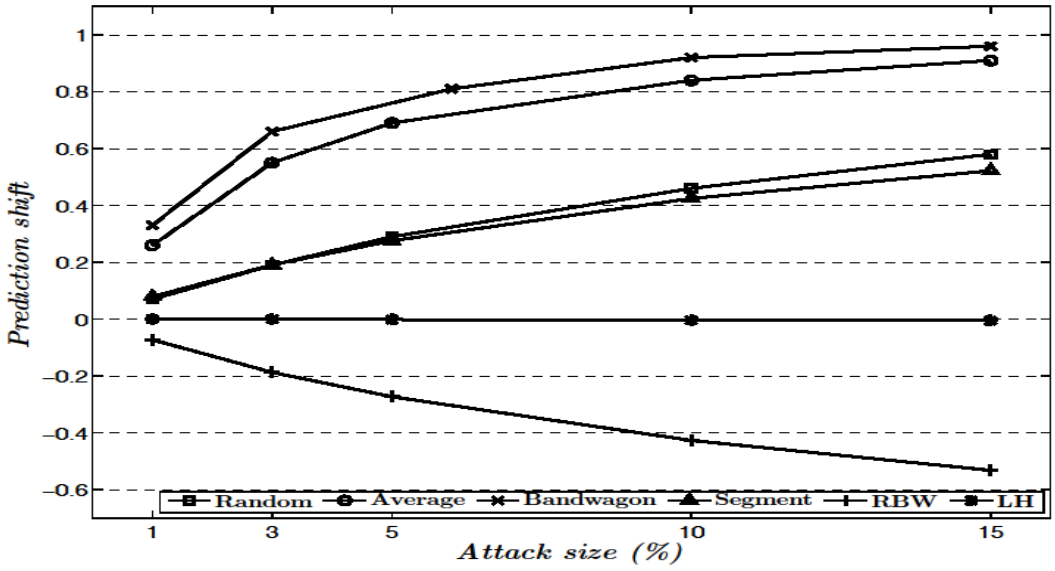

Figure 5.3. Prediction shifts for varying attack size (DWT-based scheme)

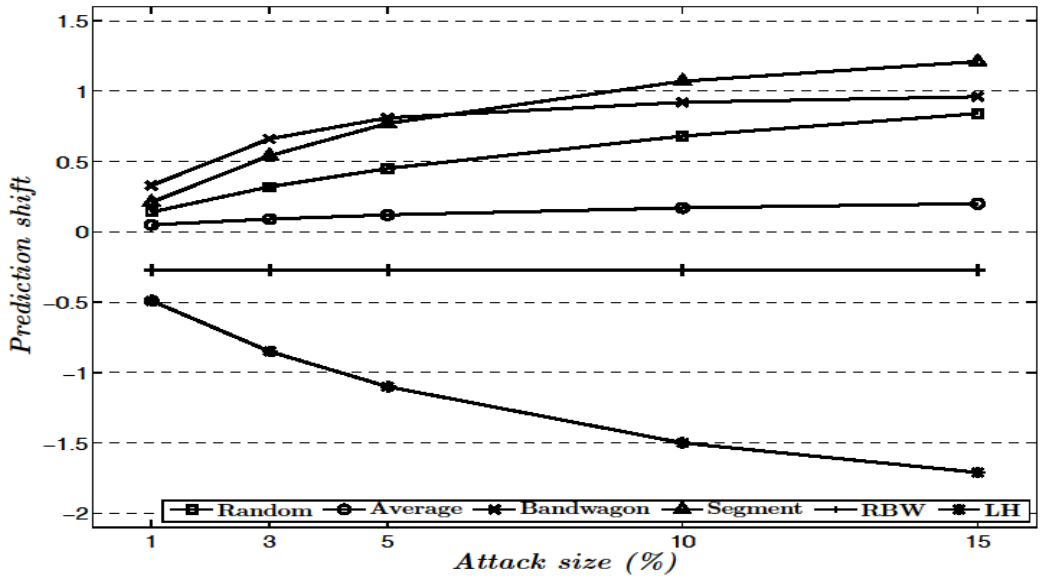

Figure 5.4. Prediction shifts for varying attack size (*k*-means-based scheme)

As shown in Fig. 5.3 and Fig. 5.4, effects of the applied attacks become more significant as the attack size expands excluding for the love/hate attack in the DWT-based scheme and reverse bandwagon attack in *k*-means clustering-based scheme. These two attacks are not effective in corresponding recommendation schemes because of already described reasons in the previous section. Besides these two

attacks, residual ones obtain a decreasingly rising trend for all of the attack models. The most effective push attack is an average attack. Since DWT procedure changes ratings by taking Haar transform of successive two votes, the loss in information becomes minimum and attack profiles can be still effective. It needs to be mentioned that random and segment attacks function very similar against DWT-based PPCF scheme for changing attack size values as indicated in Fig. 5.3. This happens because of the transformation procedure of the DWT-based scheme, which reduces the special interest given to segmented products in segment attack useless and makes it very similar to the random attack model. The highest push and nuke prediction shifts are achieved around 15% attack size. Yet, shifts at 10% are also very close to the highest ones. Therefore, employing a 10% attack size might maximize the benefit achieved from and lowers down cost of the attack. Further, adding less profiles hinders finding the attack. In particular, for *k*-means clustering-based PPCF, which has higher capacity of finding attacks, it might be more beneficial to prepare less profiles, i.e. keep attack size small. As a result, it can be finalized that DWT- and *k*-means clustering-based PPCF schemes can be exposed to profile injection attacks with significant prediction shifts.

Table 5.2. Prediction shift values for varying attack size

Attack Type	*Attack Size (%)*				
	1	3	6	10	15
SVD-based PPCF					
Random	0.0000	0.0000	0.0000	0.0001	0.0003
Average	0.0000	0.0000	0.0000	0.0001	0.0003
Bandwagon	0.0000	0.0000	0.0000	0.0001	0.0001
Segment	0.0000	0.0000	0.0000	0.0001	0.0001
Reverse BW	-0.0001	-0.0003	-0.0006	-0.0013	-0.0016
Love/Hate	-0.0000	-0.0000	-0.0000	-0.0000	-0.0000
Item-based PPCF					
Random	0.0171	0.0172	0.0172	0.0172	0.0172
Average	0.0182	0.0185	0.0188	0.0197	0.0199
Bandwagon	0.0175	0.0178	0.0179	0.0180	0.0181
Segment	0.0639	0.0701	0.0728	0.0749	0.0786
Reverse BW	-0.0160	-0.0164	-0.0167	-0.0169	-0.0174
Love/Hate	-0.0180	-0.0181	-0.0181	-0.0181	-0.0182

Since filler size parameter's utilization characteristics do not affect the SVD- and item-based PPCF schemes, as shown in the previous set of experiments, the

impact of such negligible manipulations are not much affected with varying attack size values for these algorithms, as shown in Table 5.2. Again, SVD-based PPCF scheme is extremely robust resulting in negligibly small prediction shifts for all attack models. Similarly, item-based PPCF scheme performs slightly vulnerable to manipulations; yet, such effects are insignificant, as well.

5.4.1.3. Number of number of clusters

Although experiments for the filler size and the attack size parameters were conducted with a fixed number of clusters in *k*-means clustering-based PPCF scheme, such algorithm's functioning is directly correlated to the number of clusters used. Therefore, finally a set of experiments was conducted to show how different number of clusters affect the strength of *k*-means clustering-based PPCF scheme. In this set of experiments, the filler size and the attack size parameters were fixed at 15% and the number of clusters was varied from two to 10. The experiments were performed 100 times and meaning prediction shift outcomes are shown in Fig. 5.5.

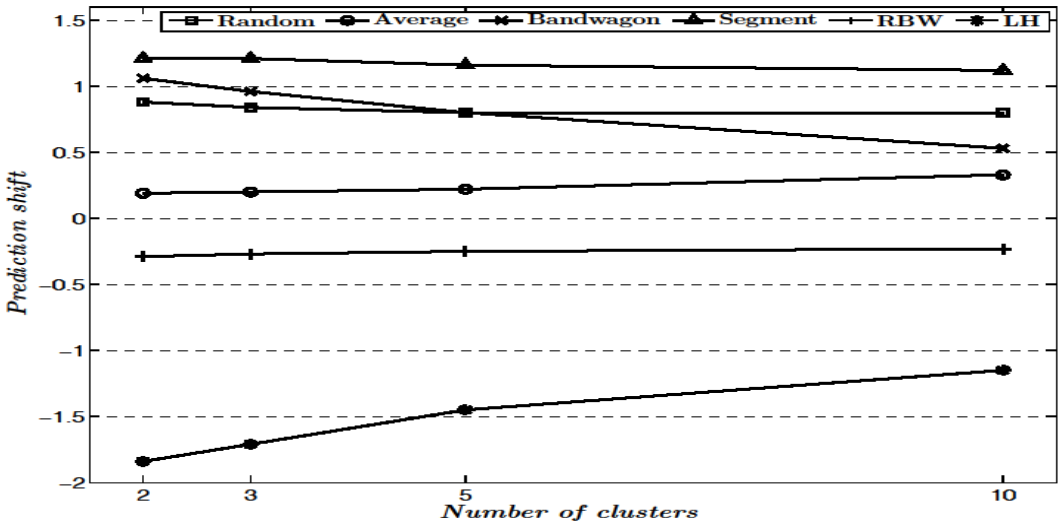

Figure 5.5. Prediction shifts for varying number of clusters

As the number of clusters expands, distinguishing the attack profiles from the genuine ones becomes easier for *k*-means algorithm. Attack profiles usually prone to demonstrate resemblance to each other due to their systematic production. In this way, *k*-means algorithm simply groups attack profiles together. Such an attack

discriminates them from original users. Therefore, as the number of clusters increase, it becomes more unlikely that a genuine profile is injected into a cluster composed of mostly of fake profiles. However, increasing the number of clusters worsens system's overall accuracy, as reported by Bilge and Polat (2013). Hence, selecting *k* for the algorithm around five according to results displayed in Fig. 5.5 might be effective. As seen in Fig. 5.5, changes in prediction shifts with increasing number of clusters are very steady for most of the attacks. With increasing number of clusters, bandwagon and love/hate attacks become less effective.

5.4.2. Overall comparison

In this part, a pairwise comparison between non-private and privacy-preserving model-based schemes is given along with privacy-preserving memory- and model-based ones with respect to strength against shilling attacks.

Because non-private DWT-based recommendation scheme is not examined previously in the literature, a comparison for *k*-means clustering-, SVD-, and item-based schemes is provided. Mobasher et al. (2006a) compare *k*-means clustering- and *k-nn*-based CF algorithms. The authors report that *k*-means functions more strongly. Furthermore, it is more resistant to segment attack than *k*-nn algorithm. Moreover, Sandvig et al. (2008) propose *k*-means clustering-based CF as a strong recommendation scheme with small influences against segment attacks. In our tests, it is also clearly seen that *k*-means clustering-based PPCF algorithm is mostly susceptible to segment attacks with a prediction shift of about 1.5. Yet, it is more resistant to other types of push attacks. Whereas, this algorithm is not examined against nuke attacks. However, it is shown that it is highly vulnerable to love/hate nuke attack model. How varying the number of clusters influences the robustness of *k*-means clustering-based scheme is explored as well. The outcomes indicated that changing number of clusters do not significantly affect the prediction shift values given by the random, average, segment, and reverse bandwagon attacks.

In terms of SVD-based CF scheme, Zhang et al. (2006) report that changes in predicted values do not go beyond 0.003, which proves it to be a very strong algorithm. Furthermore, Mehta and Nejdl (2008) suggest that SVD has greater

stability against random, average, and bandwagon attacks. According to our experimental results, among four model-based PPCF schemes, SVD-based PPCF scheme is also the most resistant one such that at most -0.0016 prediction shift happened for reverse bandwagon attack model, which is extremely negligible in a five-star rating scale. Therefore, we may state that SVD functions in a robust manner in privacy-preserving environment as it also does in non-private schemes.

As a non-private CF scheme, item-based CF algorithm is shown to be very vulnerable to segment attacks rather than random, average, or bandwagon attacks (Burke et al., 2005c; Mobasher et al., 2007b). Burke et al. (2005c) reported that segment attack is seriously more striking in item-based CF due to its profile construction protocol. It is also shown that segment attack functions more effective than broader attacks against item-based CF scheme (Burke et al., 2005c). However, according to the outcomes in privacy-preserving scheme, although a segment attack functions slightly more manipulative than other models, item-based PPCF scheme is still very resistant to all six attack models with a maximum prediction shift value of 0.0805 against disguised segment attack model.

Ultimately, four privacy-preserving model-based prediction schemes with a highly reputable privacy-reserving memory-based recommendation scheme are compared in terms of strength against the six shilling attacks. Recall that privacy-preserving *k*-nn-based CF algorithm is assessed with respect to strength previously. The most significant prediction shift values due to the six shilling attacks on four model- and one memory-based PPCF schemes are presented in Table 5.3.

Table 5.3. Prediction shift for memory- and model-based PPCF algorithms

Algorithm type	*Shilling attacks*					
	Random	*Average*	*Bandwagon*	*Segment*	*Reverse BW*	*Love/Hate*
Memory-based PPCF						
k-nn	1.343	0.545	1.377	1.523	-1.753	-0.168
Model-based PPCF						
DWT	0.600	1.032	0.877	0.601	-0.562	-0.021
k-means	1.230	0.572	1.093	1.467	-0.298	-2.083
SVD	0.000	0.000	0.000	0.000	-0.001	-0.000
Item-based	0.018	0.021	0.018	0.080	-0.017	-0.018

As shown in Table 5.3, model-based schemes are stronger than the memory-based one against the random attack. SVD-, item-, and DWT-based schemes

function successfully against the random attack. Even though *k*-means-based scheme is stronger than the *k*-nn against the random attack, the random attack produces very close prediction shift values for both schemes. SVD- and item-based schemes function better than *k*-nn against the average attack. Yet, the DWT-based method functions worse than *k*-nn while the *k*-means-based scheme almost obtains the same prediction shift as *k*-nn in case of average attack. In terms of the bandwagon and the segment attack models, the same trends are observed. The *k*-nn technique gives the worst performance against such attacks. The algorithms having the best outcomes are SVD- and item-based schemes against these two attacks. DWT-based scheme is stronger than the *k*-nn. However, *k*-means works like *k*-nn against the bandwagon and the segment attacks. As listed in Table 5.3, all four model-based methods are stronger than the memory-based technique against the reverse bandwagon attack. In terms of love/hate, only *k*-means functions worse than the memory-based algorithm. In this way, SVD- and item-based schemes are the strongest algorithms. Usually, model-based approaches are more robust than the memory-based scheme.

5.4.3. Discussion

There are several model-based methods to generate personalized prediction depending on user choices. Like their non-private forms, privacy-preserving model-based CF schemes can also be exposed to manipulations through profile injection attacks. This study gives an increase with respect to previous reports by examining the strength of such model-based PPCF techniques against well-known six shilling attack models and giving comparisons between their non-private forms and memory-based privacy-preserving methods.

According to the empirical outcomes shown above, it might be suggested that the SVD-based PPCF scheme is not fully influenced by malicious profiles and it is the strongest PPCF algorithm. This phenomenon is attributed to noise removal capability of SVD algorithm, which removes extreme influences of the added malicious profiles to manipulate a certain product's recommendation reputation. The item-based PPCF method has an implicit defense mechanism against shilling

attacks arising from its recommendation method of computing similarities among items, not users. Since added attack profiles are planned to affect user-user similarities, manipulation effects become useless against the item-based PPCF scheme like its non-private form. Whereas, the DWT-based PPCF algorithm is resistant against love/hate attack only. This is attributed to the alterations of successive items' ratings. The DWT-based scheme is influenced significantly by all other five attacks. Similarly, the *k*-means clustering-based PPCF technique is affected by all of the attacks excluding reverse bandwagon attack since clustering procedure is directly functioned on pure rating profile similarities. Although *k*-means clustering can be utilized as a finding and removal tool for malicious profiles, it is important to decide a proper number of clusters. Also, it is more successful with attacks having discriminating properties such as high filler size. Compared to memory-based *k*-nn algorithm, all of the model-based schemes are more resistant to shilling attacks, which indicates that intermediate procedures on prediction approximation process decreases the effects of such attacks.

5.5. Conclusions

Many online shopping sites broadly utilize CF algorithms. Both customers and online vendors obtain benefits from recommendation schemes. In addition to their advantages, CF techniques have their own challenges. Two most important challenges of such schemes are privacy protection and being subject to shilling attacks. Besides these, scalability is another problem that many prediction methods face with.

Because of their online efficiency, model-based recommendation techniques are preferred with respect to memory-based ones. There are privacy-preserving model-based prediction schemes, which give recommendations efficiently without disrupting customer privacy. In this thesis, first four push and two nuke shilling attack models were applied onto four broadly utilized privacy-preserving model-based techniques; namely *k*-means clustering-, SVD-, DWT-, and item-based schemes. Because disguised data are utilized by such techniques, revised versions of random, average, segment, bandwagon, reverse bandwagon, and love/hate

shilling attacks were employed. Several sets of real data-based experiments were performed to assess the strength of the prediction schemes against the six attacks.

Our experimental outcomes teach the lessons that privacy-preserving SVD-based scheme is the strongest recommendation algorithm. Privacy-preserving item-based CF algorithm comes next. Even though the DWT- and the *k*-means clustering-based schemes are not as strong as the SVD-based scheme, they are still more robust than the privacy-preserving *k*-nn scheme. This is an example of the memory-based prediction techniques. Revised average attack seemed to be the most effective attack model among all six techniques as it obtains more prediction shift against the DWT-based scheme compared to the *k*-nn algorithm.

6. ROBUSTNESS ANALYSIS OF HYBRID PIVACY-PRESERVING COLLABORATIVE FILTERING SCHEME

In this chapter, it is analyzed a privacy-preserving hybrid prediction scheme with respect to robustness. Four push and two nuke shilling attacks are applied to the algorithm to show how robust it is against them. Different sets of experiments are conducted using real data to show how varying control parameters affect the robustness. The hybrid scheme is compared with memory- and model-based schemes in terms of robustness. The analysis shows that although the scheme can be marginally considered as a robust algorithm, it is less robust than memory- or model-based prediction algorithms with privacy.

6.1. Introduction

PPCF schemes are categorized into three different schemes: memory-based, model-based, and hybrid methods. Hybrid methods can be considered as combinations of memory- and model-based methods. Memory-based techniques with privacy are the simplest heuristic methods (Polat and Du, 2005a). Using such methods for producing referrals is straightforward. It is easy to add a new user or product into the collection. The mechanism scales well with commonly rated items by any two users. However, the size of data can be a disadvantage for scaling those systems. Privacy-preserving model-based prediction algorithms generate a model relying on user ratings as well as providing predictions (Polat and Du, 2005a; Bilge and Polat, 2012; Bilge and Polat, 2013). They scale better in a sparse environment. They find item or user similarities off-line via the model. When a new item or user is added, a new and a fresh model should be established. However, this process is computationally expensive. Also, as useful data can be lost during a specific model production, accuracy may be reduced. Hybrid prediction scheme with privacy features a more effectively performance by utilizing advantages of memory- and model-based models (Renckes et al., 2012).

In addition to analyzing memory- and model-based PPCF algorithms with respect to robustness, hybrid PPCF scheme should also be analyzed in terms of robustness. Hence, a hybrid PPCF scheme is analyzed against six well-known

shilling attack models and its robustness is examined. The algorithm is also compared with other PPCF schemes with respect to robustness.

6.2. Hybrid Collaborative Filtering with Privacy

Renckes et al. (2012) propose a novel hybrid PPCF scheme. The hybrid scheme's structure is similar to that of a tree, where each node represents a user and each link depicts similarity between two corresponding users. The root of the tree indicates the initial neighbor of a target user. The authors form trees for representing the users and the similarities between them. A tree is constructed off-line after collecting users' preferences about various items, for each user *u*. The root node represents the user *u*. The server first constructs trees for each user *u* as follows:

1. Similarity weights between user *u* and each other user are computed. The user *u* is inserted into the root node. The ratings are already known and no effort is spent for finding them.
2. The most similar *s* users to user *u* are discovered and removed from the database. These *s* users represent the children (adjacent) of the user *u* and they are housed at the first level.
3. For each of the *s* users, the best similar *s* users to them among the remaining ones are found. Such users are then placed into the second level. Correspondingly, these users are the neighbors of each of the *s* users remaining at the first level.
4. The most related *s* users to each user among the remaining ones are determined until there is no one left in the records. The structure constructed for each user *u* is similar to a tree, where each node's children represent the most similar users to that user. Note that $n = 1 + s + s^2 + s^3 + \ldots + s^y$, where *y* is the number of levels and *n* is the number of users. The value of *y* is subjected to *n* and *s*.

For each tree, the following storage is done: initial user, her neighbors, her neighbors' neighbors, and so on. Further, similarity weights between neighbors and their preferences about a variety of items are stored. Similarities are saved for each

link between users. Each user is linked to the best similar users to her. They represent her neighbors.

When an active user *a* asks a prediction, the first step is to decide an initial user. There are two possible ways for selecting the initial user. In the first way, the similarities between *a* and each user in the database are estimated online. The best similar user to *a* is determined as initial user. In the second way, after collecting *n* users' data, they can be gathered in several clusters by utilizing different clustering algorithms. Since *k*-means algorithm is widely employed for clustering users for CF purposes (Marlin, 2004; Xue et al., 2005), it is used for clustering *n* users into *k* clusters off-line. When *a* asks referrals, distances between *a* and each cluster center is computed to determine her cluster online. Then, she is inserted into the closest cluster. Similarities between *a* and each user in that cluster are found and the best similar user to *a* is selected as initial user. The procedure for generating referrals online for *a* can be explained as follows:

1. *a* sends her ratings and a query to the server. The query consists of the target item *q* or items for which referrals are sought. The system first places *a* into a cluster. The initial user is chosen for *a* among the users in that cluster. The data in the tree generated for the initial user are used for finding appointments.
2. Since the tree contains *n* users' data, the optimum value of the number of users whose data to be used for PPCF should be decided. For improving the overall performance, the best-*N* neighbors can be chosen for providing recommendations and the optimum value of *N* can be calculated experimentally.
3. Finally, the system considers those *N* users' data to find referrals. The system can calculate guessing for *a* on item *q* (p_{aq}) as follows (Herlocker et al., 1999). This is one of the best memory-based CF algorithms, where z_{uq} is the *z*-score of user *u* for item *q* and *N* is the number of users involved in recommendation computation:

$$p_{aq} = \overline{v_a} + \sigma_a \times \frac{\sum_{u=1}^{N} w_{au} \times z_{uq}}{\sum_{u=1}^{N} w_{au}} \tag{6.1}$$

in which $\overline{v_u}$ and σ_u represent a's mean rating and standard deviation of her ratings, respectively and w_{au} is the similarity between a and her neighbor u. z-scores and w_{au} values based on z-scores can be computed as follows:

$$w_{au} = \sum_{j=1}^{N} Z_{aj} \times Z_{uj} \tag{6.2}$$

in which j shows co-rated items between users a and u.

6.3. Robustness of Hybrid Collaborative Filtering with Privacy

Since the hybrid scheme is popular compared to other methods, its robustness against shilling attacks should be scrutinized. Therefore, the goal is to analyze the hybrid PPCF method with respect to robustness. The most popular and successful four push attack models along with two nuke attack models are considered.

There are couple of control parameters that might affect the overall performance of shilling attacks. These are called filler size and attack size parameters. Therefore, the six attacks models and their effects on the hybrid PPCF scheme can be evaluated with varying values of filler and attack size parameters. Real data-based experiments are performed to show how varying values of filler size and attack size affect the robustness of the hybrid method. In addition to these two parameters, there also other parameters whose values might affect the overall performance of such attacks. Examples of such parameters are σ_{max}, β_{max}, and N. Their values might affect the robustness of the hybrid PPCF scheme. Hence, similar sets of experiments are conducted using real data while varying the values of such parameters. Finally, since there are memory-based, model-based, and hybrid PPCF schemes, it is vital to compare them with respect to their robustness against six popular shilling attacks. Thus, a comparative study is conducted to relate these three types of schemes in terms of robustness under the same attacks with the same settings. Real data-based empirical outcomes show that the most successful algorithms against shilling attacks are model-based PPCF schemes. Memory-based and hybrid PPCF schemes are quite vulnerable against shilling attacks.

6.4. Experimental Evaluation

To show the effects of the six shilling attack models on hybrid PPCF algorithm, real data-based experiments were performed. Effects of shilling attacks were evaluated as a function of filler size and attack size. Privacy-preserving parameters are kept constant, where β_{max} = 25% and σ_{max} = 2. Note that such values are sufficient for providing a decent level of individual privacy (Bilge and Polat, 2012). The perturbed data was divided into training and testing sets. 150 users were selected for testing and the rest of the users were assigned to the training set. Two distinct target item sets defined in Table 2.3 were formed, each consisting of 50 movies for push and nuke attacks.

6.4.1. Effects of filler size parameter

Experiments were performed for demonstrating the effects of the masked push and nuke attacks with changing filler size values on hybrid CF algorithm. Remember that filler size is directly related to the success of a performed attack because filler items comprise the base for leaking into neighborhoods of genuine users in the recommendation process. Since β_{max} was set to 25% at first, during the experiments, filler size was varied from 3% to 25%. Further, the attack size was kept constant at 15%, being the maximum value of attack size value tested. Experiments were repeated 100 times due to randomization in the perturbation process and average results are presented. Prediction shift values for push and nuke attacks are shown in Fig. 6.1 and Fig. 6.2, respectively.

As seen from Fig. 6.1 and Fig 6.2, prediction shift values show that the hybrid PPCF algorithm is not that robust against shilling attacks. In Fig. 6.1, the most successful attack seems to be bandwagon attack. Along all of the values of filler size parameter, prediction shift value for bandwagon attack does not show much variation and is realized in the vicinity of 1.58. It shows that when the filler size value increases, for the related item there is no change in the nearest neighbors of users. That is, for the values of filler size 25% compared with 5%, the first *n* nearest neighbors that will affect the prediction value were found not to change much.

There is no much change depending on the filler size value for attacks other than average attack. Only when the filler size value is 25%, there is a marked decline in the prediction value. The other successful attack is segment attack. The reason for this phenomenon is that the bandwagon and the segment attacks are specifically designed attacks. These push attacks are advanced attacks and they are similar to each other by the way they are created.

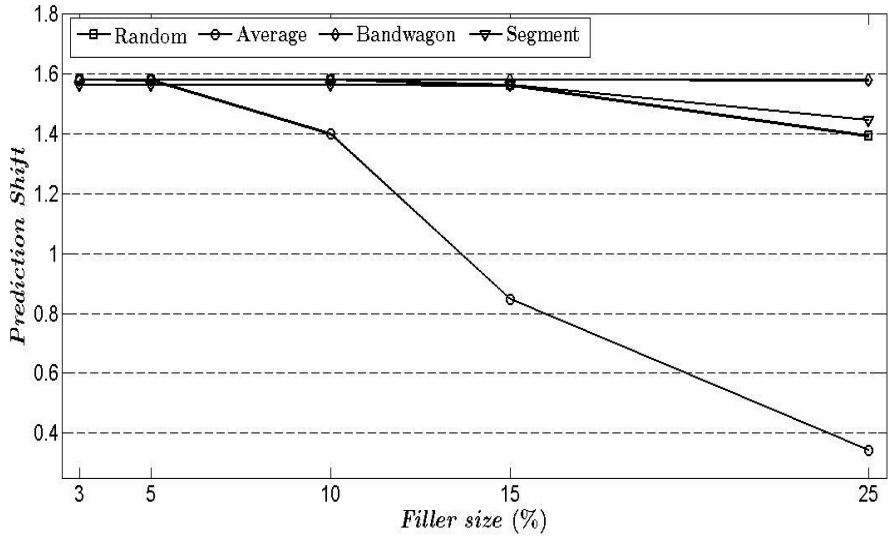

Figure 6.1. Prediction shifts for varying filler size (push attacks)

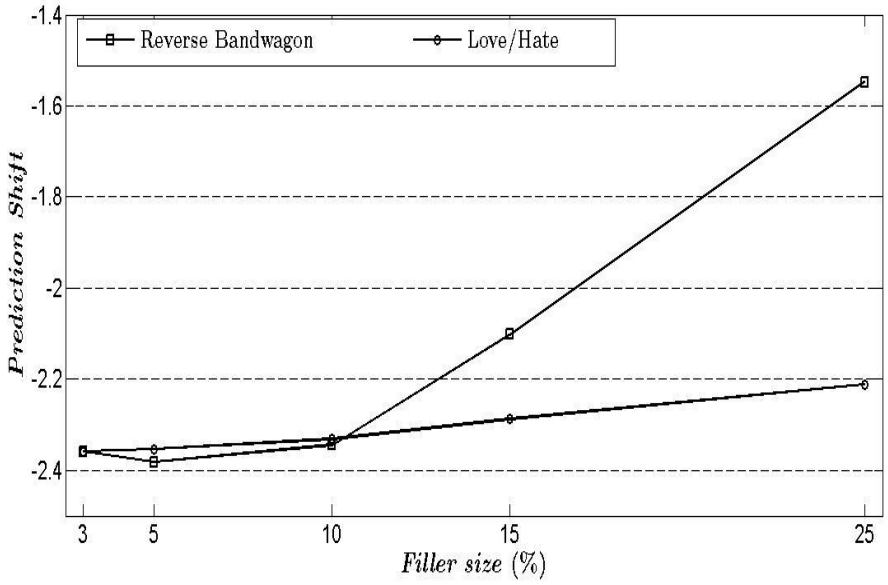

Figure 6.2. Prediction shifts for varying filler size (nuke attacks)

In Fig. 6.2, reverse bandwagon attack, which is a nuke attack, is also quite successful. In this attack, according to different filler size values, prediction values were obtained between -1.55 and -2.35. For filler size value between 3% and 15%, there has not been much of a change in the prediction shift values. The prediction shift value slightly decreases for 25% filler size. Love/hate nuke attack is quite successful as reverse bandwagon attack. According to the change of the filler size value, prediction shift values do not significantly change.

6.4.2. Effects of attack size parameter

Another set of experiments were performed for demonstrating the effects of the attacks with changing attack size values on the hybrid PPCF algorithm. While filler size parameter handles utility perspective of an attack, attack size focuses on impact of such utility by determining the number of bogus profiles. Although it is obvious that the more attack profiles inserted into the system, the larger the obtained shifts are; however, it constitutes a trade-off between the detectability and the impact of the applied attack model. Therefore, in order to define varying effects of the attack size parameter, it was varied from 1% to 15% while filler size was kept constant at 15%. The experiments were repeated 100 times due to randomization. Average prediction shifts values for push and nuke attacks were presented in Fig. 6.3 and Fig. 6.4, respectively.

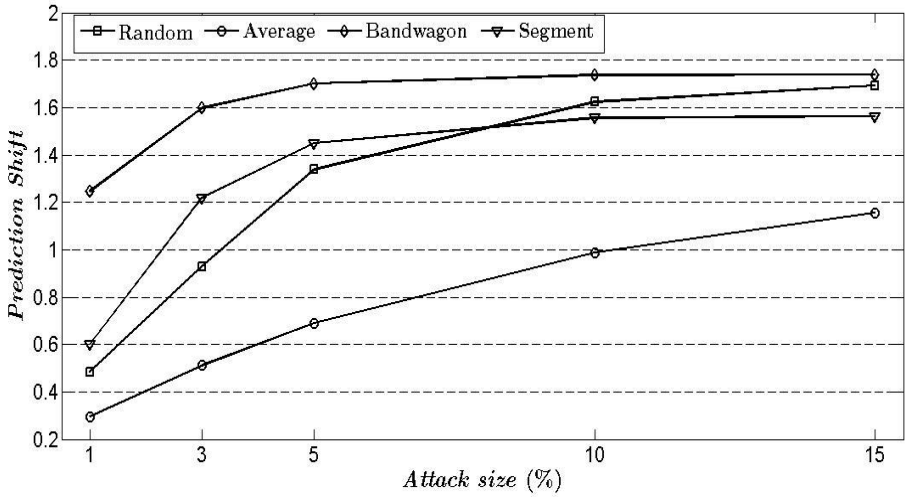

Figure 6.3. Prediction shifts for varying attack size (push attacks)

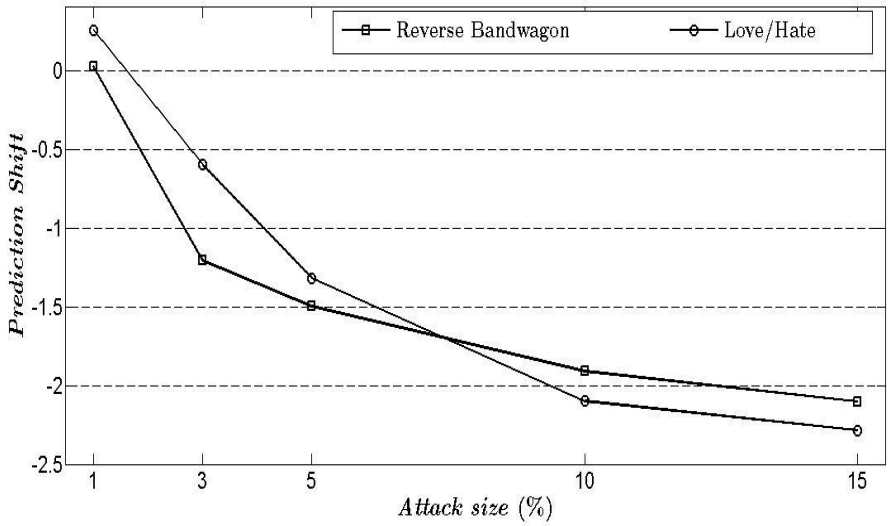

Figure 6.4. Prediction shifts for varying attack size (nuke attacks)

As shown in Fig. 6.3, the most successful push attack models are bandwagon, segment, and random attacks. With increasing attack size, the success of attacks improves. Depending on the increase in attack size, the number of profiles added to the system also increases. Along with this increase, the probability of the users of attack profiles being nearest neighbors also increase. As a result, rise of attack size is more likely to affect the users' prediction as in the previous experiment. Similarly, reverse bandwagon attack and love/hate attack models are quite successful. As shown in the Fig. 4, for these two nuke attack models, with increased attack size value, prediction shift values also increase.

6.4.3. Effects of β_{max} parameter

To show how changing β_{max} values affect the overall performance, another set of experiments were performed. As described before, during data disguise β_{max} value determines the rate of unrated item to be filled with random numbers. Each user u randomly selects β_u; and β_u percent of their unrated items to be filled with random numbers. At first, σ_{max} was set to 2 and during the experiments β_{max} parameter was varied from 5% to 25%. Furthermore, attack size and filler size were kept constant at 15%. The most successful attack models, two push (average and

bandwagon) and one nuke (reverse bandwagon-RBW) in the previous experiments were used in this and subsequent experiments. The average prediction shift values, obtained by the changing β_{max} value, were shown in Fig. 6.5. As seen from the figure, the values obtained by average and reverse bandwagon attacks are very close to one another. Average attack is a bit more successful. The most successful result obtained for push attacks based on changing values of β_{max} is 0.99 in the average attack. Prediction shift value has not significantly changed by varying the β_{max}. The reason for this finding is that more unrated cells are filled with increasing β_{max}; and fake profiles become inefficient due to smaller number of fake ratings compared to the filled ones. Prediction shift value increased to a limited extent parallel with the increasing β_{max} value in reverse bandwagon attack. The highest prediction shift value is obtained as -2.1 for this attack.

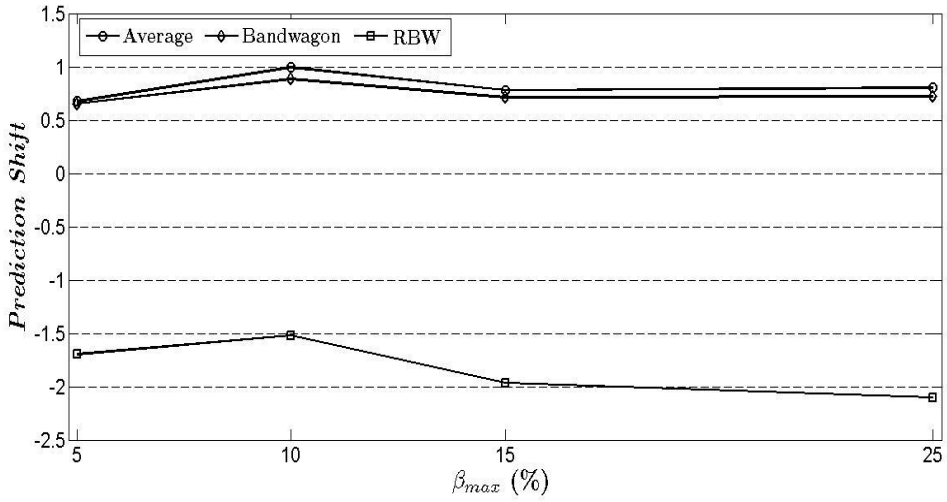

Figure 6.5. Prediction shifts for varying β_{max} parameter

6.4.4. Effects of σ_{max} parameter

In PPCF schemes, each user selects a standard deviation value σ_u from the range (0, σ_{max}] during data disguise. To examine the effects of σ_{max} value, its values were assigned from 0.5 to 3. While β_{max} was fixed at 25%, filler size and attack size values were fixed at 15%. The outcomes were displayed in Fig. 6.6.

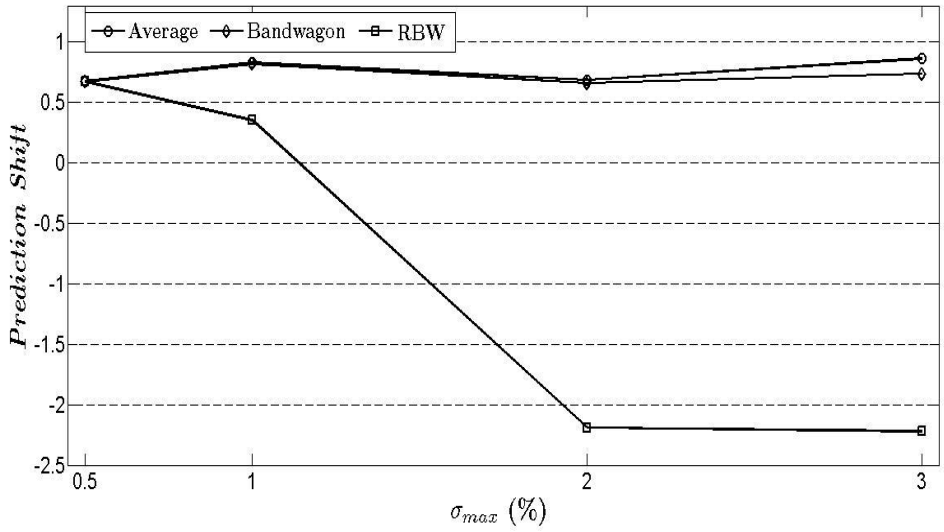

Figure 6.6. Prediction shifts for varying σ_{max} parameter

As seen from Fig. 6.6, changing σ_{max} parameter in average and bandwagon attacks does not affect prediction shift values, which are between 0.6 and 0.8 according to the changing values of the parameter σ_{max}. In reverse bandwagon attack, there is a significant increase in the prediction shift value with increasing σ_{max} parameter. The prediction shift value reached -2.25 for reverse bandwagon attack. The reason for this phenomenon is that the target item is assigned to the minimum random number in this attack; and random numbers become smaller with increasing σ_{max} values. Using smaller noise data for nuking predictions causes significant manipulations.

6.4.5. Effects of number of neighbors parameter

Number of neighbors (N) determines how many of the most similar neighbors will be included when calculating prediction in the PPCF algorithm. At first, σ_{max} and β_{max} were set to 2 and 25%, respectively. Furthermore, attack size and filler size were kept constant at 15%. During the experiments, N value was varied from 10 to 100. The most successful three attacks, average, bandwagon, and reverse bandwagon, were applied in this experiment. Fig. 6.7 shows prediction shift values for these three attack models.

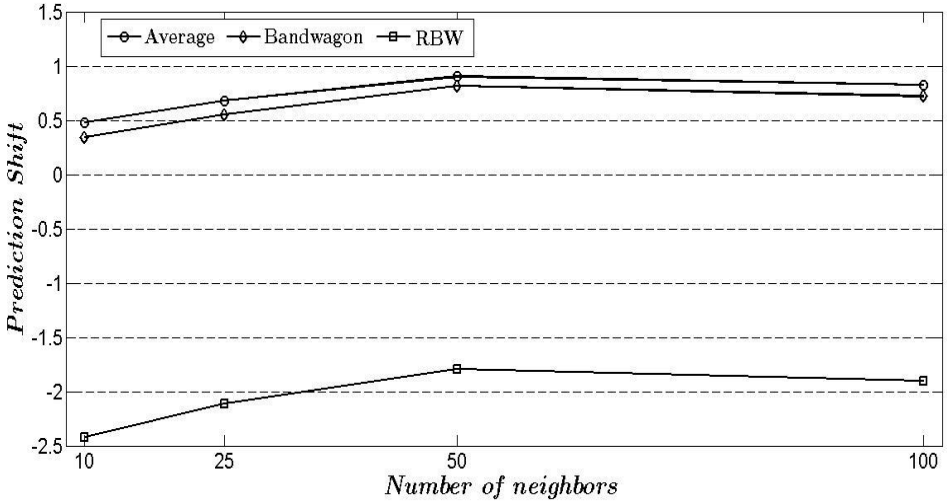

Figure 6.7. Prediction shifts for varying number of neighbors

As seen from the figure above, prediction shift values obtained via average and bandwagon push attacks increase until the value of *N* is 50 and later show a little change. Since more attack profiles will be included in the calculation according to the increase in the most similar number of neighbors, prediction shift values will increase. Therefore, as shown in Fig. 6.7, some increase in the value of *N* improves the success of average and bandwagon attack models. However, the increase after a certain value of *N* will reduce the average value because the number of users less similar will also be taken into account. The best value of *N* is considered as 50 for average and bandwagon attacks. It is considered as 10 for reverse bandwagon attack.

6.5. Comparison

The privacy-preserving hybrid prediction scheme is compared with well-known privacy-reserving memory-based and model-based recommendation schemes in terms of robustness. Recall that privacy-preserving two memory- and four model-based prediction algorithms are evaluated with respect to robustness. The comparison of the hybrid scheme with the other PPCF schemes in terms of robustness against shilling attacks is given in Table 6.1.

Table 6.1. Comparison of memory-based, model-based, and hybrid PPCF methods

Algorithm Type	*Shilling Attacks*					
	Random	*Average*	*Bandwagon*	*Segment*	*RBW*	*Love/Hate*
Memory-based PPCF						
k-nn	1.343	0.545	1.377	1.523	-1.753	-0.168
Model-based PPCF						
DWT	0.600	1.032	0.877	0.601	-0.562	-0.021
k-means	1.230	0.572	1.093	1.467	-0.298	-2.083
SVD	0.000	0.000	0.000	0.000	-0.001	-0.000
Item-based	0.018	0.021	0.018	0.080	-0.017	-0.018
Hybrid PPCF						
Hybrid	1.592	0.848	1.582	1.563	-2.102	-2.287

As seen from Table 6.1, model-based PPCF algorithms are observed more robust than memory-based and hybrid PPCF algorithms. The most robust algorithms, in general, are model-based ones against the well-known shilling attacks. The memory-based scheme is somewhat robust against such attacks. However, the hybrid method shows the worst performance in terms of robustness. Nuke attacks achieve significant success rates against the hybrid algorithm. All push attacks except average attack are also successful when they are applied to the hybrid scheme. According to the results displayed in Table 6.1, the most successful algorithm is SVD-based method. Notice that SVD is usually used to remove noise data. Thus, it is able to eliminate the effects of the fake profiles in a user-item matrix. It then makes it as a robust algorithm. As discussed before, recommendation algorithms should provide accurate predictions efficiently with privacy. They also need to be robust against shilling attacks. Therefore, users need to choose the most appropriate prediction schemes. If the only criterion is robustness, then the hybrid scheme is not a good choice.

6.6. Conclusions

After analyzing the robustness of memory- and model-based prediction schemes with privacy, privacy-preserving hybrid prediction method is examined with respect to robustness. Like memory- and model-based schemes, the hybrid scheme might be vulnerable against shilling attacks. In this thesis, the hybrid scheme with privacy exposed to shilling attacks is examined. Four push (random,

average, bandwagon, and segment) and two nuke (reverse bandwagon and love/hate) attacks are applied. Empirical results show that the hybrid scheme is vulnerable to shilling attacks. Especially bandwagon and reverse bandwagon attacks are efficient attacks for manipulating referrals. Also, some experiments are conducted to show the effects of control parameters. The outcomes show that varying values of control parameters affect prediction shift values.

7. DETECTING SHILLING ATTACKS IN PRIVATE ENVIRONMENTS

In this chapter, how to detect shilling attacks in PPCF systems is scrutinized. Four existing attack detection methods are modified in such a way to detect shilling profiles in PPCF systems. The ability of such modified methods is investigated in terms of detecting shilling profiles generated by six well-known shilling attacks on perturbed data. Also, a novel detection method, based on hierarchical clustering, is proposed. Real data-based experiments are performed. Empirical outcomes demonstrate that some of the detection methods are very successful on filtering out fake profiles in PPCF schemes. The novel scheme is also successful in detecting attacks in private environments.

7.1. Introduction

There is an increase in studies on shilling attacks in recent years. The detection of shilling attacks is essential for correct predictions by the recommender system. Chirita et al. (2005) performed the first work on the detection of the shilling profiles by checking the profile properties. They considered the simplest attacking models of the random and average methods. Their method is successful in the case of dense attack profiles, but unsuccessful with profiles having high sparsity. Burke et al. (2006a) studied different characteristics derived from user profiles for their effectiveness in attack detection. Their study shows that a machine learning sorting method including attributes derived from attack models is more successful than more widespread detection algorithms. The algorithm proposed by Burke et al. (2006a) establishes a model training with known number of real and attack profiles. The authors empirically showed that their algorithm was more successful than the algorithm proposed by Chirita et al. (2005).

In detecting the attack profiles in CF algorithm, variable selection based on PCA can be utilized (Mehta, 2007; Mehta et al., 2007a; Mehta and Nejdl, 2009). PCA method is about calculating either the correlation or the co-variation values of all users between each other. The data matrix is then listed with respect to that calculation. The correlation values of the attack profiles between each other are

high while those of the co-variation are low. The approach can only be applied to a dense user-item matrix because PCA cannot tolerate null values, which have to be replaced by estimated values.

Since the attack profiles are formed by a defined method, they are similar to each other. Bhaumik et al. (2011) reported that detection attributes values will be also closer to each other due to similarities between the attack profiles. Using these attributes, the profiles were separated into cluster by *k*-means algorithm. The authors mentioned that the attack profiles will accumulate to the same cluster and the number of profiles in the cluster of attack profiles will be less than the number of the profiles in the other clusters. While generating recommendations by a CF algorithm, the clusters with less number of profiles will not be taken into account and thus, the effect of the attack profiles to the system will be avoided. Mehta (2007) and Mehta and Nejdl (2009) attempted detecting the attack profiles using the PLSA-based clustering algorithm. The users were assigned to the clusters, where the probability of belonging was high. They mentioned that the attack profiles were distributed to the same clusters due to the similarities among each other. In order to figure out the cluster, where the attack profiles are located, the distance of the users in each cluster with respect to the center is measured (Mehta, 2007; Mehta and Nejdl, 2009).

Tang and Tang (2011) analyzed the time gaps between voting times in order to determine suspicious attitudes for affecting the top-*N* lists in the recommendation systems. Zhang (2011) focused on protecting recommendation systems based on trust from attacks. For this purpose, data genealogical tree method, following the recommendation history and placing sacrifice knots, is employed. Noh et al. (2014) proposed a novel robust recommendation algorithm called RobuNec, which provide admission control as a defense mechanism against shilling attacks. Due to the power of access control, the method provides highly trusted recommendation results. Cao et al. (2013) intended to utilize semi-supervised learning to identify attack profiles and describe how to apply semi-supervised learning to shilling attack detection in detail. Zhang et al. (2013) proposed two methods for building robust recommender system to prevent shilling attacks. These methods are CluTr and WCluTr, which combine trust information with clustering. CluTr filters out the suspicious fake

users in the formed clusters and WCluTr uses trust information to fortify the similarities among genuine users. Morid et al. (2013) proposed new attack detection method, which detects influential users, instead of the whole user set, to improve their attack detection performance. They define influential user as if an attacker succeeds, her profile is used over and over again by CF system, making her an influential user.

Zhang and Zhou (2014) built rating series for each user profile based on originality and reputation of the products. Then they employed the experimental mode decomposition method to decompose each rating series and extract Hilbert spectrum-based characteristics to describe shilling attacks. They exploited support vector machines to find shilling attacks based on the suggested characteristics. Bilge et al. (2014) recommended an original shilling attack finding technique for particular attacks based on bisecting *k*-means clustering method. Their approach is based on the fact that attack profiles are collected in a leaf node of a binary decision tree. Their experimental results indicate that the technique is exclusively successful on discovering exact attack profiles such as bandwagon, segment, and average attack. Li (2014) proposed a method, which discloses latent factors appealing missing ratings under the non-arbitrary-missing mechanism and further unites these hidden issues with Dirichlet process in the framework of probabilistic generative model. Zhuo and Kulkarni (2014) presented a technique to make recommender systems resistant to shilling attacks, where the attack profiles are highly related with each other. They expressed the issue as detecting a submatrix with the highest value in the similarity matrix. The maximum submatrix is explored by converting the issue into a graph and combining nodes by heuristic functions. Chung et al. (2013) suggested Beta-protection approach to solve the drawbacks of current detection techniques. The method grounds its theoretical basis on Beta scattering for facile calculation and has stable functioning when testing with data obtained from the public websites of MovieLens.

In the previous chapters, it is shown that PPCF algorithms can be affected by shilling attacks. In other words, such attacks might significantly affect the accuracy of the estimated predictions in PPCF schemes. Therefore, it is very important to detect these types of attacks and reduce their effects for recommendation systems

to function correctly. Different detection methods developed and applied to CF algorithms for determining fake profiles (Chirita et al., 2005; Burke et al., 2006a; Bhaumik et al., 2006; Mehta et al., 2007a; Li and Luo 2011; Zhang and Zhou 2014). However, no related work with detecting shilling profiles in PPCF algorithms has been carried so far. Therefore, in this thesis, the most commonly used detection methods are applied to PPCF schemes. For this purpose, the current detection methods are modified in such a way so that they are applicable to PPCF methods and experiments are carried out with real data. In practice, six attacking models developed previously for attacking PPCF algorithms are employed. Four of the most common detection schemes are utilized as detection technique. In addition to the existing methods, a novel detection scheme is developed and its success is investigated.

7.2. Existing Detection Methods-based Shilling Attack Detection

Chirita algorithm, *kNN* classifier, *k*-means clustering, and PCA-based variable selection methods are briefly explained as shilling attacks detection methods. Chirita et al. (2005) tried to classify profiles using generic attributes, which consist of some basic statistical formulas. Later, Burke et al. (2006a) used model-specific attributes additionally mentioning generic attributes will not be sufficient by themselves in classifying profiles.

There are basically eight generic attributes used to determine fake profiles as follows:

1. *Number of prediction-differences* (TFS): A prediction is determined for each user. TFS describes the net difference after erasing the user from the system.
2. *Standard deviation in user's ratings*: This metric shows the selecting degree above the average of a user.
3. *Degree of agreement with other users*: This metric exhibits the difference degree of a user's each selection from the average selecting degree of an item.

4. *Degree of similarity with top neighbors*: The weight of the similarities between a user and the closest *k* number of users of her.
5. *Rating deviation from mean agreement* (RDMA): This metric determines the deviation from the pre-given average values of some of the certain items.
6. *Weighted deviation from mean agreement* (WDMA): RDMA is weighted by the square of the number of votes for the item.
7. *Weighted degree of agreement* (WDA): The difference of this metric from the RDMA metric is not applying the division operation with the total number of the votes given by the user.
8. *Length variance* (lengthVar): The metric measures to what extent the length of the investigated profile (the number of items voted for) differs from the average profile length.

Model specific attributes can be briefly described as follows:

1. *Filler mean variance* (FMV): FMV calculates the variation between the average value of the item and the value of each item (filler items I_F), assumed to exist in the item set of each profile.
2. *Filler mean difference* (FMD): The major difference of FMD from model-based metric is to use the absolute value of the difference between the vote of the user and the average of the votes instead of the square of that difference value.
3. *Filler average correlation*: This metric calculates the correlation between each item value and the average item value found in the filler item set of the investigated profile.
4. *Filler mean target difference* (FMTD): FMTD calculates the difference between the average of the assumed filler item set and the average of the possible target item set.
5. *Profile variance*: This metric calculates the profile variance as this tends to be low compared to authentic users.

7.2.1. Existing shilling attack detection methods

Chirita et al. (2005) propose an algorithm (referred to as *Chirita algorithm*) based on RDMA for detecting and isolating shilling attackers. This is the first algorithm effectively detecting the most general attacks on recommender systems. The proposed algorithm is a two-step algorithm as follows:

1. The algorithm computes the average similarity with the top neighbors for all users using PCC. It then selects those users only who have an average similarity smaller than 0.5 of the maximum average similarity in the system for computing RDMA.
2. It associates with each value of RDMA a function that evaluates the probability (PA_u) that the respective user is a shilling attacker. The first s profiles, sorted based on PA_u, are considered attack profiles.

Mobasher et al. (2006b; 2007b) propose a method based on classification (known as *kNN* classifier), which utilizes a total of 15 detection attributes: six generic (WDMA, RDMA, WDA, LengthVar, DegSim with $k = 450$ and DegSim′ with $k = 2$, $d = 963$, where k is the number of neighbors and d is co-rate factor); six average attack model (FMW, FMD and profile variance; computed for both push and nuke); two bandwagon attack model (FMTD; computed for both push and nuke); one target detection model attribute (TMF). Class labels and detection attributes are generated for entire data set, which is divided into two equal-sized sub-sections of train and test data sets. A *kNN* classifier with $k = 9$ is used. The *kNN* classifiers are implemented using Weka. For each test, the second half of the data is injected with attack profiles and then run through the classifier built on the augmented first half of the data.

Clustering is a widely used technique for determining shilling profiles. In *k*-means clustering algorithm, primary objects for cluster centers are randomly selected. Each object is then appointed to the closest cluster based on similarity measure. Cluster centers are re-calculated in each iteration. This algorithm is considered completed when there is no change left in the cluster members. Attack profiles are so similar to each other because known algorithms generate them. As a result, when *k*-means algorithm is employed on the data set into which attack

profiles are added, it is expected that most of the attack profiles would be distributed to the same cluster. The most important issue at this stage is to find in which cluster the attack profiles will be gathered. Mehta and Nejdl (2009) aim to find the tightest cluster (where the elements are so similar to each other) in their study on clustered-based detection. For this reason, for each cluster, the distance of the profiles to center is calculated. The one with the shortest distance to the center is defined as the attack cluster. The determined cluster is isolated.

In a recommendation system, if users are considered as variables, there will be data with a similar number of dimensions. Thus, dimensionality reduction would discard these dimensions due to low covariance of them. Low covariance is observed between not only shilling users, but also shilling users and normal users. PCA computes principal components, which are oriented more towards real users showing the maximum variance of the data. For this reason, those users who show the least covariance with all the other users should be selected. This quantity is used to select some variables from the original data applying PCA, known as variable selection using PCA. In the algorithm below (***Algorithm 1***), Mehta and Nejdl (2009) depict the outline of their approach for variable selection. The first *s* users are selected as the attack profiles and they are isolated form the system, where *s* is considered as the number of the attack profiles added to the system. The algorithm is known as PCA-based variable selection detection algorithm.

Algorithm 1 PCA Select Users (D: user-item matrix & s: cut-off parameter)

1: $D \leftarrow$ *z-scores (D)*

2: $COV \leftarrow D^T D$ {Covariance of D^T}

4: $U\lambda U^T$ *= Eigenvalue Decomposition (COV)*

5: $PCA_1 \leftarrow U(:, 1)$ *{First Eigenvector of COV}*

6: $PCA_2 \leftarrow U(:, 2)$ *{Second Eigenvector of COV}*

7: $PCA_3 \leftarrow U(:, 3)$ *{Third Eigenvector of COV}*

8: ***for*** *all column id users in D* ***do***

9: Distance (user) $\leftarrow PCA_1(user)^2 + PCA_2(user)^2 + PCA_3(user)^2$

10: ***end for***

11: Sort Distance

Output*: Return s users with smallest Distance*

7.2.2. Shilling detection methods for PPCF schemes

All of the detection algorithms described in the previous section are performed on CF attacks. However, such methods can be used on PPCF attacks by adapting them in such a way so that they are able to determine shilling attacks on masked data. There are two confidential data in PPCF schemes: actual rating values and rated and/or unrated items. In order to protect such private data, random numbers are generated using either uniform or Gaussian distribution with zero mean and σ, which is uniformly randomly selected from (0, σ_{max}]. Such noise data are added to actual votes. Also, some of the uniformly randomly selected unrated items cells are filled with noise data. To select unrated cells, a β value is uniformly randomly selected from (0, β_{max}]. Then, β percent of empty cells are filled with random numbers. Values of privacy parameters σ_{max} and β_{max} can be determined according to privacy and accuracy levels required by CF users (Bilge et al. 2014).

Chirita algorithm computes similarities using PCC. Such similarities can be estimated with decent accuracy based on perturbed data, as well (Bilge and Polat, 2012; Bilge et al., 2014). Similarly, RDMA can be computed from masked data. Finally, the probability of being an attack profile or not can be calculated using RDMA on masked data. Therefore, Chirita algorithm can be employed to determine shilling profiles in PPCF schemes. Generic attribute values used in Chirita algorithm are calculated for disguised data. Within the study by Chirita et al. (2005), α = 10 value is defined in the formula, which calculates the possibility of profiles to be an attack profile. When Chirita algorithm is used on PPCF schemes, it is seen that the best result is gathered for the value of α = 1; and this value is used in experiments stated below.

Second detection method, *kNN* classifier, is based on detection attributes. The values of such attributes can be determined on disguised data. The modified classifier utilizes 14 detection attributes: six generic attributes (WDMA, RDMA, WDA, LengthVar, DegSim with k = 450 and DegSim′ with k = 2, d = 963); six average attack models (filler mean variance, filler mean difference, and profile variance; computed for both push and nuke); two bandwagon attack models (FMTD; computed for both push and nuke). Like in non-private environment, class

labels and detection attributes are generated for the whole data set. As a classifier, a *kNN* with $k = 9$ is used. This is the same value used in the study by Mobasher et al. (2007b). This gives the chance of comparing the results of the proposed method to the results obtained by them. All experiments in the present study are conducted using both the proposed method and the one introduced by Mobasher et al. (2007b).

k-means clustering-based detection method utilizes PCC to group users into *k* clusters. As shown by Bilge and Polat (2013), *k*-means clustering is able to group users into clusters with decent accuracy using disguised data. The success of this method mainly depends on the ability of correctly clustering users into clusters. The similarity of each profile in the clusters to the cluster center is calculated in order to determine the attack cluster. The similarity between the attack profiles is higher than that of the other profiles. The cluster with the highest average similarity is isolated from the system. The selection of the cluster number is important for the performance of the application. The results of the trials reveal that 12 can be chosen as the ideal number of clusters. In this clustering method, as the initial cluster centers are chosen randomly among the data, different results could be produced when same data are processed recursively. Choosing initial cluster centers could affect the results. The steps of the *k*-means clustering-based detection scheme employed on perturb data are defined as follows:

Algorithm 2 k-means clustering-based detection method on masked data

Let $U' = \{u_1, u_2, \ldots, u_n\}$- set of disguised data vectors & $C = \{c_1, c_2, \ldots, c_k\}$-set of cluster centers

1: Randomly select the 'k' cluster centers

2: Estimate the similarity between each data vector and cluster centers

3: Assign the data vector to the closest cluster

4: Recalculate the new cluster center

5: Recalculate the similarity between each data vector and new obtained cluster centers

6: If no data vector is reassigned then stop, otherwise, repeat from step 3

7: Determine the cluster with the highest average similarity as shilling cluster

The steps defined in ***Algorithm 1***, which is stated before for PCA-based variable selection detection method, are also used in PPCF. However, in PPCF

schemes, disguised z-score data are used as input data. During data disguising, random numbers whose average is 0 are added to *z*-scores and masked values are obtained ***(***$z'_{uj} = z_{uj} + r_{uj}$***)***. Similarly, the average of z-score data is expected to be 0. Polat and Du (2005c) state that during the scalar product and sum process, the effect of random numbers could be neglected because the average of random numbers is 0. Nevertheless, the same random numbers must be multiplied while calculating the diagonal components of the matrix, which is obtained as a result of the COV ← D^T D process stated in the third line of ***Algorithm 1***. Multiplying the same random numbers, r_{uj}^2, will create an excess value. To lessen such effects, $n\sigma_r^2$ value is extracted from the diagonal components (Polat and Du 2005c), where n shows the number of random numbers and σ_r shows the standard deviation of random numbers. After modifying the ***Algorithm 1*** as described above, it is utilized as a detection method for filtering out shilling profiles in PPCF schemes.

7.2.3. Experimental evaluation

In order to show the ability of the four modified shilling attack detection methods on disguised databases in PPCF schemes with respect to six shilling attack models, real data-based experiments are performed. The success of shilling attack models depends on two control parameters: *filler size* and *attack size*. Privacy-preserving parameters are kept constant, β_{max} = 25% and σ_{max} = 2. In this section, the empirical outcomes of the current contribution with respect to the varying control parameters are presented and the significance of these results is discussed.

7.2.3.1. Effects of filler size parameter

Experiments are performed for signifying the performance of the detection methods with varying *filler size* values while detecting fake profiles in masked databases. Filler size is varied from 5% to 50% while attack size is kept constant at 15%. The tests are repeated 100 times due to randomization in the perturbation process. Overall averages of precision and recall for Chirita algorithm with varying filler size values are shown in Table 7.1, where *RB* stands for reverse bandwagon.

Table 7.1. Performance of Chirita algorithm with varying filler size

	Precision					*Recall*				
Filler Size	5	10	15	25	50	5	10	15	25	50
Random	0.206	0.214	0.217	**0.221**	0.217	0.206	0.214	0.217	**0.221**	0.217
Average	**0.000**	0.000	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000	0.000
Bandwagon	0.143	0.181	0.199	**0.209**	0.207	0.143	0.181	0.199	**0.209**	0.207
Segment	0.169	0.141	0.143	**0.175**	0.138	0.169	0.141	0.143	0.175	0.138
RB	0.145	0.175	0.186	**0.195**	0.192	0.145	0.175	0.186	**0.195**	0.192
Love/Hate	0.135	0.177	0.197	0.204	**0.205**	0.135	0.177	0.197	0.204	**0.205**

As seen from Table 7.1, empirical outcomes with respect to precision and recall are equal for Chirita algorithm. Notice that the profiles are listed from top to bottom according to PA_u and the first s profiles are classified as the attack profiles, as explained before. Since s is considered as the number of added attack profiles into the system, precision and recall values are found equal. The increase in filler size value does not significantly change precision and recall values for all attack models. Precision and recall values for all attack models except average attack with varying filler size values range from 0.135 to 0.221. Therefore, Chirita algorithm shows weak performance on detection operation in private environments. The best outcomes are usually observed when filler size is 25%. The most successful results are obtained for random attack. For average attack, all filler size values get the value 0. Recall that Chirita algorithm does classification operation considering especially RDMA attribute value. RDMA values for attack profiles will be higher than those of real profiles. However, while forming average attack profiles since the filler items are filled with the item mean, RDMA value becomes lower. Compared to outcomes for non-private environment published by Chirita et al. (2005), Chirita algorithm provides lower results in private environments. There are couple of reasons why the results are lower than the ones published by Chirita et al. (2005). First, in the report by Chirita et al. (2005), there are simultaneous attacks to three target items. As a result, RDMA values are found higher. In these experiments, attacks are performed to each 50 item separately. The other reason might be data disguising in PPCF schemes. Selection of σ during data disguising operation affects RDMA attribute value, which may affect the results. The same experiments are then conducted for *kNN* classifier-based detection algorithm. Overall averages of precision and recall are displayed in Table 7.2.

Table 7.2. Performance of *kNN* classifier with varying filler size

	Precision					*Recall*				
Filler Size	5	10	15	25	50	5	10	15	25	50
Random	**0.987**	0.884	0.872	0.872	0.987	**0.987**	0.987	0.974	0.974	0.987
Average	**1.000**	0.927	0.938	0.938	0.987	**0.987**	0.987	0.987	0.987	0.987
Bandwagon	0.776	0.800	0.817	0.817	**0.987**	**0.987**	0.987	0.987	0.987	0.987
Segment	**1.000**	0.925	0.873	0.873	0.987	**0.987**	0.961	0.805	0.805	0.987
RB	**0.987**	0.920	0.927	0.962	0.987	0.974	0.896	0.987	0.987	**0.987**
Love/Hate	0.917	0.936	0.950	0.884	**0.987**	0.286	0.948	**0.987**	0.987	0.984

It seems that *kNN* classifier algorithm is quite successful in detection operation of the PPCF attack models. Upon a change of filler size value from 5% to 50%, almost all of the precision and recall values vary between 0.800 and 1.000 for all attack models. Precision values for reverse bandwagon and love/hate nuke attacks increase directly with the filler size value. The change in precision value depicts variability depending on the filler size for the push attacks. Precision value lower than 1.0 exhibits that some of the real profiles are classified as attack profiles. However, in general, *kNN* classifier algorithm is also successful in PPCF algorithm as in the case of CF algorithm. The disguise operation in PPCF algorithm does not have significant effect on the detection algorithm performance. Since *kNN* classifier can also create a model by using train data, which are disguised in PPCF schemes, attack profiles on masked data can be detected by test set easily using this model. As shown in Table 7.2, the recall values of *kNN* classifier algorithm depict high success rates. For all filler size values, the algorithm performs very well with respect to recall. Generally speaking, random, average, and bandwagon push attacks are similar to each other. Therefore, the recall values of such push attacks are closer to each other. The recall value of segment attack, different than other attacks due to the purpose of it, could be slightly lower than those of other attacks. For the nuke attacks at lower filler size values, the recall value is lower than that of the push attacks. Yet, as the filler size value increases, the recall value of the nuke attacks approaches to 1.0. The results are similar to the ones calculated for CF algorithm reported in the study by Mobasher et al. (2007b). Another set of experiments are conducted to evaluate the success of *k*-means clustering-based detection method in private environments. After calculating the overall averages of precision and recall for *k*-means algorithm, they are displayed in Table 7.3.

Table 7.3. Performance of *k*-means clustering with varying filler size

	Precision					*Recall*				
Filler Size	5	10	15	25	50	5	10	15	25	50
Random	**0.501**	0.361	0.298	0.245	0.192	**1.000**	0.997	1.000	1.000	1.000
Average	**0.349**	0.347	0.319	0.308	0.255	0.838	**1.000**	1.000	1.000	1.000
Bandwagon	**0.358**	0.285	0.267	0.260	0.232	**1.000**	1.000	1.000	1.000	1.000
Segment	**0.436**	0.362	0.328	0.297	0.225	0.953	**1.000**	1.000	1.000	1.000
RB	**0.350**	0.313	0.298	0.290	0.248	**1.000**	1.000	1.000	1.000	1.000
Love/Hate	**0.340**	0.275	0.243	0.231	0.229	0.995	**1.000**	1.000	1.000	1.000

As indicated in Table 7.3, recall values are very close to each other for all attack models and filler size values. Precision values decrease with increasing filler size values for all attack models. Recall that *k*-means clustering-based detection method does clustering operation by considering the similarities between profiles. For this reason, type of the attack model is not significant. Since all attack models are formed with defined algorithms, they are all naturally similar to each other. Based on this similarity, *k*-means clustering-based detection method finds the tightest cluster and isolates that cluster from the system. As the filler size value increases, the attack profiles become more similar to the real profiles. Thus, there will be more real profiles in the cluster with the attack profiles. In this situation, as shown in Table 7.3, more real profiles will be isolated from the system leading to lower precision values. The recall values for all attack models approach to 1.0. As mentioned above, the increase in filler size values only increases the number of real profiles in the cluster under search. This case does not change the recall value. As a result, *k*-means clustering-based detection method on the one hand classifies almost 100% of the attack profiles correctly, on the other hand, it pushes many of the real profiles to the outside of the system. This negatively affects the accuracy of the system.

The same experiments are performed for the last detection method, PCA-based variable selection detection algorithm. Table 7.4 shows the overall averages of precision and recall values with varying attack size values for this detection algorithm.

Table 7.4. Performance of PCA-based detection scheme with varying filler size

	Precision					*Recall*				
Filler Size	5	10	15	25	50	5	10	15	25	50
Random	0.300	0.330	**0.340**	0.340	0.270	0.300	0.330	**0.340**	0.340	0.270
Average	0.440	0.610	**0.670**	0.650	0.300	0.440	0.610	**0.670**	0.650	0.300
Bandwagon	**0.090**	0.080	0.080	0.090	0.090	**0.090**	0.080	0.080	0.090	0.090
Segment	0.090	0.080	0.080	0.090	**0.100**	0.090	0.080	0.080	0.090	**0.100**
RB	0.076	0.079	0.082	0.082	**0.088**	0.076	0.079	0.082	0.082	**0.088**
Love/Hate	**0.060**	0.058	0.057	0.057	0.057	**0.060**	0.058	0.057	0.057	0.057

As seen from Table 7.4, the best results are obtained for average attack model. While generating average attack profiles since the filler item set is filled with around the item mean, it is expected to have small covariance value of the attack profiles among each other. Mehta et al. (2007a) mention that the covariance value among the attack profiles is less than that among the real profiles. For this reason, the attack profiles might be detected by PCA-based variable selection technique. As shown in Table 7.4, both precision and recall values for the average attack model reach 0.670. While establishing the other attack models, filler item set is filled with random numbers generated with known standard deviation. In the current application, since σ_{max} is set to 2, standard deviation is randomly selected from the range of (0, 2]. If the standard deviation is high, the covariance among the profiles will be high. In this case, PCA-based variable selection detection algorithm does not yield successful results.

7.2.3.2. Effects of attack size parameter

Various sets of experiments are conducted for scrutinizing the success of shilling attack detection methods with changing attack size values on private environments. Attack size is the second control parameter exactly affecting overall success of a detection method. It emphasizes the impact of determining the number of bogus profiles to be inserted into a database. Furthermore, it touches the utility perspective of an attack. It is clear that more attack profiles inserted into the system refers to the situation of the larger obtained shifts. However, it establishes an adjustment between the delectability and the impact of the applied attack. Thus, experiments are conducted while varying attack size from 1% to 15%, where filler

size is kept constant at 25%. The trials are repeated 100 times due to randomization. Overall averages of precision and recall values are displayed with varying attack size values for Chirita algorithm, *kNN* classifier, *k*-means clustering, and PCA-based scheme in Table 7.5, Table 7.6, Table 7.7, and Table 7.8, respectively.

Table 7.5. Performance of Chirita algorithm with varying attack size

	Precision					*Recall*				
Attack Size	1	3	5	10	15	1	3	5	10	15
Random	0.018	0.051	0.084	0.157	**0.221**	0.018	0.051	0.084	0.157	**0.221**
Average	**0.000**	0.000	0.000	0.000	0.000	**0.000**	0.000	0.000	0.000	0.000
Bandwagon	0.015	0.049	0.077	0.148	**0.209**	0.015	0.049	0.077	0.148	**0.209**
Segment	0.013	0.041	0.067	0.127	**0.175**	0.013	0.041	0.067	0.127	**0.175**
RB	0.014	0.043	0.070	0.132	**0.195**	0.014	0.043	0.070	0.132	**0.195**
Love/Hate	0.017	0.049	0.075	0.144	**0.204**	0.017	0.049	0.075	0.144	**0.204**

Chirita algorithm is successful in detecting shilling attacks with dense attacker profiles, and unsuccessful against attacks with small size and high sparsity (Williams et al., 2006). As seen from Table 7.5, as the attack size increases, this algorithm becomes more successful towards the attacks excluding average attack. Since RDMA values of random attack profiles are higher, the most successful precision and recall values are obtained for random attack. On the contrary, average attack profiles have lower RDMA values due to establishing methodology, and thus, Chirita algorithm cannot detect these attack profiles. Although detection performance of the algorithms becomes better with increasing attack size, they are not successful in detecting shilling profiles. The best result for this method is 0.22 for both precision and recall; and such values are observed for random attack when the attack size is 15%.

Table 7.6. Performance of *kNN* classifier with varying attack size

	Precision					*Recall*				
Attack Size	1	3	5	10	15	1	3	5	10	15
Random	0.000	0.750	0.852	0.847	**0.872**	0.000	0.706	0.852	0.962	**0.974**
Average	**1.000**	0.842	0.839	0.895	0.938	0.857	0.941	0.963	0.981	**0.987**
Bandwagon	0.000	0.375	0.815	**0.850**	0.817	0.000	0.176	0.815	0.981	**0.987**
Segment	0.000	0.750	0.833	0.872	**0.873**	0.000	0.706	**0.926**	0.788	0.805
RB	**1.000**	0.938	0.929	0.927	0.962	0.143	0.882	0.963	0.981	**0.987**
Love/Hate	**1.000**	1.000	0.897	0.836	0.884	0.571	0.941	0.963	0.981	**0.987**

As depicted in Table 7.6, precision and recall values are usually ranging between 0.8 and 1.0 for *kNN* classifier. Especially, when the attack size exceeds 5%, this algorithm becomes more successful. Number of attack profiles in train and test data cannot be enough to make a stable classification when the attack size is low. Thus, zero precision and recall values were gathered for attack size being 1% for random, bandwagon, and segment attacks, while better results are acquired for other attacks. Actually, since train data set is used in all attack models within this method, it does not matter whichever attack model is used. As long as there are enough train data, this method proves out to be successful. Therefore, attack size parameter plays an important role for the success of this method.

Table 7.7. Performance of *k*-means clustering with varying attack size

	Precision					*Recall*				
Attack Size	1	3	5	10	15	1	3	5	10	15
Random	0.095	0.152	0.189	0.248	**0.289**	0.883	0.915	0.949	0.927	**0.962**
Average	0.116	0.229	0.279	0.357	**0.396**	0.874	0.902	**0.943**	0.879	0.919
Bandwagon	0.108	0.226	0.276	0.316	**0.344**	0.885	0.913	0.950	0.937	**0.975**
Segment	0.103	0.179	0.243	0.329	**0.386**	0.910	0.920	**0.961**	0.922	0.941
RB	0.109	0.243	0.302	0.371	**0.401**	**1.000**	0.999	0.998	0.975	0.951
Love/Hate	0.110	0.212	0.244	0.264	**0.288**	**1.000**	0.991	0.997	0.999	1.000

In Table 7.7, it is seen that there is a direct correlation between attack size and precision value of the *k*-means detection algorithm towards the attacks. As attack size increases, number of attack profiles in the cluster of interest increases. This leads to improvement in precision value. Since the tightest cluster is found and isolated from the database, a lot of real profiles are left out in this cluster. As a result of leaving real profiles out of the user-item matrix, precision value turns out to be lower than recall value. As in the application of the previous section, recall metric value for *k*-means clustering-based detection method is seriously meaningful. In this situation, as discussed previously, since attack profiles are so similar to each other, they are located in the same cluster. Therefore, recall value becomes successful in all attack models. As seen from Table 7.7, recall value changes between about 0.9 and 1.0 for all attacks. Differences in attack size can only increase the number of attacks in cluster, and they do not significantly affect recall value.

Table 7.8. Performance of PCA-based detection scheme with varying attack size values

	Precision					*Recall*				
Attack Size	1	3	5	10	15	1	3	5	10	15
Random	0.120	0.180	0.220	0.290	**0.340**	0.120	0.180	0.220	0.290	**0.340**
Average	0.030	0.220	0.370	0.560	**0.650**	0.030	0.220	0.370	0.560	**0.650**
Bandwagon	0.040	0.050	0.050	0.070	**0.090**	0.040	0.050	0.050	0.070	**0.090**
Segment	0.110	**0.150**	0.140	0.060	0.090	0.110	**0.150**	0.140	0.060	0.090
RB	0.078	0.067	0.041	0.057	**0.082**	0.078	0.067	0.041	0.057	**0.082**
Love/Hate	0.006	0.015	0.021	0.038	**0.057**	0.006	0.015	0.021	0.038	**0.057**

Like in the case of the application based on filler size parameter explained above, the application based on attack size parameter yields the best results for average attack as shown in Table 7.8. The reason why PCA-based detection technique is more successful for average attack model is described previously. As seen in Table 7.8, the increase in attack size value causes an increase in precision and recall values of almost all of attack models. Precision and recall values of average attack approaches as high as 0.65, and this is an acceptable success. This method is unsuccessful for other attack models. The reason for this is that the profiles have higher covariance values according to their generating algorithms. As mentioned before, PCA method can make categorization by using low covariance qualification among attack profiles.

7.2.4. Discussion

Many methods have been developed to detect the attacks performed for manipulating recommendation systems. PPCF schemes can also be exposed to these attacks as well as non-private CF algorithms. As a matter of fact, literature review shows that Gunes et al. (2013b) and Bilge et al. (2014) developed attack models for PPCF methods and they reported the effects of these attacks on PPCF techniques using real data-based experiments. However, there has not been a study on detecting the attacks against PPCF schemes. Hence, the most widely used four detection methods on non-private schemes are adapted and used on PPCF schemes.

When the empirical results, shown in the tables above, are examined, it can be seen that *kNN* classifier method is the most successful method for all attack models. The disguise operation in private environments does not have a significant

effect on the detection algorithm performance. *kNN* classifier detection algorithm calculates a number of generic and model-specific attribute values for each profile, and creates a new data table as well as making classifications by using this new data table. It divides this attribute table into two groups under the headlines of train and test data. Since it creates a model by using train data, data masking does not have significant effects for *kNN* classifier. By using a train set generated from perturbed data, a new model can be formed to detect PPCF attack profiles on test set.

Upon comparing Chirita algorithm performed on attacks in private environments with the implementations on non-private environments, it is seen that it is not a highly successful practice. Chirita et al. (2005) states that due to the high standard deviation among the rating dispersions in attack profiles, RDMA attribute value will be higher than real profiles. In addition, they increase the deviation among ratings by making push or nuke attacks on three target items during the experiments. Here in this thesis, each item listed in target items of experiments is attacked respectively and one by one. Attack profiles generated on PPCF schemes are filled with random numbers, which are generated with σ chosen over the range $(0, \sigma_{max} = 2]$. If the σ value is small, RDMA value for attack profiles turn out to be small, and Chirita algorithm cannot detect these PPCF attacks successfully.

It is hypothesized that with increasing σ values, Chirita algorithm might become more successful. In order to verify this hypothesis, a set of experiments are conducted while varying σ values, where σ values are selected from the ranges [0, 2], [1, 2], and [2, 3]. Thus, the expected mean values for σ are 1, 1.5, and 2.5 for each range, respectively. In this set of experiments, filler size is fixed at 25% and attack size at 15%. Similar methodology is followed for all attack models and overall averages are displayed in Table 7.9. The outcomes show that varying σ values definitely affect the success of Chirita algorithm. Especially when σ value is chosen from the range [2, 3], precision and recall values are over 0.9 for all attacks except average attack. Since item mean is used when filling profiles in average attack, RDMA value turns out to be low even if we use larger σ values. In this case, as seen from Table 7.9, this model will be unsuccessful for average attack.

Table 7.9. Performance of Chirita-based detection scheme with varying standard deviation

	Precision				*Recall*	
σ	[0, 2]	[1, 2]	[2, 3]	[0, 2]	[1, 2]	[2, 3]
Random	0.221	0.396	**0.982**	0.221	0.396	**0.982**
Average	**0.000**	0.000	0.000	**0.000**	0.000	0.000
Bandwagon	0.209	0.384	**0.970**	0.209	0.384	**0.970**
Segment	0.175	0.368	**0.971**	0.175	0.368	**0.971**
RB	0.195	0.395	**0.975**	0.195	0.395	**0.975**
Love/Hate	0.204	0.395	**0.979**	0.204	0.395	**0.979**

Even though the precision value of *k*-means algorithm is not quite good, the recall value is highly successful. The modified *k*-means algorithm makes clustering process by considering the similarities between profiles. A certain profile is created for each attack model in non-private environments; therefore, these profiles are similar to each other. The same result applies for the attack profiles on PPCF. Since the attack profiles are similar due to their generation methods, they are expected to be dispersed in the same cluster. However, one of the main goals of shilling attacks is to manipulate the system. Thus, they try to look similar to real profiles. As a result of this goal, in *k*-means clusters, there might be many real profiles in the same cluster together with the attack profiles. Consequently, a lot of real profiles are left out of the database as soon as the isolation of defined attack clusters, which leads to smaller precision values.

PCA-based variable selection-based detection method is successful for average attack only. This situation can be explained with the fact that smaller covariance values among profiles due to completing filler items set with item mean when creating average attack profiles. In other attack models, filler items set in profiles are completed with random numbers generated with a certain σ value. If the σ value of the random numbers are high, covariance value among profiles also becomes high; due to which PCA algorithm may not be able to detect these attack profiles. In the studies proposed by Mehta (2007), Mehta et al. (2007a) and Mehta and Nejdl (2009) the authors state that it is expected to have lower covariance value than real profiles because filler items set is generally completed with item mean or system overall mean while creating attack profiles. Hence, it is hypothesized that the success of PCA-based detection algorithm can be improved if smaller σ values are used. To verify this hypothesis experimentally, another set of trials are

performed using the same methodology while varying σ values, where σ values are selected from the ranges [0, 2], [0, 1], and [0, 0.5]. Filler size is fixed at 25% and attack size at 15%. Table 7.10 shows the empirical outcomes. As seen from Table 7.10, smaller σ values definitely improves the success of PCA-based algorithm. With decreasing σ values, precision and recall values become larger. Especially for the range [0, 0.5], quite remarkable results are achieved for all attacks except for love/hate attack model.

Table 7.10. Performance of PCA-based detection scheme with varying standard deviation

	Precision				*Recall*		
σ	[0, 2]	[0, 1]	[0, 0.5]		[0, 2]	[0, 1]	[0, 0.5]
Random	0.340	0.519	**0.721**		0.340	0.519	**0.721**
Average	0.650	**0.740**	0.738		0.650	**0.740**	0.738
Bandwagon	0.090	0.215	**0.411**		0.090	0.215	**0.411**
Segment	0.090	0.491	**0.713**		0.090	0.491	**0.713**
RB	0.082	0.289	**0.613**		0.082	0.289	**0.613**
Love/Hate	0.057	0.068	**0.093**		0.057	0.068	**0.093**

Finally, the detection methods used for PPCF schemes are compared with the respective ones utilized in CF schemes under the same conditions. In Table 7.11 and Table 7.12, precision and recall values are compared, respectively, where filler size is 25% and attack size is 1%. The results of experiments carried on CF schemes are gathered from related studies. If the comparison with the results taken from the study proposed by Burke et al. (2006a) for Chirita algorithm is considered, it can be concluded that precision value is the same with the results on modified algorithm; however, the practice on CF algorithm is more successful for recall value. When a comparison is made with *kNN* classifier algorithm taken from the study proposed by Mobasher et al. (2007b), the results from both studies vary from each other. However, at a higher filler size and attack size values, the results of both studies are similar. Bhaumik et al. (2011) utilize *k*-means algorithm on CF in a different way than the way in this thesis. They define some generic attribute values and perform cluster process with these values. When the results are compared, it is seen that their results are more successful than the modified one. Precision and recall values achieved for PCA method in the study proposed by Mehta and Nejdl (2009) are quite higher than the results of this thesis. The reason behind this

difference is the higher σ values chosen during the generation of PPCF attack profiles, as mentioned before. As seen from Table 7.11, in the experiments with decreasing σ values, successful outcomes are observed in private environments.

Table 7.11. Comparison of detection algorithms on precision

	No Privacy				*Privacy*			
	Chirita	*kNN*	*k-means*	*PCA*	*Chirita*	*kNN*	*k-means*	*PCA*
Random	0.020	0.350	0.980	0.960	0.018	0.000	0.095	0.120
Average	0.020	0.330	0.920	0.900	0.000	1.000	0.116	0.030
Bandwagon	0.020	0.350	0.900	0.960	0.015	0.000	0.108	0.040
Segment	0.050	0.280	0.980	-	0.013	0.000	0.103	0.110
RB	-	-	-	-	0.014	1.000	0.109	0.093
Love/Hate	0.010	0.350	-	-	0.017	1.000	0.110	0.058

Table 7.12. Comparison of detection algorithms on recall

	No Privacy				*Privacy*			
	Chirita	*kNN*	*k-means*	*PCA*	*Chirita*	*kNN*	*k-means*	*PCA*
Random	0.680	1.000	1.000	1.000	0.018	0.000	0.883	0.120
Average	0.620	1.000	1.000	1.000	0.000	0.857	0.874	0.030
Bandwagon	0.650	1.000	1.000	1.000	0.015	0.000	0.885	0.040
Segment	0.650	0.920	1.000	-	0.013	0.000	0.910	0.110
RB	-	-	-	-	0.014	0.143	1.000	0.093
Love/Hate	0.670	1.000	-	-	0.017	0.571	1.000	0.058

7.3. A Novel Detection Algorithm

In addition to modifying existing detection scheme, novel methods should be proposed. In this section, a novel algorithm is described, which is based on hierarchical clustering. Hierarchical clustering creates a hierarchy of clusters (Johnson, 1967; Sembiring et al., 2011; Madhulatha, 2012). It is an unsupervised clustering algorithm, which does not contain any of experimental variables. It can be thought as a tree structure called a dendogram. It is a tree that represents how clusters are combined/divided hierarchically. A key step is selecting a distance measure. Hierarchical clustering algorithms may be agglomerative or divisive. Agglomerative one starts at the leaves and successively merges clusters together. It uses each element as a separate cluster at the beginning and tries to convert them into successively larger clusters. Divisive algorithm is a top-down clustering.

Agglomerative clustering, on the other hand, is known as botom-up clustering. Divisive one selects the whole set as a starting point and divides it into successively smaller clusters. Any metric can be used to measure the similarity between pairs of observations. The choice of which clusters to merge or split is determined by a linkage criterion, which is a function of the pair wise distances between observations. If there is a need of certain number of clusters, it will be sufficient to stop accordingly. Hierarchical methods suffer from the act that once the merge or split is done, it can never be undone. The basic process of hierarchical clustering can be described as follows (Johnson, 1967):

1. Appoint each item to a cluster. If there are *n* items, then there will be *n* clusters with one item.
2. Find the closest (most similar) pair of clusters and combine them into a single cluster leading a case with one cluster less.
3. Calculate distances (similarities) between the new cluster and each of the old clusters.
4. Repeat steps 2 and 3 until all items are collected into a single cluster of size *n*.

7.3.1. Hierarchical clustering-based detection algorithm

Clustering is a broadly used technique for determining shilling profiles in CF schemes (Mehta and Nejdl, 2009; Bhaumik et al., 2011; Bilge et al., 2014). Clustering methods group similar entities in to the same clusters. Shilling profiles are very similar to each other because they are generated using the same methodology. Therefore, if clustering methods are used to group profiles, shilling profiles will be placed in the same cluster with high probability. There are different clustering algorithms, where each one has its own advantages and disadvantages. In addition to successfully grouping shilling profiles into the same cluster, clustering algorithms should have other advantages. It is proposed to use hierarchical clustering method as a shilling attack detection technique. Such clustering algorithm has not been used for detection process yet and also has some advantages compared to other clustering methods (Manning et al., 2008).

Hierarchical clustering presents a more informative structure than flat clustering. It provides a hierarchic structure instead of an unstructured set of clusters. There is no need to pre-specify the number of clusters. Since many of them are deterministic, they lead to the cost of lower efficiency. Although common hierarchical clustering algorithms have a complexity that is at least quadratic, they produce better clusters than flat clustering.

Note that users disguise their z-scores and send perturbed data to the server. Let v_{uj} be the rating for user u on item j, σ_u be the standard deviation of her ratings, and V_u be the mean rating of her ratings. The related z-score (z_{uj}) can be estimated as $z_{uj} = (v_{uj}-V_u)/ \sigma_u$. Suppose that user u's ratings vector including z-scores is $\mathbf{U} = (z_{u1}, z_{u2}, \ldots, z_{um})$, where m is the total number of items and note that $\mathbf{U}$ is a very sparse vector. User u disguises her rating vector and obtains masked vector as follows: $\mathbf{U'} = (z_{u1}+r_{u1}, z_{u2}+r_{u2}, \ldots, z_{um}+r_{um})$, where r_{uj} values are random numbers. Users mask their z-scores similarly and send the server. The server needs to cluster the users based on their perturbed data. According to the steps for hierarchical clustering explained above, similarities between any two entities need to be estimated. Such similarities based on masked z-scores can be estimated as follows:

$$w'_{uv} = z'_{u1}z'_{v1} + z'_{u2}z'_{v2} + \ldots + z'_{um}z'_{vm} \tag{7.1}$$

$$w'_{uv} = (z_{u1} + r_{u1})(z_{v1} + r_{v1}) + (z_{u2} + r_{u2})(z_{v2} + r_{v2}) + \ldots + (z_{um} + r_{um})(z_{vm} + r_{vm}) \tag{7.2}$$

Note that

$$(z_{7u1} + r_{u1})(z_{v1} + r_{v1}) = z_{u1}z_{v1} + z_{u1}r_{v1} + r_{u1}z_{v1} + r_{u1}r_{v1} \tag{7.3}$$

Similarly, other multiplications can be extended. Since the random numbers are generated with zero mean and a standard deviation using uniform or Gaussian distribution, expected values of the sum of $z_{u1}r_{v1}$, $r_{u1}z_{v1}$, and $r_{u1}r_{v1}$ values will be zero. Therefore, the similarities between any two users can be estimated with decent accuracy from masked z-scores.

User profiles including fake or shilling profiles are clustered using hierarchical clustering according to the similarity weights on disguised data. Due to the nature of the random numbers and the similarity weights, hierarchical clustering is able to put users into clusters with good accuracy using disguised data. The performance of this method mainly relies on the ability of correctly grouping users into clusters. It is expected that attack or shilling profiles will be found in the same cluster due to high similarities between fake profiles.

The cluster having the most similar elements is considered as the attack cluster. In order to determine such cluster, *DegSim* metric is used. Therefore, *DegSim* for each profile in a cluster needs to be calculated (Burke et al., 2006a). This metric is the average similarity weight with the top-*k* neighbors of a user *u* and can be calculated as follows: $DegSim_u = (w_{u1} + w_{u2} + \ldots + w_{uk})/k$. Since similarities are estimated on masked data, *DegSim* values for each user *u* can be estimated as follows:

$$DegSim'_u = (w'_{u1}w'_{v1} + \ ... \ + \ w'_{uk})/k \tag{7.3}$$

$$DegSim'_u = [(w_{u1} + R_{u1}) + (w_{u2} + R_{u2}) + \ ... \ + \ (w_{uk} + R_{uk}))/k] \tag{7.4}$$

where R_u values are noise data introduced due to random numbers in similarity computations. The equation can be written as follows:

$$DegSim'_u = [(w_{u1} + w_{u2} + \ ... \ + w_{uk})/\mathrm{k} + \ [(R_{u1} + R_{u2} + \ ... \ + R_{uk})/\mathrm{k} \tag{7.5}$$

Due to the same reasons described previously, the expected value of $(R_{u1} + R_{u2} + \ldots + R_{uk})/k$ will be zero. Therefore, *DegSim* values for all users can be estimated with decent accuracy based on perturbed data. After estimating such values for all users in each cluster, the cluster with the highest average *DegSim* value is considered as the attack cluster (the cluster including shilling profiles). This cluster is then isolated from PPCF system, which prevents the system to be manipulated.

As discussed before, attack profiles are so similar to each other because shilling profiles are generated by a certain algorithm. At the same time, shilling attacks aim to be similar with real profiles to manipulate PPCF systems. As a result, there might be many real profiles in the attack cluster together with the attack profiles. Herewith, depending on the isolation of defined attack clusters, a lot of real profiles might be left out of database and this leads to smaller precision values. To save real profiles in the determined cluster, it is proposed to analyze target items in such a way so that the real profiles in the attack cluster are distinguished. Hence, all profiles are analyzed in the attack cluster and separate real profiles.

While generating attack profiles regarding PPCF schemes, target items are given maximum and minimum values of randomly generated random numbers for push and nuke attacks, respectively. In this way, the attack profiles will be distinguishable from other attack profiles. In other words, the target items might have the maximum or the minimum values for push and nuke attack profiles, respectively. However, the target item is unknown during detection process; and therefore, the target item needs to be determined first. The target item for push (nuke) attacks can be determined as follows:

1. Find the maximum (minimum) value in each profile in the database.
2. Determine the corresponding items in each profile.
3. The item holding the most number of maximum (minimum) values is determined as the target item.

After determining the target item, those profiles whose corresponding values for the target item are maximum (minimum) in the attack cluster are marked as shilling profiles. The remaining profiles are marked as authentic profiles. In this way, if there are some real profiles in the attack cluster, they then can be determined, which leads to better precision.

7.3.2. Experimental evaluation

To show the ability of the new shilling attack detection method on disguised databases in PPCF schemes with respect to six shilling attacks, various experiments are conducted using real data. The success of shilling attacks depends on two

control parameters: *filler size* and *attack size*. Privacy parameters, β_{max} and σ_{max} are fixed at 25% and 2, respectively.

7.3.2.1. Effects of filler size parameter

Experiments are made for determining the performance of the detection methods with varying *filler size* values while detecting fake profiles in masked databases. During the experiments, filler size is ranged from 5% to 50% while attack size is kept constant at 15%. The tests are run again 100 times due to randomization in the perturbation process. Overall averages of precision and recall for hierarchical clustering algorithm with varying filler size values are shown in Table 7.13, where *RB* stands for reverse bandwagon attack model.

Table 7.13. Performance of hierarchical clustering algorithm with varying filler size

	Precision					*Recall*				
Filler Size	5	10	15	25	50	5	10	15	25	50
Random	**0.947**	0.097	0.003	0.004	0.006	**0.653**	0.057	0.002	0.003	0.004
Average	**0.903**	0.834	0.739	0.658	0.635	0.843	0.843	0.830	0.904	**0.968**
Bandwagon	0.779	0.806	0.818	**0.841**	0.397	**1.000**	1.000	0.999	0.971	0.391
Segment	**0.997**	0.997	0.993	0.992	0.988	**0.999**	0.989	0.967	0.911	0.915
RB	0.990	0.997	0.**998**	0.997	0.278	**1.000**	1.000	1.000	0.994	0.253
Love/Hate	**0.995**	0.947	0.578	0.165	0.031	**0.883**	0.645	0.329	0.083	0.014

Due to the properties of the attack profiles, it is easy to classify them. In random and love/hate attacks, items are randomly selected and randomly filled. Thus, it is difficult to categorize them. For this reason, hierarchical clustering method is not successful in these attacks compared to the other ones. In this experiment, generally speaking, precision value is better than the recall value. Since making them similar to the real profiles forms the attack profiles, while classifying them, most of the real profiles are also placed into the same profiles with the attack profiles. Isolating this group makes most of the real profiles attack profiles, and this lowers down the precision. In particular, when the filler size reaches 50%, the similarities between real and attack profiles become closer, and this lowers down the precision even further. Usually the attack profiles are dropped down to the same group; and therefore, the recall is higher. The best outcome is obtained in the push

attack model when segment attack filler size is 5% and with the precision value of 0.997. The best result is obtained in the nuke attack, when RB attack filler size is 15%, and the precision is about 0.998. Bandwagon obtains the best recall value for push attack with 1.000, while for nuke attack, RB obtains the best value with 1.000. As the filler size value increases, precision value decreases. Average, segment, bandwagon, and reverse bandwagon attacks obtain better precision and recall values.

7.3.2.2. Effects of attack size parameter

Series of experiments are administered for examining the success of the shilling attack detection method with modifying attack size values on private environments. Attack size is the second parameter fully affecting entire success of a detection method. It stresses the effect of defining the number of bogus profiles to be included into a database. Further, filler size parameter touches the utility way of an attack. It is obvious that more attack profiles inserted into the system refers to the situation of the larger obtained shifts. Whereas, it sets up an adjustment between the delectability and the impact of the applied attack model. For this reason, to explain the varying effects of the attack size parameter, it is varied from 1% to 15% while the filler size is kept constant at 25%. The experiments are repeated 100 times due to randomization. The complete averages of precision and recall values with varying attack size values for the proposed shilling attack detection method are shown. Table 7.14 shows the performance of the hierarchical clustering-based detection method in terms of precision and recall, respectively. Due to the same reasons, average, bandwagon, segment, and reverse bandwagon attacks are more successful compared to the others. Reverse bandwagon attack is the most successful one for 15% attack size with 0.998 precision and 1.000 recall value. Bandwagon and segment attacks obtain values closer to the ones obtained by reverse bandwagon. Random and love/hate attacks are not that successful due to the reasons explained in the previous experiment.

As the attack size increases, the number of profiles inserted into the system will also increase. Therefore, it will be easier to classify them by the clustering

method. For example, when the attack size is at 1%, there will be only 10 attack profiles. For this reason, it will be difficult to do classification. Yet, when the attack size is at 15%, there will be 150 attack profiles added. In this way, it will become easier to do classification of them. For this reason, the best outcomes are obtained when the attack size is 15%. In push attack, segment attack obtains the best precision value of 0.993, and in nuke attack, reverse bandwagon obtains the best precision value of 0.998. In the case of recall value, bandwagon yields 0.999 for push attack, for nuke attacks it is reverse bandwagon obtaining the best value of 1.000.

Table 7.14. Performance of hierarchical clustering algorithm with varying attack size

	Precision					*Recall*				
Attack Size	1	3	5	10	15	1	3	5	10	15
Random	0.001	0.001	0.002	**0.003**	**0.003**	**0.005**	0.003	0.003	0.003	0.002
Average	0.017	0.055	0.140	**0.401**	**0.739**	0.156	0.183	0.310	0.548	**0.830**
Bandwagon	0.018	0.139	0.432	**0.737**	**0.818**	0.186	0.434	0.802	0.998	**0.999**
Segment	0.001	0.074	0.793	**0.982**	**0.993**	0.004	0.105	0.836	0.955	**0.967**
RB	0.000	0.000	0.348	**0.980**	**0.998**	0.000	0.000	0.408	1.000	**1.000**
Love/Hate	0.000	0.001	0.002	**0.221**	**0.578**	0.003	0.003	0.003	0.149	**0.329**

It is hypothesized that performance of the proposed hierarchical clustering-based detection scheme with respect to precision might be improved by analyzing the values of target items. After evaluating our method in terms of precision and recall, some experiments are finally performed to show how target item analysis affects precision. The same methodology is followed and the trials are conducted. Overall averages of precision values with varying filler and attack size are displayed in Table 7.15 and Table 7.16, respectively.

Table 7.15. Performance of improved hierarchical clustering method with varying filler size

Filler Size	5	10	15	25	50
Random	**0.977**	0.111	0.005	0.005	0.006
Average	**1.000**	0.994	0.918	0.850	0.795
Bandwagon	**1.000**	1.000	1.000	1.000	0.525
Segment	**1.000**	1.000	1.000	1.000	1.000
RB	**1.000**	1.000	1.000	1.000	0.180
Love/Hate	**1.000**	0.959	0.640	0.150	0.002

Table 7.16. Performance of improved hierarchical clustering method with varying attack size

Attack Size	1	3	5	10	15
Random	0.000	0.001	0.001	0.003	**0.003**
Average	0.018	0.063	0.137	0.536	**0.918**
Bandwagon	0.018	0.122	0.635	0.968	**1.000**
Segment	0.000	0.141	0.904	1.000	**1.000**
RB	0.000	0.024	0.889	1.000	**1.000**
Love/Hate	0.001	0.000	0.002	0.240	**0.640**

The base results in terms of precision for the proposed method are displayed in Table 7.13 and Table 7.14 for varying filler and attack size values, respectively. After determining the attack cluster, the target items are determined first and their values are analyzed. If the results for the improved method displayed in Table 7.15 and Table 7.16 are compared with the base results, it is seen that analyzing the target items improves the outcomes. Since random attack is designed randomly, target item analysis does not make any difference. However, improvements are observed for specifically designed attacks. For example, precision value increases from 0.818 to 1.000 for bandwagon attack when attack size is 15%. Similarly, precision value for love/hate attack also improves from 0.578 to 0.640.

7.4. Conclusions

Since PPCF methods can be subjected to shilling attacks, it is also imperative for them to determine shilling profiles. Thus, four widely used detection algorithms are utilized in private environments in order to figure out shilling profiles created using six different attack models.

kNN classifier and *k*-means methods are more successful in comparison to other methods. Even though *kNN* classifier algorithm is successful, it needs a train data set, which can be defined as a disadvantage. PCA algorithm works faster than all the others. One disadvantage of PCA algorithm can be explained as the first *s* pieces should be classified as attack profiles because it ranges profiles according to the covariance value. *k*-means algorithm is also quite successful in isolating the attack profiles. However, a great number of real profiles can be left out. In consequence of this situation, system accuracy will be affected in a negative way.

When all detection methods are analyzed in general, *kNN* classifier provides the best results because a sample train data is used. It can be recommended for both non-private and private environments. *k*-means algorithm also provides good results for all attacks and especially for recall value. It is easy to use *k*-means algorithm; but choosing the cluster number has an important role. The best cluster number can be defined in accordance with the experiments carried out by considering existing data. It is crucial to choose the right cluster and to isolate it from the system. If a wrong cluster is chosen and isolated from the system, so many real profiles could also be left out. However, many attack profiles still stay in the system. Similar results can be generated for all attack models because profiles are similar due to attack generation algorithms. *k*-means algorithm makes cluster process by using this similarity. PCA algorithm is successful for average attack due to its low covariance feature. In this attack, filler items set is filled with item mean value when creating profiles to increase the resemblance to real profiles. This makes the covariance value among the created profiles low. In other attack models, random numbers are generated with a certain standard deviation value to fill filler items. Thus, it is more succeeding at average attack than the others. Chirita algorithm's success is low when compared to others, especially at attacks with lower attack size.

A novel shilling attack detection scheme is proposed. The hierarchical clustering-based shilling attack detection method basically clusters user profiles into various clusters. Due to the nature of the attack profiles, it is expected that they are grouped into the same cluster. The scheme then determines that cluster, named the attack cluster. Since it is expected that all fake profiles are grouped in this cluster, attack cluster is isolated. The real data-based empirical outcomes demonstrate that the proposed scheme is able to detect shilling profiles designed for private environments.

To improve the detection performance of the proposed scheme, the values of target items are analyzed. The target items are usually assigned to maximum or minimum random values for push and nuke attacks, respectively while designing them for PPCF schemes. Different sets of experiments are performed and the results verify the hypothesis. In other words, analyzing the values of the target items improves precision values for specifically designed attacks.

8. CONCLUSIONS AND FUTURE WORK

There are various studies proposed to provide recommendations while preserving data confidentiality. Similarly, there are different works aiming at enhancing the robustness of CF systems. However, as in CF schemes without privacy concerns, PPCF schemes can also be subjected to shilling attacks. Malicious users and/or sites might try to insert fake profiles to achieve nuke and push attacks; and make the robustness of such schemes worse. The objective of the dissertation is to study PPCF schemes in terms of shilling attacks. Firstly, in this dissertation, six attack models are proposed for PPCF schemes. Then, this dissertation analyzes the robustness of selected memory-based, model-based, and hybrid PPCF algorithms against these attack models. Four widely used detection methods are utilized in such a way to detect shilling profiles in privacy environment. Finally, a novel detection method is developed to filter out shilling profiles in PPCF schemes. Main conclusions and future works of the dissertation can be listed as follows:

1. Based on the extensive literature review about PPCF schemes, the current generation of PPCF requires extra improvements to make predictions more effective and privacy-preserving methods more protective and understandable. Upcoming PPCF systems should be resistant to data losses, more user-friendly, transparent, and effectively deployed into mobile devices. Another important issue that future PPCF techniques should be able to deal with is to make accurate predictions in the presence of shilling attacks. The so-called issues and concerns should be discussed in the notion of privacy and PPCF community about their future generations.
2. The comprehensive survey including research that has been carried out on the issue of shilling attacks will lead to researchers. In addition, detailed separate surveys about shilling attacks strategies, detection algorithms, robustness analysis, and robust algorithms should be conducted. Also, known attacks should be classified according to various classification dimensions and the related attributes. In addition, more work need to be done to create new attack strategies and the related detection algorithms to detect them. Moreover,

additional study should be done to enhance the robustness of well-known model-based and hybrid CF algorithms.

3. Privacy-preserving prediction methods can also be subjected to shilling attacks. In this thesis, PPCF systems are evaluated in terms of their robustness against profile injection attacks. Based on this, these systems are also vulnerable to profile injection. Modified bandwagon, segment, and reverse bandwagon attacks achieved significant alterations in produced predictions. 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 attacks, similar to traditional CF schemes.
4. The importance of empirical results is that they confirm the applicability of some attacks on recommendation schemes with privacy, which lead researchers to question the robustness of other methods.
5. Comprehensive real data-based experiments are conducted to evaluate the robustness of the four model-based PPCF algorithms against the six attack models. It is experimentally shown that privacy-preserving SVD- and item-based PPCF algorithms schemes are the most robust recommendation algorithms. Values of privacy-preserving control parameters might affect the overall performance of the attack models. Thus, investigating how varying values of such parameters affect the robustness of privacy-preserving model-based schemes warrants future work. The robustness of binary ratings-based privacy-preserving recommendation schemes against profile injection attacks can also be a future work.
6. In this thesis, privacy-preserving hybrid prediction methods have been evaluated in terms of robustness against shilling attacks. It is empirically shown that the hybrid scheme is vulnerable to shilling attacks. Especially bandwagon and reverse bandwagon attacks are efficient attacks for manipulating referrals. Based on the outcomes of the experiments, prediction shift values are affected by varying values of control parameters. Other hybrid recommendation algorithms should be investigated with respect to privacy and robustness. Extensive analysis should be performed to compare different

types of CF algorithms in terms of accuracy, efficiency, privacy, and robustness.

7. Four widely used detection techniques are utilized in such a way to detect shilling profiles created using six different attack models on masked data in PPCF systems' databases. According to experimental results, it is clearly seen that *kNN* classifier and *k*-means methods are more successful in comparison with the other methods. It is shown that values of privacy-preserving control parameters such as standard deviation might affect the detection method. In addition, success of *k*-means method is affected by the selected cluster size. As a future work, ways of improving the success of the existing detection algorithms should be investigated. Detection algorithms, which can filter out binary ratings-based shilling profiles should also be developed.
8. The hierarchical clustering-based detection method, which is utilized in such a way to discover shilling profiles on masked data in PPCF systems' databases is developed. It is experimentally verified that analyzing the values of the target items improves precision values for specifically designed attacks. It is experimentally shown that this novel detection method is quite successful on detection process. Other clustering algorithms can be employed for detecting shilling attacks in private environments as a future work. The detection methods used in non-private environments might be modified and utilized in private environments.

REFERENCES

Adomavicius, G. and Tuzhilin, A. (2005), "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*, **17** (6), 734-749.

Aggarwal, C.C. and Yu, P.S. (2008), "A general survey of privacy-preserving data mining models and algorithms," *Privacy-Preserving Data Mining* (Ed: Aggarwal, C.C. and Yu, P.S.), Springer US, New York, NY, USA, 11-52.

Agrawal, R. and Srikant, R. (2000), "Privacy-preserving data mining," *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, Dallas, TX, USA, 439-450.

Bhaumik, R., Mobasher, B. and Burke, R. (2011), "A clustering approach to unsupervised attack detection in collaborative recommender rystems," *Proceedings of the 7th IEEE International Conference on Data Mining*, Las Vegas, NV, USA, 181-187.

Bhaumik, R., Williams, C., Mobasher, B. and Burke, R. (2006), "Securing collaborative filtering against malicious attacks through anomaly detection," *Proceedings of the 4th Workshop on Intelligent Techniques for Web Personalization*, Boston, MA, USA.

Bilge, A., Gunes, I. and Polat, H. (2014), "Robustness analysis of privacy-preserving model-based recommendation schemes," *Expert Systems with Applications*, **41** (8), 3671-3681.

Bilge, A., Gurmeric, S. and Polat, H. (2012), "An enhanced collaborative filtering scheme via recursive clustering," *Proceedings of the Workshop on Knowledge Discovery, Data Mining and Machine Learning*, Dortmund, Germany.

Bilge, A., Kaleli, C., Yakut, I., Gunes, I. and Polat, H. (2013), "A survey of privacy-preserving collaborative filtering schemes," *International Journal of Software Engineering and Knowledge Engineering*, **23** (8), 1085-1108.

Bilge, A. and Polat, H. (2012), "An improved privacy-preserving DWT-based collaborative filtering scheme," *Expert Systems with Applications*, **39** (3), 3841-3854.

Bilge, A. and Polat, H. (2013), "A comparison of clustering-based privacy-preserving collaborative filtering schemes," *Applied Soft Computing*, **13** (5), 2478-2489.

Billsus, D. and Pazzani, M.J. (1998), "Learning collaborative information filters," *Proceedings of the 15th International Conference on Machine Learning*, Madison, WI, USA, 46-54.

Bobadilla, J., Ortega, F., Hernando, A. and Gutiérrez, A. (2013), "Recommender systems survey," *Knowledge-Based Systems*, **46**, 109-132.

Breese, J.S., Heckerman, D. and Kadie, C. (1998), "Empirical analysis of predictive algorithms for collaborative filtering," *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, Madison, WI, USA, 43-52.

Burke, R., Mobasher, B. and Bhaumik, R. (2005a), "Limited knowledge shilling attacks in collaborative filtering systems," *Proceedings of the 3rd IJCAI Workshop in Intelligent Techniques for Personalization*, Edinburgh, UK.

Burke, R., Mobasher, B., Bhaumik, R. and Williams, C. (2005b), "Collaborative recommendation vulnerability to focused bias injection attacks," *Proceedings of the Workshop on Privacy and Security Aspects of Data Mining*, Houston, TX, USA, 35-43.

Burke, R., Mobasher, B., Bhaumik, R. and Williams, C. (2005c), "Segment-based injection attacks against collaborative filtering recommender systems," *Proceedings of the 5th IEEE International Conference on Data Mining*, Houston, TX, USA, 577-580.

Burke, R., Mobasher, B., Zabicki, R. and Bhaumik, R. (2005d), "Identifying attack models for secure recommendation," *Proceedings of the WebKDD Workshop on the Next Generation of Recommender Systems Research*, San Diego, CA, USA, 19-25.

Burke, R., Mobasher, B., Williams, C. and Bhaumik, R. (2006a), "Classification features for attack detection in collaborative recommender systems," *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Philadelphia, PA, USA, 542-547.

Burke, R., Mobasher, B., Williams, C. and Bhaumik, R. (2006b), "Detecting profile injection attacks in collaborative recommender systems," *Proceedings of the 8th IEEE Conference on E-commerce Technology*, San Francisco, CA, USA, 23-30.

Burke, R., O'Mahony, M.P. and Hurley, N.J. (2011), "Robust collaborative recommendation," *Recommender Systems Handbook* (Ed: Ricci, F., Rokach, L., Shapira, B. and Kantor, P.B.), Springer US, New York, NY, USA, 805-835.

Canny, J. (2002a), "Collaborative filltering with privacy," *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Oakland, CA, USA, 45-57.

Canny, J. (2002b), "Collaborative filtering with privacy via factor analysis," *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Tampere, Finland, 238-245.

Cheng, Z. and Hurley, N.J. (2010a), "Robust collaborative recommendation by least trimmed squares matrix factorization," *Proceedings of the 22nd IEEE International Conference on Tools with Artificial Intelligence*, Arras, France, 105-112.

Cheng, Z. and Hurley, N.J. (2010b), "Robustness analysis of model-based collaborative filtering systems," *Lecture Notes in Computer Science*, **6206**, 3-15.

Chirita, P.-A., Nejdl, W. and Zamfir, C. (2005), "Preventing shilling attacks in online recommender systems," *Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management*, Bremen, Germany, 67-74.

Choi, K. and Suh, Y. (2013), "A new similarity function for selecting neighbors for each target item in collaborative filtering," *Knowledge-Based Systems*, **37**, 146-153.

Chung, C.-Y., Hsu, P.-Y. and Huang, S.-H. (2013), "βP: A novel approach to filter out malicious rating profiles from recommender systems," *Decision Support Systems*, **55** (1), 314-325.

Dellarocas, C. (2000), "Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior," *Proceedings of the 2nd ACM Conference on Electronic Commerce*, Minneapolis, MN, USA, 150-157.

Desrosiers, C. and Karypis, G. (2011), "A comprehensive survey of neighborhood-based recommendation methods," *Recommender Systems Handbook* (Ed: Ricci, F., Rokach, L., Shapira, B., and Kantor, P.B.), Springer US, New York, NY, USA, 107-144.

Dokoohaki, N., Kaleli, C., Polat, H. and Matskin, M. (2010), "Achieving optimal privacy in trust-aware social recommender systems," *Proceedings of the 2nd International Conference on Social Informatics*, Laxenburg, Austria, 62-79.

Ekstrand, M.D., Riedl, J.T. and Konstan, J.A. (2011), "Collaborative filtering recommender systems," *Foundations and Trends in Human-Computer Interaction*, **4** (2), 81-173.

García-Cumbreras, M.Á., Montejo-Ráez, A. and Díaz-Galiano, M.C. (2013), "Pessimists and optimists: Improving collaborative filtering through sentiment analysis," *Expert Systems with Applications*, **40** (17), 6758-6765.

Goldberg, D., Nichols, D., Oki, B.M. and Terry, D. (1992), "Using collaborative filtering to weave an information Tapestry," *Communication of the ACM*, **35** (12), 61-70.

Grčar, M. (2004), "User profiling: Collaborative filtering," *Proceedings of the 7th International Multiconference Information Society IS*, Ljubljana, Slovenia, 75-78.

Gunes, I. and Polat, H. (2015a). "Robustness analysis of privacy-preserving hybrid recommendation algorithm," *International Journal of Information Security Science*, **4** (1), 13-25.

Gunes, I. and Polat, H. (2015b). "Hierarchical clustering-based shilling attack detection in private environments," *Proceedings of the 3rd International Symposium on Digital Forensics and Security*, Ankara, Turkey.

Gunes, I., Bilge, A., Kaleli, C. and Polat, H. (2013a), "Shilling attacks against privacy-preserving collaborative filtering," *Journal of Advanced Management Science*, **1** (1), 54-60.

Gunes, I., Bilge, A. and Polat, H. (2013b), "Shilling attacks against memory-based privacy-preserving recommendation algorithms," *KSII Transactions on Internet and Information Systems*, **7** (5), 1272-1290.

Gunes, I., Kaleli, C., Bilge, A. and Polat, H. (2014), "Shilling attacks against recommender systems: A comprehensive survey," *Artificial Intelligence Review*, **42** (4), 767-799.

Han, J., Kamber, M. and Pei, J. (2011). *Data mining: Concepts and techniques*, Morgan Kaufmann.

Herlocker, J.L., Konstan, J.A., Borchers, A. and Riedl, J.T. (1999), "An algorithmic framework for performing collaborative filtering," *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, CA, USA, 230-237.

Herlocker, J.L., Konstan, J.A., Terveen, L.G. and Riedl, J.T. (2004), "Evaluating collaborative filtering recommender systems," *ACM Transactions on Information Systems*, **22** (1), 5-53.

Hill, W., Stead, L., Rosenstein, M. and Furnas, G. (1995), "Recommending and evaluating choices in a virtual community of use," *Proceedings of the*

SIGCHI Conference on Human Factors in Computing Systems, Denver, CO, USA, 194-201.

Hurley, N.J., O'Mahony, M.P. and Silvestre, G.C.M. (2007), "Attacking recommender systems: A cost-benefit analysis," *IEEE Intelligent Systems*, **22** (3), 64-68.

Johnson, S.C. (1967), "Hierarchical clustering schemes," *Psychometrika*, **32** (3), 241-254.

Kim, J., Kwon, E., Cho, Y. and Kang, S. (2011), "Recommendation system of IPTV TV program using ontology and *k*-means clustering," *Ubiquitous Computing and Multimedia Applications* (Ed: Kim,T.-h, Adeli, H, Robles, J.R. and Balitanas, M.), Springer Berlin Heidelberg, Berlin, Germany, 123-128.

Konstan, J.A., Miller, B.N., Maltz, D., Herlocker, J.L., Gordon, L.R. and Riedl, J.T. (1997), "GroupLens: Applying collaborative filtering to Usenet news," *Communications of the ACM*, **40** (3), 77-87.

Lam, S.K. and Riedl, J.T. (2004), "Shilling recommender systems for fun and profit," *Proceedings of the 13th International Conference on World Wide Web*, New York, NY, USA, 393-402.

Li, D., Lv, Q., Shang, L. and Gu, N. (2014), "Item-based top-*N* recommendation resilient to aggregated information revelation," *Knowledge-Based Systems*, **67**, 290-304.

Li, M. (2014), "Shilling attack detection algorithm based on non-random-missing mechanism," *International Journal of Security and Its Applications*, **8** (6), 115-126.

Lindell, Y. and Pinkas, B. (2002), "Privacy preserving data mining," *Journal of Cryptology*, **15** (3), 177-206.

Linden, G., Smith, B. and York, J. (2003), "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Computing*, **7** (1), 76-80.

Madhulatha, T.S. (2012), "An overview on clustering methods," *IOSR Journal of Engineering*, **2** (4), 719-725.

Manning, C.D., Raghavan, P. and Schütze, H. (2008), *Introduction to information retrieval*, Cambridge University Press, New York, NY, USA.

Marlin, B. (2004), *Collaborative filtering: A machine learning perspective*, Ph.D. Dissertation, University of Toronto, Canada.

Mehta, B. (2007), "Unsupervised shilling detection for collaborative filtering," *Proceedings of the 22nd National Conference on Artificial Intelligence*, Vancouver, Canada, 1402-1407.

Mehta, B. and Hofmann, T. (2008), "A survey of attack-resistant collaborative filtering algorithms," *Bulletin of the Technical Committee on Data Engineering*, **31** (2), 14-22.

Mehta, B., Hofmann, T. and Fankhauser, P. (2007a), "Lies and propaganda: Detecting spam users in collaborative filtering," *Proceedings of the 12th International Conference on Intelligent User Interfaces*, New York, NY, USA, 14-21.

Mehta, B., Hofmann, T. and Nejdl, W. (2007b), "Robust collaborative filtering," *Proceedings of the 2007 ACM Conference on Recommender Systems*, Minneapolis, MN, USA, 49-56.

Mehta, B. and Nejdl, W. (2008), "Attack resistant collaborative filtering," *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Singapore, 75-82.

Mehta, B. and Nejdl, W. (2009), "Unsupervised strategies for shilling detection and robust collaborative filtering," *User Modeling and User-Adapted Interaction*, **19** (1-2), 65-97.

Mobasher, B., Burke, R., Bhaumik, R. and Sandvig, J.J. (2007a), "Attacks and remedies in collaborative recommendation," *IEEE Intelligent Systems*, **22** (3), 56-63.

Mobasher, B., Burke, R., Bhaumik, R. and Williams, C. (2007b), "Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness," *ACM Transactions on Internet Technology*, **7** (4), 23-60.

Mobasher, B., Burke, R. and Sandvig, J.J. (2006a), "Model-based collaborative filtering as a defense against profile injection attacks," P*roceedings of the 21st National Conference on Artificial Intelligence*, Boston, MA, USA, 1388-1393.

Mobasher, B., Burke, R., Williams, C. and Bhaumik, R. (2006b), "Analysis and detection of segment-focused attacks against collaborative recommendation," *Lecture Notes in Computer Science*, **4198**, 96-118.

Noh, G., Kang, Y.-m., Oh, H. and Kim, C.-k. (2014). "Robust sybil attack defense with information level in online recommender systems," *Expert Systems with Applications*, **41** (4), 1781-1791.

O'Mahony, M.P., Hurley, N.J., Kushmerick, N. and Silvestre, G.C.M. (2004), "Collaborative recommendation: A robustness analysis," *ACM Transactions on Internet Technology*, **4** (4), 344-377.

O'Mahony, M.P., Hurley, N.J. and Silvestre, G.C.M. (2002a), "Promoting Recommendations: An attack on collaborative filtering," *Lecture Notes in Computer Science*, **2453**, 494-503.

O'Mahony, M.P., Hurley, N.J. and Silvestre, G.C.M. (2002b), "Towards robust collaborative filtering," *Lecture Notes in Computer Science*, **2464**, 87-94.

O'Mahony, M.P., Hurley, N.J. and Silvestre, G.C.M. (2005), "Recommender systems: Attack types and strategies," *Proceedings of the 20th National Conference on Artificial Intelligence*, Pittsburgh, PA, USA, 334-339.

O'Mahony, M.P., Hurley, N.J. and Silvestre, G.C.M. (2006), "Attacking recommender systems: The cost of promotion," *Proceedings of the Workshop on Recommender Systems, in Conjunction with the 17th European Conference on Artificial Intelligence*, Riva del Garda, Trentino, Italy, 24-28.

O'Mahony, M.P. (2004). *Towards robust and efficient automated collaborative filtering*, Ph.D. Dissertation, University College Dublin, Ireland.

Ortega, F., Sánchez, J.-L., Bobadilla, J. and Gutiérez, A. (2013), "Improving collaborative filtering-based recommender systems results using Pareto dominance," *Information Sciences*, **239**, 50-61.

Pennock, D.M., Horvitz, E., Lawrence, S. and Giles, C.L. (2000), "Collaborative filtering by personality diagnosis: A hybrid memory and model-based approach," *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, Stanford, CA, USA, 473-480.

Polat, H. (2006), *Privacy-preserving collaborative filtering*, Ph.D. Dissertation, SyracuseUniversity, Syracuse, NY, USA.

Polat, H. and Du, W. (2005a), "Privacy-preserving collaborative filtering," *International Journal of Electronic Commerce*, **9** (4), 9-35.

Polat, H. and Du, W. (2005b), "Privacy-preserving top-*N* recommendation on horizontally partitioned data," *Proceedings of the EEE/WIC/ACM International Conference on Web Intelligence*, Compiègne, France, 725-731.

Polat, H. and Du, W. (2005c), "SVD-based collaborative filtering with privacy," *Proceedings of the 2005 ACM Symposium on Applied Computing*, Santa Fe, NM, USA, 791-795.

Polat, H. and Du, W. (2006), "Achieving private recommendations using randomized response techniques," *Lecture Notes in Computer Science*, **3918**, 637-646.

Polat, H. and Du, W. (2007), "Effects of inconsistently masked data using RPT on CF with privacy," *Proceedings of the 2007 ACM Symposium on Applied Computing*, Seoul, Korea, 649-653.

Polezhaeva, E. (2011), "Incremental methods in collaborative filtering for ordinal data," *Lecture Notes in Computer Science*, **6744**, 452-457.

Ray, S. and Mahanti, A. (2009a), "Filler item strategies for shilling attacks against recommender systems," *Proceedings of the 42nd Hawaii International Conference on System Sciences*, Big Island, HI, USA, 1-10.

Ray, S. and Mahanti, A. (2009b), "Strategies for effective shilling attacks against recommender systems," *Lecture Notes in Computer Science*, **5456**, 111-125.

Renckes, S., Polat, H. and Oysal, Y. (2012), "A new hybrid recommendation algorithm with privacy," *Expert Systems*, **29** (1), 39-55.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J.T. (1994), "GroupLens: An open architecture for collaborative filtering of netnews," *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, Chapel Hill, NC, USA, 175-186.

Russell, S. and Yoon, V. (2008), "Applications of wavelet data reduction in a recommender system," *Expert System with Applications*, **34** (4), 2316-2325.

Sandvig, J.J., Mobasher, B. and Burke, R. (2008), "A survey of collaborative recommendation and the robustness of model-based algorithms," *IEEE Data Engineering Bulletin*, **31** (2), 3-13.

Sandvig, J.J., Mobasher, B. and Burke, R. (2007), "Robustness of collaborative recommendation based on association rule mining," *Proceedings of the 2007 ACM Conference on Recommender Systems*, Minneapolis, MN, USA, 105-112.

Sarwar, B.M., Karypis, G., Konstan, J.A. and Riedl, J.T. (2000a), "Analysis of recommendation algorithms for e-commerce," *Proceedings of the 2nd ACM Conference on Electronic Commerce*, Minneapolis, MN, USA, 158-167.

Sarwar, B.M., Karypis, G., Konstan, J.A and Riedl, J.T. (2002), "Recommender systems for large-scale e-commerce: Scalable neighborhood formation using clustering," *Proceedings of the 5th International Conference on Computer and Information Technology*, Dhaka, Bangladesh.

Sarwar, B.M., Karypis, G., Konstan, J.A. and Riedl, J.T. (2000b), "Application of dimensionality reduction in recommender system - A case study," *Proceedings of the ACM WebKDD Workshop*, Boston, MA, USA.

Sarwar, B.M., Karypis, G., Konstan, J.A, Riedl, J.T. (2001), "Item-based collaborative filltering recommendation algorithms," *Proceedings of the 10th International Conference on World Wide Web*, Hong Kong, China, 285-295.

Schafer, J.B., Frankowski, D., Herlocker, J.L. and Sen, S. (2007), "Collaborative filtering recommender systems," *Lecture Notes in Computer Science*, **4321**, 291-324.

Schafer, J.B., Konstan, J.A. and Riedl, J.T. (2001), "E-commerce recommendation applications," *Data Mining and Knowledge Discovery*, **5** (1-2), 115-153.

Sembiring, R.W., Zain, J.M. and Embong, A. (2011), "A comparative agglomerative hierarchical clustering method to cluster implemented course," *Journal of Computing*, **2** (12), 1-4.

Steinbach, M., Karypis, G. and Kumar, V. (2000), "A comparison of document clustering techniques," *Proceedings of the KDD Workshop on Text Mining*, Boston, MA, USA, 525-526.

Su, X. and Khoshgoftaar, T.M. (2009), "A survey of collaborative filtering techniques," *Advances in Artificial Intelligence*, **2009**, 2-21.

Troiano, L. and Díaz, I. (2014), "A model for preserving privacy in recommendation systems," *Information Processing and Management of Uncertainty in Knowledge-Based Systems* (Ed: Laurent, A., Strauss, O., Bouchon-Meunier, B. and Yager, R.), Springer International Publishing, Switzerland, 56-65.

Wen, J. and Zhou, W. (2012), "An improved item-based collaborative filtering algorithm based on clustering method," *Journal of Computational Information Systems*, **8** (2), 571-578.

Williams, C., Bhaumik, R., Burke, R. and Mobasher, B. (2006), "The impact of attack profile classification on the robustness of collaborative recommendation," *Proceedings of the 2006 WebKDD Workshop*, Philadelphia, PA, USA.

Williams, C., Mobasher, B. and Burke, R. (2007), "Defending recommender systems: Detection of profile injection attacks," *Service Oriented Computing and Applications*, **1** (3), 157-170.

Wu, J., Chen, L., Feng, Y., Zheng, Z., Zhou, M.C. and Wu, Z. (2013), "Predicting quality of service for selection by neighborhood-based collaborative filtering," *IEEE Transactions on Systems, Man, and Cybernetics*, **43** (2), 428-439.

Xue, G.-R., Lin, C., Yang, Q., Xi, W., Zeng, H.-J., Yu, Y. and Chen, Z. (2005), "Scalable collaborative filtering using cluster-based smoothing," *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Salvador, Brazil, 114-121.

Yu, K., Schwaighofer, A., Tresp, V., Xu, X. and Kriegel, H.-P. (2004), "Probabilistic memory-based collaborative filtering," *IEEE Transactions on Knowledge and Data Engineering*, **16** (1), 56-69.

Zhang, F. (2009a), "Average shilling attack against trust-based recommender systems," *Proceedings of the Information Management, Innovation Management and Industrial Engineering*, Xi'an, China, 588-591.

Zhang, F. (2009b), "Reverse bandwagon profile inject attack against recommender systems," *Proceedings of the 2nd International Symposium on Computational Intelligence and Design*, Changsha, China, 15-18.

Zhang, F. (2009c), "A survey of shilling attacks in collaborative filtering recommender systems," *Proceedings of the International Conference on Computational Intelligence and Software Engineering*, Wuhan, China, 1-4.

Zhang, F. (2011), "Analysis of bandwagon and average hybrid attack model against trust-based recommender systems," *Proceedings of the 5th International Conference on Management of E-commerce and E-government*, Hubei, China, 269-273.

Zhang, S., Ouyang, Y., Ford, J. and Makedon, F. (2006), "Analysis of a low-dimensional linear model under recommendation attacks," *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Seattle, WA, USA, 517-524.

Zhang, X.-L., Lee, T. and Pitsilis, G. (2013), "Securing recommender systems against shilling attacks using social-based clustering," *Journal of Computer Science and Technology*, **28** (4), 616-624.

Zhang, Z., Tang, X. and Chen, D. (2014), "Applying user-favorite-item-based similarity into slope one scheme for collaborative filtering," *Proceedings of the 2014 World Congress on Computing and Communication Technologies*, Tiruchirappalli, India, 5-7.

Zhang, F. and Zhou, Q. (2014), "HHT–SVM: An online method for detecting profile injection attacks in collaborative recommender systems," *Knowledge-Based Systems*, **65**, 96-105.

Zhuo, Z. and Kulkarni, S.R. (2014). "Detection of shilling attacks in recommender systems via spectral clustering," *Proceedings of the 17th International Conference on Information Fusion*, Salamanca, Spain, 1-8.