IMPROVING PERFORMANCE OF PRIVACY-PRESERVING COLLABORATIVE FILTERING SCHEMES

Alper BİLGE

PhD Dissertation

Graduate School of Sciences
Computer Engineering Program
April, 2013

ANADOLU ÜNİVERSİTESİ

ABSTRACT

PhD Dissertation

IMPROVING PERFORMANCE OF PRIVACY-PRESERVING COLLABORATIVE FILTERING SCHEMES

Alper BİLGE

Anadolu University
Graduate School of Sciences
Computer Engineering Program

Supervisor: Assoc. Prof. Dr. Hüseyin POLAT
2013, 117 pages

Privacy-preserving collaborative filtering methods offer useful filtering skills without deeply jeopardizing individual privacy. However, they mostly suffer from accuracy, scalability, and sparseness problems. Applying privacy measures to conceal confidential data in recommendation systems causes a bias in collected data, which might make accuracy worse. As the content in recommendation domain proliferates, the size of collected data expands rapidly, which aggravates scalability challenge of those systems. In addition, since users are typically able to rate a small fraction of existing products, sparseness of collected data becomes an issue.

In this dissertation, various preprocessing methods are proposed to overcome accuracy, scalability, and sparseness challenges faced by various privacy-preserving collaborative filtering systems. Through application of the proposed preprocessing techniques like item ordering and elimination, clustering, dimensionality reduction, user profiling, profile cloning, and son on, novel privacy-preserving collaborative filtering schemes are cultivated. Essentially, the proposed enhanced systems focus on producing accurate predictions while coping with constantly growing nature of collections without jeopardizing individual privacy. The proposed schemes are analyzed in terms of privacy and overhead costs. Also, real data-based experiments are performed to scrutinize their effects on accuracy, scalability, and privacy. The analysis and experimental outcomes demonstrate that the methods preserve individual privacy and offer adequately accurate recommendations in scalable amount of time.

Keywords: Preprocessing; Privacy; Scalability; Accuracy; Sparsity; Collaborative filtering.

ÖZET

Doktora Tezi

GİZLİLİK TEMELLİ ORTAK SÜZGEÇLEME YÖNTEMLERİNİN BAŞARIMININ İYİLEŞTİRİLMESİ

Alper BİLGE

Anadolu Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Doç. Dr. Hüseyin POLAT
2013, 117 sayfa

Gizliliği koruyan ortak süzgeçleme yöntemleri bireylerin gizliliklerini tehlikeye atmadan yararlı süzgeçleme becerileri ortaya koymaktadır. Ancak bu sistemler doğruluk, ölçeklenebilirlik ve boşluklu veri sorunlarıyla karşı karşıyadır. Gizli kalması gereken tercihlerin saklı tutulması için uygulanan gizlilik ölçütleri, toplanan veride bozulmaya yol açar ve dolayısıyla gizliliği koruyan ortak süzgeçleme sistemlerinin doğruluğuna zarar verebilir. Öneri alanındaki içerik genişledikçe toplanan verinin boyutları hızlı biçimde büyür ve sistemlerin ölçeklenebilirlik sorunlarını daha da zorlaştırır. Ek olarak, kullanıcılar mevcut ürünlerin genelde küçük bir yüzdesine tercih belirtebildiklerinden dolayı derlenen verinin boşluklu yapısı bir sorun haline gelmektedir.

Bu tezde gizliliği koruyan ortak süzgeçleme sistemlerinin karşılaştığı doğruluk, ölçeklenebilirlik ve boşluklu veri sorunlarınının üstesinden gelmek üzere çeşitli önişleme yöntemleri önerilmiştir. Ürün sıralama ve eleme, kümeleme, boyut indirgeme, kullanıcı ayrımlama, profil klonlama vb. gibi önerilen önişleme yöntemlerinim uygulanmasıyla yeni gizliliği koruyan ortak süzgeçleme şemaları geliştirilmiştir. Önerilen önişleme ile iyileştirilmiş şemalar kişilerin gizliliğini tehlikeye sokmadan, verinin sürekli genişleyen yapısıyla başa çıkabilmek ve yeterli doğrulukla öneriler üretmek üzerine odaklanmıştır. Önerilen taslaklar, sağlanan gizlilik ve ortaya çıkan ek yükler açısından analiz edilmiştir. Ayrıca gerçek veri tabanlı deneyler yapılarak, bu taslakların doğruluk, ölçeklenebilirlik ve gizliliğe etkileri ölçülmüştür. Analizler ve deneysel sonuçlar yöntemlerin gizliliği koruduğunu ve ölçeklenebilir zaman dilimleri içinde yeterli doğrulukla öneriler ürettiğini göstermiştir.

Anahtar Kelimeler: Önişleme; Gizlilik; Ölçeklenebilirlik; Doğruluk; Boşluklu veri; Ortak süzgeçleme.

ACKNOWLEDGMENTS

I would like to express my sincerest gratitude to my supervisor, Assoc. Prof. Dr. Hüseyin Polat, who has supported me throughout my dissertation with his patience and unsurpassed knowledge. I thank him for his guidance and encouragement, where good advice, support, and friendship has been invaluable on both academic and personal levels.

Also, I would like to thank Prof. Dr. Yaşar Hoşcan, Assoc. Prof. Dr. Atakan Doğan, Assoc. Prof. Dr. Osman Abul, and Assist. Prof. Dr. Serkan Günal on my dissertation committee for their valuable contributions.

I gratefully acknowledge funding I received from Turkish Scientific and Technical Research Council (TÜBİTAK) under the Grant 108E221.

I would like to thank my research friends, Assist. Prof. Dr. Cihan Kaleli, Dr. İbrahim Yakut, and Serhan Gürmeriç for their scientific support.

Finally, I would like to express my eternal gratitude to my parents, my brother Alkan Bilge, and my dear Bengi Nar for their everlasting love and support. This dissertation would not have been possible without them.

Alper Bilge

April, 2013

Contents

List of Tables

List of Figures

ABBREVIATIONS

a	Active user
BDT	Binary decision tree
CA	Classification accuracy
CBF	Content-based filtering
CF	Collaborative filtering
DWT	Discrete wavelet transform
F1	F-measure
FBP	Feature-based profile
FCM	Fuzzy c-means clustering
KMC	k-means clustering
MA	Multiple arrangement
MAE	Mean absolute error
ML	MovieLens data set
MLM	MovieLens million data set
MLP	MovieLens public data set
NBC	Naïve Bayesian classifier
NF	Netflix prize data set
PBP	Purchase-based profile
PCA	Principal component analysis
PCC	Pearson's correlation coefficient
PPCF	Privacy-preserving collaborative filtering

PPNBC	Privacy-preserving NBC-based CF
PSP	Pseudo-self-prediction
RBP	Rating-based profile
RPTs	Randomized perturbation techniques
RRTs	Randomized response techniques
q	Target item
SA	Single arrangement
SOM	Self-organizing map
SVD	Singular value decomposition
T	Total elapsed time
U	User rating data
U'	Disguised user rating data

1. INTRODUCTION

From the beginning of the Industrial Age, lives of people have been constantly changing. On one hand, modernization helps people reach data easily. On the other hand, amount of data augments rapidly, which confuses people and causes anxiety. Rapid integration of the Internet services into daily lives has attracted people to use online shopping amenities rather than discretionary shopping (McLaughlin, 2003). However, as such services pervade, it also leads to a difficulty for people to make decisions caused by the presence of too much information, called *information overload* problem or *infobesity*.

Recommender systems are emerging tools to deal with infobesity by providing personalized predictions to help customers find entities they might like. Such recommendations might be about products like movies, music CDs, books, news, images, web pages, research papers, etc. or social elements like events, people, or groups (Herlocker et al., 2004). There has been a number of ways to produce automated predictions including content-based, collaborative, or knowledge-based techniques (Burke, 2002; Melville et al., 2002; Ziqiang and Boqin, 2004).

1.1 Collaborative Filtering

Within a menu of many offered dishes, people tend to get recommendations from the waiters and the cooks about the cooking styles and tastes to have delicious nourishment. Among several places available to visit, they again ask for endorsement of travel guides; and read comments and critics before buying a new book and/or a music CD. When there are too many choices to select one of them, the very nature of human being is to get a recommendation from those who have an idea about them. As the subset of available products

expands over the Internet, e-commerce companies come up with prediction generation abilities relying on previous customers' opinions on clicked and/or purchased items. Collaborative filtering (CF) is one such approach utilized for recommendation purposes, where the term was first coined by the Tapestry system serving as an e-mail filtering system (Goldberg et al., 1992).

CF filters out irrelevant content and/or ranks items to be evaluated relying on wisdom of crowds (Surowiecki, 2004) and law of large numbers principles (Van Roy and Yan, 2010). Based on the assumption that users having similar tastes in the past are tend to agree in the future, CF systems primarily attempt to predict a newbie's tastes for available products operating on the collection of previous users' rating information. There are several studies demonstrating the prosperity and the efficiency of CF systems on impressing people with successful referrals (Bobadilla et al., 2011; Symeonidis et al., 2008; Chen et al., 2009; Bilge and Polat, 2012). CF has also been deployed by many online services such as Amazon.com, YouTube, Last.fm, and so on to render online services more entertaining and boost sales (Linden et al., 2003; Cechinel et al., 2013). CF systems rely on a database, which is typically very large and sparse, containing preference information by users about items to estimate future predictions on requested products. Such database, commonly referred to as the *user-item matrix*, consists of collected ratings from n users on m products.

Memory-based, model-based, and hybrid schemes are three classes of CF algorithms (Breese et al., 1998; Al-Shamri and Bharadwaj, 2008; Herlocker et al., 2004). Memory-based algorithms typically operate on the entire data collection to produce recommendations. Model-based schemes, on the other hand, operate on a prototype derived from the original user-item matrix. Although models are helpful in practice, it is relatively hard to fine-tune their parameters and they often come with the cost of accuracy (Xue et al., 2005). Hybrid schemes utilize both schemes for improved performance. Su

and Khoshgoftaar (2009) present a detailed survey about the most common CF techniques.

Most practical deployments of CF schemes are memory-based implementations in which similarities are calculated among all pairwise entities (users or items) using a variety of similarity metrics such as Pearson's correlation coefficient (PCC), cosine similarity, distance-based similarity, or concordance (Sarwar et al., 2000; Lathia et al., 2007; Choi and Suh, 2013). Online vendors marketing over a diverse variety of products mostly prefer item-based methods (Linden et al., 2003), which performs relatively faster. Conversely, others prefer user-based systems to obtain better accuracy (Herlocker et al., 2004). In order to operate through binary ratings, Miyahara and Pazzani (2000) propose an algorithm based on naïve Bayesian classifier (NBC). Such systems are simple and they do not have complicated parameters to tune.

Model-based CF approaches depend on training data and utilize a learning model. They usually require considerably large computation power to tune model parameters. Such methods are proposed to handle shortcomings of memory-based implementations like scalability and sparseness. Model-based algorithms mostly utilize various data mining techniques such as clustering (Chen et al., 2009; Zhang and Chang, 2006), dimensionality reduction (Vozalis et al., 2010), decision trees (Breese et al., 1998), Bayesian classifiers (Miyahara and Pazzani, 2002), and association rule mining (Shyu et al., 2005) to simplify and compact original database. Such produced prototypes need to be updated periodically (Breese et al., 1998; Herlocker et al., 2004). Dimensionality reduction techniques project data into a narrower dimension handling sparseness and scalability problems; however, they generally suffer from loss of potential useful information (Xue et al., 2005; Russell and Yoon, 2008). Conventional clustering approaches, on the other hand, are preferable in relieving scalability to some extent without sacrificing accuracy much (Zhang and Chang, 2006).

Researchers propose to construct hybrid schemes to combine advantages of both memory- and model-based techniques. Pennock et al. (2000) performs a personality diagnosis analysis prior to prediction estimation. Chen et al. (2009) propose applying a clustering-based approach through nonnegative tri-factorization method along with user- and item-based methods. Russell and Yoon (2008) propose a wavelet data reduction-based model construction to reduce number of items. Jeong et al. (2009) propose a novel iterative semi-explicit rating method, which aggregates neighbor ratings and extrapolates unrated elements in a semi-supervised manner to obtain a dense matrix.

Unlike CF schemes, there are content-based filtering (CBF) systems used to provide predictions through analysis of contextual information of documents, URLs, web-logs, product descriptions, and comments of users on their purchases (Pazzani, 1999). Melville et al. (2002) introduce a content-boosted CF algorithm to improve accuracy. In addition to CF and CBF systems, some approaches also take product contents into account and propose hybrid schemes combining CBF along with CF. Although combining CBF with CF enhances recommendation accuracy, such hybrid systems require a complex implementation (Burke, 2002; Pazzani, 1999).

1.2 Privacy-Preserving Collaborative Filtering

Privacy is a variable term and it is very hard to define it succinctly. The exact definition and borders of this concept is still discussed by diverse communities. According to Westin (1968), it can be defined as "the claim of individuals, groups or institutions to determine for themselves when, how, and to what extent information about them is communicated to others." However, emerging collaborative recommendation service technologies threaten this right. In order to benefit from recommendation services, a user must provide her personal data to such systems. Such data might be used to disturb customers via

spam e-mails and phone calls for unsolicited marketing. Other examples of privacy risks might be price discrimination, profiling users, being subject to government surveillance, data transferring, and so on (Ackerman et al., 1999; Cranor, 2003; Jensen et al., 2005; Calandrino et al., 2011).

Rating values and rated and/or unrated items can be considered as confidential data in CF. Due to the increasing popularity of protecting confidential data, privacy issues have been receiving increasing attention in CF community. People become more conscious about privacy risks and avoid using such services even if they get benefit in return (Berkovsky et al., 2007). Hence, estimating accurate predictions without violating privacy becomes important (Canny, 2002a; Kaleli and Polat, 2012). Various privacy-preserving collaborative filtering (PPCF) methods have been utilized to provide CF tasks while preserving confidentiality. The most widely used methods are randomization, encryption, anonymization, aggregation, agent-based methods, and so on.

First approaches to build privacy measures on CF applications are distributed solutions by Canny (2002a,b), relying on formation of an aggregate data using cryptographic techniques to hide confidential data in distributed environments. Unlike distributed schemes, central server-based applications are more popular, where individuals submit their preferences after perturbing their confidential data up to a level for concealing their actual ratings and rated products. Data perturbation methods like randomized perturbation techniques (RPTs), randomized response techniques (RRTs), and data substitution are employed to perform filtering processes without violating privacy.

Polat and Du (2003, 2005a,b) propose to implement RPTs in order to allow customers control their privacy by employing individual level of cautions rather than joining an aggregate. They investigate providing private predictions using a memory-based CF algorithm (Polat and Du, 2003). They

also apply RPTs to achieve confidentiality in singular value decomposition (SVD)-based CF scheme, where scalability issue is addressed by reducing dimensionality (Polat and Du, 2005b). RRTs are utilized in a similar way to conceal binary rating-based preferences (Polat and Du, 2006). Like perturbation, data substitution methods are proposed to preserve individual privacy, which completely replaces individual votes (Berkovsky et al., 2012).

Although data obfuscation methods prevent data holders from disclosing confidential data, they inevitably come with reasonable loss of accuracy (Polat and Du, 2005a). Since they are more practical to assure individual user privacy against central servers, such methods have been employed extensively. Different from data masking and disguising techniques, Wim F. J. Verhaegh (2004) propose to utilize encryption techniques in order to achieve individual privacy in memory-based central server CF algorithms. They describe how to measure similarities and estimate recommendations on encrypted profiles.

1.3 Related Work

CF is the most popular and widely used information filtering method in e-commerce. In addition, providing privacy measures helps them become more widespread and more reputable. However, there are some common limitations of CF and PPCF schemes. Such limitations shall be overcome to produce qualified predictions. In this section, major challenges of CF and PPCF systems are listed and previous studies proposed to alleviate them are examined.

1.3.1 Scalability

Typically, CF and PPCF systems operate over large user-item matrices. Due to continually enlarging nature of e-commerce facilities, the number of users and/or products tend to be quite large in online shopping databases (often in terms of millions) (Chen et al., 2009). Complexity of online prediction estimation process is related to both dimensions of user-item matrix. Consequently,

the amount of time required to calculate similarities among users increases exponentially as the user-item matrix grows causing scalability problems (Zhang and Chang, 2006). CF systems are supposed to respond many customers at any given time; therefore, it becomes a challenge to offer online predictions in a reasonable amount of time. Thus, CF systems are required to become more robust against the grow in size of data to overcome such problems.

The most widely adopted approach to overcome scalability problem in CF schemes is dimensionality reduction in which the dimension of users and/or items are decreased to improve scalability consequently. Zhang et al. (2005) propose using SVD as a dimensionality reduction method and Vozalis and Margaritis (2007) implement demographic data along with SVD for enhancing scalability. Other approaches of dimensionality reduction employed to solve scalability are principal component analysis (PCA) and correspondence analysis (Kim and Yum, 2005; Vozalis et al., 2009). Another approach on scalability problem is clustering in which neighborhood selection is performed according to previously clustered groups of customers. One example of these approaches combines case-based reasoning with self organizing map (SOM) clustering (Roh et al., 2003) and another example utilizes smoothing-based clustering (Xue et al., 2005). However, these approaches are computationally expensive limiting the benefits of data reduction to be handled. Also, they tend to be ineffective with extremely large data sets. To overcome pitfalls of such approaches, Bilge et al. (2012) propose to employ a recursive clustering-based layout, which performs considerably robust against growth of data. Alternatively, Russell and Yoon (2008) propose discrete wavelet transform (DWT) to reduce data efficiently and increase performance in large data sets without compromising accuracy. In addition, Chen et al. (2009) propose to use orthogonal nonnegative matrix tri-factorization and Jeong et al. (2009) propose a novel iterative semi-explicit rating method, which aggregates neighbor ratings

and extrapolates unrated elements in a semi-supervised manner to obtain a dense preference matrix. Honda et al. (2001) offer to employ PCA and fuzzy clustering simultaneously, where they extract local principal components by using lower rank approximation of the data matrix and predict the missing values as an approximation. Also, genetic algorithms-based clustering methods are proposed by (Zhang and Chang, 2006; Georgiou and Tsapatsoulis, 2010; Bobadilla et al., 2011).

In order to relieve scalability issues in PPCF framework, Polat and Du (2005b) apply SVD-based data reduction, where RPTs are employed to achieve individual privacy. Subsequently, Yakut and Polat (2007) build privacy in Eigentaste-based CF scheme, which relies on a constant time recommendation algorithm using RPT-based approach. Tada et al. (2010) address the scalability problem of PPCF schemes by replacing similarity between users with similarity between items. Similar to non-private schemes, Bilge and Polat (2013a) employ clustering algorithms onto privacy-preserving framework to obtain scalable PPCF schemes. An item-based algorithm is presented to provide private predictions in binary data configurations utilizing RRTs (Polat and Du, 2006). When the users' preferences are represented with binary ratings, NBC-based CF scheme can be utilized while preserving privacy (Kaleli and Polat, 2007). However, due to privacy-preserving measures, overall performance of the scheme degrades. In order to enhance its performance using some preprocessing techniques, additional schemes have been proposed by Kaleli and Polat (2009) and Bilge and Polat (2010).

1.3.2 Sparsity

As explained in Section 1.3.1, there are too much items in a typical CF and/or PPCF database. However, users are only able to rate a very small fraction of such products. Therefore, resulting collections are typically highly sparse

(often with less than 5% density) (Acilar and Arslan, 2009; Bilge and Polat, 2013a). Hence, it gets more unlikely to find co-rated items between users and it gets much harder to determine accurate neighborhoods because the similarities can only be calculated through overlapping ratings (Jeong et al., 2009). Furthermore, even if the similarities can be accurately determined, it is likely that most of the neighbors might not have a rating on the item for which the prediction will be estimated. Thus, it renders the effort of finding similar users pointless, which affects quality of predictions adversely.

There are also other challenges caused by sparse structure of databases. *Cold start* problem refers to system's disability of producing predictions to new users having a limited number of ratings due to lack of information to calculate correlations with existing users (Ahn, 2008). The same condition also applies to the products, where the system is not able to produce a prediction for a given newly inserted item due to limited number of ratings on it, called *coverage* problem (Sarwar et al., 2001).

The success of CF and PPCF algorithms mainly depends on data sparsity and size of the matrix they operate on. The performance of any CF and/or PPCF approach immediately decreases as data get sparse, which habitually happens in most web applications. Researchers have long been studying on this issue and they have made a significant progress to handle it. Examples of research focusing on alleviating sparseness issues in CF applications include using artificial immune networks (Acilar and Arslan, 2009), factorization of user-item matrix via orthogonality properties (Chen et al., 2009), employing a random walk recommender (Yildirim and Krishnamoorthy, 2008), utilizing a hybrid approach consisting of both user- and item-based CF by defining a similarity weight (Liang et al., 2008), applying back-propagation neural networks to predict values of null ratings on users whose non-null ratings intersect the most (Zhang and you Chang, 2005), and a novel and efficient algorithm

to effectively predict missing ratings by setting a similarity threshold for users and items (Ma et al., 2007).

In order to lessen the effects of data sparsity in PPCF framework, Bilge and Polat (2011) propose a profiling scheme. The proposed profiling method deals with both the sparseness and the scalability problems by projecting large and sparse user vectors onto compact and dense item features-based profiles. Such profiling approach is also utilized to recover from sparseness side effects (Bilge and Polat, 2013a).

1.3.3 Accuracy

CF and PPCF systems either provide predictions for single items or top-N recommendation lists. Accuracy of produced recommendations is important for recommendation frameworks because accuracy level defines reputability of such systems. Qualified predictions should not exceed a narrow boundary of error range to make customers feel comfortable to follow such guidance. Also, top-N recommendation lists should not yield to especially false positive referrals, which might lead angry customers and damage system's credibility.

Various techniques have been utilized to enhance the quality of the predictions. The most successful implementations are memory-based solutions in which whole database is utilized supporting wisdom of crowds principle. Bogdanova and Georgieva (2008) propose an algorithm based on discovering the functional error-correcting dependencies in a data set by using the fractal dimension for tackling accuracy problem. Runran Liu (2009) propose a modified CF method computing the similarity between congeneric nodes in bipartite networks substituting standard cosine similarity and achieve a significant improvement of algorithmic accuracy. Jeong et al. (2010) propose a similarity update method that uses an iterative message passing procedure and an accuracy metric in order to minimize the predictive accuracy error and to

evenly distribute predicted ratings over true rating scales. Kim et al. (2010) perform collaborative tagging, which is employed as an approach to grasp and filter users' preferences for items and Lee et al. (2010) propose a CF recommendation approach based on both implicit ratings and less ambitious ordinal scales for mobile music recommendations. Recently, Choi and Suh (2013) propose a new similarity function to improve quality of predictions by ranking item votes according to their similarity to the target item.

In order to enhance both accuracy and online performance of PPCF schemes, Bilge and Polat (2012) discuss how to achieve predictions on reduced space using DWT without violating individual users' privacy. The same authors also employ a modified similarity function along with an item elimination preprocessing scheme on memory-based PPCF schemes to obtain better quality referrals in scalable amount of time (Bilge and Polat, 2013c). Also, Renckes et al. (2012) propose a new recommendation algorithm, which is a hybrid algorithm, to generate truthful referrals efficiently. Recently, Bilge and Polat (2013b) propose a bisecting k-means clustering PPCF approach to obtain referrals with comparable accuracy to non-private schemes even when individual privacy is preserved.

1.4 Scope and Contributions

This dissertation focuses on improving overall performance of PPCF schemes with respect to robustness against data sparsity and dimensions, quality of predictions, and online performance. In order to achieve such enhancements, novel preprocessing schemes are proposed onto either previously proposed or newly developed privacy-preserving recommendation schemes. If suggested preprocessing schemes have not been implemented in non-private schemes before, they are also evaluated in traditional CF schemes without privacy along with PPCF schemes. In addition, state-of-the-art privacy-preservation mechanisms

are investigated in detail and novel information theory-based evaluations are performed to assess their extent in provided privacy levels. Main contributions of the dissertation can be summarized in the following.

A detailed privacy analysis of the state-of-the-art PPCF recommendation scheme is performed by means of privacy protection parameters. In addition to theoretical analysis, various experiments are performed on traditional PPCF schemes to present effects of varying privacy parameters on accuracy.

Memory-based CF and PPCF schemes are addressed in terms of both accuracy and scalability. A formerly proposed target item-based similarity modification function is investigated in privacy-preserving environment. Moreover, a preprocessing method to eliminate relatively dissimilar items from prediction process is proposed over the similarity function to alleviate scalability challenges (Bilge and Polat, 2013c).

Model-based CF and PPCF schemes are analyzed and DWT-based transformations are applied on PPCF schemes to improve scalability. Also, an item ordering preprocessing is proposed to improve quality of predictions before applying DWT in privacy-preserving layout (Bilge and Polat, 2012). As another model-based approach, a novel content-based profiling method utilizing categorical information of items to alleviate sparsity-related problems is proposed. Applicability of both conventional and soft non-hierarchical clustering techniques (k-means, fuzzy c-means, and SOM) to the PPCF framework to overcome scalability issues is discussed in detail. Additionally, a comparison among utilized clustering techniques is provided. This research presents the first analysis and evaluation on integrating uncertainty-based soft computing constituents on PPCF framework (Bilge and Polat, 2013a). Also, a novel PPCF framework based on bisecting k-means clustering is proposed. A two-level preprocessing scheme is suggested to deal with scalability and accuracy

problems of PPCF in general. Effects of scalability and sparseness challenges are alleviated considerably and significantly higher prediction accuracy is obtained compared to the traditional methods (Bilge and Polat, 2013b).

Binary ratings-based systems are investigated in terms of scalability issues. Online performance of privacy-preserving NBC-based CF scheme (PPNBC) proposed by (Kaleli and Polat, 2007) is enhanced without greatly compromising accuracy of individuals by two preprocessing methods. NBC-based CF algorithm's ability to estimate predictions from a small amount of training data is utilized (Bilge and Polat, 2010).

1.5 Organization of the Dissertation

The rest of the dissertation is organized, as follows: In Chapter 2, general background and preliminaries about CF and PPCF are explained. In Chapter 3, detailed privacy analysis of the state-of-the-art PPCF scheme by means of privacy protection parameters are discussed. Also, effects of varying privacy parameters on accuracy are presented. Chapter 4 focuses on memory-based CF and PPCF schemes and proposes an item elimination preprocessing to alleviate scalability challenges. In Chapter 5, DWT-based transformations are applied on PPCF schemes to improve scalability and a preprocessing is proposed to obtain better quality of predictions. In Chapter 6, clustering methods are employed on PPCF schemes. Chapter 7 proposes a novel PPCF framework based on bisecting k-means clustering and Chapter 8 addresses scalability issues of binary ratings-based systems. Finally, in Chapter 9, concluding remarks and recommendations for further research are discussed.

2. PRELIMINARIES

In this chapter, general background and preliminaries on CF and PPCF are explained. Recommendation systems are described and CF prediction estimation algorithm is defined. Afterwards, randomization-based individual privacy-protection mechanisms are discussed. Data perturbation protocols for numerical and binary ratings-based PPCF systems are investigated. Then, PPCF prediction estimation process is introduced and formulated. Finally, real data sets and evaluation metrics used in the experiments are described.

2.1 Prediction Estimation

CF systems collect ratings and form a user-item matrix, $U_{n\times m}$, containing preference information from n users on m items. During an online interaction with a CF system, an active user (a) requests a prediction for a target item (q) after sending her available ratings. CF prediction estimation can be thought as a two-step process: (i) locating neighbors by computing similarities between a and all other users in the system and (ii) estimating a prediction as a weighted average based on preferences of neighbors on q. Such similarities between a and any user u are calculated using various methods. According to results presented in (Choi and Suh, 2013), the best similarity measure is PCC, which is given in Eq. 2.1.

$$w_{au} = \frac{\sum_{i=1}^{m'}[(v_{ai} - \overline{v_a})(v_{ui} - \overline{v_u})]}{\sqrt{\sum_{i=1}^{m'}(v_{ai} - \overline{v_a})^2}\sqrt{\sum_{i=1}^{m'}(v_{ui} - \overline{v_u})^2}}, \tag{2.1}$$

where v_{ai} and v_{ui} are the votes for item i by users a and u, respectively. Similarly, $\overline{v_a}$ and $\overline{v_u}$ are the average votes of users a and u, respectively, and m' is the number of co-rated items by both a and u. Such similarity weight is utilized in prediction estimation process as will later be explained in Eqs. 2.2

and 2.5. After calculating similarities, the most similar k users are marked as neighbors (Herlocker et al., 2004). A prediction for a on q, referred to as p_{aq}, is produced as a weighted average of neighbors' ratings on q using the formula given in Eq. 2.2.

$$PCC_{aq} = \overline{v_a} + \frac{\sum_{u=1}^{k}[(v_{uq} - \overline{v_u}) \times w_{au}]}{\sum_{u=1}^{k} w_{au}}, \tag{2.2}$$

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

2.2 Privacy Protection by Randomization

High quality predictions can only be produced from authentic data. However, customers often hesitate to submit their true preferences due to privacy concerns. Hence, the goal is to provide accurate predictions by guaranteeing confidentiality. Privacy-preserving schemes generally require a level of distortion in user profiles. Accordingly, accuracy losses are inevitable. Therefore, privacy parameters must be well-tuned as not allowing the server to extract any valuable information from user profiles and still be able to produce accurate predictions. Randomization methods can be utilized to mask numerical and binary ratings in such a way so that precise recommendations can be provided without violating privacy.

2.2.1 Disguising numerical ratings

PPCF has two key aspects in privacy perspective (Polat and Du, 2005a): (i) hiding individual preferences and (ii) concealing the list of rated items. Disclosure of the actual preferences might cause privacy violations such as profiling and price discrimination; and revelation of the rated items list can be abused to achieve unsolicited marketing. Primary approach for preserving individual privacy is to disguise personal data by randomly perturbing each vote in the profile and randomly filling some fraction of the empty cells (Polat and Du,

2005a). RPTs are useful for applying a preferred level of distortion on data to provide proper privacy intervals by obstructing disclosure of individual data items.

In terms of PPCF, RPTs offer to disguise a vote entry v by replacing it with $v + r$, where r is a random number drawn from either a uniform or Gaussian distribution. In Gaussian distribution, random numbers are generated with zero mean ($\mu = 0$) and standard deviation (σ) while in uniform distribution, random numbers are generated over the range $[-\alpha, +\alpha]$, where α is a constant and $\sqrt{3}\sigma$. Also, users insert additional random numbers to selectively or uniformly randomly chosen β percent of empty cells as fake ratings. The values of σ and β control privacy and accuracy levels. Their values can be determined based on the values of σ_{max} and β_{max}, respectively. After the data holder sets σ_{max} and β_{max}, each user determines individual σ and β values randomly from $(0, \sigma_{max}]$ and $(0, \beta_{max}]$ intervals, respectively. Users can perturb their vectors, as described in Protocol 1 (Polat and Du, 2005a).

Procedure 1 Perturbation Protocol for Numeric Data

Input: User vector $(u_{1\times m})$, σ_{max}, β_{max}

Estimate z-scores ($\rightarrow Z$):

1: $\overline{u} \leftarrow \text{MEAN}(u)$; $\sigma_u \leftarrow \text{STD}(u)$
2: **for all items** in u ($j \leftarrow 1$ to m) **do**
3: $\quad z_j = (u_j - \overline{u})/\sigma_u$
4: **end for**

Determine privacy parameters:

5: $\beta \leftarrow \text{RND}(0, \beta_{max})$; $\sigma \leftarrow \text{RND}(0, \sigma_{max})$; $\alpha \leftarrow \sqrt{3}\sigma$
6: $e \leftarrow$ # of empty cells; $g \leftarrow$ # of genuine ratings
7: $F \leftarrow e \times \beta\%$ $\quad \triangleright$ # of empty cells to be filled

Select distribution & generate random numbers:

8: $dist \leftarrow \text{RANDOM}(\text{uniform, Gaussian})$
9: $R \leftarrow dist(g + F; \mu = 0, \sigma|\alpha)$

Disguise z-scores ($\rightarrow Z'$):

10: **for all items** in u ($j \leftarrow 1$ to m) **do**
11: $\quad z_j^{'} = (z_j + R_j)$
12: **end for**
13: **return** Z'

After data disguising, users send their disguised vectors rather than true ratings vector to the data holder, which creates a disguised user-item matrix, $U'_{n\times m}$, and estimates predictions based on it. Hereafter, each active user a disguises her private data similarly and sends masked data along with a query to the server in order to get a prediction.

2.2.2 Disguising binary ratings

In binary ratings-based PPCF systems, RRTs are useful for masking individual rating entries. RRTs were first introduced by Warner (1965) as a research method to estimate the percentage of people in a population that has any particular attribute. They allow respondents to respond to sensitive issues (such as criminal behavior or sexuality) while maintaining confidentiality. The interviewer asks two questions to each respondent for which the answers are opposite to each other. Respondents choose the first question with probability θ and the second question with probability $1 - \theta$ to answer. The interviewer learns responses but does not know which question is answered.

In PPCF schemes, sensitive questions are whether a purchased product is liked or disliked by the user. An example binary ratings vector can be written as $u_{1\times m} = [11\perp 00\perp 101]$, where $\perp$ refers to unrated item. To mask ratings, u is first divided into M vectors $(u_1, u_2, \ldots, u_M)$, where $M \ll m$, and M random numbers $(r_{1\times M})$ are generated using uniform distribution over the range [0, 1]. Then, user sends either true or negated values of u_i $(i = 1, 2, \ldots, M)$ vectors to the central server. After the data holder sets θ and M, users can mask their vectors, as described in Protocol 2 (Kaleli and Polat, 2007).

After data disguising, users send their masked vectors to the data holder, which creates a masked user-item matrix and estimates predictions based on it. Hereafter, each active user a masks her vector similarly and sends masked data along with a query to the server in order to get a prediction.

Procedure 2 Perturbation Protocol for Binary Data

Input: User vector $(u_{1\times m})$, θ, M

Initialize masked vector:

$u' \leftarrow null$

Divide vector into M chunks & mask ratings:

$chunkSize \leftarrow m/M$

for all groups of u ($i \leftarrow 1$ to M) **do**

  $u_i \leftarrow u[(i-1) \times chunkSize : i \times chunkSize]$

  $r_i \leftarrow \mathcal{U}(0,1)$

  if $r_i > \theta$ **then** $u'_i \leftarrow \neg u_i$ ▷ chunk negated

  else $u'_i \leftarrow u_i$

  end if

end for

$u' = [u'_1 : u'_2 : \cdots : u'_M]$

return u'

2.3 Estimating Predictions with Privacy

Due to privacy concerns, users prefer to submit their disguised vectors instead of explicit expressions, as explained in Section 2.2.1. Therefore, the central server needs to estimate predictions based on such disguised collections with decent accuracy. The PCC (Eq. 2.1) is modified to incorporate an item-variance weight factor by Herlocker et al. (1999). The modified version is represented as the covariance of two users' rating vectors, which consist of z-scores, as shown in Eq. 2.3.

$$w_{au} = \frac{\sum_{i=1}^{m} z_{ai} \times z_{ui}}{m}. \tag{2.3}$$

Since PPCF schemes collect disguised z-scores according to previously described privacy preservation protocol, they typically employ the formula given in Eq. 2.3 to estimate similarity between any two perturbed user vectors, which converges to the non-private similarity calculation relying on the principle of producing random numbers from a zero-mean distribution, as shown in Eq. 2.4.

$$
\begin{aligned}
w'_{au} &= \frac{Z'_a \cdot Z'_u}{m} = \frac{(Z_a + R_a) \cdot (Z_u + R_u)}{m} \\
&= \frac{\sum_{i=1}^{m} z'_{ai} \times z'_{ui}}{m} = \frac{\sum_{i=1}^{m} (z_{ai} + r_{ai})(z_{ui} + r_{ui})}{m} \\
&= \frac{\sum_{i=1}^{m} z_{ai} z_{ui} + \sum_{i=1}^{m} z_{ai} r_{ui} + \sum_{i=1}^{m} z_{ui} r_{ai} + \sum_{i=1}^{m} r_{ai} r_{ui}}{m} \\
&\approx \frac{\sum_{i=1}^{m} z_{ai} \times z_{ui}}{m}.
\end{aligned} \tag{2.4}
$$

Notice that R_a and R_u vectors are noise data drawn from a zero-mean distribution, which are generated to disguise original z-score values. Similarly, the expected means of z-scores are zero, as well. Thus, the expected values of the last three summations in Eq. 2.4 converge to zero, which helps the server estimate similarities with decent accuracy on perturbed aggregate data.

Eq. 2.2 can be rewritten for producing a private prediction for a on q, as follows (Polat and Du, 2005a; Bilge and Polat, 2012) because predictions are generated relying on masked z-scores:

$$
p'_{aq} = \overline{v_a} + \sigma_a \times \frac{\sum_{u=1}^{k} z'_{uq} w'_{au}}{\sum_{u=1}^{k} w'_{au}} = \overline{v_a} + \sigma_a \times P'_{aq}, \tag{2.5}
$$

where k is the number of neighbors utilized in the prediction production process, $\overline{v_a}$ and σ_a represent a's mean vote and standard deviation values, respectively. Therefore, the server estimates P'_{aq} and sends it back to a, where she de-normalizes the received aggregate and obtains the final prediction. The server can estimate P'_{aq} based on masked data, as follows:

$$
\begin{aligned}
P'_{aq} &= \frac{\sum_{u=1}^{k} (w_{au} + V_{au})(z_{uq} + r_{uq})}{\sum_{u=1}^{k} w_{au} + V_{au}} \\
&= \frac{\sum_{u=1}^{k} w_{au} z_{uq} + \sum_{u=1}^{k} w_{au} r_{uq} + \sum_{u=1}^{k} V_{au} z_{uq} + \sum_{u=1}^{k} V_{au} r_{uq}}{\sum_{u=1}^{k} w_{au} + \sum_{u=1}^{k} V_{au}} \\
&\approx \frac{\sum_{u=1}^{k} w_{au} z_{uq}}{\sum_{u=1}^{k} w_{au}}.
\end{aligned} \tag{2.6}
$$

Eq. 2.6 holds because expected values of the last three summations in the nominator and the second one in the denominator converge to zero due to zero-mean random number distributions. In other words, the server can estimate P'_{aq} on masked data and still can produce accurate predictions.

2.4 Data Sets and Evaluation Criteria

There are a number of qualified crawls forming a variety of data sets containing preferences of real people whom they are sometimes defined by demographic information. Throughout the dissertation, experiments are performed on three well-known benchmark data sets. MovieLens data set (ML) is probably the mostly widely used data set collected by GroupLens at the University of Minnesota[1]. It has two variations by varying dimensions and densities according to the included number of ratings, i.e. 100K (MLP) and 1M (MLM) data sets. Netflix[2] is another data set of 100,480,507 ratings that 480,189 users gave to 17,770 movies. During the experiments, a subset of Netflix prize data set (NF) is utilized, where 10,000 users from differing density ranges are sampled. These data sets contain discrete preferences for movies in a 5-star rating scale. In addition, each movie in ML data set contains at least one or more genre features from predefined 18 categories. ML and NF data sets are suitable to show effects of preprocessing schemes as they are extremely sparse and large. Jester (Gupta et al., 1999) is a web-based joke recommendation system developed at University of California, Berkeley. Different from ML and NF data sets, Jester contains continuous ratings in [-10,+10] range for jokes and is much more dense than ML and NF. The data sets are summarized in Table 2.1.

Success of the proposed preprocessing schemes are evaluated with respect to the quality of produced predictions and online performance. Evaluations are conducted based on empirical results derived from real data-based experiments.

[1]http://www.grouplens.org/
[2]http://www.netflixprize.com/

Table 2.1: Descriptions of data sets

Name		Users × Items	Rating scale	Total votes	Density (%)
ML	MLP	943 × 1,682	5-star	100K	6.3%
	MLM	6,040 × 3,952	5-star	1M	4.25%
NF		10,000 × 17,700	5-star	2,337,295	1.32%
Jester		48,483 × 100	[-10,10]	3,519,448	72.59%

While statistical predictive accuracy metrics are valuable for experimenting on data sets containing numerical ratings, classification-based accuracy metrics are more useful on binary data sets.

Most widely utilized metric for numerical ratings-based systems is mean absolute error (MAE), which measures how close the predictions are to the actual ratings as an average of absolute errors, i.e., $\sum_P(e_i/n) = \sum_P(|p_i - v_i|/N)$, where p_i is the estimated prediction, v_i is the actual rating value, and P is the number of produced predictions. Thus, the smaller the MAE is, the better the results are.

Quality of predictions in binary ratings-based systems are measured using classification accuracy (CA) and F-Measure (F1) (Miyahara and Pazzani, 2002). CA measures the percentage of true classifications. F1 is generated to build precision and recall together, as follows: F1 = (2 × precision × recall)/(precision + recall). This measure combines precision and recall into one single number producing a useful metric for recommender systems. In the context of recommender systems, precision measures the probability of recommended item while recall is defined as the ratio of desirable and recommended items to all desirable items. The bigger the CA and F1 values, the better the results are.

In addition to accuracy, preprocessing approaches are proposed to improve scalability, as well. Thus, total elapsed time (T) in seconds spent on producing online referrals is recorded to demonstrate online improvements..

3. PRIVACY BY DESIGN

In this chapter, privacy-preserving mechanisms explained in Chapter 2 are analyzed by means of privacy controlling parameters to see how and to what extent they are effective in protecting confidentiality. Also, experiments are performed to show the effects of such parameters on accuracy in PPCF schemes.

3.1 Privacy Analysis

Data disguising protocols focus on preventing the central server from deducing (i) if a rating is genuine or forged and (ii) actual values of the genuine ratings. Accordingly, these two considerations are analyzed to evaluate the privacy level provided by the system.

3.1.1 Privacy analysis of rated items disguising

According to Procedure 1, each user profile contains genuine ratings along with the fake ones, which fill $\beta\%$ of the empty cells, where β is uniformly randomly chosen by the user from the interval $(0, \beta_{max}]$, as explained in Section 2.2.1. Therefore, the server first needs to guess the value of uniformly randomly chosen β and then it can predict the exact set of genuine items with some probability.

Privacy provided by uniform random selection of β is measured by utilizing Shannon entropy (Shannon, 1948) of user's apparent rating distribution. Recall that the entropy of a random variable X with possible values of $\{x_1, x_2, \cdots, x_n\}$ and distributed by a probability mass function p is defined as in Eq. 3.1, which can be interpreted as a measure of uncertainty or randomness of the outcome.

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log_2 p(x_i). \tag{3.1}$$

Intuitively, such uncertainty can be maximized through uniform selection of β over $(0, \beta_{max}]$. Inline with the perturbation scheme described in Section 2.2.1, users of a PPCF system can be modeled as random variables taking on values from normal distribution $\mathcal{N}(x; 0, \sigma^2)$ or uniform distribution $\mathcal{U}(x; 2\sigma\sqrt{3})$. Let g and b be the numbers of genuine ratings and empty cells of a particular user, respectively. Accordingly, let $P = \{p_1, p_2, \cdots, p_g\}$ defines the probability distribution of genuine ratings and $R = \{r_1, r_2, \cdots, r_{b\times\beta}\}$ defines the distribution of fake ratings. Now, user's distribution can be modeled as $\mathcal{S} = \frac{\#P+\#R}{g+(b\times\beta_{max})}$, where $\#P$ and $\#R$ indicate the number of elements in the sets P and R, respectively. Thus, privacy obtained by $Pr(\beta)$ can be quantified as the entropy of $\mathcal{S}$, i.e., $H(\mathcal{S})$. An example of provided privacy levels for varying β_{max} (accordingly β) values is depicted in Fig. 3.1 for 1M runs, where the user is assumed to have 50 genuine ratings. Note that there are 1,682, 3,952, and 17,700 ratable items in MLP, MLM, and NF data sets, respectively.

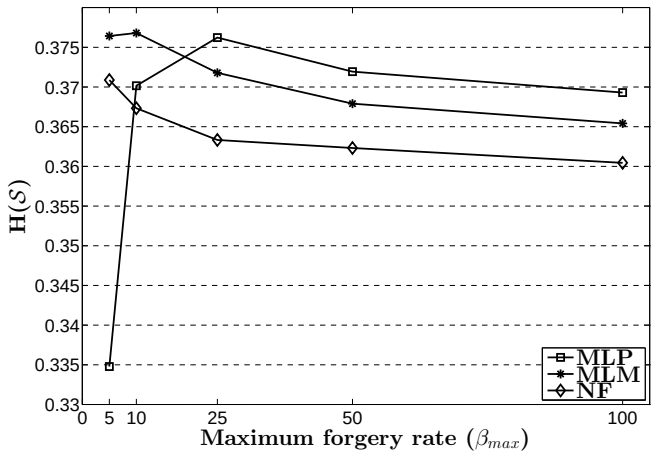


Figure 3.1: Privacy levels for varying β_{max} values

As seen from Fig. 3.1, inserting fake ratings has a definite effect on provided privacy levels. However, privacy levels differ according to β_{max}, number of ratable items, and density of data sets. While relatively small β_{max} values, such as 5% and 10%, are enough to provide the highest privacy intervals for data sets with so many items like MLM and NF, little more higher β_{max} val-

ues, such as 25%, provides the best privacy level. Accordingly, increasing β_{max} values do not provide more privacy due to loss of uncertainty between genuine and forged ratings. Therefore, it can be concluded that optimal selection of β_{max} for obtaining the highest privacy levels must be done inline with number of ratable items and data density within the data set.

After guessing β with probability 1 out of β_{max}, the server can try to extract the list of truly rated items. However, due to perturbation protocol, the server has g' and b' instead of g and b as numbers of actual ratings and empty cells, respectively. Hereafter, g can be calculated as $g = m - b$, where $b = b' \times 100/\beta$. Then, the list of genuine ratings can be predicted as one of the combinations of selecting g ratings out of g' disguised values. Combining these probabilities, the probability of determining the exact list of genuine ratings from a given disguised user vector can be estimated as 1 out of $\beta_{max} \times \binom{g'}{g}$, where $\binom{g'}{g}$ is the number of combinations of g' objects chosen g at a time.

3.1.2 Privacy analysis of ratings disguising

Even if the central server distinguishes genuine votes from faked ones, it still needs to extract real values from their perturbed z-score forms. Additionally, the privacy obtained by adding random noise on ratings must also be quantified. Agrawal and Aggarwal (2001) propose a differential entropy-based metric to quantify privacy of an additive noise-based perturbed variable, where such metric is utilized in PPCF context by (Polat and Du, 2005a; Bilge and Polat, 2012, 2013a). Let random variables P and R represent the original user vector and perturbing random data, respectively yielding $U = P + R$. Then *average conditional privacy* of P is defined as $\Pi(P|U) = 2^{H(P|U)}$, where $2^{H(P|U)}$ represents *conditional differential entropy* of P given U. Recall that P and R are independent random variables. Thus, privacy level of P after disclosing U is given by $\Pi(P|U) = \Pi(P) \times (1 - Pr(P|U))$, where $Pr(P|U) = 1 - 2^{H(U|P)-H(U)}$.

Assuming that P distributes normally, privacy levels, $\Pi(P|U)$, for various perturbation levels are presented in Fig. 3.2. Note that the distribution of R is determined by coin tosses. As seen from Fig. 3.2, provided privacy levels enhance with increasing level of perturbation, as expected. Normal distribution provides slightly better privacy.

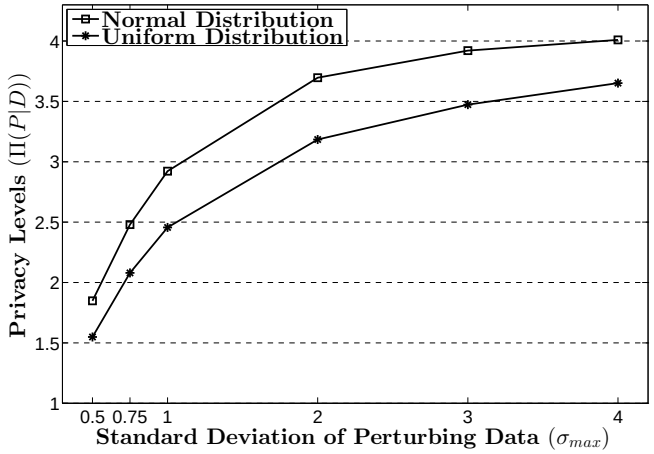


Figure 3.2: Privacy levels for varying σ_{max} values

Finally, as Eq. 2.5 demonstrates, the server needs to de-normalize extracted z-score values, which requires deducing mean and standard deviation of each user's original rating profiles.

3.2 Effects of Privacy Parameters on Accuracy

In order to demonstrate effects of privacy controlling parameters on accuracy, various experiments are performed based on three data sets using traditional k-nearest neighbor-based PPCF algorithm. During such experiments, data sets are divided into two subsets. One is used for training and the other is used for testing. A total of 1,570, 9,060, and 2,500 predictions are produced for MLP, MLM, and NF data sets, respectively. Number of neighbors to be utilized in prediction process is kept constant at 40 for all trials. While testing on β_{max} parameter, σ_{max} is also kept constant at 2. The trials are performed 100 times due to randomization and average results are presented. Changes in

error levels of produced referrals with varying β_{max} are demonstrated in Fig. 3.3.

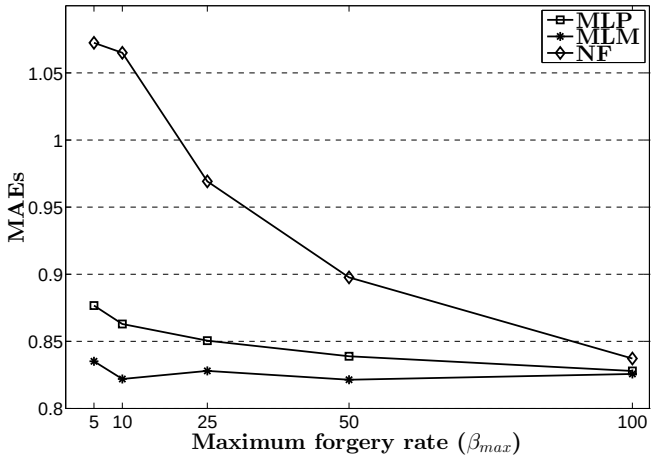


Figure 3.3: Accuracy levels for varying β_{max} values

As can be followed from Fig. 3.3, increasing β_{max} values generally have a positive effect on accuracy. Especially for extremely sparse NF data set, inserting ratings into the profiles, even if they are fake, improves accuracy. The reason for such improvement is it becomes possible to find co-rated items among users with inserted fake ratings. Therefore, as more fake ratings are inserted, accuracy enhances. However, considering the privacy levels presented in Fig. 3.1, reasonable β_{max} could be 10% or 25% for MLP, 10% for MLM, and 50% for NF. Next set of experiments investigate how accuracy changes with varying perturbation levels. Keeping optimum β_{max} values, several experiments are conducted 100 times while changing σ_{max} values. Overall averages of the outcomes are demonstrated in Fig. 3.4.

As expected, accuracy and σ_{max} are inversely correlated because perturbation level increases with increasing σ_{max}; and that makes accuracy worse. As seen from Fig. 3.4, with increasing σ_{max} values, the quality of the referrals gets worse. The bigger such values are, the more randomness added into the original data is. Although accuracy becomes worse with increasing σ_{max} values, privacy enhances due to augmented randomness, as demonstrated in Fig.

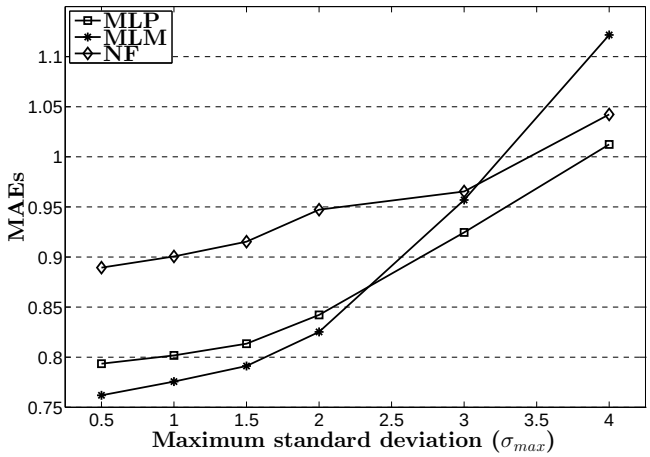


Figure 3.4: Accuracy levels for varying σ_{max} values

3.2. It is crucial to avoid data holders from estimating true rating values for individual privacy concerns. However, it is shown by (Huang et al., 2005) and (Kargupta et al., 2003) that utilizing σ less than 1 might result in recovery of actual ratings from perturbed values. Therefore, its value should be chosen carefully in order to still produce dependable and accurate recommendations while not deeply jeopardizing individual privacy.

4. AN ENHANCED RECOMMENDATION SCHEME ON TARGET ITEM-BASED SIMILARITY FUNCTION WITH PRIVACY

In this chapter, a preprocessing method is proposed to enhance the scalability of the recommendation scheme on target item-based similarity function. A formerly proposed similarity function is integrated into PPCF framework to improve quality of private referrals. The proposed preprocessing scheme finally is used to enhance the overall performance of the PPCF scheme on target item-based similarity function. Empirical outcomes demonstrate that the proposed preprocessing scheme relieves scalability issues significantly in both CF and PPCF environments and also improves accuracy in privacy-preserving frameworks.

4.1 Introduction

Memory-based CF schemes are one of the most successful recommendation technologies in terms of quality of predictions even though they commonly suffer from accuracy, scalability, and privacy issues. Such CF systems operate on entire collection to calculate similarities using a similarity metric such as PCC, cosine similarity, or distance-based similarity (Sarwar et al., 2000; Choi and Suh, 2013). Practical deployments to cover a large number of products prefer item-based similarity calculations while others perform user-based similarities to obtain better accuracy (Linden et al., 2003; Herlocker et al., 2004). Recently, Choi and Suh (2013) propose an intuitively reasonable modification in similarity function, which is proven to provide more accurate recommendations than the ones estimated by the state-of-the-art memory-based CF methods. This method introduces a ranking on neighbors' ratings by item similarities between

corresponding items and the target item. The proposed modification, which favors target items' similarity to those of other items, introduces additional computational costs on similarity measures, which further complicates scalability issues in memory-based CF schemes. Moreover, although the proposed similarity function results in better recommendation skills in CF systems, the framework does not consider privacy preservation.

In this chapter, the method proposed by Choi and Suh (2013) is investigated in terms of overhead costs and a preprocessing method to accelerate prediction production process is proposed. Preprocessing scheme aims at filtering out items that do not strongly correlate to the target item. In addition, privacy-preserving measures are employed on original and enhanced non-private schemes. Effects of the new similarity function and preprocessing method on scalability and accuracy of both non-private and privacy-preserving frameworks are investigated. Different sets of experiments are performed to evaluate proposed preprocessing scheme and similarity function with respect to scalability and accuracy.

4.2 Target Item-based Similarity Function

According to PCC (Eq. 2.1), the mostly utilized similarity calculation method in CF systems, each co-rated item has equal effect on similarity measure. Choi and Suh (2013) propose to modify the similarity metric so that each co-rated item's effect is also ranked with the item similarity between corresponding item and the target item in consideration. In other words, if an item is very similar to the target item, then it will have a superior influence on estimated prediction compared to a less similar one's effect. Therefore, they propose to apply adjustments on several similarity metrics. The best performing user-user similarity measure, a modified version of PCC, is presented in Eq. 4.1 (Choi and Suh, 2013).

$$PCC_{au}^{q} = \frac{\sum_{i=1}^{m'}[IS_{iq}^{2} \times (v_{ai} - \overline{v_a}) \times (v_{ui} - \overline{v_u})]}{\sqrt{\sum_{i=1}^{m'}[IS_{iq} \times (v_{ai} - \overline{v_a})]^2}\sqrt{\sum_{i=1}^{m'}[IS_{iq} \times (v_{ui} - \overline{v_u})]^2}}, \tag{4.1}$$

where PCC_{au}^{q} denotes PCC between a and u for q and IS_{iq} denotes the item similarity between co-rated items i and q. The best metric to calculate IS_{iq} might vary for different data sets; however, PCC (Eq. 2.1) and cosine similarity (Eq. 4.2) are shown to be the most feasible ones (Choi and Suh, 2013).

$$IS_{iq} = Cosine_{iq} = \frac{\sum_{u=1}^{n'}(v_{ui} \times v_{uq})}{\sqrt{\sum_{u=1}^{n'} v_{ui}^2}\sqrt{\sum_{u=1}^{n'} v_{uq}^2}}. \tag{4.2}$$

4.3 A More Precise and Scalable PPCF Scheme

In this section, proposed schemes to enhance scalability of CF and PPCF methods based on formerly recommended similarity function are described. First, the preprocessing approach, which eliminates dissimilar items from prediction process to alleviate scalability issues and possibly enhance accuracy of referrals in non-private scheme, is described. Then, modifications to apply the similarity function on perturbed data to improve recommendation quality are introduced. Finally, proposed preprocessing method is applied to the privacy-enhanced environment to further improve scalability and accuracy.

4.3.1 A preprocessing scheme to eliminate irrelevant items

The similarity function proposed by Choi and Suh (2013) utilizes item similarities between a commonly rated item and the target item in user similarity function. Since user-similarity calculation becomes more complicated due to new function, such computational overhead also affects online performance of the recommender system. Moreover, memory-based CF applications already suffer from scalability issues due to constantly growing size of data. Inspiring

from the idea of ranking preferences according to item similarity, an additional preprocessing step is proposed to eliminate relatively dissimilar items from prediction estimation process. Such preprocessing scheme is aimed to enhance scalability of CF system because it focuses on reducing online response time significantly by eliminating irrelevant items to each related target item.

According to the scheme proposed in (Choi and Suh, 2013), similarities, calculated between items using either Eq. 2.1 or Eq. 4.2 in off-line time, can be utilized in the online prediction estimation process, as outlined in Procedure 3. Item similarities rank each rating's effect on the estimation of prediction. Thus, co-rated items resembling more similarity to the target item have a superior effect on the estimation. However, all items still join to the process no matter they have a dissimilar manner to the target item. Therefore, such items are proposed to be eliminated from the prediction estimation process so that the process speeds up due to the reduction of dimensions in original user-item matrix.

Procedure 3 Off-line Item Similarity Calculation

Input: User-item matrix ($U_{n\times m}$)
1: **Initialize:** $IS_{m\times m} \leftarrow 0$ ▷ item similarities matrix
Calculate and sort item similarities:
2: **for all** $item_i$ in U ($i \leftarrow 1$ to m) **do**
3: **for all** $item_j$ in U ($j \leftarrow i$ to m) **do**
4: $IS(i,j)$ = SIMILARITY$(U(item_i), U(item_j))$ ▷ using Eq. 2.1 or Eq. 4.2
5: **end for**
6: **end for**
7: $[IS\_values, IS\_index]$ = SORT(IS, descending) ▷ to be used in online process
8: **return** IS_values and IS_index

According to the CF recommendation estimation process explained in Section 2.1, the bottleneck in the process is the calculation of similarities between a and each user in the system. Therefore, the proposed preprocessing scheme focuses on handling such bottleneck. If relatively dissimilar items are removed from the matrix for that particular target item, then a significant reduction can be obtained in dimensions of the original user-item matrix. Such

dissimilarity can be determined relying on a predetermined similarity threshold value (τ), so that the items having less similarity than the threshold value are eliminated. Then, a temporary user-item matrix can be formed for each corresponding target item, which is to be used in neighborhood formation process. Since similarity calculations are performed in the compact and reduced form of original user-item matrix, it will take much less time to calculate user similarities online. Pseudo code of the prediction estimation process relying on our preprocessing scheme is given in Procedure 4.

Procedure 4 Online Prediction Estimation via Preprocessing

Input: User-item matrix ($U_{n\times m}$), active user (a), neighbor count (k), target item (q), threshold value (τ)

Locate items more similar than τ:

1: $remaining\_items \leftarrow IS\_index(q, IS\_values(q) > \tau)$

Reduce dimensions of matrices:

2: $U \leftarrow U(remaining\_items)$;

3: $a \leftarrow a(remaining\_items)$;

4: $IS \leftarrow IS(remaining\_items)$

5: **Initialize:** $US_{1\times n} \leftarrow 0$ ▷ user similarities vector

Calculate and sort user similarities:

6: **for all** $user_i$ in U ($i \leftarrow 1$ to n) **do**

7: $US(i,j)$ = SIMILARITY$(U(user_i), q)$ ▷ using Eq. 4.1

8: **end for**

9: $[sim\_val, neighbor\_idx]$ = SORT(US,descending) ▷ used in prediction estimation

Estimate prediction:

10: $p_{aq} \leftarrow$ PREDICTION$(U, sim_val, neighbor_idx, k)$ ▷ estimated using Eq. 2.2

11: **return** p_{aq}

4.3.2 A PPCF scheme using target item-based similarity function

Due to privacy concerns, people prefer to submit their disguised vectors instead of explicit expressions. Therefore, the central server needs to estimate predictions based on such disguised collections with decent accuracy. How formerly proposed target item-based similarity function can be applied onto private prediction generation algorithm is explained in the following.

Neighborhood formation. The PCC equation (Eq. 2.1) can be represented as the covariance of two z-score transformed user vectors (Herlocker et al.,

1999). PPCF schemes typically employ the similarity calculation method given in Eq. 2.3 due to perturbation scheme. To employ target item-based similarity function onto such similarity calculation method, utilization of item similarities as a factor to covariance calculation is proposed, as shown in Eq. 4.3.

$$w_{au}^{q} = \frac{\sum_{i=1}^{m} IS_{iq} \times z_{ai} \times z_{ui}}{m}, \tag{4.3}$$

where w_{au}^{q} denotes covariance-based PCC weight between a and u for q, z_{ai} and z_{ui} represent z-score transformations of users' ratings on item i, respectively. However, as explained in Section 2.2, users submit their disguised z-scores, Z', due to privacy concerns. Hence, similarities between users in an RPT-based PPCF scheme are estimated on perturbed data, as shown in Eq. 4.4.

$$\begin{aligned} w_{au}^{q'} &= \frac{IS(q) \cdot Z'_a \cdot Z'_u}{m} \\ &= \frac{\sum_{i=1}^{m} IS_{iq} \times z'_{ai} \times z'_{ui}}{m} = \frac{\sum_{i=1}^{m} IS_{iq}(z_{ai} + r_{ai})(z_{ui} + r_{ui})}{m} \\ &= \frac{\sum_{i=1}^{m} IS_{iq} z_{ai} z_{ui} + \sum_{i=1}^{m} IS_{iq} z_{ai} r_{ui} + \sum_{i=1}^{m} IS_{iq} z_{ui} r_{ai} + \sum_{i=1}^{m} IS_{iq} r_{ai} r_{ui}}{m} \\ &\approx \frac{\sum_{i=1}^{m} IS_{iq} \times z_{ai} \times z_{ui}}{m}. \end{aligned} \tag{4.4}$$

Notice that R_a and R_u vectors are noise data drawn from a zero-mean distribution, which are generated to disguise original z-score values. Similarly, the expected means of z-scores are zero, as well. Thus, the expected values of the last three summations in Eq. 4.4 converge to zero, which helps the server estimate similarities with decent accuracy relying on perturbed aggregate data.

Off-line item similarity calculations are also performed on perturbed data. Without privacy concerns, it is trivial to calculate such similarities using PCC or cosine similarity. However, since users send their disguised z-scores, the data collector should be able to estimate weights between items from masked data, as well. Due to disguising mechanism, the server can similarly utilize

covariance-based PCC for item similarities, as shown in Eq. 4.4. It can estimate similarities between co-rated item i and target item q, as explained in the following: In Eq. 4.2, the numerator part performs multiplication between co-rated users, which can be treated as a dot product in privacy-preserving scheme because unrated items have a zero rating value. With increasing number of users in the system, as can be followed from Eq. 4.3, such dot product calculations can be performed with sufficient accuracy due to zero-mean nature of random number distribution. The denominator holds magnitude calculation of two vectors. The server can estimate such magnitudes for an item vector, as shown in Eq. 4.5.

$$\|Z'\|_2 = \|(Z + R)\|_2 = \sqrt{\sum_{u \epsilon S} (z_u + r_u)^2}, \tag{4.5}$$

where S is the set of users who rated corresponding item and R represents the distribution of such users' random perturbing factors added onto genuine ratings. Eq. 4.5 can be rewritten without square roots, as follows:

$$\sum_{u \epsilon S} (z_u + r_u)^2 = \sum_{u \epsilon S} z_u^2 + 2 \sum_{u \epsilon S} z_u r_u + \sum_{u \epsilon S} r_u^2 \approx \sum_{u \epsilon S} z_u^2 + \sum_{u \epsilon S} r_u^2. \tag{4.6}$$

Eq. 4.6 holds as number of users submitting a vote for the item increases due to generated random numbers distribution with zero mean. However, to get rid of the second summation, the server can subtract its contribution relying on the maximum allowed standard deviation of the random numbers, as follows:

$$\sum_{u \epsilon S} (z_u + r_u)^2 \approx \sum_{u \epsilon S} z_u^2 + \sum_{u \epsilon S} r_u^2 - S\sigma_{max}^2 \approx \sum_{u \epsilon S} z_u^2. \tag{4.7}$$

After computing the summation, the server can take the square root and estimate magnitudes of vectors and similarity weights between items based on masked data. Then, the most similar k of such users are labeled as neighbors

to be used in prediction production process.

Prediction estimation. The server estimates an aggregate based on perturbed data and sends it back to a who de-normalizes the received aggregate via her ratings mean and standard deviation. Since predictions are generated relying on masked z-scores data, Eq. 2.5 can be rewritten for producing a private prediction for a on q, as follows (Bilge and Polat, 2012):

$$p'_{aq} = \overline{v_a} + \sigma_a \times \frac{\sum_{u=1}^{k} z'_{uq} w_{au}^{q'}}{\sum_{u=1}^{k} w_{au}^{q'}} = \overline{v_a} + \sigma_a \times P'_{aq}, \tag{4.8}$$

where k is the number of neighbors utilized in the prediction production process, $\overline{v_a}$ and σ_a represent a's mean vote and standard deviation, respectively. Therefore, the server estimates P'_{aq} and sends it back to a, where she de-normalizes provided aggregation and obtains the final prediction. The server can estimate P'_{aq} based on masked data, as follows:

$$\begin{aligned} P'_{aq} &= \frac{\sum_{u=1}^{k} (w_{au}^{q} + V_{au}^{q})(z_{uq} r_{uq})}{\sum_{u=1}^{k} w_{au}^{q} + V_{au}^{q}} \\ &= \frac{\sum_{u=1}^{k} w_{au}^{q} z_{uq} + \sum_{u=1}^{k} w_{au}^{q} r_{uq} + \sum_{u=1}^{k} V_{au}^{q} z_{uq} + \sum_{u=1}^{k} V_{au}^{q} r_{uq}}{\sum_{u=1}^{k} w_{au}^{q} + \sum_{u=1}^{k} V_{au}^{q}} \\ &\approx \frac{\sum_{u=1}^{k} w_{au}^{q} z_{uq}}{\sum_{u=1}^{k} w_{au}^{q}}. \end{aligned} \tag{4.9}$$

Eq. 4.9 holds because expected values of the last three summations in nominator and the second one in denominator converge to zero due to zero-mean random number distributions. In other words, the server can estimate P'_{aq} on masked data and still can produce accurate predictions.

4.3.3 Improving performance of PPCF via preprocessing

It is possible to apply the preprocessing idea proposed in Section 4.3.1 onto PPCF framework described in Section 4.3.2. The PPCF scheme utilizes the target item-based similarity function. In this proposed framework, PPCF re-

ferrals are aimed to be produced in less amount of time to enhance scalability. In addition, it is possibly expected to obtain more qualified private referrals. Hence, accuracy-enhanced traditional CF scheme via target item-based similarity function will be further improved to provide private referrals with better accuracy and in scalable time. To do so, the private framework defined in Section 4.3.2 utilizes Procedure 3 and Procedure 4 along with privacy-preserving similarity calculations (Eq. 4.3 and Eq. 4.7) and prediction estimation equations (Eq. 4.8 and Eq. 4.9).

4.4 Overhead Costs Analysis

It is imperative to analyze the proposed preprocessing scheme, which is employed in both non-private and privacy-preserving CF schemes, with respect to off-line and online costs. During such analysis and experimental examination, traditional recommendation approach, similarity function-enhanced CF method, and preprocessing applied ultimate model are denoted as **CF**, **CF+**, and **CF++**, respectively. Overhead costs due to introduced preprocessing scheme must be analyzed by means of (*i*) communication, (*ii*) storage, and (*iii*) computational costs. An overview of the analysis is given in Table 4.1.

Table 4.1: Overview of overhead costs

	Comm.	Storage	Computational	
			Off-line	Online
CF	$O(1)$	$O(nm)$	—	$O(k+nmP)$
CF+	$O(1)$	$O(nm)+O(m^2)$	$O(nm^2)$	$O(k+nmP')$
CF++	$O(1)$	$O(nm)+O(m^2)$	$O(nm^2)+O(m\log m)$	$O(k+n\overline{m}P')$

P and P': number of calculations in Eq. 2.1 and Eq. 4.1, respectively.
$\overline{m}$: reduced number of items for corresponding target item

Compared to the traditional **CF** approach, **CF+** approach proposed by Choi and Suh (2013) and our item reduction preprocessing-based **CF++** approach scheme does not cause any extra communication overheads. All three schemes require a transfer of 1-by-m user vector, which introduces an $O(1)$

complexity in terms of communications costs. Hence, it can be concluded that both the number of communications and amount of transmitted data online and off-line phases remain the same for all three schemes.

CF scheme requires a storage cost in the order of $O(nm)$ to record preferences of n users on m items. However, **CF+** and **CF++** schemes utilize item similarities in user-user similarity computations. Therefore, **CF+** requires a total storage area in the order of $O(nm) + O(m^2)$ to hold item similarities, as well. Additionally, **CF++** approach also eliminates some items from the collection relying on item similarities, which requires to hold sorted item similarity index values in addition to item similarities, which results in a storage cost in the order of $O(nm) + O(m^2)$ in total.

Computation costs should be analyzed separately for off-line and online phases. Although off-line computations are not critical for recommender systems, it is better to provide a report on off-line work overload. Traditional **CF** scheme solely runs online and does not perform any off-line computations. However, **CF+** and **CF++** schemes calculate item-item similarities in off-line phase in the order of $O(nm^2)$, where **CF++** scheme also sorts such similarities, which requires an additional $O(m \log m)$ time using quick sort algorithm.

The important component of recommender systems' performance is determined by how fast queries are responded online. **CF** scheme runs in $O(k + nmP)$ time, where k represents the number of neighbors to be utilized and P is the complexity of computations performed in user-user similarity calculation via a similarity measure. **CF+** scheme also produces predictions in a similar manner; however, drawback of **CF+** scheme is that it further complicates online similarity calculation step by assembling item-similarity factors into similarity formulas. Such increased computational complexity is denoted with $O(k+nmP')$ in Table 4.1, where P' represents the increased complexity of

calculations and $P' > P$ all the time. The proposed preprocessing, on the other hand, reduces the number of items to be utilized in similarity calculation step and requires an online computation time in the order of $O(k + n\overline{m}P')$, where the size of $\overline{m}$ is determined for each corresponding target item separately, but making sure that $\overline{m} \ll m$ to relieve scalability issues. Data disguising procedure allows PPCF systems to collect and store preference data similar to the non-private schemes and produce predictions in an identical way. Thus, such overhead costs analysis is also valid for privacy-preserving conjugates of **CF**, **CF+**, and **CF++**.

4.5 Experimental Evaluation

Several experiments are performed on two benchmark data sets, i.e., MLM and NF, to scrutinize the effects of applying the similarity function in privacy-preserving systems and employing the proposed preprocessing scheme on non-private and privacy-preserving schemes.

Experiments are realized on uniformly randomly chosen train and test sets. The original data set (U or U') is divided into two parts, where uniformly randomly chosen 30% of all users are assigned to be test users and remaining ones as train users. After training and test sets are constructed, five rated items' actual votes are withheld for each test (active) user. Such entries are replaced with null, their values are tried to be predicted, and estimations are compared with actual values. User-user similarities are computed via PCC on both data sets and item-item similarities are calculated by PCC in MLM and by cosine similarity in NF because they are the best performing measures on those data sets, as shown by (Choi and Suh, 2013). Also, number of neighbors (k) is set to 10. Trials are performed in MATLAB 8.0 environment using a computer with an Intel Core i7 2.8 GHz dual-core processor and 4 GB RAM.

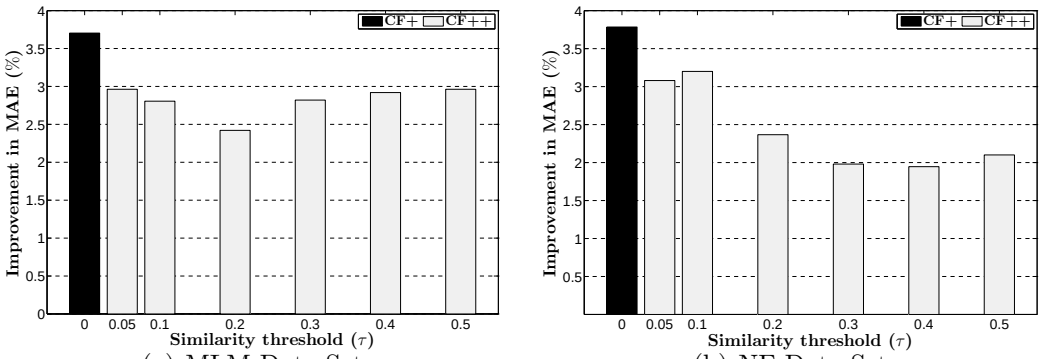
Three sets of experiments are utilized. First, **CF++** scheme is exper-

imented on by employing the proposed preprocessing method to see its effects on accuracy and scalability compared to **CF** and **CF+** schemes. Secondly, **PPCF+** scheme is derived by implementing the similarity function onto privacy-preserving scheme and investigated how it performs in terms of accuracy. Finally, **PPCF++** scheme is obtained by preprocessing **PPCF+** and its performance is examined with respect to accuracy and online performance.

4.5.1 Evaluating preprocessing technique in non-private schemes

In order to examine the effects of the proposed preprocessing scheme by means of scalability and accuracy, the method is applied to non-private scheme first. As demonstrated Choi and Suh (2013), applying the similarity function onto traditional CF schemes improves accuracy. However, they do not perform any online performance test, which is vital for recommender systems. Experimental results of accuracy and online performance for both similarity function enhanced **CF+** scheme and preprocessing enhanced **CF++** are demonstrated. While utilizing τ for **CF++** scheme, although PCC takes values in the interval [-1.0, 1.0] for item similarity calculations, such values are transformed to [0, 1.0] interval to form a common base with cosine similarity, which also takes values in the interval [0, 1.0]. Then, τ is varied from 0.05 to 0.5 in order to eliminate dissimilar items. To present a more clear comparison, improvements of **CF+** and **CF++** schemes over traditional **CF** scheme are presented in percentage. The outcomes with respect to accuracy and online performance are given in Fig. 4.1 and Fig. 4.2, respectively.

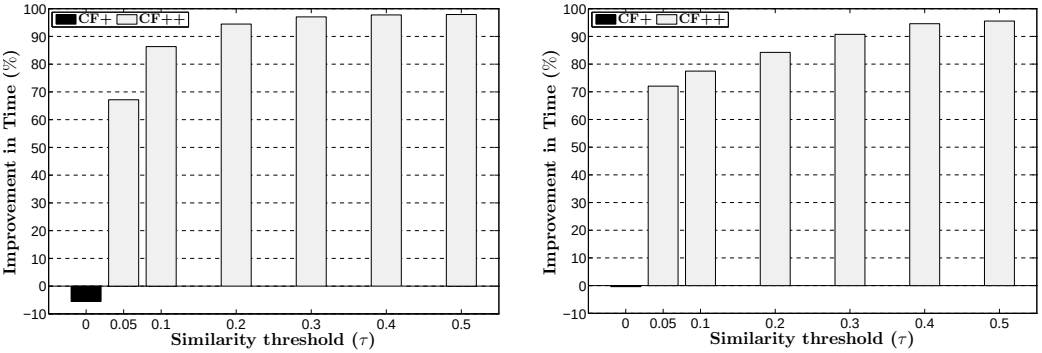
As seen from Fig. 4.1, **CF+** scheme achieves better accuracy improvements (about 4%) compared to the proposed **CF++** scheme (about 3%). Especially for MLM data set, such improvements can be obtained even for very high τ values. For MLM data set, the best improvement is obtained at $\tau = 0.5$ with 2.96%, where **CF+** scheme achieves 3.7%. Due to its extreme



(a) MLM Data Set (b) NF Data Set

Figure 4.1: Accuracy improvements with varying τ values for **CF+** and **CF++** schemes

sparse nature, increasing τ values affect accuracy adversely for NF data set. Therefore, the best improvement for NF is obtained at τ=0.1 with 3.2%, where **CF+** scheme achieves 3.8%.



(a) MLM Data Set (b) NF Data Set

Figure 4.2: Performance improvements with varying τ values for **CF+** and **CF++** schemes

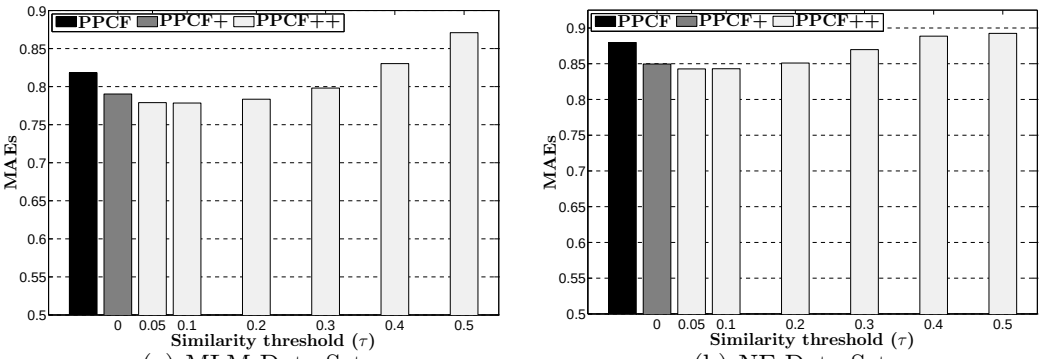
Performance improvements, on the other hand, are vast. As can be followed from Fig. 4.2, **CF+** scheme introduces extra burden compared to traditional **CF** scheme and degrades performance by 5.5% for MLM data set. However, losses due to integration of the similarity function is negligible in NF data set due to extreme sparsity. Also, improvements due to preprocessing are significant for both data sets. As τ grows, online performance enhances for **CF++** scheme, as expected. Improvements are about 98% and 77% for MLM and NF data sets, respectively, where the highest quality of predictions

are obtained, i.e., τ=0.5 for MLM and τ=0.1 for NF. It can be concluded that employing the proposed preprocessing, at a level of satisfactory quality of predictions is obtained, is also beneficial for improving scalability.

4.5.2 Evaluation of privacy-preserving schemes

After examining the effects of the proposed preprocessing scheme on non-private CF schemes and determining the optimum threshold values for different data sets, privacy-preserving environment is then evaluated experimentally. First, effects of applying the similarity function onto traditional **PPCF** algorithm, which is called **PPCF+**, is investigated. Then, the preprocessing method is implemented onto **PPCF+** and **PPCF++** scheme is derived. Similar to the experiments in non-private environment, the success of **PPCF+** and **PPCF++** schemes against traditional **PPCF** approach is scrutinized by varying τ from 0.05 to 0.5. For data disguising procedure, standard deviation of produced random values (σ_{max}) is kept constant because there is no need to investigate such parameter's effects as it obviously deteriorates accuracy inline with the distortion amount, as shown in Fig. 3.2. However, due to the results of (Huang et al., 2005) and (Kargupta et al., 2003), utilizing $\sigma \leq 1$ may permit recovery of original data from perturbed values. Thus, σ_{max} is kept at 2.0 during the experiments. Also, maximum forgery rate (β_{max}) is kept constant at 25. Effects of different distortion values and varying β_{max} values on accuracy are studied in Section 3.2 and also can be found in (Polat and Du, 2005a; Bilge and Polat, 2012). Due to randomized selection of σ and β by each user, the experiments are repeated 100 times and overall averages of the outcomes are demonstrated in Fig. 4.3.

As demonstrated in Fig. 4.3, the similarity function is effective in privacy-preserving algorithms, as well. **PPCF+** scheme manages to reduce the error values by 3.42% (from 0.818 to 0.790) in MLM data set and 3.53% (from



(a) MLM Data Set (b) NF Data Set

Figure 4.3: Accuracy improvements with varying τ values for **PPCF**, **PPCF+**, and **PPCF++** schemes

0.881 to 0.849) in NF data set compared to original **PPCF** scheme. Such improvements are similar to the ones achieved in non-private schemes. In addition, applying preprocessing causes further improvements in quality of predictions. For τ=0.05 and τ=0.1 values, **PPCF++** scheme performs better than **PPCF+** in both data sets. However, increased τ values like τ=0.5 cause too much loss of information; and therefore, accuracy diminishes. However, it can be concluded that for $\tau \leq$0.2, **PPCF++** scheme is able to perform at least as efficient as **PPCF+** scheme in terms of accuracy for both data sets. Although accuracy improvements are similar to non-private schemes' results, to present a clear overview, elapsed time (in seconds) to produce predictions are demonstrated in Table 4.2.

Table 4.2: Online performance with varying τ values for **PPCF**, **PPCF+**, and **PPCF++** schemes

	PPCF	**PPCF+**	**PPCF++**					
			τ=0.05	τ=0.1	τ=0.2	τ=0.3	τ=0.4	τ=0.5
MLM	1,285s	1,351s	416s	171s	70s	37s	28s	26s
NF	18,822s	18,891s	5,247s	4,235s	2,964s	1,740s	1,019s	838s

Table 4.2 presents elapsed time spent for producing five predictions for each active user. Thus, 9,060 and 15,000 predictions are generated for MLM and NF, respectively. It is clearly seen that **PPCF+** scheme brings extra

online computational cost and the proposed **PPCF++** scheme significantly enhances scalability with increasing τ values. Combining these experimental results with the ones presented in Fig. 4.3, it can be concluded that applying the preprocessing technique with τ=0.2 provides an optimal performance in terms of both accuracy and scalability. Since in such arrangement, quality of predictions is very close to the values achieved by **PPCF+** scheme, yet online performance is enhanced by about 94% for MLM and 84% for NF data sets.

4.6 Conclusions

In this chapter, applying a target item-based similarity function on privacy-preserving collaborative recommender systems is investigated, which was previously adapted on non-private schemes and performed well. It is theoretically examined how to integrate such similarity function onto privacy-preserving collaborative filtering architecture and its applicability is proven. It is also studied how individual privacy is preserved by following such scheme. However, applying the similarity function introduces slight extra computational costs to the existing schemes. In order to alleviate this problem, item similarities are utilized, which were already computed for the similarity function. Motivating from the same idea of ranking item ratings with the target item similarities, eliminating relatively dissimilar items from the original matrix is proposed before calculating user similarities. Such elimination reduces the size of user-item matrix, which helps scaling the system. After analyzing the proposed preprocessing scheme with respect to overhead costs, several experiments are performed to scrutinize the effects of the scheme in both non-private and privacy-preserving environments. According to overall empirical outcomes, implementing the similarity function onto privacy-preserving framework results promising as the quality of predictions are enhanced like in non-private schemes. Moreover, the proposed preprocessing scheme achieves slightly better

accuracy in privacy-preserving framework. Most importantly, improvements in terms of online performance are major, where traditional and accuracy-enhanced CF and PPCF schemes are significantly outperformed.

Although improvements are similar for both MLM and NF , the preprocessing scheme performs slightly better for MLM because it is more than three-times dense than NF. Since NF is extremely sparse, it is expected that there is already limited number of co-rated items between users, which makes harder to improve online response time by eliminating dissimilar items. Moreover, improvements in quality of predictions are also less than the values achieved in MLM. The same reason applies here, as well. Since the likelihood of finding co-rated items between users gets harder, item elimination causes too much loss of information, which makes accuracy worse. It is shown that combining the similarity function with the preprocessing method achieves better accuracy and online performance in privacy-preserving framework.

5. AN IMPROVED PRIVACY-PRESERVING DWT-BASED COLLABORATIVE FILTERING SCHEME

In this chapter, privacy-preserving schemes are proposed to produce accurate predictions efficiently based on DWT without deeply exposing customers' privacy. Also, a preprocessing method, which orders items before applying DWT is recommended to boost accuracy. Experimental results show that the proposed privacy-preserving methods are able to offer referrals with decent accuracy and the preprocessing method utilized to sort items improve accuracy.

5.1 Introduction

DWT has a wide application area in science, engineering, mathematics, and computer science. It is widely used for audio and video compression (Kumari and Sadasivam, 2007), object recognition (Cheng and Chen, 2006), and numerical analysis (Van de Plas et al., 2008). Furthermore, it has an ability to be combined with previously discussed dimensionality reduction techniques like SVD (Zhao and Ye, 2009), PCA (Korürek and Nizam, 2010), and clustering (Yu and Kamarthi, 2010) in data mining.

One approach to enhance scalability of CF systems is to apply DWT techniques on the input matrix. Russell and Yoon (2008) apply DWT to recommender systems, where input data are transformed and reduced significantly to decrease the amount of time for producing a prediction. DWT-based CF schemes significantly overcome the scalability problem. However, they fail to protect individual users' privacy. Moreover, although such schemes provide accurate predictions, the quality of the recommendations provided by DWT-based CF schemes can be further improved by applying some preprocessing methods.

In this chapter, it is investigated how to produce DWT-based referrals relying on perturbed data. In other words, privacy-preserving schemes are proposed in which users disguise their ratings and rated items before they send them to the data collector. Also, preprocessing methods are proposed to order items before applying DWT so that accuracy can be improved. It is hypothesized that items' ordering might affect the information losses in DWT. If it is possible to order the items in such a way to reduce information losses, the quality of the predictions can be improved without sacrificing on scalability. Supplementary costs due to proposed schemes are analyzed and real data-based experiments are conducted to assess accuracy changes.

5.2 DWT-based Collaborative Filtering

DWT reduction approach is applied to CF by Russell and Yoon (2008) using Haar wavelets to reduce the amount of items in order to achieve scalability in recommendation process. DWT-based CF approach basically divides the original user-item matrix into two components, approximation and detail coefficients, using Eq. 5.1:

$$C_{approximation} = \frac{u_j + u_{j+1}}{\sqrt{2}}, C_{detail} = \frac{u_j - u_{j+1}}{\sqrt{2}}. \tag{5.1}$$

While approximation coefficient is a time domain expression of original data, detail coefficient is a frequency domain perspective of it. Both coefficients consist of half of the items compared to original user-item matrix. However, there is a much difference between coefficients in the perspective of information held by them as approximation coefficient holding a very large portion of it. With this feature of DWT, consecutive transformations can be performed proceeding exclusively on approximation coefficient and hereby obtaining a very much compact form of original data with a sacrifice of negligible amount

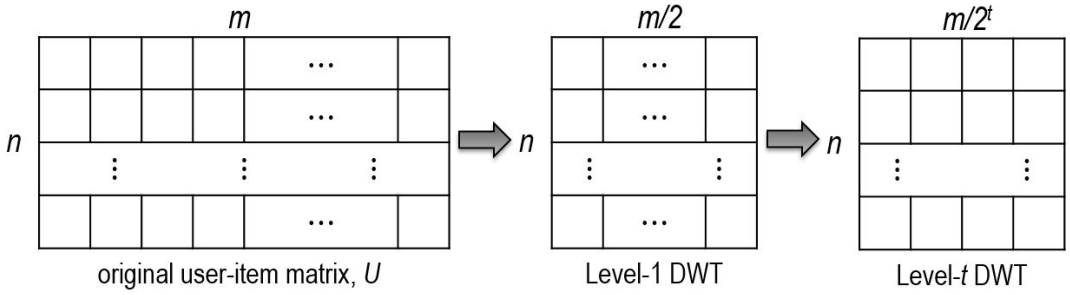


Figure 5.1: A depiction of DWT

of information. Since DWT process halves the size of an original user-item matrix of size $n \times m$, applying transformation t times, a compact form with size $n \times (m/2^t)$ is obtained as depicted in Figure 5.1.

In (Russell and Yoon, 2008), such a model (the reduced data) is used to determine similarities among users. Similarity calculation between two users is calculated using PCC. In the sense of such an approach to determine similarities, entire process is accelerated remarkably because the number of items significantly decreases, which enhances the scalability of the system. After determining similarities and selecting the nearest neighbors to a particular user, items with the highest predicted ratings are provided as a recommendation using adjusted weighted sum CF method. The approach presented in (Russell and Yoon, 2008) is shown to perform well in terms of both accuracy and performance. It is proven by experiments that DWT is a solution to overcome scalability and accuracy problems of CF systems.

5.3 Privacy-Preserving DWT-based Prediction Scheme

The proposed privacy-preserving scheme is explained in detail. Reducing perturbed user data and producing online predictions using DWT is investigated.

5.3.1 Reducing perturbed data using DWT

As explained by Russell and Yoon (2008), a Haar-based DWT produces an approximation coefficient for each pair in the distribution being transformed.

Given a user-item matrix U, transformed ratings matrix can be easily obtained. The server can transform the disguised user-item matrix U', as in the following steps, using the Haar-based DWT:

1. It fills empty cells with related user averages to obtain a dense set. To do so, it should estimate such values from disguised data. The average z-scores can be estimated from perturbed data for each user, as follows:

$$\begin{aligned} \overline{z'_u} &= \frac{\sum_{j=1}^{R_u} z'_{uj}}{R_u} = \frac{\sum_{j=1}^{R_u} (z_{uj} + r_{uj})}{R_u} = \frac{\sum_{j=1}^{R_u} z_{uj}}{R_u} + \frac{\sum_{j=1}^{R_u} r_{uj}}{R_u} \\ &\approx \frac{\sum_{j=1}^{R_u} z_{uj}}{R_u}. \end{aligned} \tag{5.2}$$

   Since the random numbers are drawn from a distribution with zero mean, the expected value of their average is zero. With increasing R_u values, the observed value of random number averages will become closer to zero. Thus, the server can estimate averages of masked z-scores, fills the empty cells, and obtains a dense set.

2. It then estimates approximation coefficients using the Haar-based DWT based on masked z-scores, as follows:

$$\begin{aligned} z'_{approximation_u} &= \frac{z'_{uj} + z'_{uj+1}}{\sqrt{2}} = \frac{(z_{uj} + r_{uj}) + (z_{uj+1} + r_{uj+1})}{\sqrt{2}} \\ &= \frac{z_{uj} + z_{uj+1}}{\sqrt{2}} + \frac{r_{uj} + r_{uj+1}}{\sqrt{2}} \approx \frac{z_{uj} + z_{uj+1}}{\sqrt{2}}. \end{aligned} \tag{5.3}$$

   For similar reasons, approximation coefficients can be estimated with decent accuracy from perturbed data. With increasing level of transformations, contributions of each pair of random numbers will converge to zero. Notice also that since users do not add or insert random numbers to all cells in their vectors, some pairs might have only one perturbed value, some may have undisguised pairs. In all cases, due to the nature

of random number distribution and data transformation, approximation coefficient can be estimated from perturbed data.

3. After performing data transformation one to three levels, it finally gets U'' with size $n \times m_t$ including transformed masked z-scores. The computations so far are done off-line. It can now estimate predictions online.

5.3.2 Online recommendation estimation

Online computations include transforming a's data, neighborhood formation, and prediction estimation, explained in detail in the following.

Transformation of a's Data. Since a normalizes her ratings into z-scores and disguises them along with rated items like other users do, the server can similarly fill her empty cells and transform her data by applying DWT.

Formation of Neighborhood. This step consists of (i) estimating similarities between a and each user in the database and (ii) selecting the nearest similar users. The similarity weights are estimated based on PCC from masked and transformed data, as formulated by Eq. 2.4.

Recommendation Estimation. Since all users, including active users, send normalized ratings to the server, p_{aq} (the prediction for a on item q) can be estimated using Eq. 2.5 and Eq. 2.6. In other words, the server is still able to estimate predictions from disguised data without violating users' privacy. Although the users are free to use either uniform or Gaussian distribution with a randomly chosen σ over a specified range, the server is still able to compute referrals. Due to z-scores, noise data, filled cells, and variably perturbing methods, the server cannot derive users' private data from received data.

5.3.3 Reducing U' with minimum information loss

DWT is used to transform a discretely sampled continuous signal into its time and frequency domain components in signal processing and coding applica-

tions. The reason why DWT is very successful in representing a signal in terms of its time and frequency components is that it produces approximation and detail coefficients from two consecutive samples, which are relatively very close to each other in the spectrum due to being the samples of a continuous signal. However, this condition is not satisfied while applying DWT to recommenders systems for the purpose of transforming and reducing a user-item matrix supposing items as discrete samples of a signal. In the transformation process, as explained previously, two consecutive samples produce an approximation coefficient through division of their sum by $\sqrt{2}$. Therefore, if two consecutive samples are far from each other in the spectrum, at the end of the transformation, those samples will neutralize each other's effect causing information loss in original data. We hypothesize that if similar items are transformed together, such transformations might reduce information losses in the matrix. For this reason, we propose to order items of the matrix in such a way so that the most similar items are transformed together. Due to the reduced information losses, the quality of the recommendations then can be improved, as well. Since the natural order is not necessarily suitable for transformation, we propose to merge the most similar items together to form non-neutralizing consecutive pairs in the matrix. Moreover, this process does not affect the online performance because such orderings can be conducted off-line. Thus, accuracy can be improved without sacrificing on online performance while protecting users' privacy.

Two items, x and y, can be thought of as two vectors in the n dimensional space and the similarity between them, $sim(x, y)$, is measured by means of the cosine of the angle between those vectors, as described in Eq. 5.4 (Sarwar et al., 2001).

$$sim(x, y) = \cos(\overrightarrow{x}, \overrightarrow{y}) = \frac{x \cdot y}{||x||_2 \times ||y||_2}. \tag{5.4}$$

Similarity weights between all items are estimated using Eq. 5.4 off-line to form proper pairs so that information losses are minimized before transformation. Two different approaches are proposed to arrange items, as follows:

Single Arrangement (SA): The most similar items are put adjacently once and DWT is repeatedly performed on arranged matrix.

Multiple Arrangements (MA): After putting the most similar items adjacently, this arrangement process is repeated for all transformations.

Without privacy concerns, estimating similarities between items using cosine similarity measure is an easy task. However, since users disguise their z-scores using RPTs, data collector should be able to estimate such weights from masked data, as well. Note that each user masks her z-scores variably, i.e. using whether Gaussian or uniform distributions with different σ values. The server can estimate $sim(x, y)$ values from perturbed data with decent approximation, as explained in Section 4.3.2.

5.4 Performance and Overhead Costs Analysis

The proposed schemes should be analyzed in terms of both online and off-line costs like storage, communication, and computation costs. Notice that off-line costs are not that critical for overall performance.

The proposed privacy-preserving schemes have no effect on storage costs. In other words, online and off-line storage costs do not change due to privacy-preserving measures. The storage cost is in the order of $O(nm)$ for both the original and the proposed privacy-preserving DWT-based CF schemes. In the proposed schemes, number of communications in online and off-line phases remains the same compared to the original algorithm. Moreover, during online interactions, amount of transferred data remains the same, as well.

Due to the nature of data disguising schemes used in the proposed schemes, online computation costs do not change. In the proposed schemes, the server performs the same tasks like it does in the original DWT-based CF algorithm. Privacy-preserving measures do not cause any additional computation costs during online phase. However, the proposed preprocessing schemes introduce extra computation costs during off-line phase. Although online computation costs are the critical ones for web-based applications; nevertheless, it is still needed to analyze off-line computation costs. Compared to the DWT-based CF algorithm, item-ordering scheme requires an extra off-line computation costs. In an $n \times m$ user-item matrix, a similarity computation between two items requires n multiplication operations according to cosine-based similarity calculation. Also, since such similarity computation is performed among all items, there will be $(m-1) + (m-2) + \cdots + 1 + 0 = \frac{(m-1)\times(m-2)}{2}$ similarity calculations for all m items. Therefore, when the server utilizes SA method to order items, this requires extra off-line computation costs in the order of $O(m^2n)$. If it employs MA approach for item ordering, it needs to apply item ordering multiple times, where each time number of calculations decreases by four times. Although total number of similarity computations in MA approach increases compared to the ones in SA, the number of similarity computations are still in the order $O(m^2n)$.

5.5 Experimental Evaluation

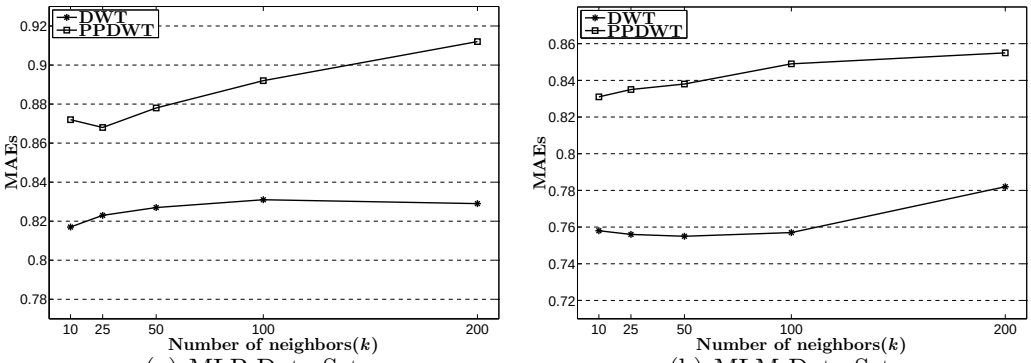
Privacy-preserving schemes definitely cause a decrease on accuracy due to conflicting goals privacy and accuracy. Item-ordering preprocessing is proposed to minimize information loss due to applying DWT on user-item matrix. Although privacy-preserving schemes make accuracy worse, item-ordering scheme might recover such losses. During experiments, the original non-private DWT-based CF scheme will be referred as **DWT** and the privacy-preserving DWT-

based CF scheme as **PPDWT**. Also, the preprocessed **PPDWT** schemes will be referred as **SA** and **MA** for single and multiple arrangements, respectively.

Experiments are performed using MLP and MLM data sets. While MLP is utilized completely, 1,000 users are uniformly randomly chosen from MLM among whom provided at least 40 ratings. Data sets are divided into train and test sets for each experiment. For test purposes, 40% of all available actual ratings are uniformly randomly chosen. Each time a rating is taken into consideration, the user whom that rating belongs to is treated as the active user and the remaining users are treated as train users. Data are transformed for three levels using DWT because it is determined to be an optimum value according to (Russell and Yoon, 2008).

5.5.1 Evaluating privacy-preserving DWT-based CF schemes

DWT and **PPDWT** schemes are compared with respect to the quality of produced predictions by varying number of neighbors (k) utilized in prediction production process from 10 to 200. Similar to the reasons referred in Chapter 4, σ_{max} and β_{max} parameters are kept constant at 2 and 25%, respectively for data disguising procedure. Due to randomization, the experiments are repeated 100 times and average of the outcomes are demonstrated. Overall accuracy results of **DWT** and **PPDWT** schemes are presented in Fig. 5.2.



(a) MLP Data Set (b) MLM Data Set

Figure 5.2: MAE values for varying k values

For **DWT** scheme, 10 and 50 neighbors provide the best results for MLP and MLM, respectively, as seen from Fig. 5.2. With increasing number of neighbors from 10 to 100, accuracy becomes stable for MLM. For neighbors larger than 100, the quality of the predictions becomes worse. For MLP, accuracy decreases with increasing number of k up to 100 and becomes stable after that. For **PPDWT** scheme, trials to determine the optimum k show that incorporating 25 neighbors for MLP with an MAE of 0.8695 produce the best results. Similarly, for MLM, 10 neighbors with an MAE of 0.8308 and 25 neighbors with an MAE of 0.8335 achieve the best results. As seen from Fig. 5.2, it is not reasonable to collaborate with more than 25 neighbors, because accuracy gets worse with more than 25 neighbors for **PPDWT** scheme.

5.5.2 Evaluating item ordering methods

Due to the randomness used to preserve users' privacy, accuracy is expected to decrease. As shown by previous experimental results, accuracy slightly becomes worse due to privacy-preserving measures. However, the items can be rearranged, as explained in Section 5.3.3, to reduce information loss caused by neglecting detail coefficients of transformation so that accuracy is improved. To investigate how item ordering affects accuracy, experiments are conducted using both data sets while utilizing the proposed two preprocessing methods to order items. After performing experiments 100 times, overall averages in terms of MAE are displayed in Fig. 5.3 for both data sets.

As can be seen from Fig. 5.3, the **SA** scheme does not have a positive effect on accuracy for MLP; however, it has a slightly encouraging achievement for MLM. Unlike the **SA** method, the **MA** scheme has much more optimistic consequences for both data sets. These results actually support the main motivation to arrange items to reduce information loss. The results presented in Fig. 5.3 belong to three-level transformation as stated previously and since

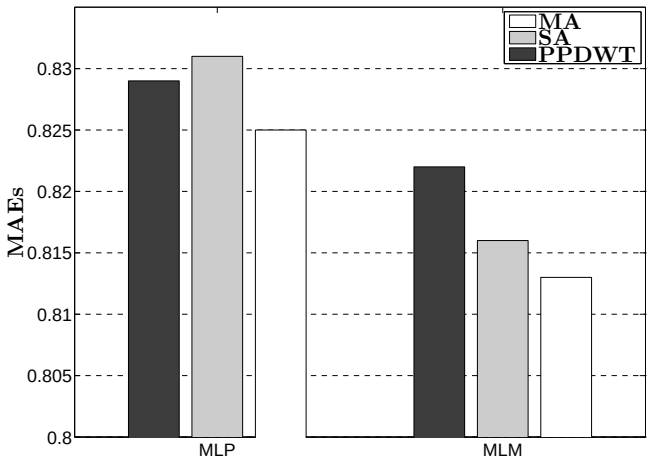


Figure 5.3: Accuracy vs. item ordering schemes for both data sets

an arrangement takes place at every step in the **MA** scheme, accuracy gets better at the final transformation step. However, in the **SA** scheme, there is one arrangement only, which takes place in the first transformation step. Therefore, the positive effect of arrangement is either lost or negated reaching up to third transformation. On the other hand, the **MA** scheme cumulates the enhancement through arrangement and produces more accurate results compared to **PPDWT**.

Although the proposed schemes, especially the **MA**, improve accuracy, such improvements do not entirely compensate the losses due to privacy concerns for MLP data set. However, as can be seen from the outcomes, they both enhance the quality of privacy-preserving scheme-based referrals. In addition, such transformation performs better in larger data sets, as expected. Especially, **MA** scheme recovers losses due to privacy protection mechanism in MLM data set.

5.6 Conclusions

DWT-based CF scheme is able to overcome the scalability problem and produce predictions with decent accuracy. However, it fails to protect privacy. Privacy-preserving schemes are proposed in order to alleviate privacy problem

that DWT-based CF scheme faces. Due to distortion on input data, the quality of the referrals becomes worse. However, compared to the original DWT-based CF scheme, relative errors due to proposed scheme are about 4.7% for MLP and 10% for MLM. Such losses can be considered acceptable because accuracy and privacy are conflicting goals. Thus, the proposed scheme is able to achieve privacy while sacrificing little from accuracy. Like accuracy and privacy, performance is also among the goals that should be accomplished. Applying privacy-preserving methods might introduce supplementary costs. Online performance is extremely critical for the success of CF systems. It is demonstrated that although the proposed scheme introduces additional costs, as expected, they are negligible and they do not immensely affect online performance.

In DWT-based CF schemes, item ordering is an important issue. DWT can be applied to naturally ordered items. However, it is hypothesized that items can be ordered based on some metrics in such a way so that accuracy might be enhanced. For this purpose, two item arrangement methods are proposed. Although single arrangement scheme does not perform well for one of the sets, it is shown that multiple arrangement approach improves accuracy. Empirical results show that multiple arrangement approach is able to reimburse some of the losses while not introducing any extra online costs.

6. A COMPARISON OF CLUSTERING-BASED PRIVACY PRESERVING COLLABORATIVE FILTERING SCHEMES

In this chapter, an item features-based profiling of users is proposed to overcome sparsity while performing clustering. Then, applying k-means clustering, fuzzy c-means method, and SOM clustering while preserving users' confidentiality is studied in order to cope with scalability and accuracy problems of PPCF schemes. Empirical outcomes demonstrate that privacy-preserving methods are able to offer precise predictions and fuzzy c-means method achieves the best low cost performance due to its approximation-based model.

6.1 Introduction

Prediction estimation process of user-based PPCF systems basically automates the old habit of "word-of-mouth". However, all so-called dependable PPCF recommendations rely on doubted past preferences of existing users, which might simply be imprecise or even inconsistent. At this juncture, due to their uncertainty and approximation-based performance, PPCF technologies can be subject to soft computing methods such as fuzzy logic and neural computing rather than conventional (hard) computing techniques.

It is proposed to employ a feature-based profiling (FBP) of users' ratings based on item categories to get rid of the effects of sparse nature of user-item matrix and empower separation skills of clustering algorithms. Such profiling performs a mapping of rating profiles of users onto a category-based profile. Feature-based profiles are much smaller and dense facilitating to determine similarities with decent accuracy and operate clustering algorithms more efficiently. Then, it is proposed to employ some well-known non-hierarchical clus-

tering algorithms, namely, k-means clustering (KMC), fuzzy c-means method (FCM), and SOM clustering to PPCF schemes and provide a comparison in terms of accuracy and performance among them. It is especially expected to enhance tractability and robustness of PPCF schemes by integrating them with fuzzy and neural computing approaches. This chapter presents the first analysis and evaluation on integrating uncertainty-based soft computing constituents on PPCF framework.

6.2 Clustering Algorithms and Clustering-based Collaborative Filtering

In CF applications, clustering is widely used to cope with the scalability problem. It reduces the size of data set involved in CF process. KMC is probably the most well-known clustering approach, where k random initial objects are chosen as centers one for each cluster (MacQueen, 1967). Then, n objects are compared with each seed by means of discrimination criterion and assigned to the closest cluster. This procedure is performed repeatedly and at each stage, cluster centers are recalculated as the average of objects assigned to corresponding cluster. The algorithm converges when the modification in cluster centers between successive stages is close to zero or less than a pre-defined value. At the end, each object is assigned to one cluster only.

Like KMC, FCM also requires c pre-defined initial objects as cluster centers (Bezdek, 1981; Yang et al., 2009). At each stage of the algorithm, a membership value for every object is estimated to each cluster center using a comparison criterion. The cluster centers are recalculated at each stage according the membership of objects to clusters as their weighted average. At the end, the algorithm provides the degree of membership to each cluster center and according to those memberships; objects either can be included in more than one cluster or to the cluster for which the degree of membership is

higher. Unlike KMC, FCM is more flexible [25] because it provides objects, which might have more than one interface with clusters. Such objects deserve more attention to figure out why they contribute to more than one cluster. In mathematical terms, given a data set $D = \{x_1, x_2, \ldots, x_n\}$, FCM minimizes the objective function given in Eq. 6.1, with respect to U (a fuzzy c-partition of the data set) and to V (a set of c prototypes), where Q is a real number greater than 1, V_i is the centroid of cluster i, u_{ij} is the degree of membership of object x_j belonging to cluster i, $d^2(\cdot,\cdot)$ is an inner product metric, and c is the number of clusters. The parameter Q controls the "fuzziness" of the resulting clusters (Gan, 2007).

$$J_Q(U,V) = \sum_{j=1}^{n}\sum_{i=1}^{c} u_{ij}^{Q} d^2(x_j, V_i). \tag{6.1}$$

SOM is a type of artificial neural network reducing dimensions by producing a map of usually one or two dimensions that plots the similarities of the data by grouping similar objects together (Kohonen, 1989). SOM architecture consists of two fully connected layers: an input layer and a Kohonen layer. The number of neurons in the input layer matches the number of attributes of the objects. Each neuron in the input layer has a feed-forward connection to each neuron in the Kohonen layer. The neuron in the Kohonen layer with the biggest input will become the winning neuron. The algorithm forms the SOM by first initializing the weights in the network by assigning them small random values (Gan, 2007). It then follows three essential procedures: competition, cooperation, and adaptation.

Aforementioned clustering algorithms can be employed as a preprocessing to group similar users in the same cluster and dissimilar ones in diverse clusters off-line in order to enhance online performance or scalability of CF schemes. After collecting users' preferences, the server constructs an $n \times m$

user-item matrix U. Then, it groups users into c clusters using different clustering approaches off-line. When an active user a requests a prediction for a target item q, she sends her known ratings and a query to the server during an online interaction. The server determines a's similarity to each cluster center using PCC (Eq. 2.1). Once a's cluster is determined, the similarities between a and all her cluster members are calculated via PCC, similarly. Notice that after clustering, the server estimates n/c similarities instead of n similarities online. The nearest k users in that cluster are chosen as neighbors. Finally, a prediction is produced as a weighted average of those neighbors having a rating on q using Eq. 2.2.

6.3 Privacy-preserving Clustering-based Recommendation Schemes

To achieve individual users' privacy, RPTs are applied, as explained in Section 2.2. Estimating users' FBPs, performing clustering on masked data, and producing predictions online on clustering are explained in the following.

6.3.1 Estimating feature-based profiles

Similarity weights between cluster centers and user profiles must be estimated in order to cluster users. Calculating the similarity weights in CF relies on comparing commonly rated fields, as seen from Eq. 2.1. However, number of items increases constantly and provided ratings on whole product range remains a very small fraction of them leading the sparsity problem. Difficulty of computing similarities accurately is even more amplified due to the sparsity. Even if there are commonly rated items, since the number of common ratings will be very few, similarity calculation cannot be accurate and thereof clustering algorithms cannot perform well. Although individual ratings are independent, items correlate among themselves as they have common features, where number of features are naturally very much less than number of items.

In order to overcome such shortcomings, instead of employing collected rating/preference profiles of users in clustering stage, it is proposed to generate additional profiles on categories of items of which a preference exists (Bilge and Polat, 2011). Individual ratings are independent, but somehow they are correlated due to common features. The objective here is to reduce typically sparse and large rating-based vectors into smaller and dense FBPs. Hence, it becomes possible to compute similarities even if there are no or very few common ratings. Moreover, since such FBPs are more compact and fixed in size, amount of time to calculate similarities becomes stabilized. To generate FBPs, it is first needed to resolve the categories of the product scale. Therefore, if such categories are determined, then histograms of absolute frequencies can be produced by giving a weight to corresponding features of a rated item. Those weights can be simply equal to each other or can also reflect rankings between preferences. Two approaches are proposed to generate FBPs, as explained in the following:

Purchase-based profiles (PBPs). PBPs are generated by checking whether an item is rated or not; and if it is rated (means previously purchased), then corresponding feature categories of that item are incremented by 1. In other words, if the user purchased an item, each category that item belongs to is increased by 1. Note that this profiling approach focuses on market basket data rather than individual opinions of users' on items.

Rating-based profiles (RBPs). RBPs are generated by checking whether an item is rated or not again; and if it is rated (means an opinion is provided on item), then regarding feature categories of that item are augmented by matching rating value. In other words, each rated item's categories are increased by as much as their matching rating values. RBP scheme focuses on how much users like or dislike certain types of item categories.

Table 6.1: A sample user-item matrix

	i_1	i_2	i_3	i_4	i_5
Alice	2	$\perp$	3	$\perp$	1
Bob	$\perp$	2	$\perp$	4	$\perp$

Table 6.2: Genre features of movies

	Comedy	Drama	Romance	Action	Fantasy
i_1	1	1	1	0	0
i_2	0	1	0	0	0
i_3	1	0	1	0	0
i_4	0	0	0	1	1
i_5	0	1	0	0	1

To understand FBP producing procedure, it is better to examine a small user-item matrix, given in Table 6.1, including ratings for movies in a 5-star scale, where $\perp$ indicates an unrated item. Suppose that genre features of movies are also provided as having at least one from the set [Comedy, Drama, Romance, Action, Fantasy] in Table 6.2. FBPs (RBPs and PBPs) are generated and depicted in Fig. 6.1 according to the example given in Tables 6.1 and 6.2.

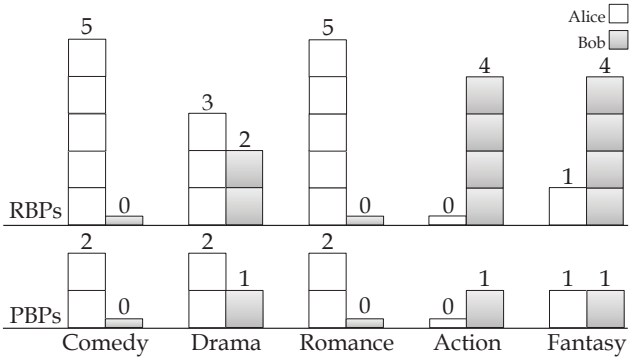


Figure 6.1: Feature-based profiles for Alice and Bob

As seen from Fig. 6.1, Alice's PBP is [2, 2, 2, 0, 1] because there are two items, i_1 and i_3, whose genre includes Comedy and she rated both items; hence, corresponding PBP value is 2; similarly, there are two items, i_4 and i_5, whose genre includes Fantasy and she rated one of them only; hence, corresponding PBP value is 1, and so on. Similarly, Bob's PBP can be obtained as [0,

1, 0, 1, 1]. For the same genres, as seen from Fig. 6.1, Alice's and Bob's RBPs are [5, 3, 5, 0, 1] and [0, 2, 0, 4, 4], respectively. Since there are two items, i_1 and i_3, whose genre is Romance and Alice rated both items as 2 and 3, respectively, the corresponding RBP value is 2 + 3 = 5. Similarly, since there is only one item (i_4) whose genre is Action and Bob rated it as 4, the corresponding RBP value is 4 for that genre. Moreover, obtained FBPs should be normalized because in real-world data sets, every user rates different number of items. Similarly, number of features might be different for various items, as i_1 has three features while i_2 having only one in this example. Due to these reasons, produced FBPs might fluctuate. In order to smooth such effects, FBPs are normalized by dividing each profile value by the sum of profile. Therefore, PBPs are updated as $[\frac{2}{7}, \frac{2}{7}, \frac{2}{7}, 0, \frac{1}{7}]$ and $[0, \frac{1}{3}, 0, \frac{1}{3}, \frac{1}{3}]$ for Alice and Bob, respectively. Similarly, RBPs are normalized as $[\frac{5}{14}, \frac{3}{14}, \frac{5}{14}, 0, \frac{1}{14}]$ and $[0, \frac{2}{10}, 0, \frac{4}{10}, \frac{4}{10}]$ for Alice and Bob, respectively. Note that similarity between these two users cannot be calculated over rating profiles as they do not have any co-rated items; however, it can be calculated over their FBPs and such circumstances are not rare on real-world data sets.

6.3.2 Performing clustering on perturbed data

After estimating FBPs corresponding to collected disguised user vectors, the server runs a clustering algorithm to group users relying on their FBPs. Given U, it is an easy task to group users; however, it is a challenge to cluster users when the server holds the disguised user-item matrix U'. Thus, it can cluster U' using aforementioned clustering methods, as follows:

KMC on perturbed data. The KMC algorithm can be considered in two distinct phases: (i) initialization phase in which the algorithm randomly assigns the users into c pre-defined clusters and (ii) iteration phase, where the distance (or similarity) between each user and each cluster center is computed

and the user is assigned to the nearest cluster. The cluster centers of any changed clusters is recomputed as the average of members of each cluster.

In the initialization phase, the server uniformly randomly assigns the users into c pre-defined clusters. It then estimates the cluster centers by taking the average of members of each cluster. Given n disguised data items, x_i' for $i = 1, 2, \ldots, n$, where $x_i' = x_i + r_i$, their average can be estimated, as follows:

$$\overline{x_i'} = \frac{\sum_{i=1}^{n}(x_i')}{n} = \frac{\sum_{i=1}^{n}(x_i + r_i)}{n} = \frac{\sum_{i=1}^{n}(x_i)}{n} + \frac{\sum_{i=1}^{n}(r_i)}{n} \approx \frac{\sum_{i=1}^{n}(x_i)}{n}. \tag{6.2}$$

Since the random numbers are drawn from a distribution with zero mean, the expected value of their mean is zero. With increasing n values, their averages will converge to zero.

In the second step, the server needs to find out similarities between users and cluster centers and updates cluster centers. PCC is employed as the discrimination criterion among users; and it can also be used to find out similarity between an individual user and a cluster center based on perturbed data, as formulated in Eq. 2.4. After estimating similarity weights, each user is assigned to the closest cluster. The server then updates clusters centers, as explained previously, using Eq. 6.3, as follows:

$$\begin{aligned} C_i &= \frac{\sum_{j=1}^{N_{C_i}} z'_{uk}}{N_{C_i}} = \frac{\sum_{j=1}^{N_{C_i}} (z_{uk} + r_{uk}}{N_{C_i}} = \frac{\sum_{j=1}^{N_{C_i}} z_{uk}}{N_{C_i}} + \frac{\sum_{j=1}^{N_{C_i}} r_{uk}}{N_{C_i}} \\ &\approx \frac{\sum_{j=1}^{N_{C_i}} z_{uk}}{N_{C_i}}, i = 1, 2, \ldots, c \end{aligned} \tag{6.3}$$

in which C_i represents cluster center of ith cluster, N_{C_i} stands for the number of users in regarding cluster and c is the number of clusters. As a result, the server can compute the similarities between users and calculate cluster centers from perturbed data; hence, it performs KMC with adequate precision.

FCM on perturbed data. FCM is an extension of KMC for fuzzy clustering and it can be roughly performed in three steps: (i) initialization step similar to KMC to choose initial c centroids for clusters, (ii) computing a membership matrix holding relationship levels of users to clusters, which is calculated as a function of similarity degrees of each user to cluster centroids, and (iii) updating each cluster centroid as weighted averages of each users' membership to each cluster centroid. It repeats steps (ii) and (iii) until a termination criterion is reached.

The only difference between FCM and KCM is that a user can belong to more than one cluster in FCM and cluster centers are estimated as weighted averages in FCM. Basically, the same estimations are conducted in FCM. Thus, the first step is easily achievable. Also, since similarity values appear in the second step and a weighted average of contributions of each member is computed in the third step of the algorithm, FCM can certainly be performed on U', relying on the equivalences expressed in Eqs. 2.4 and 6.3, respectively.

SOM on perturbed data. SOM includes three essential processes after the initialization procedure (Gan, 2007): (i) a competitive process to determine the winning neuron, (ii) a cooperative process to define a topological neighborhood for locating the center for cooperating neurons, and (iii) an adaptive process, where weight vectors are updated according to the input vectors. Let $\mathbf{u} = (z'_{u_1}, z'_{u_1}, \ldots, z'_{u_d})^T$ be a user in U' selected randomly, where d is the dimension of FBPs and the weight vector of neuron n in the Kohonen layer be $\mathbf{w}_n = (w_{n_1}, w_{n_2}, \ldots, w_{n_N})^T$, where $n = 1, 2, \ldots, N$ and N is the number of neurons in the Kohonen layer. To find the best match of user u with the weight vectors $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N$, inner products $\mathbf{w}_*^T\mathbf{u}$ is computed and the largest is chosen.

The calculations can be approximately done on U', as shown in Eq. 6.4.

$$\begin{aligned} \mathbf{w}_*^T\mathbf{u} &= \mathbf{w}_*^T z'_{u*} = \mathbf{w}_*^T(z_{u*} + r_{u*}) = w_{*1}(z_{u1} + r_{u1}) + \cdots + w_{*N}(z_{ud} + r_{ud}) \\ &= w_{*1}z_{u1} + w_{*1}r_{u1} + \cdots + w_{*N}z_{ud} + w_{*N}r_{ud} \\ &\approx w_{*1}z_{u1} + \cdots + w_{*N}z_{ud} \end{aligned} \tag{6.4}$$

in which r_{u*} terms are generated from a distribution with zero mean; hence, their multiplications with $\mathbf{w}*$ terms converge to zero and the equation holds.

In the second step, the topological neighborhood (h_{jt}) is calculated, which is a unimodal function calculated over the distance between winning neuron t and excited neuron j. It can be calculated as the simple Euclidean distance. However, since this function is computed employing none of the elements of U', but two neurons, t and j, it can easily be found unaffectedly of RPTs process. In the last step, the adaptation process, the weight vector $\mathbf{w}_j$ of neuron j changes due to user $\mathbf{u}$. Given the weight vector $\mathbf{w}_j^{(s)}$ of neuron j at iteration s, the new weight vector $\mathbf{w}_j^{(s+1)}$ is calculated, as given in Eq. 6.5

$$\mathbf{w}_j^{(s+1)} = \mathbf{w}_j^{(s)} + \eta(s)h_{j,i(u)}(s)(\mathbf{u} - \mathbf{w}_j^{(s)}), \tag{6.5}$$

where $\eta(s)$ is the learning parameter and is a constant; and $h_{j,i(u)}(s)$ is the neighborhood function defined in the previous step. Therefore, it is necessary to focus on $\mathbf{u} - \mathbf{w}_j^{(s)}$ differences. Since $\mathbf{u}$ comprises of random values added upon z-score values, the effect of random values will converge to zero in such a difference computation as they are generated from a distribution with zero mean. As these three steps of SOM clustering can be performed without greatly affected from disguising procedure, it can be concluded that SOM clustering can be performed precisely on U', without jeopardizing individuals' confidentiality.

6.3.3 Producing online recommendations privately

Typical CF-based recommendation estimation can be considered as a three-step process: (i) calculating similarities between a and train users, (ii) locating the nearest neighbors, and (iii) estimating a weighted prediction relying on the most similar users' rating on q.

After determining a's cluster, similarities are calculated between a and each user in her cluster using Eq. 2.4 based on perturbed data. After finding similarities, the most similar k users are selected as neighbors to contribute to the prediction process. Those users whose similarity weights satisfy a predefined threshold can be chosen as neighbors. Alternatively, after sorting the users decreasingly according to similarities, the first k users are selected as neighbors. After forming the neighborhood, p_{aq} can be estimated using Eq. 2.5 and Eq. 2.6.

6.4 Overhead Costs Analysis

It is imperative to analyze the proposed FBP scheme, applied clustering algorithms, and privacy preservation structure in terms of both off-line and online costs to define their effects on overall performance of offered PPCF schemes.

Storage cost of traditional CF schemes is in the order of $O(nm)$. Employed clustering algorithms and privacy preservation schemes do not introduce any extra storage costs. However, producing PBPs and/or RBPs requires an additional space in the order of $O(nd)$ to record such FBPs, where d is the constant number of item features.

Supplementary communication costs can be considered in two aspects: (i) the number of communications and (ii) the amount of transferred data. Communication costs with respect to these two aspects do not change in off-line and online phases of the proposed clustering-based PPCF schemes.

Additional computation costs are also expected. Off-line phase includes two stages: (i) generating FBPs and (ii) obtaining a model through clustering. FBPs are generated in the order of $O(nm)$ because it checks all items of each users in the matrix. Running times of KMC, FCM, and SOM clustering methods are in the order of $O(n^{dc+1} \log n)$, $O(ndc^2)$, and $O(n^2)$, respectively. Actual performance of a recommender system is defined by its response time to queries, which is determined by online computation load. In the proposed scheme, the server estimates a prediction in an identical manner to the one it does with non-private approach, which has an online running time in the order of $O(n^2m)$. However, due to clustering efforts, number of neighbors to calculate similarities between them reduces significantly. Defined clustering-based PPCF approach requires three steps to produce an online prediction. First, when a sends her data with a query, corresponding FBP of a is generated by the server in the order of $O(m)$ time because there are m items. Second, after obtaining FBP, the cluster, which a belongs to is determined by calculating similarity between a and all cluster centers. Having c clusters, calculating such similarities via dot product is performed in the order of $O(mc)$ time. Finally, since users are grouped into distinct clusters, it might be assumed that there are approximately n/c users in each cluster. Actually, this might not be the case for FCM since users may be included in more than one cluster. Therefore, it might be assumed that there are at most $n/2$ users in each cluster for FCM. In such scenario, online running time of the proposed approach is in the order of $O(n^2m/c^2)$. As can be seen from this complexity expression, the higher the number of clusters, the lesser the online running time is.

6.5 Experimental Evaluation

Experiments are conducted to investigate how explained three clustering algorithms perform along with FBP method in terms of accuracy and efficiency

on PPCF schemes. Evaluations are performed on MLP data set in which movies contain at least one of 18 predefined genre categories. Experiments are realized using a five-fold cross-validation experimentation methodology in which the data set (U or U') is uniformly randomly divided into five subsets. One of the subsets (U_i or U'_i) is used as the test set and the remaining four subsets are used as the train set in each iteration i, where $i = 1, 2, \ldots, 5$. After train and test users are uniformly randomly obtained, five rated items' votes for each test (active) user are withheld, replaced their entries with null, tried to be predicted, and compared against actual ratings. Experiments in privacy-preserving framework are repeated 100 times due to randomization and averages of results are presented. Trials are performed in MATLAB 7.9.0 environment using a computer with an Intel Xeon 2.8 GHz processor and 6 GB RAM. For clustering operations, MATLAB's built-in functions are utilized with default options except choosing '`correlation`' distance measure (utilizes PCC) for KMC, 'Q' as 2 for FCM, and one-dimensional feature-map through 500 epochs for SOM.

First, three clustering algorithms are employed along with the proposed profiling scheme in CF framework in order to show how they affect performance and accuracy without considering privacy issues. Then, such approach is employed in privacy-preserving framework in the same manner. In both frameworks, clusters are formed relying on FBPs of users via all three clustering algorithms, then original rating-based user-item matrix is utilized, either U or U', and a prediction is estimated after calculating similarities based on rating profiles.

6.5.1 Results and discussion

The parameters to test for CF and PPCF schemes throughout the experiments are number of neighbors in recommendation process (k), pre-defined number

of clusters (c) for clustering algorithms, and σ_{max}. Detailed procedures and results of conducted experiments are explained in the following.

Number of neighbors (k). First, trials are performed by varying the number of neighbors contributing to the recommendation estimation process in both traditional CF and PPCF frameworks without employing clustering algorithms. Optimum k values are searched, where k is varied from 3 to 100. For PPCF scheme, β_{max} is kept at 25 and σ_{max} at 1. MAEs are estimated and displayed in Fig. 6.2.

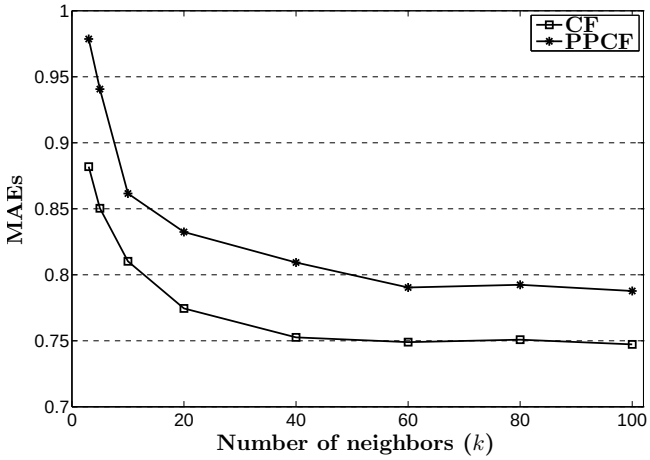


Figure 6.2: MAEs for varying number of neighbors (k)

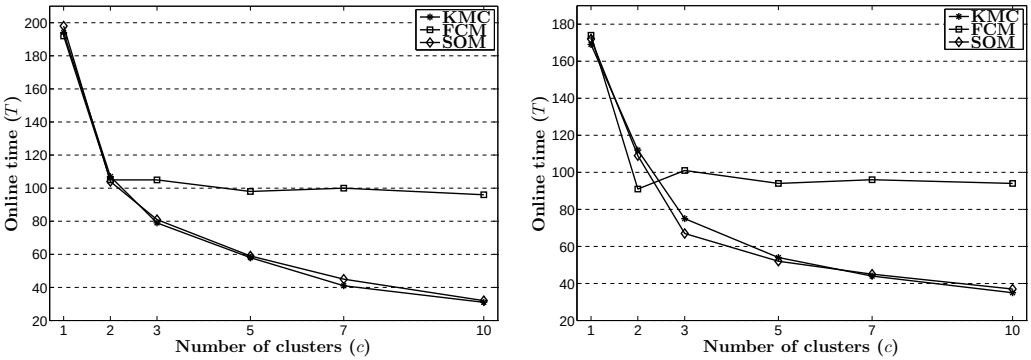
Accuracy improves with increasing number of neighbors joining to the recommendation estimation process, as seen from Fig. 6.2. However, while the improvement is significant while k moves from 3 to 60, it becomes stable around 100 neighbors for both CF and PPCF schemes. On the other hand, employing more than 100 neighbors might negatively affect the scalability problems, as stated in (Herlocker et al., 2004). Thus, k parameter is fixed at 100 neighbors to contribute the prediction estimation process in the rest of the experiments.

Number of clusters (c). Next trials are conducted to investigate effects of varying c values on performance and accuracy while running clustering algorithms. Obviously, as c increases, online time T spent on producing predictions declines. However, increasing c may also cause losses in accuracy. Therefore,

the behavior of clustering algorithms is investigated and an optimum value is tried to be found by varying c from 1 to 10 in both CF and PPCF schemes. Note that evaluating c=1 means there is no clustering at all, which constitutes a baseline to compare the performance of clustering approaches. Other parameters are kept constant at k=100 for both schemes, β_{max}=25 and σ_{max}=1 in PPCF scheme. Accuracy results are displayed in Table 6.3 and Table 6.4 including both profiling schemes for CF and PPCF frameworks, respectively. Note that since both profiling schemes (PBP and RBP) show very similar trends for the overall averages of T values, results are depicted for PBP scheme only in Fig. 6.3 for CF and PPCF schemes.

Table 6.3: Comparing PBP and RBP for varying number of clusters for CF

	c					
	1	2	3	5	7	10
PBP						
KMC	0.7519	0.7561	0.7635	0.7732	0.7881	0.8054
FCM	0.7519	**0.7543**	**0.7586**	**0.7595**	**0.7588**	**0.7620**
SOM	0.7519	0.7549	0.7668	0.7800	0.7886	0.8103
RBP						
KMC	0.7519	0.7604	0.7708	0.7837	0.8114	0.8293
FCM	0.7519	**0.7603**	**0.7634**	**0.7706**	**0.7774**	**0.7803**
SOM	0.7519	0.7604	0.7748	0.8050	0.8151	0.8379



(a) Non-private scheme (b) Privacy-preserving scheme

Figure 6.3: Online time T (in seconds) for varying number of clusters

As can be followed from Table 6.3, when all three clustering algorithms are employed along with both proposed profiling schemes without preserving privacy, all algorithms produce similar MAE values, where FCM performs

slightly better than KMC and SOM, which are depicted in bold. There is a slight decrease (from 0.7519 to 0.7543) in accuracy for c=2 while T almost halves in FCM. Furthermore, it can be seen that MAE values for both profiling schemes show similar trends, where PBP performs insignificantly better than RBP. Likewise, it can be followed from Fig. 6.3(a) that, as c increases, online time T spent on producing predictions reduces because number of users for whom it is needed to calculate similarity weights decreases. FCM produces predictions around 100 seconds for all values of c >1. Therefore, it can be concluded that c=2 is the best option for FCM because MAE value corresponding to two clusters is the smallest, which is about 0.7543.

Table 6.4: Comparing PBP and RBP for varying number of clusters for PPCF

	c					
	1	2	3	5	7	10
PBP						
KMC	0.7976	0.7956	0.8010	0.8170	0.8328	0.8580
FCM	0.7977	**0.7943**	**0.7930**	**0.7957**	**0.7941**	**0.7944**
SOM	0.7977	0.7946	0.7979	0.8128	0.8321	0.8661
RBP						
KMC	0.7980	**0.8033**	**0.8164**	**0.8488**	**0.8529**	**0.8630**
FCM	0.7973	0.8734	0.9005	0.9027	0.9029	0.9047
SOM	0.7967	0.9157	0.9139	0.9153	0.9154	0.9167

When privacy measures are employed, clustering algorithms perform even better results compared to no cluster approach. As shown in Table 6.4, all three algorithms show very close MAE values to original PPCF approach with PBP scheme, where FCM is the best among all (depicted in bold). When c=3, FCM performs the smallest MAE, which is slightly better than the base approach (0.7977 for base and 0.7930 for FCM). Moreover, KMC and SOM produce similar outcomes in terms of MAE values with high accuracy. However, RBP scheme is not able to produce accurate referrals. The reason for this consequence is that RBP is affected much from privacy protection protocol. Within the protocol, each rating is disguised with random numbers and additionally fake random ratings are forged into the profiles. Although each existing rat-

ing, whether it is genuine or fake, reflects as an increment by 1 to PBPs; they affect RBPs proportional to their values. However, such values are perturbed and they distort RBPs much more than they do PBPs. Online time T needed to produce predictions shows similar trends while it does without preserving confidentiality, as seen in Fig. 6.3(b). KMC and SOM produce predictions in a logarithmically decaying time scale for increasing c; on the other hand, FCM again shows a non-improving behavior in terms of T due to fuzzy clustering. In short, while producing predictions with privacy concerns, it can be followed that PBP scheme performs better than RBP and while FCM performs better for all levels of c in terms accuracy, but it can only improve T by twice. Thus, it might be useful to employ KMC or SOM if scalability concerns thwart accuracy.

Overall performance by varying σ_{max}. Final set of experiments are conducted to show varying effects of data distortion level for all clustering algorithms in PPCF framework. Choosing σ_{max} totally depends on individuals' demand on required privacy levels. Therefore, for this search, σ_{max} parameter is not fixed; but, it is varied to see all opportunities. However, k is set at 100, c at 2, and β_{max} at 25 for all algorithms and PBP scheme is employed. Results are presented in Table 6.5, where also MAE and T values are included for traditional CF algorithm (mentioned as Base) to form a comparison opportunity to the reader.

Table 6.5: Overall performance with varying σ_{max} values.

σ_{max}	Base	KMC	FCM	SOM
MAE				
0.5	0.7519	0.7884	0.7853	0.7868
1	0.7519	0.7935	0.7915	0.7918
2	0.7519	0.8135	0.8080	0.8108
3	0.7519	0.8622	0.8519	0.8562
4	0.7519	0.9143	0.9079	0.9055
T (s)	199	116	91	104

As seen from Table 6.5, the quality of the predictions naturally gets worse due to privacy concerns. However, although there are accuracy losses due to privacy protection scheme, such losses are not vast. It is still possible to provide recommendations with pretty good accuracy employing the proposed FBP scheme and a clustering approach. Among examined clustering approaches, FCM performs the best in terms of both accuracy and online performance. Compared to the base algorithm, it produces 4,715 predictions in 91 seconds compared to 199 seconds, which means a 45.7% enhancement in online time T. Significance of obtained MAE values are presented by statistical t-tests for all clustering algorithms. Significance tests are performed by comparing the five-fold MAE results mutually between traditional CF algorithm and one of KMC, FCM, and SOM clustering algorithms. One-tailed t-tests are performed for all σ_{max} values of 0.5, 1, 2, 3, and 4. It is hypothesized that the difference between traditional CF algorithm's and each clustering algorithms' accuracy values are not significantly diverse. Significance test results are presented in Table 6.6.

Table 6.6: Statistical significance of the differences

σ_{max}	KMC	FCM	SOM
0.5	t=1.34*	t=1.24*	t=1.27*
1	t=1.48*	t=1.43*	t=1.42*
2	t=2.29**	t=2.03*	t=2.16**
3	t=4.06	t=3.48	t=3.84
4	t=6.27	t=6.04	t=6.09

For σ_{max} values of 0.5 and 1, t statistics indicates that the difference between traditional algorithm's and each clustering algorithms' accuracy is not significantly different with a confidence level of 90% (indicated with *), as seen from Table 6.6. Also, when σ_{max} is increased to 2, the results are still not significantly different with a confidence level of 95% (indicated with **). However, for σ_{max} values of 3 and 4, the differences between MAE values seem to be significant, which is expected due to level of randomization. Since input

vectors are disguised with random numbers distributing in a considerably large range, privacy enhances with the cost of much accuracy loss. Bearing the trade-off in mind, a σ_{max} value of 2 seems harmonizing between privacy requirements and accuracy losses. In conclusion, profiling and privacy protection schemes along with clustering algorithms achieve confidentiality while sacrificing little on accuracy but bringing a lot in scalability.

6.6 Conclusions

Clustering-based CF schemes' privacy problem is relieved using RPTs. As expected, accuracy becomes worse due to privacy measures. However, compared to the non-private scheme, relative errors due to privacy are not major as shown with significance tests. Since accuracy and privacy are conflicting goals, such losses are inevitable and can be considered acceptable. Thus, the proposed scheme is able to protect confidentiality without jeopardizing on accuracy. At least as important as accuracy and privacy, performance is also among the goals that should be accomplished in CF. Applied clustering algorithms serve well for scaling PPCF systems reducing online running time. Especially the fuzzy approach offers a low cost solution for producing qualified referrals and enhances robustness of PPCF systems by its approximation-based model.

An FBP approach is proposed to overcome sparsity related similarity calculation problems in CF systems. By taking advantage of common features among items, it is possible to produce much smaller and dense feature-based profiles to be utilized in clustering processes. Although the proposed two distinct approaches of producing feature-based profiles perform very well in non-private CF approach, RBP scheme is affected much from data disguising and not able to produce predictions with high accuracy. However, PBP scheme performs well in PPCF schemes along with clustering. Additionally, due to off-line computations, they do not cause any extra online costs.

7. A SCALABLE PRIVACY-PRESERVING RECOMMENDATION SCHEME VIA BISECTING k-MEANS CLUSTERING

In this chapter, a novel PPCF scheme based on bisecting k-means clustering is proposed in which two preprocessing methods are applied. The first preprocessing scheme deals with scalability problem by constructing a binary decision tree through a bisecting k-means clustering approach while the second produces clones of users by inserting pseudo-self-predictions (PSPs) into original user profiles to boost accuracy of scalability-enhanced framework. Sparse nature of collections is handled by transforming ratings into item features-based profiles. Empirical outcomes verify that combined effects of the proposed preprocessing schemes relieve scalability and augment accuracy significantly.

7.1 Introduction

CF and PPCF schemes are classified as (*i*) memory-based, (*ii*) model-based, or (*iii*) hybrid approaches (Breese et al., 1998; Al-Shamri and Bharadwaj, 2008; Herlocker et al., 2004). Also, such methods might take product contents into account. Bisecting k-means clustering is attracting attention mostly in the fields of where lots of data present to be distinguished and analyzed, such as object tracking (Dubuisson and Fabrizio, 2009), image processing (Thilagamani, 2011), and document clustering (Steinbach et al., 2000). In those fields, traditional clustering approaches fall short to process lots of data fast; thus, researchers propose clustering objects recursively to process images for tracking movement or graphical representations.

In this chapter, novel PPCF framework is proposed aiming at alleviating scalability and accuracy problems of state-of-the-art k-nearest neighbor-

based methods by applying a two-level preprocessing scheme. Initially, effects of sparseness on neighborhood formation is aimed to be relieved by a transformation of rating data into item features-based profiles (FBPs) relying on purchase data. Since number of item features is constant and extremely less than number of items, such FBPs function as a dimension reduction module. First preprocessing method focuses mainly on scalability issue and it is at the heart of the proposed framework. In such preprocessing, a binary decision tree (BDT) is constructed by recursively clustering FBPs via bisecting k-means algorithm to locate the nearest neighbors in a significantly rapid and robust manner. Second preprocessing method is motivated to boost accuracy of scalability-enhanced scheme by enhancing discrimination skills of bisecting k-means clustering approach. For this purpose, clones of original users are reproduced by inserting PSPs into user profiles. Proposed preprocessing framework is evaluated in both CF and PPCF frameworks. Empirical results demonstrate that prediction accuracy of the proposed solution is significantly higher compared to the k-nearest neighbor-based CF and PPCF methods.

7.2 Bisecting k-means Clustering

Clustering algorithms take n objects and assemble them into c clusters so that the members of a cluster are close to each other in terms of discrimination measure and the members of different clusters are diverse. To overcome scalability challenges, clustering has been employed as an off-line preprocessing tool to narrow search space in CF applications. Although traditional one-level clustering helps finding neighbors fast; however, it creates a trade-off for accuracy as number of clusters increases due to loss wisdom of the crowd. On the other hand, small number of clusters does not facilitate well for scalability. Even if number of entities raises drastically, there is a reasonable limit to increase number of clusters to provide predictions with reasonable accuracy.

k-means clustering is a well-known center-based clustering algorithm in which each cluster has a center called the mean (Gan, 2007). Number of clusters is preset and an error function is defined as the change in cluster centers or membership of data items between two consecutive epochs. Initially, k random objects are chosen as centers, one for each cluster. It proceeds by calculating similarities between all entities and each seed, assigns them to the nearest cluster, and recalculates cluster centers. This process is repeated until the error function does not change significantly or the membership of the clusters no longer changes.

Bisecting k-means algorithm is a divisive hierarchical clustering (Steinbach et al., 2000), which starts with a single cluster and splits it into two sub-clusters using k-means algorithm (bisecting step). It repeats this process recursively for either a desired number of times or a certain criterion is met. Note that it has a time complexity of linear with the number of entities. If the number of clusters is large, then bisecting k-means is even more efficient than the regular k-means algorithm (Steinbach et al., 2000).

7.3 A Novel Scalable PPCF Scheme

This section defines the proposed novel scheme based on two preprocessing methods, where users are supposed to interact with the CF system by explicitly submitting preferences on their ratings because implicit mechanisms are often less accurate and tractable (Adomavicius and Tuzhilin, 2005). It is first explained how a BDT is formed by applying a bisecting k-means clustering. Then, it is studied how to produce clone profiles to recover accuracy losses due to the first scheme. It is also shown that applied clustering method can be run on privately collected data.

7.3.1 Forming a BDT via bisecting k-means clustering

As a model-based CF technique, clustering-based CF is a valuable approach. Yet, it needs to be utilized in a more profound manner rather than one level application to alleviate its shortcomings. While clustering algorithms are very useful in discriminating entities relying onto a criterion, they best perform while dividing a set into two halves because it is more unlikely to happen very close membership values in case of two clusters, which intensify the algorithm's sensitivity. However, forming two clusters only would not relieve scalability issues much. Thus, applying a multi-level clustering is proposed by recursively bisecting input data into clusters and forming a BDT according to clustering results. Hence, it is aimed to produce tiny clusters containing very similar users at the leaf nodes of BDT and find the nearest neighbors of a newcomer by simply traversing down the tree.

Suppose that an optimal value of number of neighbors (N) to be utilized in prediction production process in a PPCF system is known. A BDT with at most N users in the leaf nodes can be constructed to efficiently form neighborhoods, as described in Algorithm 5. Given $U_{n\times m}$, the server first estimates FBPs of users ($F_{n\times d}$), where d is the dimension of features. Then, bisecting k-means clustering is employed and FBPs are divided into two distinct clusters at each level. Cluster centers are indexed to be used as a forwarding tool for each corresponding level, as well. If the number of profiles in any cluster exceeds N, then recursive bisecting continues to divide those clusters using the same approach and indexes cluster centers of each clustering process. This procedure continues repeatedly until leaves having at most N profiles are reached; thus, N can be thought as a stopping criterion. Finally, a BDT having indexed cluster centers as branch nodes and grouped similar profiles at leaf nodes is obtained. Related rating profiles-based tree structure can be formed easily.

During an online transaction with a, the PPCF server first generates FBP of a (F_a) and determines the leaf, which F_a belongs to by traversing down the tree. While traversing, two similarity calculations are performed at each level to find relationships to both cluster centers and higher resemblance decides the next hope, which is stored in IDX variable in Algorithm 5.

Algorithm 5 BDT formation via bisecting k-means clustering

function BKM(F, N) ▷ bisecting k-means cluster
Initialize:
  $IDX(n) \leftarrow 0$ ▷ to record forwarding paths as either 'left' or 'right'
  $BDT.centers_{2\times d} \leftarrow null$ ▷ to hold cluster centers
  $BDT.left \leftarrow null$ ▷ to hold left sub-tree elements
  $BDT.right \leftarrow null$ ▷ to hold right sub-tree elements
  $BDT.LST$: a new BDT ▷ Left sub-tree
  $BDT.RST$: a new BDT ▷ Right sub-tree
Cluster:
  $[IDX, BDT.centers]$ = k-means$(F, 2)$ ▷ divide into two clusters
  for all u_i in F ($i \leftarrow 1$ to n) **do**
    if $IDX(u_i)$ = "left" **then**
      append u_i into $BDT.left$
    else
      append u_i into $BDT.right$
    end if
  end for
  if size$(BDT.left) > N$ **then**
    $BDT.LST$ = BKM$(BDT.left, N)$
  end if
  if size$(BDT.right) > N$ **then**
    $BDT.RST$ = BKM$(BDT.right, N)$
  end if
  return BDT
end function

Locating exact leaf of F_a requires at most $2 \times (h-1)$ similarity computations, where h is the height of the BDT. The leaf, F_a goes into, can reference to at most N users; however, this amount might be fewer. Therefore, the server treats all references in that particular leaf and its sibling as potential neighbors, calculates exact similarities with all such users, and eventually, marks the nearest N of them as neighbors, which requires at most $2N$ additional

similarity computations. Although h is dependent to the value of n in the system, intuitively, both h and N are much less than n in systems suffering from scalability. Hence, once the tree is formed, at most $2 \times (h - 1 + N)$ similarity computations are done instead of n to form the neighborhood. In other words, number of computations performed to find similarities significantly reduces.

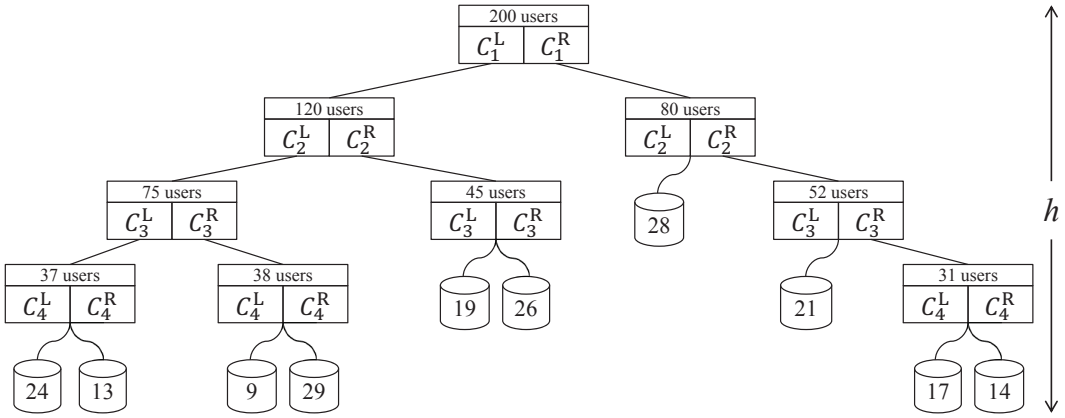


Figure 7.1: An example binary decision tree

An example BDT produced via bisecting k-means clustering can be seen in Fig. 7.1. Suppose that there are initially 200 users and N is chosen as 30 in this example. Users are divided into two clusters with 120 and 80 users at the first level. Corresponding cluster centers are indexed at the root of the tree as C_1^L and C_1^R, where superscripts indicate for which sub-tree the cluster center forwards (either left or right) and subscripts indicating the height of BDT so far, which is one for the case of root. Similarly, in the left sub-tree of root, 120 users are clustered into two groups of 75 and 45 users with branch node containing cluster centers C_2^L and C_2^R. Note that subscripts are incremented by 1 and superscripts show again the forwarding paths as left or right. Each branch node is repeatedly divided into two sub-clusters unless they contain N or fewer users. Finally, the BDT is completed with two cluster centers at each branch node to facilitate forwarding and tiny but very similar user groups at leaf nodes to enable forming accurate neighborhoods. For this specific example, locating neighbors of a new user requires at most $2(5 - 1 + 30) = 68$

similarity computations instead of 200, reducing the number of computations approximately three times. Since the time consuming part in CF algorithms is neighborhood formation, the effect of this preprocessing scheme on scalability can be seen more clearly as the number of users in the system increases.

7.3.2 Cloning users by producing PSPs

Considering the discrimination mechanism of k-means clustering over levels, correlations between users are utilized as distance criterion and they must be calculated precisely to construct accurate clusters. Such correlations are calculated over FBPs of users, which project rating preferences onto a feature-based dimension. Therefore, two users having similar tastes cannot be caught by the system unless they rate on items having the same features. Finding similar users relying on past preferences may not always result accurate findings due to sparseness. Consequently, discriminations over levels of BDT can be performed with limited accuracy. Although the first preprocessing scheme deals with scalability issues well, accuracy of recommendations cannot be jeopardized greatly. Hence, a second scheme is proposed to alleviate effects of sparseness and boost accuracy in CF systems.

Regarding similarity between users in terms of PCC, two users are similar if their preferences to co-rated items show concordance no matter what their uncommon ratings are. Even FBPs are utilized in similarity calculation, two users are required to rate on the same kind of items. The idea of this preprocessing scheme is to produce identical clones of an original user's rating profile in terms of PCC (having exact same ratings with the original user) and also having some additional ratings to unrated cells. Then, such clones' FBPs will also be included in the BDT formation process. Hence, if the system falls short to determine a high correlation between a's and an existing user's FBP, it can figure it out through one of its replicas. Since clones will have more

ratings, consequently, they will have more meaningful FBPs. Moreover, since each clone is identical to its parent in terms of PCC, it can be assumed that if a is similar to any clone profile, she also most likely be similar to the parent of which that clone is created. Although this process naturally increases the number of users in the system, the first scheme reduces the search space logarithmically. Therefore, while one scheme enhances discrimination capabilities of clustering approach, the other can still cover overheads caused by it.

The outline of the proposed cloning scheme is given as a pseudo-code layout in Algorithm 6. According to the algorithm, each produced clone includes all true ratings of the original user and additional PSPs to randomly chosen unrated cells. Such PSPs are estimated from the input-matrix itself using the traditional PPCF prediction method, i.e., the nearest neighbors of an existing user in U are determined through calculating PCC and predictions to unrated cells are estimated via Eq. 2.5. Also, each clone has PSPs to distinct random empty cells. Therefore, the correlation among a user and all of its replicas show the highest resemblance because their co-rated items are all the same. However, additional PSPs will change corresponding FBPs, which will help revealing uncovered relations between original users. To provide a base for tuning, it is offered to produce clones by inserting PSPs into the original profiles proportional to the number of existing ratings of each user. Such parameter is named as the *density*. Suppose that an original user has 100 ratings in her profile and five clones is required to be created through increasing the density by 50%, which means that 50 extra ratings per clone are added. Since five clones will be created, a total of 250 PSPs are estimated and then each replica will have initial 100 ratings along with additional 50 PSPs to distinct empty cells. After creating clones, FBPs of such clones are also generated and BDT is formed. Supposedly an original user and her clones will fall into the same leaf. However, while the BDT is created, clones will affect cluster cen-

ters, which will facilitate forwarding a new user to her neighbors. Note that the number of clones to produce and how much to increase density in clones are to be determined experimentally.

Algorithm 6 Generating clones of a user by PSPs

Input: User-item matrix ($U_{n\times m}$), User-id (id), Clone Count (ω), Density (ρ)

Initialize: $Clones_{\omega\times m} \leftarrow 0$ ▷ clones matrix
for $i \leftarrow 1$ to ω **do**
    $Clones(i) = U(id)$ ▷ create identical clones
end for
$n_r \leftarrow$ # of ratings in $U(id)$
$i_{ec} \leftarrow$ index of empty cells in $U(id)$
$r_{ec} \leftarrow$ RandomPermutation(i_{ec}) ▷ random index of empty cells
$n_{ec} \leftarrow n_r \times \rho$ ▷ # of empty cells to be filled
Calculate and sort similarities:
for all u_i in U ($i \leftarrow 1$ to n) **do**
    $similarities(i)$ = PCC($U(i), U(id)$)
end for
$sorted\_sim$ = SORT($similarities$, descending) ▷ to be used by PSP production
Produce PSPs for each clone:
for $i \leftarrow 1$ to ω **do**
    for $j \leftarrow 1$ to n_{ec} **do**
        $idx = (i-1) \times n_{ec} + j$
        $target\_item = i_{ec}(r_{ec}(idx))$
        $Clones(i, target\_item) \leftarrow$ PSP($sorted\_sim, target\_item$) ▷ estimated using Eq. 2.5
    end for
end for

7.3.3 Bisecting k-means clustering on perturbed data

After estimating FBPs of the collected disguised user vectors, the server forms a BDT by running Algorithm 5, where bisecting k-means clustering is performed through FBPs to cluster users. However, in a privacy-preserving environment, such profiles can only be estimated through disguised user vectors. Given a user-item matrix U, it is an easy task to cluster users; however, the server holds the disguised user-item matrix U' for privacy reasons. Bisecting k-means algorithm performs two different calculations using FBPs, i.e., estimating cluster

centers by taking average of members of clusters and calculating similarities between user profiles and cluster centers. Thus, how precise the server can cluster U' is analyzed.

Considering the FBP generation process for purchase-based profiles explained in Section 6.3.1, the server simply increments each vote's corresponding feature value by 1 and normalizes the vector at the end. Estimation of FBPs on perturbed data must be analyzed under two circumstances. As explained in Procedure 1, to protect their privacy, users (i) disguise their individual ratings and (ii) insert fake ones into their profiles. Indeed, disguising actual ratings does not affect the estimation of an FBP, because the server increments corresponding feature values by 1 no matters the related rating values are disguised. Hence, an original ratings vector and its disguised version shall produce the same FBP.

Remember that users forge additional ratings to hide their truly rated items. Nevertheless, due to normalization procedure of profiles, effects of those artificial ratings are diminished. Given m_g disguised genuine ratings for user u (u_i for $i = 1, 2, \ldots, m_g$), each element of F_u is estimated by incrementing the related feature values of u_i's, normalized by the $\sum_{j=1}^{m_g} d_j$, where d_j is the number of features for item j. Then, random artificial ratings are inserted into profile by $\beta\%$ of m_g and as m_g grows, expected value of increase in each element of F_u is also $\beta\%$. Also, as m_g increases, it can be assumed that $\sum_{j=1}^{m_g} d_j \approx m_g \times \delta$, where δ is the average number of features per item. Due to the proportional increase in elements of FBP and normalization coefficient, negligible differences occur in estimated FBPs. While calculating cluster centers, average of those FBPs will be taken, which will further diminish such difference. In addition, when clones of users are included in the process, number of users will increase drastically and effects of random filling procedure will become further insignificant.

7.4 Overhead Costs Analysis

It is imperative to analyze the proposed schemes with respect to both off-line and online costs like storage, communication, and computation costs. Although off-line costs do not have acute effects on performance compared to online ones, they are still needed to be analyzed to provide a report on off-line overloads.

Compared to the traditional k-nearest neighbor-based CF approaches, neither of the proposed preprocessing schemes causes any extra communication costs in terms of both number of communications and amount of transferred data. Thus, such costs remain the same in online and off-line phases.

Traditional CF schemes require a storage cost in the order of $O(nm)$ to collect n users' data for m items. Additionally, FBPs of users are utilized, which requires an extra storage cost in the order of $O(nd)$, where note that d is the constant number of features and $d \ll m$. Due to the BDT generation process, there will be two 1-by-d vectors being the cluster centers to be recorded after each recursive call. Therefore, at each level, there will be at most 2^k such vectors, where $k = 1, 2, \ldots, h-1$ and h is the height of BDT. Also notice that $h \ll n$ in systems suffering from scalability. Moreover, the second preprocessing scheme increases number of users by producing clones; however, the upsurge is linear; and thus, the storage cost is linear, as well. Consequently, total storage costs of both schemes are in the order of $O(nm)$.

Additional computation costs can be analyzed, as follows. Off-line phase includes three stages: (i) estimating FBPs, (ii) generating clones, and (iii) building a model through bisecting k-means clustering. FBPs are estimated in $O(nm)$ time because every item of every user is checked through. Then, the server creates clones of all n original users by predicting PSPs in an identical manner it produces actual recommendations, which has an online running

time in the order of $O(n^2m)$. Thus, total cost of producing clones is in the order of $O(n^3m)$. Finally, it constructs a BDT by applying bisecting k-means clustering, which runs in the order of $O(n^3 \log n)$ for each iteration. At each level for BDT, at most 2^k clustering operations ($k = 0, 1, \ldots, h-1$) will be performed yielding a total computation cost in the order of $O(2^h n^3 \log n)$ to form a BDT.

Actual performance of a recommender system is determined by its response time to queries. After forming neighborhood, data disguising schemes allow PPCF systems to produce predictions identical to traditional CF schemes. Unlike the PPCF approach, the proposed scheme constructs neighborhood by traversing the BDT. For this purpose, during an online transaction, FBP_a of a new user is estimated initially, which requires $O(m)$ time. While traversing the tree, PCC is calculated between cluster centers and F_a, where each calculation requires $O(d)$ time. After locating user's neighbors, exact similarities between rating profiles are determined. Therefore, online predictions can be estimated in the order of $O(Nm)$ because h and d are small constants, where typically $N \ll n$ in systems suffering from scalability.

7.5 Experimental Evaluation

Experiments are conducted on benchmark data sets collected for CF purposes to investigate how proposed schemes perform with respect to accuracy and efficiency on both CF and PPCF schemes. First, BDT produced by bisecting k-means approach is evaluated solely against state-of-the-art k-nearest neighbor-based CF method to see its effects on scalability clearly. Then, the second preprocessing scheme is built on scalability enhanced structure to determine its impact on accuracy. Finally, combined effects of the proposed schemes are evaluated on PPCF architecture and obtained enhancements are analyzed in terms of their significance.

Trials are performed on MLP and MLM data sets, which contain preferences for movies in a 5-star rating scale and each movie in the sets contain at least one or more genre features from predefined 18 categories. Data sets are suitable to show effects of preprocessing schemes as they both are extremely sparse and especially MLM is very large. Experiments are realized using a 10-fold cross-validation experimentation methodology. The original data set (U or U') is uniformly randomly divided into ten subgroups and at each iteration i ($i = 1, 2, \ldots, 10$), corresponding subset (U_i or U'_i) is considered as the test users and the remaining ones are as the training users. After training and test sets are constructed, five rated items' actual votes are withheld for each test (active) user. Such entries are replaced with null, their values are tried to be predicted, and estimations are compared with actual values. Trials are done in MATLAB 7.9.0 environment using a computer with an Intel Xeon 2.8 GHz processor and 6 GB RAM. For k-means clustering operations, MATLAB's built-in function is used with parameters to take head k user vectors in the input matrix as initial centers and utilizes '`correlation`' as the distance measure based on PCC.

Several trials are performed to assess the effects of preprocessing schemes with different parameters. The proposed schemes are evaluated in non-private and private environments separately. Mainly, distinct performances of schemes are assessed in non-private architecture and optimized configuration of both preprocessing methods is evaluated through varying privacy parameters in privacy-preserving architecture. Details of experimental procedures and results of conducted tests are explained in the following.

7.5.1 Evaluating non-private schemes

In order to examine improvements with respect to scalability, first, predictions are produced relying on the BDT constructed by bisecting k-means clustering

approach and empirical outcomes are compared against the traditional CF scheme, where these schemes will hereafter be referred to as BKM and CF methods, respectively. Then, the second preprocessing method is employed onto the first one, denoted as BKM+, by accompanying clones into user-item matrix and its effects on accuracy and online performance are demonstrated.

BKM Model. BKM model is experimented to examine its effects on accuracy and online performance. For all folds, FBPs are initially produced, training sets are recursively clustered, as described in Section 7.3.1, and corresponding BDTs are constructed. Then, belonged leaf of each test user is located by forwarding them in relation to cluster centers and the nearest neighbors are determined by calculating similarities to the users in that particular leaf node and its sibling. Next, predictions are produced for withheld items, MAEs are estimated, and T for whole online process is measured. Trials are repeated by varying N from 20 to 100 because employing more than 100 neighbors might have adverse effects on scalability, as stated in (Herlocker et al., 2004). Outcomes are displayed in Table 7.1 for both data sets.

Table 7.1: MAE and T values by varying N for CF and BKM

	N		20	40	60	80	100
MLP	CF	MAE	0.790	0.773	0.772	0.773	0.774
		T	149s	150s	150s	150s	151s
	BKM	MAE	0.831	0.793	0.779	0.758	0.758
		T	6s	11s	14s	18s	25s
MLM	CF	MAE	0.766	0.754	0.749	0.747	0.747
		T	1,516s	1,520s	1,523s	1,526s	1,531s
	BKM	MAE	0.819	0.781	0.762	0.749	0.742
		T	68s	108s	147s	233s	311s

According to Table 7.1, CF scheme obtains its best accuracy values for MLP and MLM as 0.772 and 0.747 with N values of 60 and 80, respectively. BKM scheme achieves 0.758 and 0.742 MAE values with N values of 80 and 100, respectively. It can be concluded that there is not a significant improvement in accuracy for MLM and a slight progress for MLP. However, there is a

significant improvement in T. Through overall trials, five predictions are produced for each test user, which yields a total of 4,715 and 30,200 predictions for MLP and MLM, respectively. Compared to CF scheme, BKM model produces such predictions in 18 seconds instead of 150 for MLP and 311 seconds instead of 1,526 for MLM, which means an enhancement of approximately 88% and 80% on the online performance, respectively. Moreover, since BKM model operates by simply traversing over the BDT, which is constructed off-line, as the number of users in the system increases linearly, such online performance enhancements will be much greater due to logarithmic grow of height of the tree. This phenomenon can be observed better in the next set of experiments, where the number of users in the system increases dramatically due to additional clone profiles injected into the system.

BKM+ Model. BKM model proves that it can handle scalability issues well; however, it is not very successful at boosting accuracy. BKM+ model is proposed as reinforcement onto BKM model to improve precision of predictions, which are restricted due to information losses caused by narrowing down search space through a BDT. However, BKM+ model must be studied in terms of its controlling parameters to get the most possible gain of accuracy. Such parameters are experimentally analyzed in the following.

Number of clones ($\boldsymbol{\omega}$): It is hypothesized that cloning original user profiles shall improve determination skills of BDT and hence, accuracy of the system. However, ω must be fine-tuned because it can influence scalability of the system. While producing clones, 100 nearest neighbors are located to each train user in the system and PSPs are produced to uniformly randomly chosen empty cells using such neighbors' data. While ω is varied from 1 to 5, other parameters are held at N = 80 and ρ = 1, i.e., each clone has twice as much ratings as the original user. After estimating FBPs of original and clone users, clones-added profiles are recursively clustered and corresponding BDTs are

produced. In order to avoid U to converge to itself and lose personalization, predictions are produced through original user profiles only. Due to the randomness while inserting PSPs, those experiments are repeated 100 times and average results are computed to obtain more dependable outcomes. Estimated MAEs and measured T values for both data sets are given in Table 7.2.

Table 7.2: MAE and T values for varying ω

ω		1	2	3	4	5
MLP	MAE	0.735	0.730	0.726	0.724	0.722
	T	18s	18s	19s	20s	20s
MLM	MAE	0.730	0.728	0.725	0.724	0.723
	T	237s	239s	243s	246s	249s

As can be followed from Table 7.2, adding more clones has positive effects on accuracy; however, enhancements are getting smaller as ω grows. The highest accuracy values are obtained at ω=5 for both data sets, being around 0.720. Also note that even if there are five times more users in the system, online performance is negligibly affected, i.e., T increases by approximately 11% (from 18s to 20s) and 7% (from 233s to 249s) in this sample case for MLP and MLM, respectively. These outcomes present the robustness of the proposed BKM model in terms of scalability.

Density coefficient ($\boldsymbol{\rho}$): Cloning process aims to boost accuracy of the system. Thus, it is imperative to define effects of PSP amount to be inserted into replicated profiles. PSPs are produced like before and other parameters are held at N = 80 and ω = 1 while ρ is varied from 0.5 to 4, which created one clone per user with ρ times much ratings. After cloning users, FBPs are produced and preprocessing steps are applied. Experiments are repeated 100 times due to randomness, as before. Estimated MAE values are given in Table 7.3.

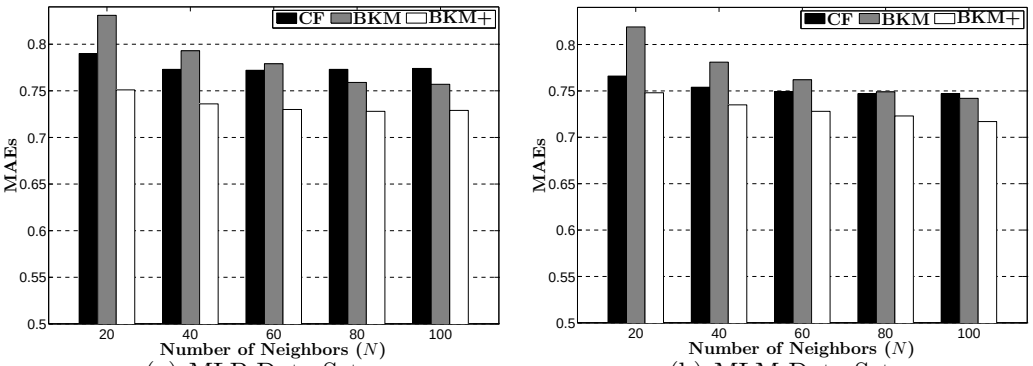
Inserting as much PSPs as existing ratings into clone profiles helps boosting accuracy; however, further insertions cause losing personalization and decrease accuracy. Online performance evaluation is not given in Table 7.3. Since

Table 7.3: MAE values for varying ρ

ρ	0.5	1	2	4
MLP	0.737	0.726	0.742	0.748
MLM	0.730	0.729	0.729	0.729

$\omega = 1$, for all values of ρ, the models require approximately 18 and 240 seconds for MLP and MLM, respectively. As displayed in Table 7.3, $\rho = 1$ maximizes accuracy. Note that the reached maximum accuracy values of about 0.73 are higher than both CF and BKM models' accuracy, which concludes that by combining two preprocessing methods, both more precise predictions can be produced and a significant enhancement in online performance can be achieved.

Overall comparison: After experimenting how accuracy and efficiency changes by utilizing the proposed schemes, another experiment is conducted to present a clear picture of comparison among CF, BKM, and BKM+ in terms of accuracy. Thus, joint effects of controlling parameters are demonstrated. All three models are ran while varying N from 20 to 100. For BKM+ model, $\omega = 5$ and $\rho = 1$ are set for both data sets because those values are verified to maximize accuracy. MAE values are presented in Figure 7.2.



(a) MLP Data Set (b) MLM Data Set

Figure 7.2: Accuracy with varying N values for CF, BKM, and BKM+ schemes

As can be seen from Fig. 7.2, for both data sets and all N values, BKM+ clearly outperforms CF scheme in terms of accuracy. Statistical significance t-tests are performed to compare the results of CF and BKM+ schemes. Overall

MAE results are handled by taking average of 10-fold experimental accuracy values. A series of paired t-tests are done for varying N values and statistics are displayed in Table 7.4. The results of one-tailed hypotheses show that all of the differences appear to be statistically significant at 99% confidence level except for $N = 40$ in MLP, which is also significant at 95% confidence level.

Table 7.4: Statistical significance tests for CF and BKM+ schemes

	MLP	MLM
N	CF vs. BKM+	CF vs. BKM+
20	$t = 2.900$ $p = 0.004^{**}$	$t = 3.439$ $p = 0.001^{**}$
40	$t = 2.332$ $p = 0.016^{*}$	$t = 3.792$ $p = 0.000^{**}$
60	$t = 2.618$ $p = 0.009^{**}$	$t = 4.762$ $p = 0.000^{**}$
80	$t = 2.918$ $p = 0.005^{**}$	$t = 5.335$ $p = 0.000^{**}$
100	$t = 3.231$ $p = 0.002^{**}$	$t = 5.031$ $p = 0.000^{**}$

Degree of freedom = 18
* For significance at 95%
** For significance at 99%

7.5.2 Evaluating privacy-preserving schemes

After examining the effects of the preprocessing schemes in non-private CF schemes and determining optimum values of parameters ($\omega = 5$ and $\rho = 1$), experiments are performed in privacy-preserving environment. Since prediction production process in a privacy-preserving architecture is the same with non-private schemes, BDT generation step achieves the same improvements in online performance. Therefore, T values are not presented for those experiments, as they are similar with non-private experimental outcomes. Also, experiments are proceeded only with BKM+ scheme in privacy-preserving parts because distinct effects of preprocessing are investigated in previous section. Privacy-preserving form of the proposed scheme (P^{2}BKM+) is examined against the PPCF method (P^{2}CF) in terms of accuracy. For disguising schemes, σ_{max} is

kept constant at 2 in the experiments as it is appropriate to see effects of data perturbation and still provides pretty sufficient level of privacy. Nevertheless, effects of varying forgery rates (β_{max}) are studied.

Maximum forgery rate (β_{max}). Data disguising protocol proposes users to choose β value uniformly randomly over $(0, \beta_{max}]$ to perturb genuine ratings. Effects of different β_{max} levels on privacy and accuracy are studied in Sections 3.1.1 and 3.2. Now, effects of this parameter is examined on cloning- and FBP-based schemes' accuracy. In the experiments, N is set at 80 for both data sets while β_{max} is varied from 5 to 100. Due to randomized selection of β and cells to insert PSPs by each user, the experiments are repeated 100 times and average of outcomes are demonstrated in Table 7.5.

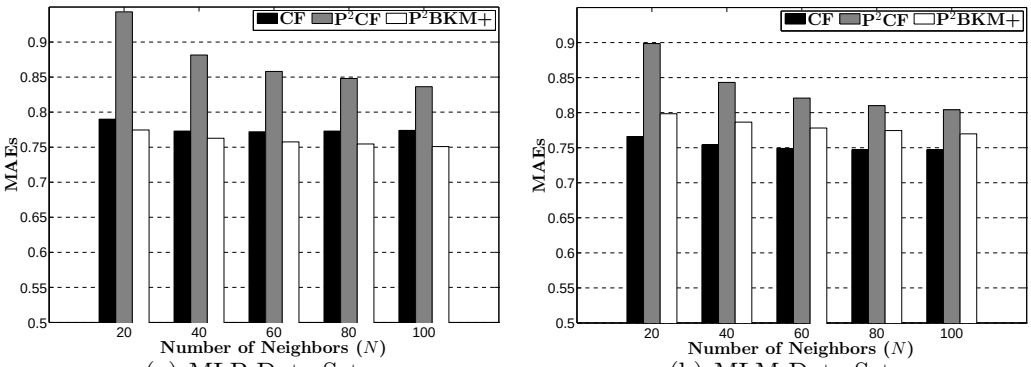
Table 7.5: MAE values by varying β_{max} for P^{2}CF and P^{2}BKM+

	β_{max}	5%	10%	20%	50%	100%
MLP	P^{2}CF	0.850	0.850	0.848	0.853	0.869
	P^{2}BKM+	0.737	0.739	0.747	0.752	0.775
MLM	P^{2}CF	0.798	0.800	0.810	0.840	0.882
	P^{2}BKM+	0.774	0.776	0.781	0.811	0.843

Although forged ratings deteriorate accuracy with P^{2}CF model especially for MLP, losses are not that much for the proposed P^{2}BKM+ scheme. Even for β_{max} value of 100, accuracy losses are approximately 0.06 for MLP and 0.12 for MLM. Thus, it can be inferred that P^{2}BKM+ scheme is more resistant to changes in β_{max} compared to the P^{2}CF method. Considering provided privacy levels due to maximum forgery rate discussed before, choosing a β_{max} value of 20 for both data sets seems optimal because it balances the trade-off between conflicting goals of accuracy and privacy for this case.

Overall comparison. The last experiment is conducted to demonstrate a clear comparison among CF, P^{2}CF, and P^{2}BKM+ schemes in terms of accuracy, like the one in non-private scheme. For this purpose, β_{max} is set at 20 and P^{2}CF and P^{2}BKM+ models are ran while varying N from 20 to 100.

Combined outcomes in terms of MAE values are presented in Figure 7.3.



(a) MLP Data Set (b) MLM Data Set

Figure 7.3: Accuracy with varying N values for CF, P^{2}CF, and P^{2}BKM+ schemes

As seen from Fig. 7.3, P^{2}BKM+ scheme performs significantly better than P^{2}CF for both data sets. Paired t-tests are done to compare the accuracy of P^{2}CF and P^{2}BKM+ schemes and statistics are presented for varying N values in Table 7.6. The results of one-tailed hypotheses claim that all enhancements appear to be statistically significant at 99% confidence level.

Table 7.6: Statistical significance tests for P^{2}CF and P^{2}BKM+ schemes

	MLP	MLM
N	P^{2}CF vs. P^{2}BKM+	P^{2}CF vs. P^{2}BKM+
20	t = 9.918 p = 0.000*	t = 18.718 p = 0.000*
40	t = 7.071 p = 0.000*	t = 12.442 p = 0.000*
60	t = 5.588 p = 0.000*	t = 10.770 p = 0.000*
80	t = 5.334 p = 0.000*	t = 7.654 p = 0.000*
100	t = 5.017 p = 0.000*	t = 5.830 p = 0.001*

Degree of freedom = 18
* For significance at 99%

In fact, P^{2}BKM+ performs even better than non-private CF scheme for MLP as seen from Fig. 7.3(a); however, these differences are not significant. Also, it can be seen from Fig. 7.3(b) that P^{2}BKM+ performs slightly worse than CF scheme while providing privacy measures to individuals. Thus, it can

be concluded that the combined effects of the proposed preprocessing schemes allow providing accurate private referrals in considerably less amount of time.

7.6 Conclusions

A novel CF scheme is proposed based on two off-line preprocessing methods to improve online performance and accuracy. In the first approach, applying a bisecting k-means clustering algorithm is offered on item category-based histograms of users to construct a BDT and it is utilized to determine neighbors of new users, which significantly improves scalability. The second preprocessing scheme focuses on enhancing the accuracy of the predictions produced by the first scheme by alleviating effects of sparseness. Replication of original user profiles is proposed and their density is increased by inserting PSPs into some randomly chosen empty cells. Privacy concerns of schemes are solved by utilizing RPTs. Empirical outcomes show that employing the proposed preprocessing schemes significantly outperforms the state-of-the-art k-nearest neighbor-based PPCF scheme in terms of online performance and accuracy, as demonstrated by significance tests. However, relative errors due to randomization are not significant and the proposed privacy-preserving scheme is able to produce predictions with comparable accuracy to original non-private scheme. Moreover, the proposed bisecting k-means clustering approach promises much better relative improvements on online performance as input matrix gets larger, which is vital to scale CF systems.

In this chapter, it is essentially tried to provide a different perspective on data configuration and prediction production process. More important than the observed empirical achievements, it can be claimed that the proposed preprocessing ideas are modular, easy to manipulate, and effective. They can be easily integrated into real life deployments by adjusting them to particular needs.

8. IMPROVING NBC-BASED PRIVATE RECOMMENDATION SCHEME

In this chapter, improving both efficiency and accuracy of NBC-based private recommendation scheme is studied by utilizing two preprocessing techniques. Masked data is preprocessed by constructing the best similar item sets for each item in the set. Also, some of the unrated items' cells are filled by self predictions to improve density. Experimental results show that efficiency and accuracy improves due to preprocessing.

8.1 Introduction

It might be crucial to determine the likelihood of a user will either like or dislike an item rather than estimating the exact rating. This approach may either facilitate to filter out possible items, which will be disliked by user or indicate items, which user will possibly tend to purchase. Thus, the crucial point is to determine whether a user will like or dislike an item, which requires transforming the numerical votes into binary ones. NBC is a simple yet very effective classifier to determine class labels of recommendation items. Miyahara and Pazzani (2000) propose a CF approach using NBC to offer binary ratings-based referrals.

Kaleli and Polat (2007) propose a privacy-preserving scheme to offer binary ratings-based predictions while achieving privacy. Although their method achieves privacy and is able to produce referrals with decent accuracy, efficiency significantly degrades with increasing number of users and number of groups. By selecting the most similar items to each item by preprocessing the masked data off-line, online performance is more likely to improve because amount of data involved in CF process decreases. Due to privacy concerns, accuracy

losses are inevitable. Although such losses are expected and could be said that acceptable, it is hypothesized that the quality of referrals might be improved by filling some of the unrated items' cells before applying CF process.

In this chapter, it is aimed to enhance performance of PPNBC proposed by (Kaleli and Polat, 2007) without greatly compromising (or sometimes even increasing) accuracy by preprocessing the masked data so that prediction processes can be completed in a reasonable amount of time for online applications. In order to achieve such goal, NBC-based CF algorithm's ability to estimate predictions from a small amount of training data is utilized.

8.2 NBC-based Prediction Schemes

In the proposed framework, referrals are offered using NBC-based approach during online interactions. Despite its simplicity, NBC has the ability to often outperform more sophisticated classification methods. Miyahara and Pazzani (2000) utilize NBC by classifying ratings to two class labels, i.e., *like* or *dislike*. An active user for whom the prediction will be produced is having ratings utilized as class labels and U holds ratings from other users corresponding to the feature values, where other users are treated as features. Relying on the naïve assumption that features are independent given the class labels, the conditional probability that an item belongs to $class_j$, where $j \in \{like, dislike\}$ given its n feature values, is formulated as in Eq. 8.1:

$$p(class_j|f_1, f_2, \ldots, f_n) \propto p(class_j) \prod_i^n p(f_i|class_j), \tag{8.1}$$

where f_i is the feature of target item (q) for user i and $p(class_j)$ and $p(f_i|class_j)$ can be determined from training data. The conditional probabilities of both classes are calculated and q is assigned to the class with the highest probability. Kaleli and Polat (2007) propose PPNBC to offer NBC-based predictions with

preserving individual privacy, where their protocol of data masking is explained in Section 2.2.2. Authors propose feasible parameters as three groups with a perturbation level of 0.7 (M=3 and θ=0.7). Although their scheme preserves confidentiality, its performance decreases considerably with privacy concerns.

8.3 Improving NBC-based Private Prediction Scheme

Due to privacy-preserving measures, extra costs, especially online computation costs, are inevitable. Although PPNBC scheme proposed by (Kaleli and Polat, 2007) provides accurate referrals while preserving privacy, efficiency significantly decreases. Moreover, accuracy diminishes due to data masking schemes. It is hypothesized that online performance and even accuracy of PPNBC can be improved by applying off-line preprocessing to masked data.

If only a subset of items are used in the recommendation generation process, (i) recommendation accuracy would enhance with most information providing items and (ii) performance of the algorithm would increase considerably due to reduced number of elements utilized in online process. Two preprocessing methods are applied: In the first method, referred to as *item extraction*, the most similar items to any corresponding item is determined using a binary similarity measure. In the second method, which is called *densifying*, some of the randomly chosen empty cells in the masked user-item matrix (U') is filled with PSPs estimated by PPNBC. The overall procedure can basically be described, as follows: The vendor first forms U' by collecting perturbed ratings from its customers. It then applies densifying preprocessing scheme to U'. It finally applies item extraction preprocessing to each item. After conducting preprocessing steps off-line, the data owner starts offering predictions online using the preprocessed U', referred to as U'_p. To provide truthful and dependable referrals, number of commonly rated items between a and those users who rated q is also imperative. To increase such items, it is also proposed

to fill some of a's unrated items' cells with a's mean vote. The details of such preprocessing methods are given in the following.

8.3.1 Item extraction

In order to improve online performance, amount of data involved in CF process should be decreased. Extracted items list for each item in U is determined by selecting the most similar ones. The size of extraction is imperative for overall performance. Similarly, an item's extraction list significantly influences accuracy. Extraction size (e) is determined by locating the most e similar items to any related item and note that the optimum value of e can be determined experimentally, where e is a constant and $e \ll m$. To determine such neighbors, Tanimoto coefficient similarity measure is preferred due to its given importance to commonly and concordantly rated items (Bilge et al., 2010).

Let S_{ij} be the number of occurrences of commonly rated items with i in the first pattern and j in the second pattern, where $i, j \in \{0, 1\}$. Given two binary feature vectors X and Y, $T(X, Y)$ denotes Tanimoto coefficient between X and Y (similarly, between two items' ratings vectors) and it is calculated as given in Eq. 8.2:

$$T(X,Y) = \frac{(S_{11} + S_{00}) - (S_{10} + S_{01})}{S_{11} + S_{00} + S_{10} + S_{01}}, \tag{8.2}$$

where S_{11} is the number of users rated both items as "1", S_{10} represents the number of users rated item i as "1" and item j as "0", S_{01} is the number of users rated item i as "0" and item j as "1", and S_{00} shows the number of users rated both items as "0".

Tanimoto similarity measure computes the similarity between two binary vectors; however, U representing true users' ratings, does not present. Without privacy concerns, it is trivial to estimate similarities between binary vectors.

Due to underlying data masking methods described in Procedure 2, it becomes a challenge to estimate the same similarities from perturbed data. U' is obtained after collecting data from many users. Therefore, actual rates cannot be determined exactly, but according to masking protocol (number of groups, M and disguising rate, θ), an inference can be made to calculate similarities between features. If it is assumed that S_{ij} be the occurrence of commonly rated items for two items' ratings vector, $S_{i'j'}$ represents the exact opposite of the ratings, where $i', j' \in \{0, 1\}$. Thus, corresponding variable, S_{ij}, cannot be simply incremented due to masking procedure. All the values are correct with a probability of θ in U'. Thus, to estimate S_{ij} values, all possible combinations of the match should be considered, as explained in Eq. 8.3:

$$\begin{aligned} S_{i,j} &= S_{i,j} + (\theta \times \theta) = S_{i,j} + \theta^2 \\ S_{i',j} &= S_{i',j} + ((1-\theta) \times \theta) = S_{i',j} + \theta - \theta^2 \\ S_{i,j'} &= S_{i,j'} + (\theta \times (1-\theta)) = S_{i,j'} + \theta - \theta^2 \\ S_{i',j'} &= S_{i',j'} + (1-\theta) \times (1-\theta) = S_{i',j'} + (1-\theta^2) \end{aligned} \tag{8.3}$$

After estimating S_{ij} values from masked data, similarity weights between two items can be estimated using Eq. 8.2. For each item j, the similarities between j and the remaining m-1 items should be estimated. Since $T(X,Y) = T(Y,X)$, $[(m-1) \times (m-2)/2]$ number of $T(X,Y)$ similarity values are estimated. For each item j, m-1 similarity weights are sorted decreasingly, and the first e items are appended into extraction list. Likewise, extraction lists for all items are formed. Since all of these computations are conducted offline, they are not critical for online efficiency. When an a asks for a prediction on q, rather than using entire items' ratings, only corresponding e items' rates are used for recommendation generation. The quantity of data used in recommendation processes decreases; thus, online performance enhances. Since

$T(X, Y)$ values are estimated from perturbed data and U' is usually a sparse set, it might not be possible to improve accuracy. In order to prevent accuracy losses, densifying preprocessing method is proposed.

8.3.2 Densifying

To be able to find a meaningful correlation between two users or items, there should be sufficient amount of common ratings. It is hypothesized that it might be possible to enhance accuracy of scalability-enhanced PPNBC scheme if sparsity is relieved by filling some of the randomly chosen unrated items' cells via PSPs. There are basically two ways to obtain PSPs. Estimating non-personalized ratings or calculating personalized votes from masked data off-line. User, item, or overall mean votes are considered non-personalized ratings; and they can be used to fill unrated items' cells for densifying. Although it is trivial to estimate such non-personalized ratings from U or even from U', they might not reflect users' true preferences. Thus, accuracy might become worse. Unlike non-personalized votes, personalized ratings are more likely to represent customers' proper preferences. Therefore, it is proposed to employ personalized votes estimated from U' using PPNBC, as outlined as a pseudo-code layout in Algorithm 7.

According to Algorithm 7, a densification rate (d) is determined by the server. Note that such d value, which happens to give the optimum results for the data holder, can be determined experimentally and should be associated with the density of U'. Then, for each user in the database, the server uniformly randomly chooses ρ over the range $(0, d]$ and uniformly randomly selects $\rho\%$ of the unrated items' cells of corresponding user. It estimates personalized predictions for those chosen unrated items' cells using PPNBC scheme, as explained in (Kaleli and Polat, 2007) and fills such cells with corresponding personalized ratings.

Algorithm 7 Densifying U' by PSPs

Input: User-item matrix ($U'_{n\times m}$), Densification rate (d)

Initialize:

1: $DU = U'$ ▷ create a backup of U' to be densified
2: **for all** u_i in U' ($i \leftarrow 1$ to n) **do**
3: **Define densifying parameter:** $\rho \leftarrow \text{RND}(0, d)$
Locate empty cells to be filled:
4: $n_r \leftarrow$ # of ratings in $U(u_i)$
5: $i_{ec} \leftarrow$ index of empty cells in $U(u_i)$
6: $r_{ec} \leftarrow \text{RandomPermutation}(i_{ec})$ ▷ random index of empty cells
7: $n_{ec} \leftarrow n_r \times \rho$ ▷ # of empty cells to be filled
Produce PSPs for chosen cells:
8: **for** $j \leftarrow 1$ to n_{ec} **do**
9: $target\_item = i_{ec}(r_{ec}(j))$
10: $DU(u_i, target\_item) \leftarrow \text{PSP}(target\_item)$ ▷ by PPNBC
11: **end for**
12: **end for**
13: **return** DU

8.4 Overhead Costs Analysis

Due to preprocessing methods, supplementary storage costs are in the order of $O(me)$ because for each item, index of the most similar e items are stored in a database. The proposed schemes do not change communication costs in terms of number of communications and amount of data to be transmitted.

Computation costs must be analyzed for off-line and online steps, separately. Densifying preprocessing introduces predicting PSPs to at most $d \times m$ items of any existing user in the database. Considering that time complexity of PPNBC algorithm is $O(km)$, where k is the number of nonzero elements in n-dimensional input features vector and generally $k \ll n$, the total computation cost for densifying preprocessing is in the order of $O(dk^2m^2)$. In addition, item extraction preprocessing requires computing item similarities among all items. Estimating Tanimoto coefficient requires n comparisons, so total off-line computation cost for item extraction preprocessing is in the order of $O(nm^2)$. However, online performance, which is critical for recommender

systems, is improved by in the order of $O(m/e)$, where $e \ll m$, due to reduced number of items utilized in recommendation process.

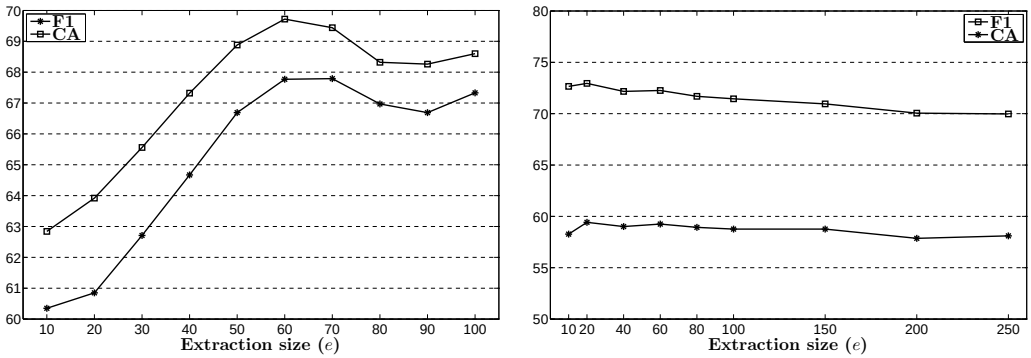
8.5 Experiments

Experiments are performed on Jester and MLP in order to demonstrate performance of the proposed schemes on partially dense and mostly sparse data sets. Numerical ratings in the data sets are converted into two labels (Miyahara and Pazzani, 2000). Ratings are labeled as "1" if its numerical value is greater than 3 for MLP and 2.0 for Jester, as "0" otherwise. Test users are sampled as 243 and 500 users from MLP and Jester, respectively. Number of train users utilized in experiments are determined based on the experimental settings. For each active user, five rated items' ratings are withheld, their entries replaced with null, tried to be predicted, and estimations are compared against true ratings. Trials are performed using MATLAB 7.6.0 environment on a computer with Intel Core2Duo, 2.2 GHz processor, and 2 GB RAM.

8.5.1 Effects of item extraction

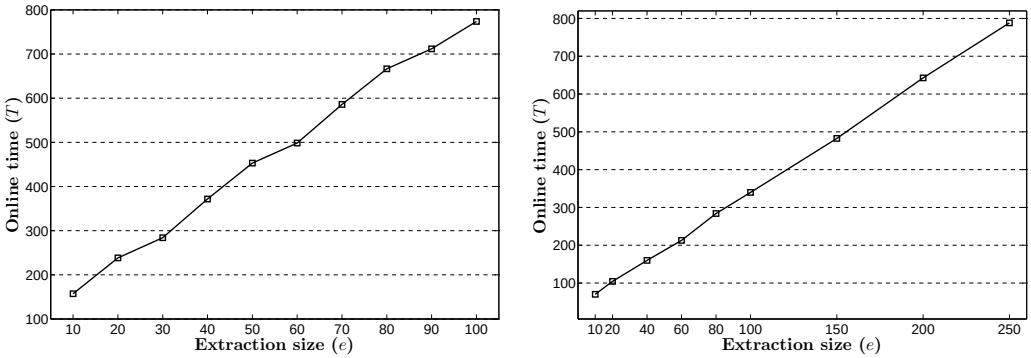
First, experiments are performed to investigate the effects of varying e values on accuracy and efficiency. Uniformly randomly chosen 250 users are formed as test sets for Jester and MLP. During experiments, e is varied from 10 to 1,682 for MLP and from 10 to 100 for Jester. Note that results for e being 1,682 and 100 for MLP and Jester, respectively, corresponds to outcomes without preprocessing. Since the results for values of e greater than 250 are not promising in terms of accuracy and especially online performance, only the outcomes up to e being 250 are demonstrated. Estimated CA and F1 values are presented in Fig. 8.1.

As seen from Fig. 8.1, optimum values of e are 20 and 60 for MLP and Jester, respectively. When all items are utilized in MLP, i.e., e=1,682, F1 and CA values are 61.04% and 59.34%, respectively. However, when preprocessing



(a) Jester Data Set (b) MLP Data Set

Figure 8.1: Quality of recommendations by varying e values

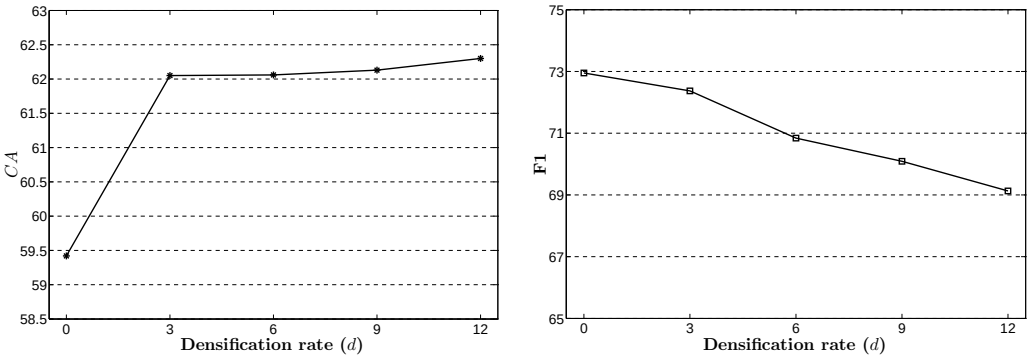


(a) Jester Data Set (b) MLP Data Set

Figure 8.2: Elapsed time (in seconds) by varying e values

is applied, F1 increases to 72.95% and CA stays at 59.42%. In other words, preprocessing significantly improves F1 while CA slightly enhances. For Jester, the results for e being 60 are approximately the same compared to the non-preprocessed scheme, where F1 enhances from 67.33% to 67.77% and CA from 68.60% to 69.72%. On the other hand, effects of item extraction method on online performance is significant. During the experiments, 1,215 and 2,500 predictions are produced for MLP and Jester, respectively. Estimated online times in seconds required to offer such of referrals are displayed in Fig. 8.2.

As seen from Fig. 8.2, gains on efficiency due to preprocessing linearly increases as explained in Section 8.4 and they are significantly high for both data sets. For MLP, results up to 250 items are displayed. When all items are utilized in MLP, T is 5,135 seconds. T values increase with increasing e and



(a) Classification Accuracy (CA) (b) F-Measure (F1)

Figure 8.3: Quality of recommendations by varying d values

they finally reach their peaks when $e = m$. For optimum e values, T decreases from 5,135 seconds to 104 seconds for MLP, while it decreases from 773 seconds to 498 seconds for Jester. Due to preprocessing, online performance enhances by 49.37 and 1.55 times for Jester and MLP, respectively. The reason for improvements being relatively small for Jester is that there already exists very few number of items in the data set.

8.5.2 Effects of densifying

In order to demonstrate the effects of densifying preprocessing method, only MLP is utilized because Jester is already a very dense data set. During the trials, d is varied from 0 to 12 and e is set to previously determined optimal value 20. Note that d is associated with the density of MLP, which is about 6%. Estimated F1 and CA values are presented in Fig. 8.3. As shown in Fig. 8.3, increasing d from 0 to 3 significantly improves CA. For values of d larger than 3, CA enhances slightly. On the other hand, this preprocessing reduces obtained improvements in F1 slightly with increasing d values. However, note that they are still better than the results of non-preprocessed scheme which is around 70%. When d is 3, CA increases from 59.42% to 62.05%, while F1 decreases from 72.95% to 72.37%. For the sake of gains in CA, losses in F1 are acceptable.

It is also imperative to investigate the effects of densifying preprocessing method on online performance. In order to show such effects, online performance results for producing 1,215 predictions in MLP for varying d values are presented in Table 8.1. As can be followed from Table 8.1, T faintly becomes worse due to densification. When d is 3, which seems an optimal selection for MLP, T gets worse only about 1 second. Thus, it can be concluded that densification preprocessing scheme has negligible effects on online performance and combined effects of item extraction and densification methods improve scalability and accuracy of PPNBC algorithm.

Table 8.1: Online performance by varying d values

d	0	3	6	9	12
T	104.435s	105.122s	105.839s	106.458s	107.096s

8.6 Conclusions

Although PPNBC scheme offers truthful predictions with privacy, online performance significantly degrades. Due to privacy concerns, accuracy losses are also inevitable, even if they are small. Two preprocessing methods are presented to enhance efficiency and accuracy of PPNBC scheme. First method extracts the most similar items to each one so that those items providing the most useful information join in recommendation process. Since the most similar items are utilized, efficiency definitely improves, even accuracy enhances. In order to increase density so that similarity weights become more reliable and truthful, a densifying preprocessing scheme is utilized. According to such method, some of the empty cells are filled with personalized ratings. This scheme helps binary ratings-based CF systems provide more truthful predictions by sacrificing from online performance slightly.

9. CONCLUDING REMARKS

This dissertation proposes various non-trivial solutions to overcome on-going challenges of privacy-preserving collaborative filtering schemes like accuracy, scalability, and accuracy. The proposed schemes alleviate at least one, sometimes even more than one challenge without jeopardizing individual privacy. Besides privacy-preserving schemes, the proposed solutions are also employed on traditional non-private collaborative filtering schemes to investigate their extent. Preprocessing methods are analyzed in terms of privacy levels, overhead costs, accuracy, and online performance. Performed real-data based experiments demonstrate each preprocessing scheme's success on targeted problem of privacy-preserving collaborative filtering. Main contributions of the dissertation can be listed, as follows.

- In order to preserve confidentiality, randomized perturbation techniques for numerical ratings-based data and randomized response techniques for binary ratings-based data are utilized. Relying on the privacy preservation principles of randomization techniques, a novel privacy level quantification method based on information theory is proposed. Such method helps users better understand how applied random filling approach within numerical ratings vector perturbation protocol aid concealing private information of individuals.

- Memory-based collaborative filtering and privacy-preserving collaborative filtering methods are the most successful schemes in terms of accuracy. However, they deeply suffer from scalability issues since they operate on whole user-item matrix. An item reduction model is proposed to alleviate scalability challenge for both non-private and privacy-preserving memory-based collaborative filtering schemes which eliminates dissimi-

lar items to each target item. It is shown that proposed preprocessing method performs well in terms of online performance while not jeopardizing accuracy.

- Discrete wavelet transformation is utilized in privacy-preserving collaborative filtering approach as a model-based approach. In addition to dimension reduction facilities provided by such transformation, an item ordering method is proposed to recover accuracy losses due to distortion in input vectors for privacy preservation. It is shown that multiple item ordering method performs well in recovering losses especially in large scale data sets.

- Clustering algorithms are utilized in privacy preserving environment to enhance scalability along with a novel user profiling method to deal with sparsity issues. User profiling method projects large and sparse vectors to a compact and dense form which is then given as an input to various clustering algorithms. It is demonstrated that clustering algorithms perform well up to 2-level clustering in terms of accuracy.

- Traditional clustering approach is concluded insufficient to relieve scalability issues. Therefore, a recursive 2-level clustering approach is proposed along with a novel user cloning technique. Such approach produces a binary decision tree to accelerate neighborhood formation process after cloning users to discover hidden similarities among users. It is both theoretically and experimentally shown that proposed bisecting k-means clustering-based privacy-preserving collaborative filtering scheme achieves comparable accuracy with non-private collaborative filtering schemes.

- In addition to numerical ratings-based system, algorithms relying on binary collections is studied to be improved in scalability and sparsity

perspective. Naïve Bayesian classifier based privacy-preserving collaborative algorithm is enhanced by applying item extraction and data condensation preprocessing methods. Experimental outcomes confirm that scalability problem of the algorithm is relieved without jeopardizing from accuracy by applying both preprocessing schemes.

Although applied privacy measures distort preference information and conflict with the goal of providing accurate referrals, the proposed preprocessing schemes manage to relieve such losses significantly. Overall experimental outcomes demonstrate that each distinct accuracy-improving preprocessing technique has specific effects relying on the data configuration of utilized collaborative filtering approach.

9.1 Recommendations for Future Research

The proposed item features-based user profiling scheme in Chapters 6 and 7 is effective in transforming rating profile dimensions. However, such features might not always present distinctively in product range. Therefore, creation and/or extraction of new features to transform user data remains an open problem.

Although user cloning techniques proposed in Chapter 7 is effective extracting hidden concordances among users, different cloning methodologies might be developed to obtain better quality of predictions. This subject warrants future work.

In this dissertation, scalability problem of privacy-preserving collaborative filtering schemes is handled in general due to its negative effects to produced referrals' quality. However, there remains work to handle other minor challenges originating from sparse collections such as *cold start* problem and *coverage*.

REFERENCES

Acilar, M., Arslan, A., (2009), "A collaborative filtering method based on artificial immune network," *Expert Systems with Applications*, **36** (4), 8324–8332.

Ackerman, M. S., Cranor, L. F., Reagle, J., (1999), "Privacy in e-commerce: Examining user scenarios and privacy preferences," In: *Proceedings of the 1st ACM Conference on Electronic Commerce*, Denver, Colorado, USA, pp. 1–8.

Adomavicius, G., 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.

Agrawal, D., Aggarwal, C. C., (2001), "On the design and quantification of privacy preserving data mining algorithms," In: *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, Santa Barbara, California, USA, pp. 247–255.

Ahn, H. J., (2008), "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Information Sciences*, **178** (1), 37–51.

Al-Shamri, M. Y. H., Bharadwaj, K. K., (2008), "Fuzzy-genetic approach to recommender systems based on a novel hybrid user model," *Expert Systems with Applications*, **35** (3), 1386–1399.

Berkovsky, S., Borisov, N., Eytani, Y., Kuflik, T., Ricci, F., (2007), "Examining users' attitude towards privacy preserving collaborative filtering," In: *Proceedings of the Workshop on Knowledge Discovery for Ubiquitous User Modeling*.

Berkovsky, S., Kuflik, T., Ricci, F., (2012), "The impact of data obfuscation on the accuracy of collaborative filtering," *Expert Systems with Applications*, **39** (5), 5033–5042.

Bezdek, J. C., (1981), *Pattern recognition with fuzzy objective function algorithms*, Kluwer Academic Publishers, Norwell, MA, USA.

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

Bilge, A., Kaleli, C., Polat, H., (2010), "On binary similarity measures for privacy-preserving top-N recommendations," In: *Proceedings of the 5th International Conference on Software and Data Technologies*, Athens, Greece, pp. 299–304.

Bilge, A., Polat, H., (2010), "Improving privacy-preserving NBC-based recommendations by preprocessing," In: *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, Toronto, Ontario, Canada, pp. 143–147.

Bilge, A., Polat, H., (2011), "An improved profile-based CF scheme with privacy," In: *Proceedings of the 2011 IEEE 5th International Conference on Semantic Computing*, San Francisco, California, USA, pp. 133–140.

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

Bilge, A., Polat, H., (2013a), "A comparison of clustering-based privacy-preserving collaborative filtering schemes," *Applied Soft Computing*, doi: 10.1016/j.asoc.2012.11.046.

Bilge, A., Polat, H., (2013b), "A scalable privacy-preserving recommendation scheme via bisecting k-means clustering," *Information Processing & Management*, **49** (2013), 912–927.

Bilge, A., Polat, H., (2013c), "Target item-based similarity function in privacy-preserving collaborative filtering," submitted to a journal.

Bobadilla, J., Ortega, F., Hernando, A., Alcalá, J., (2011), "Improving collaborative filtering recommender system results and performance using genetic algorithms," *Knowledge-Based Systems*, **24** (8), 1310–1316.

Bogdanova, G., Georgieva, T., (2008), "Using error-correcting dependencies for collaborative filtering," *Data & Knowledge Engineering*, **66** (3), 402–413.

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

Brun, A., Boyer, A., (2009), "Towards privacy compliant and anytime recommender systems," In: *Proceedings of the 10th International Conference on E-Commerce and Web Technologies*, Linz, Austria, pp. 279–287.

Burke, R., (2002), "Hybrid recommender systems: Survey and experiments," *User Modeling and User-Adapted Interaction*, **12** (4), 331–370.

Calandrino, J. A., Kilzer, A., Narayanan, A., Felten, E. W., Shmatikov, V., (2011), "“You might also like:“ Privacy risks of collaborative filtering," In: *Proceedings of the 2011 IEEE Symposium on Security and Privacy*, Oakland, California, USA, pp. 231–246.

Canny, J., (2002a), "Collaborative filtering with privacy," In: *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Oakland, California, USA, pp. 45–57.

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

Cechinel, C., Sicilia, M.-A., Sánchez-Alonso, S., García-Barriocanal, E., (2013), "Evaluating collaborative filtering recommendations inside large learning object repositories," *Information Processing & Management*, **49** (1), 34–50.

Chen, G., Wang, F., Zhang, C., (2009), "Collaborative filtering using orthogonal nonnegative matrix tri-factorization," *Information Processing & Management*, **45** (3), 368–379.

Cheng, F.-H., Chen, Y.-L., (2006), "Real time multiple objects tracking and identification based on discrete wavelet transform," *Pattern Recognition*, **39** (6), 1126–1139.

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

Cranor, L. F., (2003), "'I didn't buy it for myself' privacy and e-commerce personalization," In: *Proceedings of the 2003 ACM Workshop on Privacy in the Electronic Society*, Alexandria, Virginia, USA, pp. 111–117.

Dubuisson, S., Fabrizio, J., (2009), "Optimal recursive clustering of likelihood functions for multiple object tracking," *Pattern Recognition Letters*, **30** (6), 606–614.

Gan, G., M. J., (2007), *Data clustering: Theory, algorithms, and applications*, ASA-SIAM Series on Statistics and Applied Probability, Alexandria, Virginia, USA.

Georgiou, O., Tsapatsoulis, N., (2010), "Improving the scalability of recommender systems by clustering using genetic algorithms," In: *Proceedings of the 20th International Conference on Artificial Neural Networks*, Thessaloniki, Greece, pp. 442–449.

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

Gupta, D., Digiovanni, M., Narita, H., Goldberg, K., (1999), "Jester 2.0: Evaluation of a new linear time collaborative filtering algorithm," In: *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Berkeley, California, USA, pp. 291–292.

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

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

Honda, K., Sugiura, N., Ichihashi, H., Araki, S., (2001), "Collaborative filtering using principal component analysis and fuzzy clustering," In: *Proceedings of the First Asia-Pacific Conference on Web Intelligence: Research and Development*, Maebashi, Japan, pp. 394–402.

Huang, Z., Du, W., Chen, B., (2005), "Deriving private information from randomized data," In: *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, Baltimore, Maryland, pp. 37–48.

Jensen, C., Potts, C., Jensen, C., (2005), "Privacy practices of internet users: Self-reports versus observed behavior," *International Journal of Human-Computer Studies*, **63** (1-2), 203–227.

Jeong, B., Lee, J., Cho, H., (2009), "An iterative semi-explicit rating method for building collaborative recommender systems," *Expert Systems with Applications*, **36** (3), 6181–6186.

Jeong, B., Lee, J., Cho, H., (2010), "Improving memory-based collaborative

filtering via similarity updating and prediction modulation," *Information Sciences*, **180** (5), 602–612.

Kaleli, C., Polat, H., (2007), "Providing private recommendations using naïve Bayesian classifier," *Lecture Notes in Computer Science*, **43**, 168–173.

Kaleli, C., Polat, H., (2009), "Similar or dissimilar users? or both?," In: *Proceedings of the 2009 2nd International Symposium on Electronic Commerce and Security*, Nanchang, China, pp. 184–189.

Kaleli, C., Polat, H., (2012), "Privacy-preserving SOM-based recommendations on horizontally distributed data," *Knowledge-Based Systems*, **33** (0), 124–135.

Kargupta, H., Datta, S., Wang, Q., Sivakumar, K., (2003), "On the privacy preserving properties of random data perturbation techniques," In: *Proceedings of the 3rd IEEE International Conference on Data Mining*, Melbourne, Florida, USA, pp. 99–106.

Kim, D., Yum, B.-J., (2005), "Collaborative filtering based on iterative principal component analysis," *Expert Systems with Applications*, **28** (4), 823–830.

Kim, H.-N., Ji, A.-T., Ha, I., Jo, G.-S., (2010), "Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation," *Electronic Commerce Research and Applications*, **9** (1), 73–83.

Kohonen, T., (1989), *Self-organization and associative memory*, Springer-Verlag Inc., New York, NY, USA.

Korürek, M., Nizam, A., (2010), "Clustering MIT–BIH arrhythmias with ant colony optimization using time domain and PCA compressed wavelet coefficients," *Digital Signal Processing*, **20** (4), 1050–1060.

Kumari, R. S. S., Sadasivam, V., (2007), "A novel algorithm for wavelet based ECG signal coding," *Computers & Electrical Engineering*, **33** (3), 186–194.

Lathia, N., Hailes, S., Capra, L., (2007), "Private distributed collaborative filtering using estimated concordance measures," In: *Proceedings of the 2007 ACM Conference on Recommender Systems*, Minneapolis, Minnesota, USA, pp. 1–8.

Lee, S. K., Cho, Y. H., Kim, S. H., (2010), "Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations," *Information Sciences*, **180** (11), 2142–2155.

Liang, Z., Bo, X., Jun, G., (2008), "A hybrid approach to collaborative filtering for overcoming data sparsity," In: *Proceedings of the 9th International Conference on Signal Processing*, Beijing, China, pp. 1595–1599.

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

Ma, H., King, I., Lyu, M. R., (2007), "Effective missing data prediction for collaborative filtering," In: *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Amsterdam, The Netherlands, pp. 39–46.

MacQueen, J., (1967), "Some methods for classification and analysis of multivariate observations," In: *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability* Berkeley, California, USA, pp. 281–297.

McLaughlin, R., (2003), "Big box retail in the new economy: Place-making in the era of the electronic milkman," CharretteCentre.Com, July, 2003.

Melville, P., Mooney, R. J., Nagarajan, R., (2002), "Content-boosted collaborative filtering for improved recommendations," In: *Proceedings of 18th National Conference on Artificial Intelligence*, Edmonton, Alberta, Canada, pp. 187–192.

Miyahara, K., Pazzani, M. J., (2000), "Collaborative filtering with the simple Bayesian classifier," In: *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, Melbourne, Australia, pp. 679–689.

Miyahara, K., Pazzani, M. J., (2002), "Improvement of collaborative filtering with the simple Bayesian classifier," *The Information Processing Society of Japan Journal*, **43** (1), 3429–3437.

Pazzani, M. J., (1999), "A framework for collaborative, content-based and demographic filtering," *Artificial Intelligence Review*, **13** (5-6), 393–408.

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

Polat, H., Du, W., (2003), "Privacy-preserving collaborative filtering using randomized perturbation techniques," In: *Proceedings of the 3rd IEEE International Conference on Data Mining*, Melbourne, Florida, USA, pp. 625–639.

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

Polat, H., Du, W., (2005b), "SVD-based collaborative filtering with privacy," In: *Proceedings of the 2005 ACM Symposium on Applied Computing*, Santa Fe, New Mexico, USA, pp. 791–795.

Polat, H., Du, W., (2006), "Achieving private recommendations using randomized response techniques," Lecture Notes in Artificial Intelligence, **3918**, pp. 637–646.

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

Roh, T. H., Oh, K. J., Han, I., (2003), "The collaborative filtering recommendation based on SOM cluster-indexing CBR," *Expert Systems with Applications*, **25** (3), 413–423.

Runran Liu, Chunxiao Jia, T. Z. D. S. B. W., (2009), "Personal recommendation via modified collaborative filtering," *Physica A: Statistical Mechanics and its Applications*, **388** (4), 462–468.

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

Sarwar, B., Karypis, G., Konstan, J., Riedl, J. T., (2000), "Analysis of recommendation algorithms for e-commerce," In: *Proceedings of the 2nd ACM Conference on Electronic Commerce*, Minneapolis, Minnesota, USA, pp. 158–167.

Sarwar, B., Karypis, G., Konstan, J., Riedl, J. T., (2001), "Item-based collaborative filtering recommendation algorithms," In: *Proceedings of the 10th*

International Conference on World Wide Web, Hong Kong, China, pp. 285–295.

Shannon, C. E., (1948), "A mathematical theory of communication," *Bell System Technical Journal*, **27**, 379–423.

Shyu, M.-L., Haruechaiyasak, C., Chen, S.-C., Zhao, N., (2005), "Collaborative filtering by mining association rules from user access sequences," In: *Proceedings of the International Workshop on Challenges in Web Information Retrieval and Integration*, Tokyo, Japan, pp. 128–135.

Steinbach, M., Karypis, G., Kumar, V., (2000), "A comparison of document clustering techniques," In: *Procedings of KDD Workshop on Text Mining*, Boston, Massachusetts, USA.

Su, X., Khoshgoftaar, T. M., (2009), "A survey of collaborative filtering techniques," *Advances in Artificial Intelligence*, *2009*, 1–19.

Surowiecki, J., (2004), *The wisdom of crowds*, Anchor books.

Symeonidis, P., Nanopoulos, A., Papadopoulos, A. N., Manolopoulos, Y., (2008), "Collaborative recommender systems: Combining effectiveness and efficiency," *Expert Systems with Applications*, **34** (4), 2995–3013.

Tada, M., Kikuchi, H., Puntheeranurak, S., (2010), "Privacy-preserving collaborative filtering protocol based on similarity between items," In: *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications*, Perth, WA, pp. 573 –578.

Thilagamani, S., S. N., (2011), "A novel recursive clustering algorithm for image oversegmentation," *European Journal of Scientific Research*, **52**, 430–436.

Van de Plas, R., De Moor, B., Waelkens, E., (2008), "Discrete wavelet transform-based multivariate exploration of tissue via imaging mass spectrometry," In: *Proceedings of the 2008 ACM Symposium on Applied computing*, Fortaleza, Ceará, Brazil,pp. 1307–1308.

Van Roy, B., Yan, X., (2010), "Manipulation robustness of collaborative filtering," *Management Science*, **56** (11), 1911–1929.

Verhaegh, W. F. J., Duijnhoven, A. E. M., Tuyls, P., Korst, J., (2004), "Privacy protection in memory-based collaborative filtering," *Lecture Notes in Computer Science*, **3295**, 61–71.

Vozalis, M., Markos, A., Margaritis, K., (2009), "On the performance of SVD-based algorithms for collaborative filtering," In: *Proceedings of the 2009 4th Balkan Conference in Informatics*, Thessaloniki, Greece, pp. 245–250.

Vozalis, M. G., Margaritis, K. G., (2007), "Using SVD and demographic data for the enhancement of generalized collaborative filtering," *Information Sciences*, **177** (15), 3017–3037.

Vozalis, M. G., Markos, A., Margaritis, K. G., (2010), "Collaborative filtering through SVD-based and hierarchical nonlinear PCA," In: *Proceedings of the 20th International Conference on Artificial Neural Networks*, Thessaloniki, Greece, pp. 395–400.

Warner, S. L., (1965), "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, **60** (309), 63–69.

Westin, A. F., (1968), "Privacy and freedom," *Washington and Lee Law Review*, **25** (1), 1–5.

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

Yakut, I., Polat, H., (2007), "Privacy-preserving eigentaste-based collaborative filtering," In: *Proceedings of the Security 2nd International Conference on Advances in Information and Computer Security*, Nara, Japan, pp. 169–184.

Yang, Z., Chung, F.-L., Shitong, W., (2009), "Robust fuzzy clustering-based image segmentation," *Applied Soft Computing*, **9** (1), 80–84.

Yildirim, H., Krishnamoorthy, M. S., (2008), "A random walk method for alleviating the sparsity problem in collaborative filtering," In: *Proceedings of the 2008 ACM Conference on Recommender Systems*, Lausanne, Switzerland, pp. 131–138.

Yu, G., Kamarthi, S. V., (2010), "A cluster-based wavelet feature extraction method and its application," *Engineering Applications of Artificial Intelligence*, **23** (2), 196–202.

Zhang, F., Chang, H., (2006), "A collaborative filtering algorithm employing genetic clustering to ameliorate the scalability issue," In: *Proceedings of the IEEE International Conference on e-Business Engineering*, Shanghai, China, pp. 331–338.

Zhang, F., you Chang, H., (2005), "A collaborative filtering algorithm embedded BP network to ameliorate sparsity issue," In: *Proceedings of 2005 International Conference on Machine Learning and Cybernetics*, Guangzhou, China, pp. 1839–1844.

Zhang, S., Wang, W., Ford, J., Makedon, F., Pearlman, J., (2005), "Using singular value decomposition approximation for collaborative filtering," In: *Proceedings of the 7th IEEE International Conference on E-Commerce Technology*, Munich, Germany, pp. 257–264.

Zhao, X., Ye, B., (2009), "Similarity of signal processing effect between Hankel matrix-based SVD and wavelet transform and its mechanism analysis," *Mechanical Systems and Signal Processing*, **23** (4), 1062–1075.

Ziqiang, W., Boqin, F., (2004), "Collaborative filtering algorithm based on mutual information," *Lecture Notes in Computer Science*, **3007**, pp. 405–415.