DERIVING PRIVATE NUMERIC DATA IN PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEMS

Ph.D. Dissertation

Burcu DEMİRELLİ OKKALIOĞLU

Eskişehir 2017

DERIVING PRIVATE NUMERIC DATA IN PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEMS

Burcu DEMİRELLİ OKKALIOĞLU

Ph.D. Dissertation

Computer Engineering Program
Supervisor: Assoc. Prof. Dr. Cihan KALELİ

Eskişehir
Anadolu University
Graduate School of Science
November 2017

This dissertation is supported by TUBİTAK under grant 113E262

FINAL APPROVAL FOR THESIS

This thesis titled "**Deriving Private Numeric Data in Privacy-Preserving Collaborative Filtering Systems**" has been prepared and submitted by Burcu DEMİRELLİ OKKALIOĞLU in partial fulfillment of the requirements in "Anadolu University Directive on Graduate Education and Examination" for the Degree of Doctor of Philosophy (Ph.D.) in Computer Engineering Department has been examined and approved on 13/11/2017.

<u>**Committee Members**</u>		<u>**Signature**</u>
Member (Supervisor)	: Assoc. Prof. Dr. Cihan KALELİ	………………..
Member	: Asst. Prof. Dr. Mehmet KOÇ	………………..
Member	: Asst. Prof. Dr. Alper BİLGE	………………..
Member	:Asst. Prof. Dr. Adem TUNCER	………………..
Member	:Asst. Prof. Dr. Ahmet ARSLAN	………………..

………………………………

Director

Graduate School of Science

ABSTRACT

DERIVING PRIVATE NUMERIC DATA IN PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEMS

Burcu DEMİRELLİ OKKALIOĞLU

Computer Engineering Program

Anadolu University, Graduate School of Science, November 2017

Supervisor: Assoc. Prof. Dr. Cihan KALELİ

There are a great number of methods introduced to offer reliable and accurate recommendations in privacy-preserving collaborative filtering systems without disclosing individual or data holders' private data. Privacy has become a fundamental principle in privacy-preserving collaborative filtering systems. Several data disguising methods have been proposed to alleviate users' privacy concerns. Randomization, which is one of the such proposed methods, is commonly utilized to protect individual private data while allowing collaborative filtering systems to produce decent recommendations. However, recent studies show that it is likely to reconstruct private data from the data disguised by randomization.

The dissertation scrutinizes whether the private numeric data can be reconstructed from central, partitioned and distributed data-based privacy-preserving schemes by planning privacy attack scenarios. In central data-based schemes, noise from z-scores must be eliminated first by employing the noise elimination methods. A new method is proposed to derive the original ratings from the estimated z-score values utilizing public data. It is also crucial to reconstruct rated items when users inconsistently perturb their confidential data. In partitioned data based-schemes, deriving other party's private data from the aggregate results are investigated. In distributed data-based schemes, data holders might coalesce against one holder to derive its private data. In addition, solutions are proposed by utilizing various domain-related auxiliary information and characteristic features of collaborative filtering systems to enhance the reconstructions. Theoretical analysis and experimental outcomes demonstrate that the proposed approaches help attackers derive a meaningful amount of private data in certain circumstances.

Keywords: Privacy, data reconstruction, auxiliary information, privacy attack, collaborative filtering.

ÖZET

GİZLİLİK TABANLI ORTAK FİLTRELEME SİSTEMLERİNDE GİZLİ NÜMERİK VERİLERİN ELDE EDİLMESİ

Burcu DEMİRELLİ OKKALIOĞLU

Bilgisayar Mühendisliği Anabilim Dalı

Anadolu Üniversitesi, Fen Bilimleri Enstitüsü, Kasım 2017

Danışman: Doç. Dr. Cihan KALELİ

Gizlilik tabanlı ortak filtreleme sistemlerinde kişilerin veya veri sahiplerinin mahrem verilerini ifşa etmeden güvenilir ve doğru öneriler sunmak için çok sayıda yöntem uygulanmıştır. Mahremiyet gizlilik tabanlı ortak filtreleme sistemlerinde temel prensiptir. Kullanıcıların mahremiyet endişelerini gidermek için farklı veri gizleme metotları sunulmuştur. Bu metotlardan biri olan rasgeleleştirme ortak filtreleme sistemlerinin iyi sonuçlar üretmesini mümkün kılarken aynı zamanda mahrem veriyi korumak için yaygın biçimde faydalanılan bir yöntemdir. Fakat, yakın zamanda ki çalışmalar rasgeleleştirme ile saklanmış veriden mahrem veriyi imar etmenin olası olduğunu göstermiştir.

Tez merkezi, bölünmüş ve dağıtık veri-tabanlı gizliliği koruyan ortak filtreleme şemalarından saldırı senaryoları planlayarak mahrem nümerik veri imarının yapılıp yapılamayacağını inceler. Merkezi veri-tabanlı şemalarda, gürültü ayıklama yöntemleri kullanılarak z-skorlardan gürültü giderilmelidir. Herkese açık veri kullanarak z-skorlardan orijinal değerlemeleri elde etmek için yeni bir metot önerilmiştir. Kullanıcılar kişisel mahrem verilerini tutarsız şekilde sakladıklarında oylanan ürünleri imar etmek de çok önemlidir. Bölünmüş veri-tabanlı şemalarda, toplam sonuçlardan diğer partinin mahrem verisinin elde edilmesi incelenmiştir. Dağıtık veri-tabanlı şemalarda, bir veri sahibinin mahrem verisini elde etmek için diğer veri sahipleri ona karşı iş birliği yapabilirler. Bunun yanında, imar sonuçlarını iyileştirmek için alanla ilgili yardımcı bilgiler ve ortak filtreleme sistemlerinin karakteristik özelliklerini kullanan çözümler sunulmuştur. Teorik analizler ve deneysel sonuçlar göstermiştir ki sunulan yaklaşımlar saldırganlara belli durumlarda anlamlı ölçüde mahrem bilgi çıkarmasına yardımcı olur.

Anahtar Sözcükler: Gizlilik, veri imarı, yardımcı bilgi, gizlilik saldırısı, ortak filtreleme.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Assoc. Prof. Dr. Cihan Kaleli, for the support, encouragement and guidance throughout the dissertation. It has been a pleasure to work with him.

I would like to thank my dissertation committee members, Asst. Prof. Dr. Mehmet Koç, Asst. Prof. Dr. Alper Bilge, Asst. Prof. Dr. Adem Tuncer, and Asst. Prof. Dr. Ahmet Arslan for their valuable contributions.

I would also like to thank all my family members, especially my beloved mother and father, for their endless support. You always encourage me and provide emotional support in my life.

A special thanks to my beloved husband, Murat Okkalıoğlu. I cannot express my appreciation to you. You are always there when I need the support.

Burcu Demirelli Okkalıoğlu
November, 2017

13/11//2017

STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES

I hereby truthfully declare that this thesis is an original work prepared by me; that I have behaved in accordance with the scientific ethical principles and rules throughout the stages of preparation, data collection, analysis and presentation of my work; that I have cited the sources of all the data and information that could be obtained within the scope of this study, and included these sources in the references section; and that this study has been scanned for plagiarism with “scientific plagiarism detection program” used by Anadolu University, and that “it does not have any plagiarism” whatsoever. I also declare that, if a case contrary to my declaration is detected in my work at any time, I hereby express my consent to all the ethical and legal consequences that are involved.

Burcu DEMİRELLİ OKKALIOĞLU

TABLE OF CONTENTS

TITLE PAGE i
FINAL APPROVAL FOR THESIS ii
ABSTRACT iii
ÖZET iv
ACKNOWLEDGMENTS v
STATEMENT OF COMPLIANCE WITH ETHICAL PRINCIPLES AND RULES vi
LIST OF TABLES x
LIST OF FIGURES xii
ABBREVIATIONS xiii
1. INTRODUCTION 1
- **1.1. Collaborative Filtering** 1
- **1.2. Privacy-Preserving Collaborative Filtering** 4
- **1.3. Related Works** 8
- **1.4. Problem Definitions** 12
- **1.5. Contributions** 13
- **1.6. Organization of the Dissertation** 15

2. PRELIMINARIES 16
- **2.1. Data Disguising by Randomization** 16
  - **2.1.1. Randomized perturbation techniques** 16
  - **2.1.2. Random filling** 18
- **2.2. Providing Predictions in PPCF** 18
  - **2.2.1. Providing predictions in the central server-based PPCF** 19
  - **2.2.2. Providing predictions in the partitioned data-based PPCF** 20
  - **2.2.3. Providing clustering-based predictions in the distributed data-based PPCF** 25

3. DERIVING THE ORIGINAL DATA PERTURBED BY RANDOMIZATION IN CENTRAL DATA-BASED PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEM 32

3.1. Introduction ... 32

3.2. Data Reconstruction Methods ... 33

3.2.1. k-means clustering-based reconstruction ... 33

3.2.2. SVD-based reconstruction with expectation-maximization ... 34

3.3. Utilizing Auxiliary Information to Improve Data Reconstruction Methods ... 35

3.3.1. Popular and unpopular movies ... 37

3.3.2. Prestigious awards ... 37

3.3.3. Users' demographic information ... 38

3.3.4. Prior knowledge about mean of each item ... 39

3.3.5. Prior knowledge about mean of each user's ratings ... 40

3.4. The Proposed Method, Reconstruction from the Estimated z-score Values ... 40

3.5. Complexity Analysis ... 42

3.6. Experiments ... 43

3.6.1. Data set and evaluation metric ... 43

3.6.2. Methodology ... 44

3.6.3. Experimental results ... 44

3.7. Conclusions ... 60

4. RECONSTRUCTING RATED ITEMS FROM INCONSISTENTLY PERTURBED DATA IN CENTRAL DATA-BASED PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEM ... 62

4.1. Introduction ... 62

4.2. Noise Elimination Methods ... 63

4.2.1. Singular value decomposition ... 63

4.2.2. Discrete cosine transformation ... 64

4.2.3. Principal component analysis ... 64

4.2.4. QR factorization ... 65

4.3. Filling the Missing Entries ... 65

4.4. Estimating Number of Real Ratings from Inconsistently Perturbed Data ... 66

4.5. Utilizing Auxiliary Information 67

4.5.1. Features of random data distributions 68

4.5.2. Total number of votes 68

4.5.3. User demographic information 69

4.6. Complexity Analysis 70

4.7. Experiments 71

4.7.1. Data set and evaluation metric 72

4.7.2. Methodology 72

4.7.3. Experimental results 72

4.8. Conclusions 86

5. ESTIMATING THE PRIVATE DATA IN DISTRIBUTED DATA-BASED PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEM 88

5.1. Introduction 88

5.2. Privacy Attack Scenarios on Partitioned Data 89

5.2.1. Privacy attack scenarios on horizontally partitioned data 89

5.2.2. Privacy attack scenarios on vertically partitioned data 95

5.3. Privacy Attack Scenarios on Distributed Data 97

5.3.1. Privacy attack scenarios on horizontally distributed data 97

5.3.2. Privacy attack scenarios on vertically distributed data 101

5.4. Experiments 103

5.4.1. Data set and evaluation metric 103

5.4.2. Methodology 103

5.4.3. Experimental results 104

5.5. Conclusions 107

6. CONCLUSIONS AND FUTURE WORKS 109

REFERENCES 113

CIRRUCULUM VITAE 122

LIST OF TABLES

Table 3.1. Genre preferences with age 39
Table 3.2. The computational complexity of auxiliary information 43
Table 3.3. k-means clustering-based reconstruction results with popular movies 45
Table 3.4. k-means clustering-based reconstruction results with unpopular movies ... 47
Table 3.5. k-means clustering-based reconstruction with popular movies, the Oscars, and merged movies 49
Table 3.6. k-means clustering based reconstruction results with items’ mean 50
Table 3.7. k-means clustering-based reconstruction results with users’ mean 51
Table 3.8. k-means clustering-based results with genre preferences with gender 52
Table 3.9. k-means clustering-based results with genre preferences with age 53
Table 3.10. k-means clustering-based reconstruction results with combined auxiliary information 54
Table 3.11. Error rates of true ratings after SVD-based reconstruction is applied 56
Table 3.12. Error rates of true ratings after SVD-based reconstruction is applied based on different auxiliary information 57
Table 3.13. Error rates of true ratings after the proposed method is applied to SVD-based reconstruction with different auxiliary information 58
Table 4.1. The computational complexity of the proposed solution 71
Table 4.2. The computational complexity of used auxiliary information 71
Table 4.3. Recall and precision results for different reconstruction methods when β = 6% 74
Table 4.4. Recall and precision results for SVD when σ is 1 75
Table 4.5. Recall and precision results for SVD when β is 6% 76
Table 4.6. Recall and precision results based on IMDb data when σ is 1 77
Table 4.7. Recall and precision results based on distribution type’s feature when σ is 1 79
Table 4.8. Recall and precision results based on distribution type’s feature when β is 6% 79
Table 4.9. Recall and precision results with σ being 1 when auxiliary information and SVD is used together 81

Table 4.10. Recall and precision results based on SVD with σ being 1 when the β is random 83

LIST OF FIGURES

Figure 2.1. Random filling scenarios 18
Figure 2.2. A general view of the knn-based CF algorithm 19
Figure 2.3. The horizontally partitioned data-based PPCF 23
Figure 2.4. The vertically partitioned data-based PPCF 25
Figure 3.1. Comparison of the existing k-means clustering-based reconstruction 46
Figure 3.2. Comparison of the existing k-means clustering-based reconstruction with the Oscars winner movies and the Oscars winner and popular movies 49
Figure 3.3. Comparison of the existing k-means based reconstruction with different auxiliary information 55
Figure 3.4. Comparison of estimation of true ratings after the proposed method is applied to SVD-based reconstruction with different auxiliary information 59
Figure 4.1. Recall with different default values 73
Figure 4.2. Recall with β = 6% for utilizing IMDb data 78
Figure 4.3. Precision values for SVD, auxiliary information (total votes) and experiment 5 when uniform distribution with β=6% is used 82
Figure 4.4. Precision values for SVD, auxiliary information (total votes) and experiment 5 when mix distribution with β=6% is used 82
Figure 4.5. Precision with fixed and random β for SVD 84
Figure 4.6. Precision with β = 6% for SVD and demographic information with SVD . 85
Figure 4.7. Precision with σ = 1 for SVD and demographic information with SVD 86
Figure 5.1. A sample of transforming the original matrix 90
Figure 5.2. An example of the targeted protocol 91
Figure 5.3. Auxiliary information: locations of the one-time-rated items 95
Figure 5.4. Auxiliary information: locations of the two times-rated items 96
Figure 5.5. An example of the targeted distributed PPCF protocol 99
Figure 5.6. Accuracy on horizontally partitioned data 104
Figure 5.7. Accuracy on random horizontally partitioned data 105
Figure 5.8. Accuracy on vertically partitioned data 106
Figure 5.9. Accuracy on random vertically partitioned data 107

ABBREVIATIONS

Σ	: Standard Deviation
A	: Active user
CF	: Collaborative Filtering
D	: Density
DCT	: Discrete Cosine Transformation
EM	: Expectation-Maximization
HDD	: Horizontally Distributed Data
HE	: Homomorphic Encryption
HPD	: Horizontally Partitioned Data
IMDb	: Internet Movie Database
k	: Number of Clusters
knn	: Number of Nearest Neighbors
m	: Number of Items
MAE	: Mean Absolute Error
MP	: Master Party
n	: Number of Users
PCA	: Principal Component Analysis
PCC	: Pearson Correlation Coefficient
PPCF	: Privacy-Preserving Collaborative Filtering
PSPCP	: Private Scalar Product Computation Protocol
q	: Target Item
RPT	: Randomized Perturbation Technique
SOM	: Self-Organizing Map
SVD	: Singular Value Decomposition
VDD	: Vertically Distributed Data
VPD	: Vertically Partitioned Data
z	: Number of Parties

To my grandmother

1. INTRODUCTION

With the increasing development and use of technology, many people have begun to shop over the Internet nowadays. There are numerous e-commerce sites which sell a great number of goods online. People can compare what they desire from different e-commerce sites within minutes and buy it in a few seconds. On the other hand, the Internet is like the ocean and finding out the right product that meets the expectation sometimes takes a long time. Although purchasing a new product through the Internet accomplishes in the blink of an eye, deciding which product to buy takes a long time for individuals due to overwhelmingly many options to choose from. This phenomenon causes to emerge a new term, which is called *information overload*. The information overload has become the primary problem for users to deal with. To overcome the information overload problem, recommender systems have emerged. Before the Internet became widespread and popular, people had shared their recommendations about movies, books, music, restaurants, and sightseeing trips with like-minded friends, families, and relatives. Today, recommender systems take like-minded friends' and families' place and assist people with their choices whenever they need.

1.1. Collaborative Filtering

Collaborative filtering (CF) is a relatively recent technique used by recommender systems. CF was first introduced by Golberg et al. (1992), who proposed a system called Tapestry for email filtering. CF helps users narrow down the number of choices they have to deal with. The underlying assumption of the CF is to collect users' preferences or tastes and then finds a subset of users with similar preferences in the past. It is possible that two users with the same or similar preferences in the past will prefer other similar items in the future as well. Although a single vendor can collect users' preferences, it is likely that users' preferences are distributed vertically or horizontally among different parties. There are a great number of e-commerce sites, and the new one is added to them every day. Thus, collecting users' preferences by a single vendor is not always possible in a CF system.

The type of user preference is another property of CF systems. Such preferences can be represented as numeric or binary ratings. In the numeric ratings, a rating scale indicates a minimum and maximum rating values. A user chooses his rating from this range in the numeric scale while users either rate items as like or dislike in the binary ratings. The numeric ratings-based CF data set is employed in the dissertation.

There are two goals in CF systems, which either recommend a list of new items called *recommendation* (or *top-N recommendation*) or predict a particular item for a specific user (referred to as an active user) called *prediction* based on other like-minded users and their previous ratings. In order to achieve such goals, the proposed CF algorithms are divided into two main classes, which are memory-based and model-based (Breese, Heckerman and Kadie, 1998). In addition, hybrid schemes are introduced by combining different type of recommendation techniques (Burke, 2002).

Memory-based schemes are the most popular algorithms used in CF systems, which are also called as *neighborhood-based* CF (Breese, Heckerman and Kadie, 1998; Resnick et al., 1994; Konstan et al., 1997; Herlocker et al., 1999; Sarwar et al., 2000; 2001). Memory-based algorithms are divided into two parts in terms of the implementation, either user-based CF (Resnick et al., 1994) or item-based CF (Sarwar et al., 2001). While user-based algorithms discover a correlation between users, item-based algorithms examine relationships between items. The first memory-based algorithm for a CF system utilizing user-based approach is introduced by GroupLens (Resnick et al., 1994; Konstan et al. 1997). According to memory-based algorithms, the whole user-item matrix is used to produce a prediction. The scheme consists of two steps. The first step calculates similarity among all users (or items) and an active user. There are various similarity metrics used in measuring correlation. The most popular and prevalent one is the Pearson correlation coefficient (PCC) (Resnick et al., 1994). PCC is given in Eq. 1.1.

In Eq. 1.1, v_{aj} and v_{uj} are the ratings of user *a* and user *u* for item j, $\overline{v_a}$ and $\overline{v_u}$ are the averages of user *a* and user *u* and *j* illustrates the item sets both user *a* and user *u* rate commonly.

$$w_{au=}\frac{\sum_j (v_{aj}-\overline{v_a})(v_{uj}-\overline{v_u})}{\sqrt{\sum_j (v_{aj}-\overline{v_a})^2}\sqrt{\sum_j (v_{uj}-\overline{v_u})^2} \tag{1.1}$$

Instead of measuring the correlation between two users, similarities between two items can also be computed. Sarwar et al. (2001) propose three methods to calculate similarities between items, which are a cosine-based similarity, correlation-based similarity, and adjusted-cosine similarity. Their experimental results show that the adjusted-cosine similarity gives the best results to calculate the similarities between items. The adjusted-cosine similarity is written as in Eq. 1.2,

$$sim_{ij} = \frac{\sum_{u \in U}(v_{ui} - \overline{v_u})(v_{uj} - \overline{v_u})}{\sqrt{\sum_{u \in U}(v_{ui} - \overline{v_u})^2}\sqrt{\sum_{u \in U}(v_{uj} - \overline{v_u})^2}} \tag{1.2}$$

where *U* refers to the set of users, who rated both items *i* and *j*. v_{ui} and v_{uj} are ratings of user *u* on item *i* and *j*, respectively. $\overline{v_u}$ is the average of ratings of user *u*.

After measuring similarities, the next step is to choose a subset of users or items (also known as neighbors). There are two techniques to identify the neighbors (Herlocker et al., 1999). The first technique called *best k-neighbors* is to select the best *k* users from all user (*k* << *n*) based on similarity weights. The second one called *correlation-thresholding* is to determine users based on the predefined threshold value. If the similarity weight of a user is greater than the predefined threshold value, the user is picked as a neighbor. After choosing neighbors, the prediction is produced based on the preferences of selected neighbors.

Model-based schemes have emerged to overcome the sparsity problem in memory-based schemes. Model-based schemes aim to build a model off-line by utilizing the preferences of users. Then, the prediction is produced using the constructed model. Unlike the memory-based schemes, the model-based schemes do not need the whole data set to compute predictions every time. There are different approaches used in model-based schemes. Some of the studies utilize probabilistic approaches, for instance; Bayesian networks (Breese, Heckerman and Kadie, 1998), while some of them use various machine learning techniques such as clustering (Breese, Heckerman and Kadie, 1998; Ungar and Foster, 1998; Kim and Ahn, 2008; Bilge and Polat, 2013a), support vector machines (Grčar et al., 2006), and matrix factorization (Koren, Bell and Volinsky, 2009; Hernando, Bobadilla and Ortega, 2016; Ortega et al. 2016). Although there are different approaches used in model-based schemes, the main idea of all used approaches is to construct a model before producing predictions or recommendations.

Hybrid-model schemes are proposed to cope with the weaknesses of any individual recommender methods. Hybrid-model combines two or more recommender methods to increase the quality of recommendations. A hybrid model may be a combination of CF with content-based filtering (Claypool et al., 1999; Barragáns-Martínez et al., 2010;

Campos et al., 2010), CF with demographic filtering (Vozalis and Margaritis, 2007), CF with content-based and demographic filtering (Pazzani ,1999), CF with knowledge-based filtering [Burke, 2002; Martinez, Perez and Barranco, 2009; Zanker and Jessenitschnig, 2009), CF with other CF algorithms (Pennock et al., 2000; Wang et al., 2014), and so on. Most of the hybrid-model schemes rely on the probabilistic methods such as neural networks (Christakou, Vrettos and Stafylopatis, 2007; Hsu et al.,2007), Bayesian networks (Campos et al., 2010), genetic algorithms (Gao and Li, 2008; Salehi, Pourzaferani and Razavi, 2013), fuzzy association rules (Lucas et al., 2012), latent features (Maneeroj and Takasu, 2009), and clustering (Shinde and Kulkarni, 2012). A detailed survey about the classification of hybrid recommender systems is demonstrated by Burke (2002).

1.2. Privacy-Preserving Collaborative Filtering

CF systems need reliable and genuine preferences of users in order to make accurate recommendations. However, users' data is a valuable asset and it can be sold in case of bankruptcy (Canny, 2002a; 2002b). Moreover, CF is a potential threat to privacy and it poses different privacy risks like price discrimination, government surveillance, unsolicited marketing, and so on (Cranor, 2003). Cranor, Reagle and Ackerman (2000) conduct a survey about the privacy concerns of users. They indicate a couple of major results according to the survey outcomes. When users do not need to express personally identifiable information, they are much more enthusiastic to share their information; otherwise, they might hesitate to share their private data or give false information. Besides, users might think that some types of data are more sensitive than others. For example; users can share their e-mail address as their contact information whereas they may not want to share their phone number. They assume that their phone number is more confidential data than the e-mail address. Unsolicited communication is another problem. Users would be willing to share some personal information such as postal code, address, or age in order to get free coupons; however, they do not want this kind of information to be shared with other companies for different purposes. As seen in the survey results, privacy has become a major concern in CF systems. Since CF systems fail to protect users' privacy, *privacy-preserving collaborative filtering* (PPCF) has emerged. The underlying approach of PPCF is to alleviate users' or data holders' privacy concerns while providing accurate recommendations.

In PPCF schemes, the private data is masked before users send it to the CF system. There are different data disguising methods used to protect individuals' privacy. *Anonymization* is one of them, which prevents identity disclosure (Pfitzmann and Hansen, 2010; Yang and Qiao, 2010). However, it cannot guarantee the quality of the data. The data set may contain some made-up information due to the anonymous communication. The second method is a *secure multi-party computation* that allows parties to perform distributed calculations based on their private data without revealing anything except the output (Canny 2002a; Bogetoft et al. 2009; Lindell and Pinkas, 2009; Li et al., 2016). Secure multi-party computation protocols should be efficient when large data sets are used for computation. Since cryptography is commonly used, it requires heavy calculations for very large data sets to carry out the secure multi-party computation. Another method is called *randomized perturbation technique (RPT)* (Polat and Du, 2003; 2005a). RPT is a widespread technique in PPCF schemes, which disguises user ratings while allowing the data collector (or the recommender system) to perform CF services without knowing the original ratings and/or the rated items.

PPCF schemes can be classified as centralized or decentralized according to the location of data. The main distinction between these two classifications in terms of privacy is that the centralized PCCF schemes receive perturbed user vectors while the decentralized PPCF schemes hold original user vectors. For this reason, the primary privacy concern for the centralized PPCF scheme is to protect individual data from the server, which is called *individuals'* or *users' privacy*. On the other hand, the objective in the distributed PPCF schemes is to protect original user vectors from unauthorized access of other parties while collaborating for CF purposes. This privacy protection effort is called *data holder's*, *institutional*, or *corporate privacy*.

When the data is collected on the central server, Polat and Du (2003; 2005a) first introduce two PPCF schemes, which utilize the neighborhood-based algorithm and singular value decomposition (SVD)-based algorithm, using RPTs. According to their schemes, users first disguise their original ratings based on the server's instruction and send their concealed z-score values to the server. In this way, the server cannot learn the actual ratings from the disguised data while still producing accurate and reliable predictions. Zhang, Ford and Makedon (2006b) propose a two-way communication PPCF scheme which users mask their original ratings according to the server's perturbation guidance instead of the same perturbation level. Their scheme consists of two parts, communication, and perturbation. In the communication part, when a user wants to send his private data to the server, the user first needs to send a message to the server for receiving the current disclosure level. The importance of each item can differ, so the server determines the current disclosure level. If the user agrees with the received disclosure level, the server sends the perturbation guide. In the perturbation part, the user disguises his ratings according to the perturbation guide and sends them to the server. Yakut and Polat (2007) propose a modified Eigentaste-based CF algorithm utilizing *k*-means clustering algorithm to produce predictions and recommendations. They employ randomization to perturb the private data. Polatidis et al. (2017) propose a multi-level method for the centralized PPCF. Their method allows users to select their privacy level and ranges of random values from a fixed range of privacy level. Each rating is perturbed using the selected privacy level by users before the disguised rating is sent to the server. Five different data sets are used in the experiments, and privacy level for each data set is defined first. After determining a fixed range of privacy level for the data set such as the first two level for MovieLens, each user selects his privacy level (***p***) whenever he wants to perturb one of his ratings. In this case, the user randomly chooses either 1 or 2. Then, the user generates ranges of random values [-*p*, …,0, … *p*]. For each rating, a value is randomly selected from the range of random values [-*p*, …,0, … *p*] and the selected random value is added to the original data. The method is repeated until each rating is disguised. Polat and Du (2007) also propose inconsistent data disguising methods in PPCF. They claim that each user has a different level of privacy concerns. Some users only want to hide their rated items' values while some users desire to conceal their rated and/or unrated items. Depending on users concerns, several scenarios are introduced. While some users can prefer to perturb some of the unrated items together with rated items, some other users can perturb all unrated and rated items together. Besides, users can define their perturbation level independently instead of using the same perturbation level. Bilge and Polat (2012) propose to use discrete wavelet transform (DWT)-based data reduction to solve privacy issue while applying inconsistent data disguising method. The researchers also propose a privacy-preserving scheme based on bisecting *k*-means clustering (Bilge and Polat, 2013b). The proposed scheme includes two preprocessing stages. The first stage is to build a binary decision tree using bisecting *k*-means algorithm while the second stage is to produce clones of users by inserting pseudo-self-predictions into the original user data. Zhang, Zhu and Zhang (2014) propose a privacy-preserving scheme with a new similarity function which is a modified algorithm in (Polat and Du, 2007). They introduce a new term called *privacy-preserving intensity weight* to measure similarities. They claim that the level of perturbation can be changed based on the parameters (β, σ). If more perturbation is added to the original data, the accuracy decreases. Therefore, they propose to calculate privacy-preserving intensity weight based on the parameters. Users send their disguised data as in (Polat and Du, 2007) and their value of intensity weight to the server. The server uses the value of intensity weight with users' disguised data in order to measure similarities and produces a prediction based on the adjusted similarity function.

There are also different studies proposed to protect privacy in decentralized PPCF. Canny (2002a; 2002b) first describes a privacy term in CF systems using peer-to-peer protocol. The proposed scheme allows users to compute the aggregate data using their preferences without exposing their privacy. With the help of the homomorphic encryption (HE), recommendations are performed safely in distributed PPCF. Polat and Du (2005b) propose a privacy-preserving scheme to produce a prediction when data is vertically partitioned among two-parties. Their scheme consists of two stages: off-line and online. The parties need to exchange their data to produce an accurate prediction. While interchanging data, the properties of randomization, permutation, and HE are used to protect the data holder's privacy. Yakut and Polat (2010) describe a new privacy-preserving CF algorithm by utilizing SVD on partitioned (horizontally and vertically) data. Their experimental results show that it is likely to produce predictions while preserving data holders' privacy. Zhan et al. (2010) propose an encryption-based scheme to provide recommendations on horizontally partitioned data (HPD). Jeckmans, Tang and Hartel (2012) also focus on a HE schemes with neighborhood-based CF on HPD between two parties. Parties use the HE to hide their private data. All users are selected as neighbors instead of the similar ones in their scheme. They claim that selecting all users as neighbors are to improve the performance in the encrypted domain. Yakut and Polat (2012a) claim that data might be cross distributed (or arbitrary partitioned) which is the combination of vertical and horizontal partitioning between two companies. They propose a hybrid CF algorithm on cross distributed data to protect data holders' privacy.

Their algorithms combine model-based CF and memory-based CF. After data holders' mask their data, each of data holder constructs a model off-line. Then, a prediction is produced based on the constructed model using the proposed memory-based algorithm. In another study (Yakut and Polat, 2012b), researchers propose an item-based prediction on arbitrary distributed data between two companies to preserve data holders' privacy. Kaleli and Polat (2012a; 2012b) introduce a privacy-preserving Self-Organizing Map (SOM)-based recommendation on horizontally (HDD) and vertically distributed data (VDD) among multiple companies. Their scheme is divided into two parts: off-line and online. In off-line part, SOM-based clustering is applied to cluster users held by parties, and the required aggregate data for recommendation process is computed. In order to compute the aggregate data, the scheme may show differences such as randomization, permutation and HE regarding data distribution. In the online case, after determining the target user's cluster, the prediction is produced using the nearest neighbor approach based on the received aggregate data. Kaleli and Polat (2011) also describe a trust-based privacy-preserving scheme on VDD among multiple parties. Shmueli and Tassa (2017) introduce a secure multi-party computation for vertically distributed item-based CF between various parties. According to their scheme, there is a mediator, and all vendors communicate with the mediator. Vendors do not need to communicate with other vendors while producing predictions.

1.3. Related Works

Randomization is a conventional method to preserve individuals' or data holders' privacy. Although the idea behind randomization is simple, it is an efficient method to hide the private information from the third party that has no right to access the confidential data. There are several works focusing on using the randomization approach in the privacy-preserving data mining in general (Agrawal and Srikant, 2000; Agrawal and Aggarwal, 2001; Evfimievski et al., 2004). Randomization is also used as a data disguising method in PPCF methods. Polat and Du (2003; 2005a) show that the data collector can perform CF computations with decent accuracy without knowing the original ratings. All users mask their ratings before sending them to the server. Although the server cannot learn the actual ratings due to RPTs, it is still able to estimate recommendations with decent accuracy.

Randomization seems to be a successful privacy measure; however, some studies show that it is possible to derive the original data from the perturbed data that is disguised by randomization. A recently published survey paper presents a detailed review of existing reconstruction methods and shows how to derive confidential data from the perturbed data, which is not only masked by randomization but also various data disguising methods (Okkalioglu B.D. et al., 2015). Agrawal and Srikant (2000) first proposed a reconstruction method, which reconstructs the distribution of the original data from the perturbed data. However, they do not consider that the attackers might want to retrieve the individual data. Agrawal and Aggarwal (2001) extend the work in (Agrawal and Srikant, 2000) using expectation maximization (EM) algorithm for reconstructing data. They show that when the data is large enough, the EM algorithm provides perfect estimates of the original distribution. Kargupta et al. (2003; 2005) first challenge that randomization may not hide the private information. Researchers claim that randomization preserves very little data privacy. They propose a technique, called a random matrix-based spectral filtering, to show how the original data can be derived from the perturbed data. The properties of random matrices are utilized in their proposed technique. Random matrices have very interesting properties and they are easily exploited. Therefore, the properties of random matrices are applied to a noise matrix. With the features of random matrices, the noise part can be removed from the perturbed data. Their results illustrate that the spectral properties of the data allow attackers to extract the noise easily from the perturbed data. They also show that when the amount of noise added to the original data increases, it causes less accurate estimation. Huang, Du and Chen (2005) scrutinize why and how correlations affect privacy. They claim that the prominent thing causing a security breach is the correlations among attributes. They propose two data reconstruction methods. One of them is based on the principal component analysis (PCA) and the other is based on Bayes estimation (BE). Their experimental outcomes illustrate that the higher the correlations among attributes are, the more the original data can be reconstructed accurately. Guo and Wu (2006) and Guo, Wu and Li (2008) improve the spectral filtering method. Their purpose is to achieve optimal performance. They define an upper bound, which helps the attacker compare how close their estimate to the original data. They show that attackers can exploit some private data using the upper bound with the spectral filtering method. Guo, Wu and Li (2006; 2008) propose SVD-based data reconstruction method to define a lower bound for the reconstruction error, which is helpful for users and data miners in order to evaluate how much noise should be added. They also show that the approach behind their SVD-based data reconstruction method is equal to the spectral filtering method. Therefore, the upper bound or lower bound can be used in both methods to achieve better performance by attackers or data miners. None of these reconstruction methods consider that the matrix is incomplete, and most of the data sets used are usually artificial. None of items or ratings is unrated, and all used data sets are complete. However, CF systems hold many unrated cells because of their nature. Zhang, Ford and Makedon (2006a) are the first to challenge random data perturbation in central server-based PPCF systems. Researchers divide their study into two parts according to the targeted studies (Polat and Du, 2003; 2005a): either only rated items or all items are used in the experiments. Researchers develop two data reconstruction methods, *k*-means clustering- and SVD-based scheme. Their empirical results exhibit that a considerable amount of original data can be derived using their proposed methods.

Most existing studies utilize reconstruction methods to derive the original ratings; however, some recent works have offered to utilize auxiliary information and/or characteristic properties of CF systems to improve accuracy. Calandrino et al. (2011) employ auxiliary information to infer users' transactions performed on a CF system. Researchers claim that fair amount of auxiliary information is enough to discover transaction by tracking temporal changes in the public output of CF systems. Some public data from popular services are used to show that their method works. They demonstrate that auxiliary information can be applied to discover user-related data from CF systems. Users' demographic data is also employed as a source of auxiliary information to improve recommendation quality for existing or new users. All these studies (Vozalis and Margaritis, 2007; Santos, Garcia Manzato and Goularte, 2014; Gogna and Majumdar, 2015; Al-Shamri 2016) exhibit different approaches by utilizing users' demographic information to generate more accurate rating predictions. In the dissertation, users' demographic information is utilized as auxiliary information like the previous studies; however, the aim is to improve the accuracy of reconstruction results instead of recommendation quality.

Besides deriving the original data, there are also some studies related to the reconstruction of the rated items from inconsistently perturbed data. Okkalioglu, Koc and Polat (2015a) utilize auxiliary information to discover originally rated items from randomly appended fake items. Okkalioglu, Koc and Polat (2015a) demonstrate how to reconstruct rated items among fake and rated items by utilizing only auxiliary information for binary ratings central data-based PPCF systems. There is one more study, which focuses on deriving the original rated items from the perturbed numeric data which is disguised by inconsistent data perturbation (Demirelli Okkalioglu, Koc and Polat, 2016). The authors claim that each user has different privacy concerns, so that each user may want to disguise his/her ratings as well as some unrated cells by employing inconsistent data perturbation approaches. Researchers utilize various domain-related auxiliary information besides existing matrix factorization methods to disclose the list of rated items in numeric ratings central data-based PPCF systems. Matrix factorization methods are widely employed in CF systems for predictions. However, instead of offering predictions, Demirelli Okkalioglu, Koc and Polat (2016) utilize matrix factorization methods to reconstruct items are genuinely rated in the original CF data before applying data disguising method.

There are also some studies that target partitioned data-based PPCF systems. Okkalioglu, Koc and Polat (2015b) first challenge to recover the original binary data on HPD. The authors propose three attack scenarios against partitioned data-based PPCF schemes offered by Polat and Du (2005c; 2008). Perfect match attack introduced by the authors tries to estimate the original data by comparing the similarity of the target query and the users. If the similarity score is 1 or -1, the authors agree that the perfect match is provided. With the perfect match, the original data can be estimated. In another study (Okkalioglu, Koc and Polat, 2016), the authors show how private binary data can be reconstructed when the data is vertically partitioned between two parties. They employ three attack scenarios as in (Okkalioglu, Koc and Polat, 2015b). There is also one more study targeting partitioned-data-based PPCF systems (Demirelli Okkalioglu, Koc and Polat, 2017). The study differs from the previous ones (Okkalioglu, Koc and Polat, 2015b; 2016). The study aims to derive the real numeric data when data is partitioned vertically or horizontally among two parties. Researchers demonstrate that it is not possible to estimate the actual data from the targeted schemes for numeric ratings-based PPCF (Polat and Du, 2005b; Polat, 2006) due to the precautions taken by schemes. However, if additional knowledge about the system is used, it can be likely to derive the private data based on the partitioning type.

1.4. Problem Definitions

PPCF has two key points to preserve privacy: (*i*) hiding the value of the original ratings and (*ii*) masking the rated and unrated items. In order to achieve the first key point, users first convert their original ratings $(x_1, x_2, \dots, x_m)$ into z-score values $(z_1, z_2, \dots, z_m)$. Then each user applies RPT to their confidential data. A random number (r_i) is added to a z-score value (z_i) of users and the value of $z_i + r_i$ is stored by users instead of x_i. The PPCF scheme does not allow the disclosure of the original data (Polat and Du, 2003). However, it is likely to reconstruct actual ratings $(x_1, x_2, \dots, x_m)$ from the perturbed data $(z_1 + r_1, z_2 + r_2, \dots, z_m + r_m)$. Since z-score values are used in the PPCF scheme, the actual ratings should be reconstructed from those values. Hence, the first problem is *how can the random values be eliminated from the disguised data to estimate the real z-score values?* After estimating the real z-sore values, user averages and standard deviation of ratings are needed to be transformed into the actual ratings. Then, the second problem becomes *whether average and standard deviation of each user can be estimated to reconstruct the original ratings because those values are not known in advance*? After the first two problems are tackled, the last issue is to improve the reconstruction predictions. Therefore, the third problem can be stated as: *is it feasible to improve such reconstruction results by applying suitable auxiliary information about the targeted data set?*

Only perturbing the original data may not be enough for some users in order to preserve their privacy. Those users may not want to reveal their rated items too. Users might believe that items they vote are also confidential. Therefore, users desire to perturb the rated items as well as their real ratings. However, someone or an attacker may want to discover which items are voted from the perturbed data which includes real and fake ratings. Suppose that the user u rates p items among m items. Besides, β% of unrated items are selected to be filled with the random numbers. $\big((m - p)x\beta\big)/100$ number of fake items are picked besides the original ratings before those values are sent to the server. The server is not informed what the value of p is or which items are voted. In addition, users may choose own β value in some cases. Then, the problem can be stated as: *Is it likely to find out the location of* p *real items among* $(p+((m-p)x\beta)/100)$*disguised items?*

Data in a CF system might be partitioned (vertically or horizontally) between two parties instead of a single server. In such cases, privacy-preserving schemes are proposed by researchers in order to protect data holders' privacy. However, privacy attack scenarios can be planned against the proposed schemes as well. Whenever an active user asks for a prediction, parties need to interact with each other to produce a prediction. During the interaction, one of the parties may intend to discover the other party's confidential data. The attack of acting as an active user is commonly performed against such schemes. The problem when data is partitioned between two parties can be described as: *Is it possible to disclose the private data by performing such privacy attack? Does the auxiliary information help to estimate the real ratings?* Besides, data may be distributed among multiple parties. In such cases, parties might coalesce against one party to derive its private data. In terms of data distribution type, parties may exchange different aggregate results to provide predictions. The problem when data is distributed among multi parties can be described as: *Is it likely to estimate the targeted party's private data from the interim or final results despite privacy if parties coalesce against the targeted party?*

1.5. Contributions

Privacy has become a fundamental issue in PPCF systems. The data collectors need to offer more secure and straightforward services to their customers day by day to preserve privacy. There are a vast number of studies focused on producing accurate predictions and recommendations in PPCF. As mentioned before, RPTs are commonly used to protect individuals' privacy in PPCF. However, the recent study (Zhang, Ford and Makedon, 2006a) shows that the original data can be derived from the perturbed data despite privacy. Malicious people or attackers may try to estimate the original data and want to use the private information for own advantages. There are a few studies for reconstructing the actual data from the perturbed data. However, some of the unrated cells of original data might be filled with random numbers. Besides, data might be distributed among two or more companies. In the dissertation, several approaches with the help of auxiliary information are proposed to reconstruct the private data from such mentioned cases.

The first contribution of the dissertation is to estimate the private data when the original data is disguised by RPT. There is one study focused on reconstructing the original data in central numeric data-based PPCF (Zhang, Ford and Makedon, 2006a). However, the results can be improved by helping auxiliary information in addition to the existing reconstruction methods. Everyone can easily obtain some public information; for example; the data range of the dataset is usually known by everyone even if RPT is used. In addition, the information of popular and unpopular items can be retrieved from prestigious websites. These websites have millions of users and provide information about items, such as popular or unpopular, depending on the type of datasets such as movies, books or music. Besides, most of CF systems contain user demographic information. Domain related auxiliary information together with user demographic information is used to contribute the reconstruction results better. The other important point is to estimate the original data from the disguised z-score values. For this reason, a method to derive the original data from the estimated z-score values without knowing the users' average rating and the standard deviation is proposed. Several experiments are performed to examine how much contribution achieved over the existing reconstruction methods using the auxiliary information.

In addition to hiding the original data, sometimes data owners do not want to reveal which items they vote. Therefore, another contribution of the dissertation is to find the real rated items when the rated and unrated items are disguised by users independently and to analyze that how the reconstruction results are changed. The study is the first to reconstruct the rated items in central numeric data-based PPCF systems (Demirelli Okkalioglu, Koc and Polat, 2016). Before marking the rated items, it is required to estimate the number of the real ratings from the disguised data masked inconsistently. Two formulas are introduced to guess the number of the rated items based on the method to fill unrated items. The existing matrix factorization methods are used to decrease the effect of the noise. Matrix factorization methods are used for the reconstruction rather than prediction in the dissertation. Also, the joint impact of auxiliary information and the existing matrix factorization methods are studied.

The last part of the dissertation is related to types of data distribution. PPCF algorithms can be divided into two main parts based on data distribution: central server-based systems and decentralized-based systems. Deriving private data might be different in terms of data distribution. The partitioned-based systems are divided into two parts, which are vertical and horizontal partitioning among two-parties. The work is the first one to derive numeric data in VPD- and HPD-based PPCF systems (Demirelli Okkalioglu, Koc and Polat, 2017). There might be different attack scenarios based on targeted schemes. In the dissertation, the attack of acting as an active user is studied to derive the numeric data. There is no way to attack the targeted schemes (Polat and Du, 2005b; Polat, 2006). The targeted schemes containing randomization, encryption, and permutation take precautions against such attacks. Therefore, it is assumed that additional knowledge about the system is known. The experiments show that it is likely to derive the private data in some circumstances when the auxiliary information is utilized. Data might be also distributed vertically or horizontally among multiple parties. The targeted schemes in the dissertation are analyzed mathematically to show whether the confidential data is estimated or not. Since aggregate values are used in the prediction process as in the partitioned-based system, analysis show it is not possible to derive data holders' private data even if acting as an active user attack is employed.

1.6. Organization of the Dissertation

The rest of the dissertation is structured as follows: In Chapter 2, the detailed information about RPT and inconsistent data disguising scenarios are represented as preliminaries. Also, general information about prediction process regarding the targeted PPCF schemes is demonstrated briefly. In Chapter 3, estimating the original data from the disguised one in central data-based PPCF is represented. In Chapter 4, when the inconsistent data disguising method is chosen to protect individuals' privacy in central data-based PPCF, how the real rated items from the disguised data can be reconstructed is studied. In Chapter 5, while data is vertically or horizontally distributed among two or more parties, how much accurate information can be derived is exploited. Finally, conclusions and future works are discussed in Chapter 6.

2. PRELIMINARIES

In this chapter, preliminaries on numeric-data based PPCF systems are represented. There are two fundamental concepts that a PPCF system deals with. The first one is how the original data is disguised without disclosing private data while the second concept is how predictions are produced accurately. First, the data disguising methods utilizing randomization are introduced, and the parameters of such methods are explained according to individual privacy concerns. Then, the prediction process provided by the related PPCF systems is demonstrated. These systems employ a neighborhood-based CF algorithm in order to provide predictions. The prediction generation is divided into two parts with regard to how data is stored, either central or distributed between two or more parties.

2.1. Data Disguising by Randomization

A CF system needs adequate and real data to produce accurate predictions. However, users may refrain from sharing their original data because of privacy concerns. To alleviate users' concerns as well as producing accurate predictions, privacy-preserving schemes are proposed (Polat and Du, 2003; 2007). Randomization techniques often utilize in the privacy-preserving schemes. Such randomization techniques allow users to hide the value of original ratings and ensure that the server cannot learn the real values from the users' disguised data while producing predictions. Besides hiding the real values, it is crucial to conceal rated and unrated items of users because some users may think that the information which items are rated is also a privacy hole. The proposed privacy-preserving schemes which are explained in the next sub-sections relieve all concerns based on user demands.

2.1.1. Randomized perturbation techniques

Randomization is one of the most widely used data disguising methods, which is first introduced by Agrawal and Srikant (2000). According to the randomization, a random number is added to an original data value. The random value is generated from either Gaussian or a uniform distribution. The basic idea behind the randomization is to hide sensitive information so that the server cannot learn what actual values are. On the other hand, the perturbed data should preserve properties of the original data. When the perturbed data is big enough, meaningful estimations based on the aggregate information can be inferred.

Polat and Du (2003) first introduce to utilize the RPT to develop a data disguising method in PPCF so that users do not need to send the actual data to the server. In the first proposed RPT, only the value of rated items is perturbed to protect individuals' privacy. Then, different scenarios for data perturbation based on users' privacy concerns are proposed (Polat and Du, 2007). The authors describe two main scenarios: *(i)* all users either use the same parameters values (distribution type, and σ) or *(ii)* select different parameters values in terms of privacy expectations. The basic and first scenario in data disguising method to generate random numbers is that all users use the same parameter values. The perturbation level of each user is the same in the proposed RPT. The first proposed scheme can be summarized as follows:

i. The server first decides the value of standard deviation (σ) and the type of random number distribution (uniform or Gaussian), and let each user know them.

ii. Users compute their mean and σ values and then calculate their z-score values for each rated item.

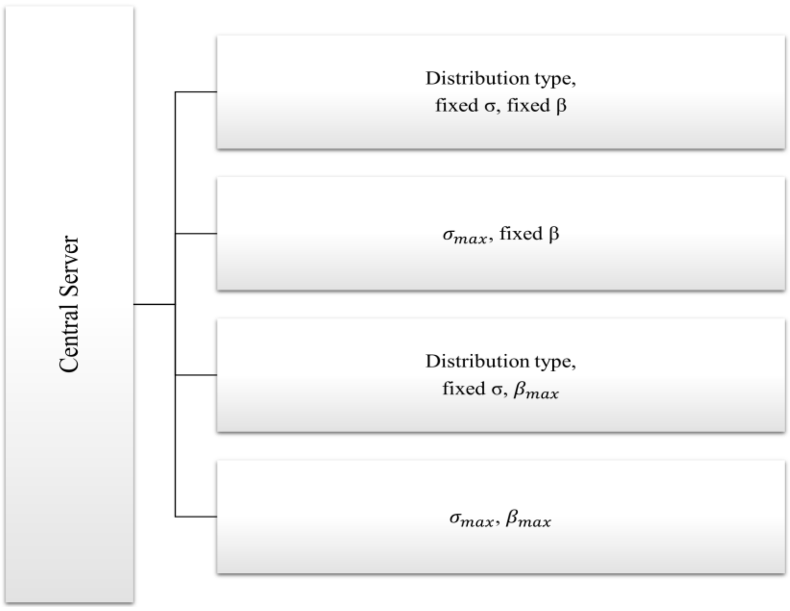
iii. Users then generate random data based on the selected σ value and type of the distribution given by the server. The number of the random data should be equal to the number of items the user has voted.

iv. Before sending z-score values to the server, each user adds the generated random numbers to his z-score values to produce disguised z-score ones. Then, users send their disguised z-score values instead of the original ratings to the server.

It would be wrong to expect all users have the same privacy concerns. Instead of using the same parameters to perturb the original ratings, each user might desire to select the value of own σ as mentioned before. In order to meet user demands, the server only decides σ_{max} and sends the predefined σ_{max} value to the users. Besides, each user determines distribution type by coin tosses. The rest of steps of the first proposed scheme except the first one (i) are applied by the users to disguise their original ratings in this case.

2.1.2. Random filling

The rated and unrated items are also crucial besides masking original ratings in PPCF. Polat and Du (2007) introduce several inconsistent data disguising scenarios by utilizing randomization based on privacy expectations of users. Individuals can fill some their unrated items with the random numbers independently as they wish based on the proposed scheme. To preserve privacy, the server determines a parameter called β which depicts the number of unrated cells that should be filled with the random numbers in order to alleviate users' concerns. There is a similar case as in selecting σ, either the server selects the fixed β and each user employs the selected β value to mask their unrated items, or the server determines β_{max} and each user uniformly randomly selects β_u, $0 < \beta_u \leq \beta_{max}$. Then, each user uniformly randomly picks *the fixed* β *or* β_u percent of unrated items from his rating vector. According to selected σ, each user generates random numbers for all rated and selected unrated items. In the last step, each user adds the random numbers to his corresponding z-score values and the picked unrated items. Figure 2.1 summarizes random filling scenarios in terms of selected parameters.

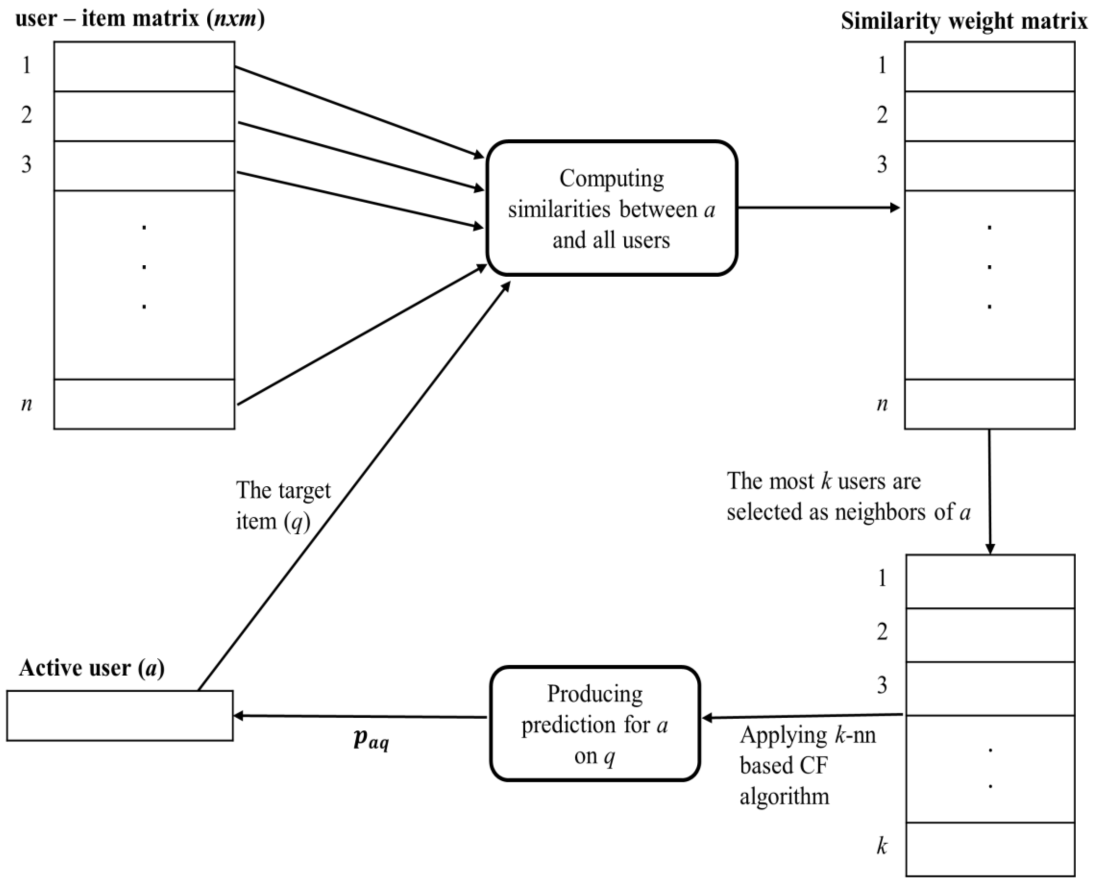


Central Server
Distribution type, fixed σ, fixed β
σ_{max}, fixed β
Distribution type, fixed σ, β_{max}
σ_{max}, β_{max}

Figure 2.1. *Random filling scenarios*

2.2. Providing Predictions in PPCF

There are various CF algorithms providing predictions based on numeric ratings in PPCF. In this section, one of the most widely used CF algorithms related to a neighborhood-based approach is explained briefly. Herlocker et al. (1999) introduce the k-nearest neighbor (*knn*)-based CF algorithm with the z-score notations to produce estimations, which is commonly utilized in the PPCF systems. According to the proposed algorithm, an active user asks for a prediction for a target item (q), referred to as p_{aq}, by sending own rating vector and q. Then, the server selects the most similar k users by calculating similarities between the active user and all users in the data set. The most similar k users form the neighbors of the active user. p_{aq} is estimated from the preferences of the selected neighbors. Figure 2.2 simply illustrates how the *knn*-based CF algorithm works.



user – item matrix (nxm)
1
2
3
n
Similarity weight matrix
1
2
3
n
Computing similarities between a and all users
The target item (q)
The most k users are selected as neighbors of a
1
2
3
k
Active user (a)
Producing prediction for a on q
p_aq
Applying k-nn based CF algorithm

Figure 2.2. *A general view of the knn-based CF algorithm*

2.2.1. Providing predictions in the central server-based PPCF

Due to the privacy concerns, users send their disguised data instead of the original ratings to the server. The data collected in the central server is used to produce predictions. Since the original ratings are converted into z-scores, the central server provides predictions by utilizing the received disguised data. The PCC algorithm is modified by Herlocker et al. (1999) to incorporate an item variance weight factor. Since the z-score values are stored in the PPCF system, the similarity weights between an active user and all users are calculated as in Eq. 2.1.

$$w_{au} = \frac{\sum_{i=1}^{m} z_{ai}\ x\ z_{ui}}{m} \tag{2.1}$$

After calculating similarities between the active user and all users using Eq. 2.1 and selecting the most k similar users for the active user, the server needs to compute the prediction, p_{aq} using the *knn*-based CF algorithm. The algorithm proposed by Herlocker et al. (1999) is rearranged as seen in Eq. 2.2,

$$p_{aq} = \overline{v_a} + \sigma_a\ x\ P\ =\ \overline{v_a} + \sigma_a\ x\ \frac{\sum_{u=1}^{k} w_{au}\ x\ z_{uq}}{\sum_{u=1}^{k} w_{au}} \tag{2.2}$$

where $\overline{v_a}$ refers to a's mean vote and σ_a represents to a's standard deviation and k is the number selected neighbors in the prediction process. z_{uq} is the z-score value of user *u* for item q.

2.2.2. Providing predictions in the partitioned data-based PPCF

Private predictions are proposed (Polat, 2006; Polat and Du, 2005b) by utilizing the algorithm introduced by Herlocker et al. (1999) according to data partitioning. When data is partitioned either vertically or horizontally between two parties, Eq. 2.2 is modified to be utilized for such cases (Polat, 2006; Polat and Du, 2005b). Furthermore, rather than selecting the most k similar users for the active user, all users held by parties join the calculation. In this case, k refers commonly rated items between the active user and the user. When the data is partitioned vertically or horizontally among two parties (*A* and *B*), the required values for the equation are computed separately by parties to provide predictions.

2.2.2.1. Providing predictions on horizontally partitioned data

It is difficult to provide reliable and accurate recommendations when there is a limited number of user vectors in CF systems. Besides, these systems may encounter such situations that cannot even offer recommendations for some items owing to limited users and ratings. Although the merged data is useful for two parties, parties may not volunteer to combine their data due to privacy, legal and financial issues. Polat (2006) proposes a scheme to overcome these issues.

A scheme is first demonstrated for HPD to achieve private predictions while preserving data holders' privacy. In HPD, two parties have disjoint sets of users' ratings for the same items. The proposed scheme consists of two parts: off-line and online computation. Polat (2006) shows how to provide an accurate prediction using the Eq. 2.3, as follows:

$$P = \frac{\overbrace{\sum_k z_{ak}\left[\sum_{i=1}^{n_A} z_{ik} z_{iq}\right]}^{A_N} + \overbrace{\sum_k z_{ak}\left[\sum_{i=1}^{n_B} z_{ik} z_{iq}\right]}^{B_N}}{\underbrace{\sum_k z_{ak}\left[\sum_{i=1}^{n_A} z_{ik}\right]}_{A_D} + \underbrace{\sum_k z_{ak}\left[\sum_{i=1}^{n_B} z_{ik}\right]}_{B_D}} = \frac{A_N + B_N}{A_D + B_D} \tag{2.3}$$

where $n = n_A + n_B$, n_A refers to the number of users of Party *A* while n_B depicts the number of users of Party *B* and *n* represents total number of users after merging data of two parties. *k* refers common item sets both user *i* and *a* have voted.

Off-line Computation: Each party can compute the denominator and the numerator part separately as seen from Eq. 2.3. Since the data is partitioned horizontally, parties do not need extra data to calculate their parts. For this reason, the values of denominator and numerator are calculated independently using the data held by parties. Polat (2006) proposes that each party should compute their parts and store them into matrices to reduce the online computation time. Since processes that two parties have done are the same, the steps that Party *A* performs can be explained as follows (Polat, 2006):

i. Party *A* first converts his ratings into z-score values.

ii. For each *j*, *j = 1, 2, ..., m*, party *A* calculates $\sum_{i=1}^{n_A} z_{ij}$. For denominator part, the results for each item are stored in a matrix named AD, $\sum AD = [\sum AD_1, \sum AD_2, \dots, \sum AD_m]$.

iii. Then, Party *A* calculates the required data for numerator part. For each *q*, *q = 1, 2, ..., m*, $\sum_{j=1}^{m} \sum_{i=1}^{n_A} z_{ij} z_{iq}$ are calculated and the results for each target item are stored in a matrix named AN, $\sum AN = [\sum AN_1, \sum AN_2, \dots, \sum AN_m]$.

Party *B* also performs same steps and computes the denominator and the numerator parts in Eq. 2.3. Then Party *B* stores them into $\sum BD$ and $\sum BN$ matrices.

Online Computation: One party is selected as a master site during the online computation. Assume the Party A is the master site. The algorithm steps can be summarized as follows (Polat, 2006):

i. An active user sends his data and requests a prediction for a target item (q) to both parties.

ii. Since Party A is the master site, Party B calculates B'_D and B'_N using the *private scalar product computation protocol* and then sends these results to Party A.

iii. After getting the results from Party B, Party A calculates own A_N and $A_{D.}$ Then, Party A produces the value of P'.

iv. In the final step, Party A computes the prediction and compares the predefined threshold value and tells whether the active user likes q or not.

Private Scalar Product Computation Protocol: The party (B) which is not the master site wants to protect own denominator and numerator values from the master site (Party A). Polat (2006) introduces a protocol to achieve secure online computation. The proposed protocol is summarized as follows:

i. The total number of rated items of the active user is calculated (C_B).

ii. If C_B is less than $\lfloor m_B/2 \rfloor$ (m_B refers to the total number of items of B), Party B finds the items that the active user does not rate and calculates $(m_B - C_B)$. Party B defines a random integer (S_{Ba}) from the range $(1,\ m_B - C_B)$. Then, Party B randomly selects (S_{Ba}) unrated items among the items it owns and fills those selected items in the active user's rating vector with the corresponding item's mean.

iii. If C_B is bigger than $\lfloor m_B/2 \rfloor$, in that case, Party B defines a random integer (S_{Br}) from the range $(1,\ C_B)$. Then, Party B randomly selects (S_{Br}) rated items among the items in the active user's rating vector and removes them.

While Party A does not know the value of S_{Ba} and S_{Br} as well as which items are added or removed, Party A does not estimate B_N and B_D values from B'_D and B'_N even if Party A acts as an active user. The *Private Scalar Product Computation Protocol (PSPCP)* achieves privacy while producing predictions.

Figure 2.3 depicts how a prediction is produced on HPD. While an active user sends his data and q to both parties, the final prediction is calculated by the master site which is Party *A* in this example.

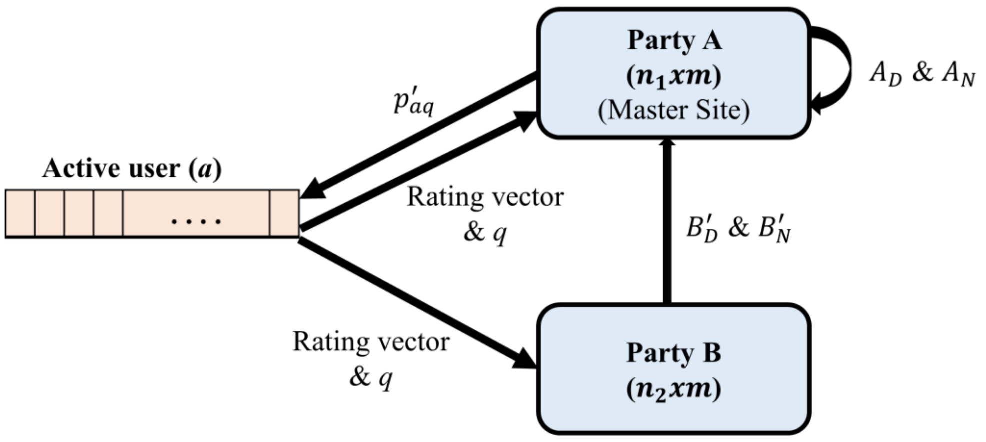


Figure 2.3. *The horizontally partitioned data-based PPCF*

2.2.2.2. Providing prediction on vertically partitioned data

The number of electronic websites has been increasing day by day, and users prefer different sites for online shopping. This situation causes vertically partitioned data (VPD) between parties. Two parties can offer better recommendations for their users when they combine vertically split data without revealing their confidentiality. Polat and Du (2005b) propose a scheme for VPD which consists of off-line and online computation as in HPD. Unlike the HPD, two parties have disjoint sets of item ratings for the same users in the VPD. In such a case, Eq. 2.4 is utilized to provide predictions (Polat and Du, 2005).

$$P = \frac{\overbrace{\sum_{k_A} z_{ak_A}\left[\sum_{i=1}^{n} z_{ik_A} z_{iq}\right]}^{A_N} + \overbrace{\sum_{k_B} z_{ak_B}\left[\sum_{i=1}^{n} z_{ik_B} z_{iq}\right]}^{B_N}}{\underbrace{\sum_{k_A} z_{ak_A}\left[\sum_{i=1}^{n} z_{ik_A}\right]}_{A_D} + \underbrace{\sum_{k_B} z_{ak_B}\left[\sum_{i=1}^{n} z_{ik_B}\right]}_{B_D}} = \frac{A_N + B_N}{A_D + B_D} \tag{2.4}$$

In Eq. 2.4, k_A depicts the common items both user *i* and *a* voted among items of *A* while k_B refers to the common items both user *i* and *a* rated among items of *B*. $k = k_A + k_B$, *k* illustrates the total number of items of two parties and *n* is the total number of users in both parties.

Off-line Computation: The denominator parts (A_D and B_D) can be easily computed as seen from Eq. 2.4. The party who does not have q needs to have z_{iq} to calculate $\sum_{i=1}^{n} z_{ik_j} z_{iq}$. Before the off-line computation step, each party horizontally divides its data, and then applies the algorithm independently for each part. The algorithm steps for Party *A* are summarized, as follows (Polat and Du, 2005b):

i. All columns of each horizontally divided part are permuted using a permutation function $\prod A_i$.

ii. For *j = 1, 2, …, m_A*, where m_A is the number of items of Party *A*, the permuted column vector is divided into a random number of vectors so that the sum of the value of these random vectors are equal itself.

iii. A new permutation function again permutes all permuted random vectors.

iv. Party *A* sends all permuted random vectors to the other party (Party *B*). Party *B* computes the scalar products between these permuted random vectors and its column vectors and encrypts the scalar product results using a HE. The encrypted values are sent back to Party *A*.

v. As Party *A* knows own perturbation functions and the used HE, Party *A* finds encrypted scalar results and merges all horizontal parts using the HE property to obtain the final encrypted scalar results

vi. Then, Party *A* adds random numbers into the final encrypted scalar product results to protect its data from Party *B* and stores them into $\sum{}'A$.

vii. Finally, the matrix $\sum{}'A$ is sent to Party *B* which decrypts the encrypted scalar product results and stores them into $\sum{}''A$.

Online Computation: Since both parties can act as an active user and try to derive the data of the other party, the online calculation is done, as follows:

i. An active user sends his rating vector and requests a prediction for a *q* to the party owning the *q*.

ii. Suppose that Party *A* has q. When Party *A* calculates $B'_N + A'_N$ and A'_D , Party *A* employs the proposed protocol (PSPCP explained in section 2.2.2.1) which helps Party *A* protect himself from Party *B*. Also, the active user's new mean vote, standard deviation is calculated by Party *A* and all results are sent to the active user. The active user forwards them to Party *B*.

iii. Party *A* can compute the value of B'_N using the data of the *q-th* row of matrix $\sum{}''B$ with the active user's data. The value of B'_N is formed as $B_N + R_q$. R_q is a random number and is added to B_N while calculating $\sum{}'B$.

iv. Party B subtracts the predefined random number (R_q) from the received result $(B'_N + A'_N)$ and calculates the final value and tells the active user whether he will like q or not.

Figure 2.4 illustrates an example of prediction process when an active user asks for a prediction for item q on VPD-based PPCF. As seen from the Figure 2.4, the final prediction calculation is performed by Party B that does not own item q.

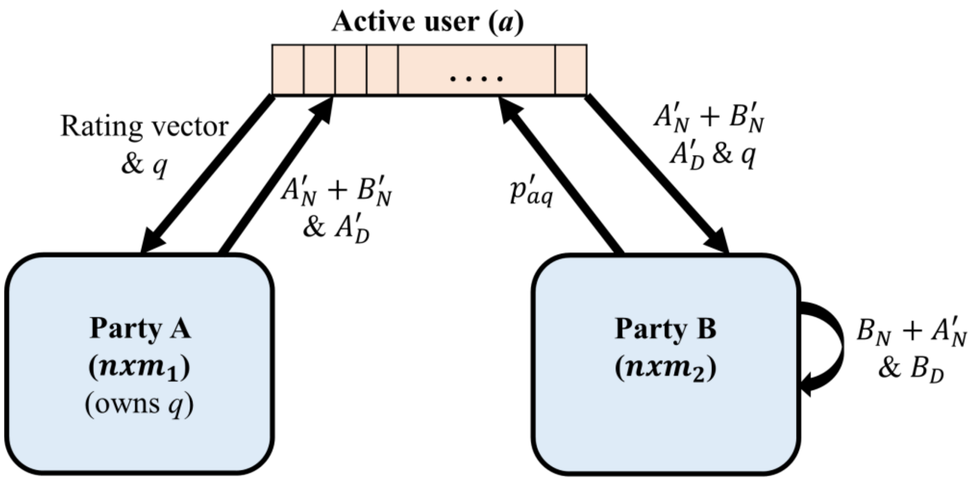


Figure 2.4. *The vertically partitioned data-based PPCF*

2.2.3. Providing clustering-based predictions in the distributed data-based PPCF

Data may be split between more than two companies in CF systems. Since users or items are divided between companies, it is important to find similar users and to generate a prediction using those users' data when a new prediction is requested. Because determining similar users (k-nearest neighbors) is time-consuming in distributed environments, Kaleli and Polat (2012a; 2012b) propose to cluster users held by parties before producing predictions. Clustering methods are widely utilized to increase the performance of CF schemes. SOM clustering is one of the clustering methods used in CF systems and is employed by Kaleli and Polat (2012a; 2012b) for the selection of the similar users before the *knn*-based CF algorithm is implemented. A detailed information of SOM applied to CF is demonstrated by Roh, Oh and Han (2003).

2.2.3.1. Providing clustering-based predictions on horizontally distributed data

Kaleli and Polat (2012b) introduce a privacy-preserving clustering-based scheme for HDD to preserve data holders' privacy. Their scheme consists of two steps as in the partitioned data: off-line and online. After users are clustered among parties off-line, *knn*-based CF algorithm is applied online to produce predictions. Eq. 2.5 illustrates how the prediction is provided when data are horizontally distributed among *z* parties.

$$\mathrm{P} = \frac{\sum_{j=1}^{J} z_{aj}\left[\sum_{u=1}^{k_1} z_{uj}v_{duq} + \sum_{u=1}^{k_2} z_{uj}v_{duq} + \cdots + \sum_{u=1}^{k_z} z_{uj}v_{duq}\right]}{\sum_{j=1}^{J} z_{aj}\left[\sum_{u=1}^{k_1} z_{uj} + \sum_{u=1}^{k_2} z_{uj} + \cdots + \sum_{u=1}^{k_z} z_{uj}\right]} \qquad (2.5)$$

In Eq. 2.5, $k_1, k_2, \ldots, k_z$ represents the number of similar users who rated q owned by the first, the second, …, zth party, respectively. *J* is the commonly rated items between an active user and user *u*. v_{duq} is the deviation from mean ratings of q and can be calculated, as follows*:* $v_{duq} = v_{uq} - \overline{v_u}$, where v_{uq} is the rating of *user u* for item q.

Off-line: Kaleli and Polat (2012b) first propose the *Private Distributed SOM Clustering Protocol (PDSOM)* to cluster users distributed between among *z* parties horizontally while preserving parties' privacy. This process aims to gather similar users into a cluster, so these users in a specific cluster can be used as nearest neighbors in the prediction process. The authors claim that defining similar users off-line improves the performance of the prediction process. The details of PDSOM is given in Kaleli and Polat (2012b). Since the online computation time is critical, the parties calculate z_{uj} and v_{duq} values and store them off-line.

Online: Kaleli and Polat (2012b) introduce *Private Distributed k-nn CF Protocol (PDKNN)* to provide predictions while preserving the parties' privacy. According to the protocol, one of the parties is selected as Master Party (MP). An active user sends his data and the q to the MP. The steps of the proposed protocol are explained, as follows:

i. After receiving the active user's rating vector and q, the MP calculates the active user's cluster by computing distance between the active user and each cluster center. Then, the closest cluster is selected as the active user's cluster. The MP lets each party know the selected cluster and the target item. After that, users in that cluster are utilized to produce a prediction.

ii. The MP and each party calculate $\sum_{u=1}^{k} z_{uj}$ and $\sum_{u=1}^{k} z_{uj}v_{duq}$ values for all items except the target item based on users' data in the selected cluster. Each party sends its aggregate values to the MP.

iii. The MP calculates the final prediction using Eq. 2.5 and sends the result to the active user.

Kaleli and Polat (2012b) have mentioned two drawbacks based on PDKNN. The first one is that if a new active user is in the same cluster with the previous ones and asks for a prediction for the same target item, the MP computes the final prediction results using the existing partial aggregate values without asking to compute the required data from the other parties. The second drawback is that the MP may collect the required aggregate values from parties using the fake active users for its interests, and then may use those values to derive the private data about parties. In order to overcome these weaknesses, a new protocol called *Improved Private Distributed k-nn CF Protocol (IPDKNN)* is proposed by Kaleli and Polat (2012b). IPDKNN is used to calculate the required aggregate values instead of the second step (*ii*) of PDKNN. The steps of IPDKNN are summarized, as follows:

i. Each party selects uniformly randomly a random number (*R*) over the predefined range (0, γ]. After that, each party selects uniformly randomly *R%* of users who did not vote *q*.

ii. Parties fill the selected unrated cells for *q* with non-personalized ratings.

iii. Then, each party selects uniformly randomly some of the rated cells and removes their values before computing the required aggregate data.

iv. Finally, each party sends its disguised aggregate values to the MP.

IPDKNN guarantees that whenever an active user asks for a prediction, all parties must participate in the prediction process. Besides, the MP will not be able to store the partial results to offer recommendations independently in the future because of randomness that the parties add.

2.2.3.2. Providing clustering-based predictions on vertically distributed data

Kaleli and Polat (2012a) also propose a privacy-preserving scheme for VDD that utilizes clustering and the nearest neighbor prediction algorithm as in HDD. Their schemes consist of two parts as off-line and online calculations. Off-line computation includes different protocols to protect data holders' privacy when the parties collaborate with each other. The parties need to exchange their data while producing a prediction because they do not own all data required to provide predictions.

Off-line: The first prominent step in off-line computation is the clustering process. Hence, the first protocol which is called *private SOM clustering on multi-party vertically* *distributed data (SOMP)* is employed to cluster users held by parties off-line. Then, the parties calculate item, user averages, and vector lengths. As each party has all the values of the items it holds, the item averages can be performed independently by parties. However, the parties need the data of each other when they want to calculate user averages. While exchanging the required data, the parties can derive the private data. A new protocol which is called *data perturbation protocol (DPP)* is proposed to prevent the parties from estimating the confidential data (Kaleli and Polat, 2012a). According to DPP, each party first uniformly randomly selects a random value (ϴ) over a predefined range, and then uniformly randomly chooses a random value (β) from the selected range (0, ϴ]. β % of the unrated cells are picked randomly, and the picked cells are filled with personalized ratings. The parties can compute their personalized ratings without the help of the other parties.

After parties disguise their data, they can compute user averages off-line by employing *user mean ratings protocol (UMRP)*, as given in Eq. 2.6,

$$\overline{v_u} = \frac{\sum_{j \in J_1} v_{uj} + \sum_{j \in J_2} v_{uj} + \cdots + \sum_{j \in J_z} v_{uj}}{|J_1| + |J_2| + \cdots + |J_z|} \tag{2.6}$$

where $J_1, J_2, \ldots, J_z$ are the number of ratings of user *u* held by the first, the second, …, the zth party. As seen from Eq. 2.6, each party computes $\sum_{j \in J_i} v_{uj}$ and $|J_i|$ values based on its disguised data. Then, each party exchanges them with other parties to compute the overall user mean.

According to the adjusted cosine measure used for computing similarities, vector lengths of users are needed. The parties can compute vector lengths off-line using *vector length protocol (VLP)* (Kaleli and Polat, 2012a). Vector lengths of user *u*, $(VL\|X_u\|)$ can be calculated with Eq. 2.7 when the data is distributed vertically among *z* parties.

$$\|X_u\| = \sqrt{\sum_{j \in J_1} \left(v_{uj} - \overline{v_j}\right)^2 + \sum_{j \in J_2} \left(v_{uj} - \overline{v_j}\right)^2 + \cdots + \sum_{j \in J_z} \left(v_{uj} - \overline{v_j}\right)^2} \tag{2.7}$$

Since the parties are able to compute their item averages, they can quickly normalize their ratings by subtracting them. Then, each party calculates $\sum_{j \in J_z} \left(v_{uj} - \overline{v_j}\right)^2$ values and exchanges them with other parties. For all *n* users, the steps of *VLP* are repeated.

After demonstrating how to calculate item and user averages and vector lengths, the next step is to show how to produce a prediction in VDD. Kaleli and Polat (2012a) describe a new equation to provide a prediction when data is vertically distributed among *z* parties, as given Eq.2.8,

$$\mathrm{P} = \frac{\sum_{j\in J_1} v''_{aj} \overbrace{\left[\sum_{u\in S} v''_{uj} v_{duq}\right]}^{P_{N_1}} + \sum_{j\in J_2} v''_{aj} \overbrace{\left[\sum_{u\in S} v''_{uj} v_{duq}\right]}^{P_{N_2}} + \cdots + \sum_{j\in J_z} v''_{aj} \overbrace{\left[\sum_{u\in S} v''_{uj} v_{duq}\right]}^{P_{N_z}}}{\sum_{j\in J_1} v''_{aj} \underbrace{\left[\sum_{u\in S} v''_{uj}\right]}_{P_{D_1}} + \sum_{j\in J_2} v''_{aj} \underbrace{\left[\sum_{u\in S} v''_{uj}\right]}_{P_{D_2}} + \cdots + \sum_{j\in J_z} v''_{aj} \underbrace{\left[\sum_{u\in S} v''_{uj}\right]}_{P_{D_z}}} \qquad (2.8)$$

where v''_{uj} and v''_{aj} depict the normalized ratings of user *u* and *a* which can be obtained by first subtracting the item averages, and then dividing the result of user's vector lengths. v_{duq} is a deviation from mean ratings of *q* as in Eq. 2.5, and *S* is the users who voted *q* in the selected cluster.

In order to decrease online computation time, parties can compute P_N and P_D values for all target items off-line. Parties need to exchange their partial results to compute P_N values without violating their privacy. Therefore, Kaleli and Polat (2012a) propose a new protocol called P_N *protocol* ($P_N P$). Their protocol ensures that parties compute their P_N values in a private way. Since the items are divided vertically among the parties, each party should be selected as a MP in turn to compute P_N values for all target items belonged to itself. The steps of $P_N P$ are repeated for each selected MP. Note that the party with item *q* is chosen as the MP. The steps of $P_N P$ are summarized, as follows:

i. For each target item, *q* = 1, 2, 3, …, m_i, where m_i is the number of item sets owned by the MP, the MP first encrypts each value of item *q* with a HE, which contains real and fake ratings because of DPP. Note that the public key is only known by the MP.

ii. The MP sends those encrypted values to the other parties.

iii. Then, for all items *j* = 1, 2, 3, m_i held by each party, the encrypted values are multiplied by the values of *j*th item using the property of HE, respectively.

iv. Since only the MP has the public key, it can derive the confidential data of the parties by decrypting the received values. Permutation functions are employed by parties to protect their private data. Each party first permutes the computed values of each item using a row permutation function.

v. Then, all items' vectors are permutated by a column permutation function. The permutated results are sent to the MP.

vi. After the MP decrypts the received permuted results, it finds column sums ($\sum_{u \in S} v''_{uj} v_{duq}$). The aggregate data is sent back to the corresponding party.

vii. Since each party knows its column permutation function, it can easily order the received aggregate values (P_N).

Parties also need to compute and store P_D values off-line. Since item averages and user vector lengths are computed previously, the parties easily calculate the normalized ratings (v''_{uj}) for each q independently. To compute $\sum_{u \in S} v''_{uj}$, parties have to know which users rated q (denoted S) in advance because they have to use those users' data in the prediction process. With the help of $P_N P$, each party derives the location of users from the encrypted values. After identifying which users are joined to the prediction process, their data is used to compute P_D values for each item.

All the protocols mentioned so far are to create a model off-line. Parties can produce a prediction using the constructed model anymore according to the active user requests.

Online: After an active user sends his rating vector and q to the MP, it first determines the active user's cluster and assigns the active user to the closest cluster as in HDD. Kaleli and Polat (2012a) introduce *private recommendation protocol (PRP)* to perform the prediction generation online. The steps of *PRP* is explained, as follows:

i. The MP disguises the active user's rating vector using DPP. The purpose here is to prevent other parties from deriving the data of the MP by acting as an active user.

ii. Then the MP calculates the normalized ratings (v''_{aj}). Using the HE, each value of the normalized ratings of the active user is encrypted ($E_{KPC}(v''_{aj})$) before the corresponding part of encrypted items' values are sent to the corresponding party.

iii. Each party also fills some unrated cells of the received normalized data to protect itself from the MP.

iv. Then, each party computes encrypted aggregate results of $E_{KPC}(v''_{aj})*P_D$ and $E_{KPC}(v''_{aj})*P_N$ using the constructed model off-line for all rated items in the active user's rating vector including fake ones.

v. Each party permutes all encrypted aggregate results with a new permutation function for numerator and denominator separately. Then, all permuted encrypted aggregate results are sent back to the MP.

vi. The MP decrypts all permuted encrypted aggregate results received from other parties. It computes the final prediction using Eq. 2.8 and sends it to the active user.

3. DERIVING THE ORIGINAL DATA PERTURBED BY RANDOMIZATION IN CENTRAL DATA-BASED PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEM

In this chapter, two data reconstruction methods are applied to derive the original data from the perturbed one. Then, auxiliary information such as popular items, unpopular items, rating range, and users' demographic information is utilized for the improvement of the reconstruction results. Besides, a new method is proposed to derive the original ratings from the estimated z-score values by utilizing only the rating range. The proposed approaches are evaluated by performing several experiments with a real data set. Empirical outcomes show that the proposed approaches and auxiliary information with the existing reconstruction methods help attackers derive a meaningful amount of the original data.

3.1. Introduction

Privacy-preserving collaborative filtering schemes have emerged to preserve individuals' privacy. For this reason, there are different methods proposed to overcome the privacy problem such as anonymization (Pfitzmann and Hansen, 2010; Yang and Qiao, 2010), cryptography (Canny 2002a; Li et al., 2016), RPT (Polat and Du, 2003; 2005a; Polatidis et. al, 2017) and so on. RPT is a conventional method to preserve privacy in PPCF schemes. Although the idea behind RPT is simple, it is a robust method to hide the private information from the third party that has no right to access the confidential data. There are several works focusing on using the randomization approach as a data disguising method in PPCF methods (Polat and Du, 2003; 2005a). Even though randomization is a widespread technique to preserve the confidential data, the existing studies show that the original data can be reconstructed from the disguised one (Kargupta et al., 2005; Huang, Du and Chen, 2005; Zhang, Ford and Makedon, 2006a).

In this chapter, the methods proposed by Zhang, Ford and Makedon (2006a) are investigated to improve the reconstruction of private data. There might be some public information depending on the data set such as the rating range, popular or unpopular items, prestigious awards, and demographic information of users, which help attackers or non-expert adversaries improve the reconstruction results. Polat and Du (2003; 2005a) operate RPT on z-scores instead of original ratings. Therefore, original z-scores must be reconstructed first. Deriving the original ratings from the reconstructed z-score values is also a significant issue. It is difficult to estimate the original ratings from the disguised z-scores values without knowing the average and the standard deviation of ratings for each user. Hence, a new method is proposed to derive the actual values from the estimated z-scores using rating range of the data set, which can be considered as auxiliary information as well, without knowing the values of such parameters. Finally, how much contribution over the existing methods using the proposed approaches can be achieved is analyzed by conducting several experiments.

3.2. Data Reconstruction Methods

In this section, two existing data reconstruction methods, *k*-means clustering-based reconstruction scheme and SVD-based reconstruction with EM procedure, are explained briefly. These two methods have been used to derive the original data from the disguised z-score data in (Zhang, Ford and Makedon, 2006a).

3.2.1. *k*-means clustering-based reconstruction

k-means clustering is one of the most widely used clustering algorithms. The algorithm is mostly utilized for predictions in PPCF systems. However, the aim of the *k*-means clustering-based reconstruction method in the dissertation is to extract the original ratings from the perturbed ratings. In the discrete-valued rating scenario, there are *k* unique ratings in the recommendation system. Thus, the number of the unique ratings is selected as a cluster number. One crucial point in the *k*-means clustering algorithm is to determine the initial cluster centroids. Selecting different initial cluster centroids each time causes different results of the *k*-means clustering algorithm. While the initial cluster centroids can be identified randomly, an algorithm can be employed to pick them. Therefore, it is essential to determine how the initial cluster centroids should be selected. In the dissertation, the initial centroid positions are picked by applying an algorithm instead of choosing them randomly. The steps of the selected algorithm can be explained, as follows. After sorting all disguised z-score values for each user, the mean of the lowest *x%* of all disguised z-score values is assigned as the first cluster centroids, and the mean of the highest *x%* of all disguised z-score values is assigned as the *kth* cluster centroids. Then, *k-2* equidistant centroids to each other are selected as the rest of centroids. After applying a *k*-means clustering algorithm to the disguised z-score values, the *jth* cluster is assigned as the *jth* original value. Pseudocode for selecting the initial cluster centroids is listed as Algorithm 1.

Algorithm 1: Identifying the initial centroid positions

```
Input:
    $D = \{d1, d2, \dots, dm\}$ set of $m$ items

    $k$ // number of clusters
Output:
    A set of $k$ initial centroid positions
Steps:
    1. Sort all disguised z-score values of the user
    2. The mean of the lowest x% of all disguised z-score values is assigned as the first cluster centroid
    3. The mean of the highest x% of all disguised z-score values is assigned as the $k^{th}$ cluster
    4. Then, k-2 equidistant centroids to each other are defined the rest of centroids
```

3.2.2. SVD-based reconstruction with expectation-maximization

SVD-based reconstruction is also applied to derive the original z-score values from the disguised z-score values. Zhang, Ford and Makedon (2006a) indicate that one of the dominant methods for describing the rating matrix is a low-dimensional linear model. Hence, the sum of a low-dimensional linear model (denoted as *X*) and a Gaussian distributed noise matrix (*Z*) constitute a normalized matrix (Y_{norm}). R is generated using a Gaussian distribution. The disguised data matrix can be represented as:

$$Y = Y_{norm} + R = X + Z + R \tag{3.1}$$

The log-likelihood of the z-scores of the perturbed data given *X*, *logPr(Y|X)*, helps discover *X*, because *X* maximizes the log-likelihood, *logPr(Y|X)*. The solution to finding out *X*, which maximizes *logPr(Y|X)*, is a low-rank approximation. If the rank of the linear model assumes *r*, then the top *r* right singular vectors of *Y* can be produced by applying SVD.

In the real-world recommendation systems, most of the rating matrices are usually sparse. Users usually vote a small number of items as compared with all items in the data set. On the other hand, SVD-based reconstruction needs complete data to be applied. Unrated entries are usually filled by user averages or item averages to cope with the sparse matrix and to find out the linear model, which fits the filled-in rating matrix. Nevertheless, it is not a right way to employ user averages or item averages as a value for unrated entries.

A better way to find out X, which maximizes the log-likelihood of the disguised z-scores or the original z-scores of the rated entries using, is to utilize an EM procedure (Srebro and Jaakkola, 2003; Zhang et al., 2005). In the EM procedure, the first step is the expectation step that fills zero for each unrated entry in the first iteration. When the iteration number is greater than or equal to two, the expectation step calculates each unrated entry in the disguised z-scores matrix by replacing with $X_{ij}^{(t-1)}$ to construct a filled-in rating matrix. In the maximization step, SVD is applied to the updated model $X^{(t)}$ to find a new low-dimensional linear model X.

3.3. Utilizing Auxiliary Information to Improve Data Reconstruction Methods

Auxiliary information in PPCF systems might help the reconstruction schemes to improve the outcomes. Recommender systems can be applied in different domains such as movies, books, music, news, restaurants, e-commerce, e-learning, and tourism recommendations. Auxiliary data, which is usually public, can be obtained depending on the data set. For example; if the data set is related to books, Amazon or Barnes & Noble e-commerce sites help anyone derive auxiliary information to determine such as which books are popular or in top 100. The application domain may be related to tourism in another example. The data set might consist of restaurants, hotels, countries, sightseeing tour, historical places, or activities. When the private information from a tourism-related data set is estimated, all related information from restaurants, hotels or activities becomes essential. For instance, most of the people enjoy visiting favorite historical places or eating the regional foods. If an attacker wants to estimate where a person visits or likes, the list of popular places can be utilized, which can be obtained from the websites or even tourist map guides. Assume that most of the people like to visit these places and they likely give higher ratings. For example, if a person goes to Rome where is one of the world’s most visited cities, it is expected that the person goes to and likes The Colosseum, Trevi Fountain, Pantheon and Vatican museums, where all the most famous places are in Rome. Besides, income ranges, which can be one of the users’ demographic information, guides to estimate the visited places or age ranges give someone an opinion about the activities that users like or perform. Several demographic information can be stored in recommender systems and some of them might be utilized to guess user interests. Selecting correct user demographic information in the reconstruction method depends on the used data set. Recommender systems usually contain demographic data in addition to ratings. Even if the recommender system does not have any demographic data, obtaining the demographic information from the users would not be a difficult job for companies. Every user needs to sign in a new system before logging in. During this process, the user usually enters birthdate, sex, income, address, or some trivial information related to himself. The user may think that this information is harmless; however, this kind of information is prominent for companies, and it can be used to derive the private information. Many auxiliary information examples can vary based on the selected data set, and they are usually obtained easily.

Nowadays, with the rapid development of technology, social networks, such as Facebook, Twitter, or YouTube are widespread and popular among people. Most existing studies in the literature use social networks as a source of axillary information (Konstas, Stathopoulos and Jose, 2009; Yuan et al., 2015; Yu et al., 2015; Jiang et al., 2015). Companies or researchers match users among different domains and use their comments or tweets as auxiliary information owing to social networks. As most of the studies in the literature are interested in movie recommendation, a movie-related data set is selected to evaluate the proposed approaches. However, the selected data set does not contain users' social data. Therefore, social networks data cannot be integrated into the experiments. If another type of data set were chosen to show the effect of auxiliary information, the auxiliary information could differ as mentioned in the previous examples according to the selected data set.

Various public information might have a different contribution to the existing reconstruction results. Different public information for *k*-means clustering-based and SVD-based reconstruction methods is utilized to determine how much their effect is. In general, there might be several public information that can be exploited to improve the reconstruction results. In this dissertation where the movie-related data set is used, the auxiliary information is categorized into four main groups, popularity and unpopularity, prestigious award-winners, user demographic information, and prior knowledge about the mean of each item or user.

3.3.1. Popular and unpopular movies

The auxiliary information can be gathered online depending on the data sets. If a targeted data set contains well-known items such as movie, music, or books, discovering auxiliary information about its items will not be difficult for an attacker or a non-expert person. The Internet has been building countless communities with the contribution of people all over the world. There are many prestigious websites with millions of registered users. Such websites collect user preferences based on its topic of interest (movie, music, or books). The reviews and ratings made by a significant number of people is a serious source of auxiliary information. For example, the Internet Movie Database (IMDb, www.imdb.com) is a reference website accepted by movie lovers and authorities. Websites like IMDb reveal many useful public auxiliary information about items in a related targeted data set. The most obvious public information that can be inferred as auxiliary information is popular and unpopular items obtained from such specialized websites. In general, such websites provide ratings or list of popular and unpopular items to inform its users. If one can find a public and online source of trusted information, then he can collect popular and unpopular items and match this auxiliary information with the items of the targeted data sets. Hence, such auxiliary information can be utilized to improve the results of a reconstruction method. After a reconstruction method is applied, public information of popular and unpopular items related to the targeted data set is exploited to improve the reconstruction results. Algorithm 2 is proposed to utilize popular (or unpopular) movies public information:

3.3.2. Prestigious awards

Another trusted source of auxiliary information can be prestigious awards. For instance, winning an Oscar award for a movie, Nobel Prize for a book, or a Grammy award for a song is an indicator that many of the people will like that item. In other words, it is likely a favorable rating for award winner items in the data set. When an item wins a prestigious award, users might be biased toward liking that item. Considering all these assumptions, it would be reasonable to integrate them into the dissertation. IMDb also provides awards for movies. Awards for each item in the targeted data set are collected. The approach is the same as popular items. It is anticipated that users like items more as well as vote items higher. An algorithm illustrating abovementioned notion is given in Algorithm 3.

Algorithm 2: Utilizing the popular (or unpopular) movies

```
    $P = \{p1, p2, \dots, pm\}$, rating value for each item in the targeted data set is collected from IMDb
    A predefined value to assume a movie is popular (or unpopular) in P
    A threshold value for popular (or unpopular) set

Output:
    Movies with a rating greater than or equal to the threshold value are considered as popular
Steps:
    1. Reconstruct the perturbed data
    2. For each popular (or unpopular) items, compare the estimated rating with its threshold
        a. If the estimated rating is less than the threshold, add (or decrease) one to the estimated rating
        b. Otherwise, let the estimated rating remain the same
```

Algorithm 3: Utilizing the prestigious award winner set

```
Input:
    $O = \{o1, o2, \dots, ok\}$, awards for each item in the targeted data set are collected from IMDb

    A threshold value for prestigious award winner set
Output:
    Movies with a rating greater than or equal to the threshold value are considered as a prestigious award winner
Steps:
    1. Reconstruct the perturbed data
    2. For each prestigious award winner items, compare the estimated rating with its threshold
        a. If the estimated rating is less than the threshold, add one to the estimated rating
        b. Otherwise, let the estimated rating remain the same
```

3.3.3. Users' demographic information

A recommender system might hold user-related information such as age, gender, and occupation in addition to ratings. Such information can be utilized to exploit users' demographic data and item relationship (Vozalis and Margaritis, 2007; Santos, Garcia Manzato and Goularte, 2014; Gogna and Majumdar, 2015; Al-Shamri 2016). Imagine a data set including votes for various items like jewelry. Females are expected to rate higher than males for jewelry because males are usually not interested in jewelry like necklace, ring, and bracelet. Thus, demographic information can be a factor to improve the reconstruction results as auxiliary information. Since the reconstruction method is tested on a movie-related data set, users' demographic information with movie genre preferences is combined.

British Film Institute conducted an online survey of 2,036 UK adults whose ages are between 15 and 74 to illustrate the contribution of movies to British culture (Alliance, 2011). Table 416 in the report depicts the movie genre preferences with age and gender. The hypothesis is that age, or gender preference of a specific movie genre can differ so that sub-matrices based on gender-genre or age-genre are created. Thus, the impact of gender-genre sub-matrix is first examined. Based on the survey results, top genre preferences for each gender are selected. A threshold value is determined for movies in top genres. If any estimated movie rating in the top genre is less than the threshold, the estimated rating is increased by one as in the previous approaches. Some movies might have more than one genres; however, they are only increased once. The same process is repeated for age-genre sub-matrix, as well. For age-genre sub-matrix, the age in Table 416 (Alliance, 2011) is divided into five groups, which are 15-24, 25-34, 35-44, 45-54, and 55+. Since the targeted data set includes users that is younger than 15, those users are added into the first group. An excerpt from Table 416 (Alliance, 2011) displaying the distribution of top movie genres is seen in Table 3.1 based on different age groups.

Table 3.1. *Genre preferences with age.*

Until 25	**25-34**	**35-44**	**45-54**	**55+**
Comedy	Comedy	Comedy	Comedy	Drama
Action	Action	Drama	Thriller	Thriller
Adventure	Adventure			

3.3.4. Prior knowledge about mean of each item

Another considered auxiliary information is prior knowledge about the mean of each item. The idea is that item means may not violate individuals' privacy. Assume that the mean of each item is known although the original data is disguised by RPT. The question is whether this information is enough to make a better estimation. After the estimated data is obtained using the reconstruction method, the mean of an estimated item with the related original item's mean can be compared, which is known in advance. If the mean of an estimated item is less than the original one, some of the estimated items' ratings are randomly increased by one; otherwise, they are decreased by one. The purpose of this process is to close the gap between the mean of estimated and original items so that the reconstruction results get better.

3.3.5. Prior knowledge about mean of each user's ratings

What if are the mean of each user's ratings known publicly? It is clear that the mean of each user's ratings is invaluable auxiliary information to enhance the outcome after the reconstruction method is applied. Assume that the mean of each user's ratings is publicly available. If the mean is known, the only essential element to calculate the original ratings is to estimate the standard deviation for each user. Below is the equation displaying how the estimated standard deviation ($\tilde{\sigma}$) is calculated in terms of the estimated minimum ($\bar{x}_{min}$) and maximum ($\bar{x}_{max}$) rating calculated from *k*-means clustering algorithm. Maximum ($\tilde{z}_{max}$) and minimum ($\tilde{z}_{min}$) z-score values are obtained from the disguised matrix. Eq. 3.2 shows how the standard deviation of a user is estimated.

$$\tilde{\sigma} = \frac{(\bar{x}_{max} - \bar{x}_{min})}{(\tilde{z}_{max} - \tilde{z}_{min})} \tag{3.2}$$

In Eq. 3.2, the disguised z-score values are used due to having only the disguised matrix. Using z-score values from the disguised matrix instead of the original z-score values might cause an error. However, knowing the mean of each user can make a significant contribution to the reconstruction result, and the error can be acceptable.

3.4. The Proposed Method, Reconstruction from the Estimated z-score Values

After recovering the estimated z-score data from SVD EM-based reconstruction, the original ratings must be predicted. Zhang, Ford and Makedon (2006a) only show how to derive the estimated z-score values from the disguised data using SVD-EM algorithm. They assume that the average and standard deviation for each user are known. Then, the error rate between the original ratings and the estimated ratings are calculated using prior knowledge. However, it is not a realistic assumption to have both of prior knowledge about averages and standard deviations of the user ratings while reconstructing the original ratings from the estimated z-score values because these parameters are not made available by users. Therefore, a new method is offered to estimate the original ratings when averages and standard deviations are unknown.

A proposed method utilizes the extreme values, which are the upper and lower bound of the rating scale, of the targeted data set. The extreme values or rating range of the data set is public information and is explicitly known by the server. For example, the maximum and minimum rating for MovieLens data set, which is used in the experiments, are 5 and 1, respectively. Although the server holds only the disguised z-score values, the server can easily find the extreme values of the targeted data set. Moreover, even if a data set is disguised by any data disguising methods, the rating range of data set is the most common public information that is known by everyone.

The proposed method estimates the average and the standard deviation of the user ratings after the reconstructed data is obtained by SVD-EM. The proposed method utilizes the original formula of z-score. As seen from Eq. 3.3, if the original rating and its z-score value are known, the average and standard deviation can be computed. However, the server has neither original ratings nor their z-score correspondences.

$$z = \frac{x - \mu}{\sigma} \tag{3.3}$$

When the extreme values belonging to the data set are utilized, two z-score equations are obtained, one for the minimum and one for the maximum extreme values. The problem is turned into two equations with two unknowns as seen from Eq. 3.4. All that is necessary is to solve these two equations to find out the estimated average and the standard deviation of the user.

$$\begin{aligned} x_{min} &= \tilde{z}_{min} \times \tilde{\sigma} + \tilde{\mu} \\ x_{max} &= \tilde{z}_{max} \times \tilde{\sigma} + \tilde{\mu} \end{aligned} \tag{3.4}$$

The values of $\tilde{z}_{min}$ and $\tilde{z}_{max}$ are available in the reconstructed data. Also, using the extreme values, the lowest value in the rating range is assigned to x_{min} and the highest value in the rating range is assigned to x_{max}. The hypothesis is that the minimum z-score ($\tilde{z}_{min}$) among the estimated z-score values is likely to belong to the minimum rating of the original data. Likewise, the maximum rating of the original data is expected to be associated with the maximum z-score ($\tilde{z}_{max}$). Since the estimated $\tilde{z}_{min}$ and $\tilde{z}_{max}$ values are employed in Eq. 3.4, the results might differ from the original ones.

When there are two equations with two unknown parameters, which are the estimated standard deviation and the estimated average, there is only one solution. This method makes the calculation of the estimated average and standard deviation easy. The proposed method demonstrates that even if the average and standard deviation for each user is not known, it is not difficult to roughly estimate them.

3.5. Complexity Analysis

Complexity analysis of the reconstruction of the original data from the disguised one is given briefly. There are two key steps in the complexity analysis. The first one explains the computational complexity of the reconstruction methods, and the second step shows how much computational complexity is required when auxiliary information is utilized to improve the reconstruction results.

Complexity analysis begins with reconstructing the original ratings from the disguised data. Assume that a matrix consists of *n* rows (users) and *m* columns (items).

i. The computational complexity of a full SVD is $O = (m^2n)$. However, in the reconstruction method, SVD is divided into two steps which are expectation and maximization. In the expectation step, each unrated item is filled with the corresponding previous reconstruction result, hence, the computational complexity is $O = (nm)$. In the maximization step, SVD is applied. Since top k singular values are selected to reconstruct the data, the computational time of the SVD is calculated as $O = (nmk)$. The iteration number is also an important factor to calculate the total complexity time. As a result, the total computational complexity of SVD-EM algorithm is $O = \big(t(nm + nmk)\big)$, where t is the iteration number in the reconstruction method. The computational complexity of SVD-EM requires more time compared with only the SVD taking $O(nmk)$; however; the success of SVM-EM algorithm to reconstruct the original data is remarkable.

ii. The complexity of k-means clustering algorithm is analyzed. The operations in each step should be evaluated correctly. The algorithm takes $O = (Ikmt)$, where k is the number of clusters, m is the number of data points (items), t is the time to calculate the distance between items and centroids, and I is the number of iterations until the clustering does not change. In the data set, since there are n rows (users), the total computational complexity of k-means is $O = (Ikmtn)$.

Complexity analysis of auxiliary information utilized to improve the results of the existing reconstruction methods should also be considered briefly. The proposed approaches such as popular and unpopular items, prestigious awards, or prior knowledge about the mean of each item run in linear time. In other words, the time complexity grows linearly based on the number of users or items. The time complexity of auxiliary information is summarized in Table 3.2.

Table 3.2. *The computational complexity of auxiliary information*

The auxiliary information	Computational complexity
Popular movies (assume that *p* is the number of popular items)	*O(np)*
Unpopular movies (assume that *u* is the number of unpopular items)	*O(nu)*
Prestigious awards (assume that *a* is prestigious award winner set of items)	*O(na)*
Users' demographic information	*O(nd)*
Prior knowledge about mean of each item	*O(m)*
Prior knowledge about mean of each user's ratings	*O(nm)*

It is also analyzed the complexity of the proposed method which estimates the average and the standard deviation of the user ratings from the reconstructed z-score values. These values are calculated from the proposed method for each user. Therefore, the complexity of the estimating the average and standard deviation of the users take $O = (n)$. After estimating these values, complexity analysis of the predicting the original ratings is $O = (nm)$. Overall complexity of the proposed method is $O = (nm)$.

3.6. Experiments

Throughout the experiments, the proposed and existing reconstruction methods are analyzed to display whether auxiliary information can improve the reconstruction results. Various public information is available and can be utilized to get better the reconstruction results. Different experiments are conducted to test the proposed approaches and demonstrate how different kinds of auxiliary data can contribute to the results of the data reconstruction.

3.6.1. Data set and evaluation metric

A well-known recommender system data set, MovieLens is used in the experiments. MovieLens is a movie recommendation website (www.grouplens.org), and each rating is based on a 5-star scale. The standard 100K MovieLens data set including 943 users and 1862 items is selected. At least 20 items have been rated by users.

Mean Absolute Error (MAE) is used to measure the reconstruction accuracy for *k*-means clustering- and SVD EM-based reconstruction, which is denoted MAE and zscore-MAE, respectively. MAE measures how close the estimated ratings are the original ratings. Eq. 3.5 shows how to calculate MAE, as follows:

$$\mathrm{MAE} = \frac{1}{R}\sum_{i=0}^{R} |e_i - o_i| \tag{3.5}$$

where e_i is the estimated rating, o_i is the original rating, and R is the number of all rated items.

In addition, the ratio of correct estimated ratings to the original ratings is calculated for only *k*-means clustering-based reconstruction and denoted as accuracy in Eq. 3.6.

$$accuracy = \frac{the\ number\ of\ correct\ estimated\ ratings}{the\ number\ of\ all\ ratings\ in\ MovieLens} \tag{3.6}$$

3.6.2. Methodology

Several experiments are performed to show how MAE changes with the varying value of parameters based on uniform or Gaussian distribution using *k*-means clustering based and SVD EM-based reconstruction. The experiments are run 100 times. The number of clusters is set to 5 in *k*-means clustering-based method since MovieLens data set has five distinct ratings. The number of dimensions is set to 10 in the SVD-EM reconstruction. The estimated ratings are compared with the real ratings to see how auxiliary information affects the results.

3.6.3. Experimental results

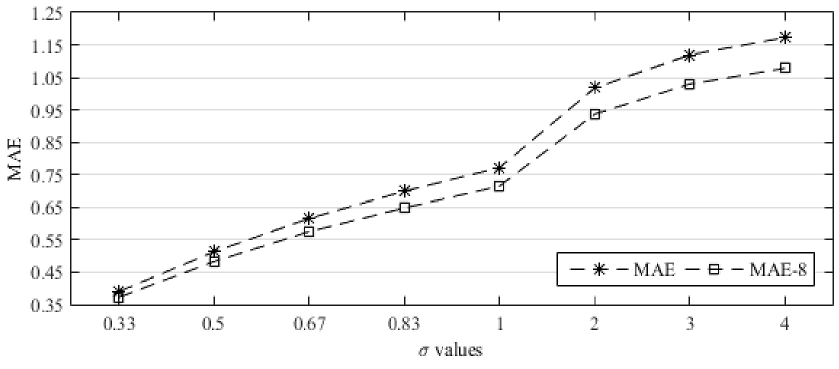
Experiment 1 – *k*-means clustering-based reconstruction results with popular movies. This experiment evaluates how popular items (or movies in the data set) affect the results after *k*-means clustering-based algorithm is applied. The hypothesis is that some popular movies from the public and trusted sources can contribute to the data reconstruction results, and better results will be achieved by increasing number of popular movies. In this regard, all movie information in MovieLens data set is collected from IMDb website, where people all around the world vote for movies. Movies are put in descending order by their ratings from IMDb, movies with a rating greater than or equal to 8 are considered popular in the assumption. The threshold value is set to 4. After the existing method is applied, a method of improving the results is to increase the estimated popular movies' ratings by one if the related popular movie is estimated less than the threshold. If the related popular movie is estimated either 4 or 5 in the experiment, its rating is preserved because it is already considered popular. In brief, the purpose of this process is to increase the estimated popular movies' rating because it is anticipated that these movies are voted higher. Three cases are defined to test how a different number of movies affects the result. The first, second, and third case for this experiment include top 62, 112, and 188 popular movies whose IMDb ratings are higher than or equal to 8.4, 8.2, and 8.0, respectively. MAE and accuracy are calculated. MAE-8_4, MAE-8_2, and MAE-8 illustrate the results when popular movies whose ratings are higher than or equal to 8.4, 8.2, and 8.0 in IMDb, respectively are used. Similar notation is used for accuracy. Note that the original data is masked using either Gaussian or uniform distributions with varying the standard deviation (σ) values. After estimating overall averages of the values, Table 3.3 displays the results.

Table 3.3. *k-means clustering-based reconstruction results with popular movies*

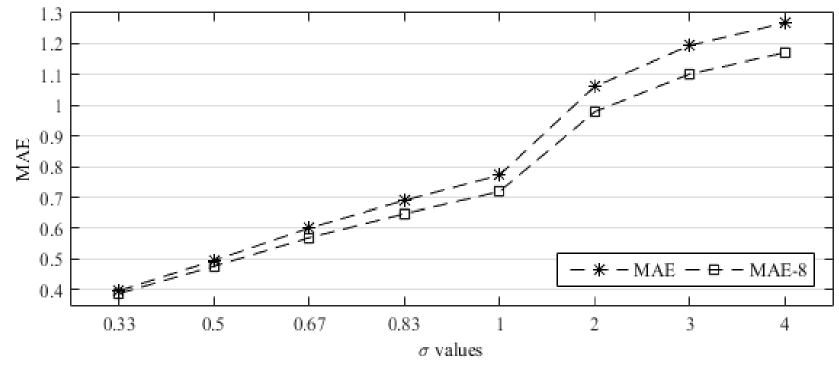
	σ	**MAE**	**Acc**	**MAE-8_4**	**Acc-8_4**	**MAE-8_2**	**Acc-8_2**	**MAE-8**	**Acc-8**
Gaussian Dist.	0.33	0.38972	0.79202	0.38206	0.78030	0.37697	0.77464	0.37204	0.77161
	0.5	0.51328	0.62423	0.50123	0.63690	0.49177	0.62571	0.48284	0.62964
	0.67	0.61605	0.52685	0.59757	0.53595	0.58641	0.54101	0.57518	0.53089
	0.83	0.69924	0.46702	0.67609	0.48327	0.66256	0.48500	0.64812	0.48714
	1	0.77149	0.41190	0.74584	0.43560	0.72998	0.44173	0.71457	0.43940
	2	1.01822	0.30839	0.97997	0.32762	0.95782	0.33280	0.93714	0.33381
	3	1.11984	0.27661	1.07730	0.30018	1.05256	0.29970	1.03020	0.29786
	4	1.17210	0.26530	1.12729	0.28083	1.10155	0.27976	1.07866	0.28411
Uniform Dist.	0.33	0.39616	0.69351	0.39158	0.68381	0.39022	0.68940	0.38747	0.69304
	0.5	0.49549	0.54405	0.48684	0.56482	0.48171	0.57339	0.47639	0.56964
	0.67	0.60111	0.46750	0.58638	0.48036	0.57842	0.48399	0.56963	0.48542
	0.83	0.68991	0.41054	0.67082	0.42565	0.65839	0.42673	0.64725	0.43244
	1	0.77201	0.37030	0.74775	0.38994	0.73341	0.39548	0.71857	0.39548
	2	1.06181	0.28143	1.02297	0.29024	1.00069	0.30411	0.97891	0.30440
	3	1.19372	0.25750	1.14977	0.27125	1.12401	0.27446	1.10246	0.27256
	4	1.26793	0.24268	1.22186	0.25685	1.19458	0.24893	1.17058	0.25815

Table 3.3 shows that auxiliary information, popular movies in the experiment, is helpful for deriving the original ratings from the masked ratings. As the number of popular movies increases, the results are getting better. Likewise, the improvement is much more for larger σ values. For example, MAE for Gaussian distribution when σ is 1 is 0.77149 while MAE-8 for the same distribution with the same σ is 0.71457 when 188 movies, which are rated greater than or equal to 8 are used as popular. Similar achievement for the uniform distribution is found.

Figure 3.1. illustrates the comparison of the existing *k*-means clustering-based reconstruction results with the popular movies. MAE-8 is selected because its contribution is more significant than others as can be seen in Table 3.3. MAE-8 represents the results for *k*-means clustering with popular movies while MAE shows the results of the existing *k*-means clustering-based reconstruction algorithm. As seen from Figure 3.1, *k*-means with popular movies provides better results for both types of distribution for each value of σ.



a) Gaussian Distribution



b) Uniform Distribution

Figure 3.1. *Comparison of the existing k-means clustering-based reconstruction*

Experiment 2 - *k*-means clustering-based reconstruction results with unpopular movies. This experiment assesses whether unpopular movies improve the results or not. In this case, it is expected that users rate movies that are not popular with a lower rating. IMDb has a movie list, which shows IMDb Bottom 100 movies voted by users as unpopular movies. However, there are a few movies in MovieLens from IMDb Bottom 100 list. Thus, IMDb ratings are used in place of IMDb Bottom 100 list to define unpopular movies. Unlike Experiment 1, movies are put in ascending order by their ratings from IMDb. Movies with ratings less than or equal to 5 are considered unpopular in the experiments. Also, the number of unpopular movies is increased to see how results change as is the Experiment 1. Three groups are defined, called unpopular movies, for this experiment which includes 41, 83, and 154 movies with ratings are less than or equal to 4.0, 4.5, and 5 in IMDb, respectively.

The assumption is that the ratings of unpopular movies should be gathered around the values of 1, 2, and 3. Thus, the threshold value is selected 3. The strategy is to decrease unpopular movies' ratings by one if the estimated unpopular movie's rating is found greater than the threshold by the *k*-means clustering algorithm. The results for movies with ratings are less than or equal to 4, 4.5, and 5 are displayed as MAE-U4, MAE-U4_5,

Table 3.4. *k-means clustering-based reconstruction results with unpopular movies*

	σ	MAE	Acc	MAE-U4	Acc-U4	MAE- U4_5	Acc-U4_5	MAE-U5	Acc-U5
Gaussian Distr.	0.33	0.38972	0.79202	0.38973	0.78744	0.39074	0.78893	0.39247	0.78780
	0.5	0.51328	0.62423	0.51439	0.62315	0.51495	0.62470	0.51650	0.62696
	0.67	0.61605	0.52685	0.61617	0.52000	0.61706	0.52583	0.61729	0.52274
	0.83	0.69924	0.46702	0.69808	0.47012	0.69861	0.46387	0.69959	0.47030
	1	0.77149	0.4119	0.77248	0.42125	0.77201	0.42399	0.77225	0.41500
	2	1.01822	0.30839	1.01765	0.30655	1.01754	0.31202	1.01538	0.31530
	3	1.11984	0.27661	1.11896	0.27702	1.11773	0.28411	1.11501	0.27720
	4	1.17210	0.26530	1.17145	0.26095	1.17080	0.25869	1.16728	0.26113
Uniform Distr.	0.33	0.39616	0.69351	0.39587	0.69976	0.39660	0.69000	0.39774	0.68774
	0.5	0.49549	0.54405	0.49595	0.55423	0.49527	0.54399	0.49734	0.55179
	0.67	0.60111	0.46750	0.60093	0.45524	0.60086	0.46869	0.60108	0.46089
	0.83	0.68991	0.41054	0.68987	0.40685	0.68992	0.40988	0.69000	0.40946
	1	0.77201	0.37030	0.77208	0.37274	0.77154	0.37923	0.77115	0.36958
	2	1.06181	0.28143	1.06019	0.28137	1.05987	0.28369	1.05706	0.29018
	3	1.19372	0.25750	1.19201	0.26143	1.19133	0.26256	1.18770	0.25571
	4	1.26793	0.24268	1.26654	0.23946	1.26583	0.23750	1.26219	0.24238

and MAE-U5, respectively. Similar notation is used for accuracy. The overall averages are shown in Table 3.4. A similar improvement is expected as in Experiment 1. However, Table 3.4 illustrates that the reconstruction results for three groups in this experiment are almost the same although increasing the number of unpopular movies is used. This approach fails because the number of votes given for unpopular movies is small in comparison to the number of votes given popular movies. Therefore, the impact of unpopular movies is not significant as much as believed.

Experiment 3 - *k*-means clustering-based reconstruction results with a prestigious award. This experiment is conducted to test the assumption about prestigious award. Since the data set is about movies, the Oscar winner movies are used as auxiliary information. The Oscars winner movies like the popular movies are expected to be liked by most of the users who watch them. In other words, it is likely these movies have favorable ratings. When a movie wins the Oscars, users might be willing to watch it and think positive about the movie before watching it. There are 172 movies among all MovieLens data set, which won the Oscars. After reconstructing data using the *k*-means clustering algorithm, the Oscars winner movies are determined. The threshold value is set to 4 as in Experiment 1. If the movie marked as the Oscars winner is estimated less than the threshold, its rating is increased by one. Otherwise, its rating remains the same. For the comparison, Table 3.5 shows the result for *k*-means clustering algorithm without auxiliary information and popular movies rated higher than or equal to 8 in the IMDb, MAE, and MAE-P, respectively. MAE-O denotes the result for only the Oscar winner movies' ratings updated after *k*-means clustering algorithm. Besides, popular and Oscar winner movies are used together as auxiliary information to get better results. After merging these two set of auxiliary movie information, 277 movies are obtained which are popular and the Oscar winner movies. The result is demonstrated as MAE-P_O. Similar notation is used for accuracy. As seen from Table 3.5, the results for popular movies and the Oscars winner movies are similar. The results support the hypothesis about the Oscars winner movies. Moreover, when two pieces of auxiliary information (popular and Oscar winners) are combined, the combined set of movies give better results than using only one piece of auxiliary information.

Figure 3.2 gives a picture of three kinds of *k*-means clustering algorithms' results. Since popular movies give almost same results with the Oscar winners, the results of the

Table 3.5. *k-means clustering-based reconstruction with popular movies, the Oscars, and merged movies*

	σ	MAE	Acc	MAE-P	Acc-P	MAE-O	Acc-O	MAE-P_O	Acc P_O
Gaussian Distr.	0.33	0.38972	0.79202	0.37204	0.77161	0.37524	0.74827	0.36797	0.74333
	0.5	0.51328	0.62423	0.48284	0.62964	0.48510	0.61161	0.47233	0.61089
	0.67	0.61605	0.52685	0.57518	0.53089	0.57685	0.52101	0.55909	0.53488
	0.83	0.69924	0.46702	0.64812	0.48714	0.65063	0.47220	0.62932	0.47750
	1	0.77149	0.41190	0.71457	0.43940	0.71611	0.43673	0.69287	0.44095
	2	1.01785	0.30399	0.93787	0.33079	0.93850	0.32952	0.90673	0.34086
	3	1.12020	0.27596	1.03144	0.30198	1.03173	0.30089	0.99666	0.31186
	4	1.17238	0.26274	1.07924	0.28834	1.07972	0.28720	1.04297	0.29807
Uniform Distr.	0.33	0.39616	0.69351	0.38747	0.69304	0.39287	0.65958	0.38889	0.67268
	0.5	0.49549	0.54405	0.47639	0.56964	0.48004	0.55232	0.47222	0.56345
	0.67	0.60111	0.46750	0.56963	0.48542	0.57267	0.47298	0.55932	0.47958
	0.83	0.68991	0.41054	0.64725	0.43244	0.64874	0.42548	0.63171	0.43548
	1	0.77201	0.37030	0.71857	0.39548	0.71939	0.39292	0.69805	0.40357
	2	1.06180	0.28039	0.97967	0.30468	0.97860	0.30435	0.94577	0.31478
	3	1.19477	0.25268	1.10282	0.27483	1.10147	0.27450	1.06482	0.28404
	4	1.26692	0.23923	1.17071	0.26051	1.16902	0.26028	1.13058	0.26946

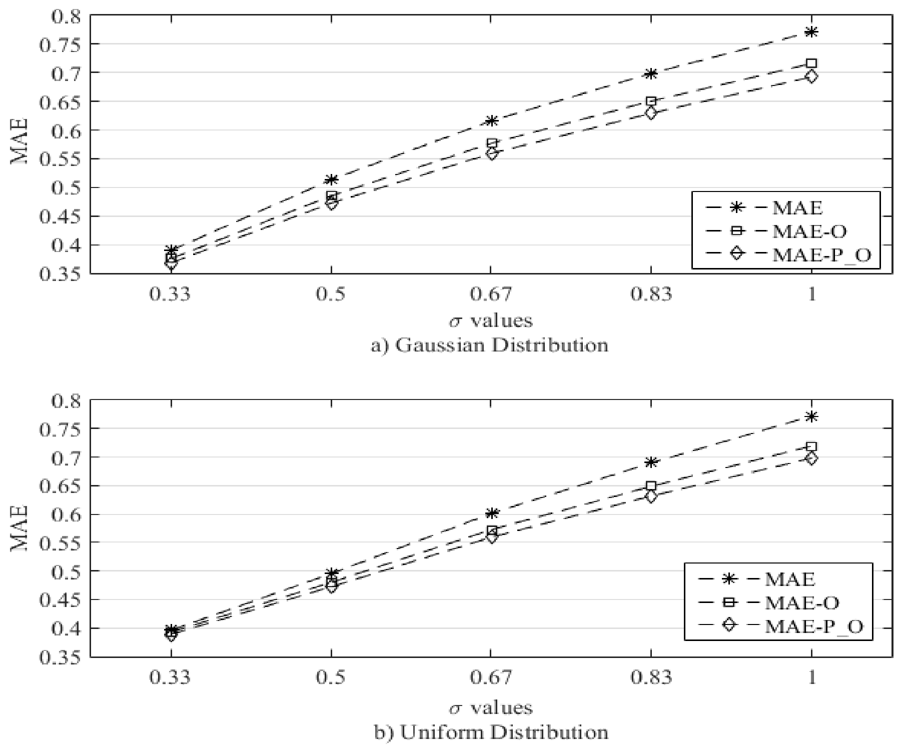


Figure 3.2. *Comparison of the existing k-means clustering-based reconstruction with the Oscars winner movies and the Oscars winner and popular movies*

Oscars winners (MAE-O) is only selected to illustrate in Figure 3.2. In other respects, using two sets of auxiliary information together (MAE-P-O) makes a significant influence on the results by comparison with *k*-means clustering without auxiliary information (MAE).

Experiment 4- *k*-means clustering-based reconstruction results with prior knowledge about the mean of each item. Another auxiliary information is prior knowledge about the mean of each item. Assume that MovieLens data set is disguised and the mean of each item is known. MAE and accuracy are again used in the same manner. MAE-I and Acc-I denote if the mean is known. The results are compared with MAE. Table 3.6 illustrates the results. One can realize in Table 3.6 that the approach of item means does not work when σ has a small number. However, when it gets larger, the assumption can be valuable to improve the performance. In brief, if the mean of items is known, it can make a substantial contribution to the results for larger σ values for both Gaussian and uniform distribution. For example, when it is 1 for the Gaussian distribution, MAE-I is 0.72484 while MAE is 0.77149 without applying the approach.

Table 3.6. *k-means clustering based reconstruction results with items' mean*

	σ	**MAE**	**Acc**	**MAE-I**	**Acc-I**
Gaussian Distr.	0.33	0.38972	0.79202	0.43483	0.64887
	0.5	0.51328	0.62423	0.52950	0.54488
	0.67	0.61605	0.52685	0.60707	0.48357
	0.83	0.69924	0.46702	0.66883	0.45024
	1	0.77149	0.41190	0.72484	0.41911
	2	1.01822	0.30839	0.91603	0.33458
	3	1.11984	0.27661	0.99882	0.31048
	4	1.17210	0.26530	1.04120	0.29310
Uniform Distr.	0.33	0.39616	0.69351	0.43535	0.61768
	0.5	0.49549	0.54405	0.51601	0.52262
	0.67	0.60111	0.46750	0.60207	0.46464
	0.83	0.68991	0.41054	0.67500	0.42220
	1	0.77201	0.37030	0.73992	0.39024
	2	1.06181	0.28143	0.96309	0.30911
	3	1.19372	0.25750	1.06265	0.27815
	4	1.26793	0.24268	1.11899	0.27155

Experiment 5- *k*-means clustering-based reconstruction results with prior knowledge about the mean of each user's ratings. An experiment is also conducted based on different σ values in order to observe how MAE changes when the mean of each user's rating is known. Similar to previous experiments, MAE and Accuracy results are shown in Table 3.7 to be compared with the results of the new assumption. MAE-U and Acc-U show the results when the users' means are known publicly. As seen from Table 3.7, this approach outputs the best results among all experiments conducted so far. It means that using mean of each user's ratings as publicly available data definitely enhances reconstruction outcomes. As mentioned before, the standard deviation of users remains the only required parameter to estimate the original ratings while users' mean is known in advance. Users' standard deviation can be quickly estimated utilizing available data as explained in Section 3.3.5.

Table 3.7. *k-means clustering-based reconstruction results with users' mean*

	σ	**MAE**	**Acc**	**MAE-U**	**Acc-U**
Gaussian Distr.	0.33	0.38972	0.79202	0.16255	0.83805
	0.5	0.51328	0.62423	0.29751	0.70733
	0.67	0.61605	0.52685	0.39969	0.61611
	0.83	0.69924	0.46702	0.47429	0.55670
	1	0.77149	0.41190	0.53754	0.51171
	2	1.01822	0.30839	0.74032	0.39903
	3	1.11984	0.27661	0.82335	0.36394
	4	1.17210	0.26530	0.86624	0.34781
Uniform Distr.	0.33	0.39616	0.69351	0.14548	0.85479
	0.5	0.49549	0.54405	0.31357	0.68812
	0.67	0.60111	0.46750	0.42929	0.57963
	0.83	0.68991	0.41054	0.50659	0.51452
	1	0.77201	0.37030	0.57168	0.46748
	2	1.06181	0.28143	0.81240	0.35207
	3	1.19372	0.25750	0.93115	0.31408
	4	1.26793	0.24268	0.99891	0.29545

Experiment 6- *k*-means clustering-based reconstruction results with user demographic information. User demographic information can be utilized as auxiliary information as well. MovieLens data set contains user demographic information such as gender and age besides ratings of all movies. In addition, each movie has at least one movie genre such as drama, action, thriller, or comedy with its name, release date, and other extra information. User demographic information is combined with genre preferences of movies in this experiment. Movie preference is compared with gender and age. The relationship between movie genre and gender is investigated first. Top two genre preferences for each gender are chosen to constitute gender-genre sub-matrix. In the experiment, comedy and drama are chosen for female; while comedy, action, and adventure are selected for a male. Adventure and action are clustered in one group in the survey (Alliance, 2011); however, they are different genres in the MovieLens data set. Therefore, three movie genres are selected for the male group. The hypothesis is that a male votes the specific type of movies higher, as a female does. Like the previous experiments, a threshold value is first selected. Its value is three for this experiment. Thus, if the estimated movie rating is below than the threshold based on movie genres, the estimated rating is increased by one. As mentioned before, some movies might have more than one genres, but their ratings are increased once. The results are seen in Table 3.8. MAE-D and Acc-D are the results when genders are divided in terms of genre preferences of movies. The assumption provides an advantage when compared with the implementation of *k*-means clustering algorithm's result.

Table 3.8. *k-means clustering-based results with genre preferences with gender*

	σ	**MAE**	**Acc**	**MAE-D**	**Acc-D**
Gaussian Distr.	0.33	0.38972	0.79202	0.37459	0.65429
	0.5	0.51328	0.62423	0.48564	0.54274
	0.67	0.61605	0.52685	0.57642	0.46798
	0.83	0.69924	0.46702	0.64915	0.41506
	1	0.77149	0.41190	0.71328	0.38387
	2	1.01822	0.30839	0.92766	0.30036
	3	1.11984	0.27661	1.01706	0.27411
	4	1.17210	0.26530	1.06293	0.25458
Uniform Distr.	0.33	0.39616	0.69351	0.39620	0.58863
	0.5	0.49549	0.54405	0.48536	0.48446
	0.67	0.60111	0.46750	0.57521	0.42696
	0.83	0.68991	0.41054	0.65092	0.38167
	1	0.77201	0.37030	0.71844	0.35446
	2	1.06181	0.28143	0.95632	0.27768
	3	1.19372	0.25750	1.06793	0.25768
	4	1.26793	0.24268	1.13206	0.24286

Another experiment is also performed to show the result based on different age groups. Note that Table 3.1 shows the age range, divided into five groups that are until

25, 25-34, 35-44, 45-54, and 55+, and movie genre based on various age group. The proposed assumption is the same with the previous one. The previous assumption is applied to the different age groups rather than genders. The outcomes are displayed in Table 3.9. Two experiments, as seen from Table 3.8 and Table 3.9, demonstrate very similar results. Therefore, either of two approaches can be utilized to improve the reconstruction results.

Table 3.9. *k-means clustering-based results with genre preferences with age*

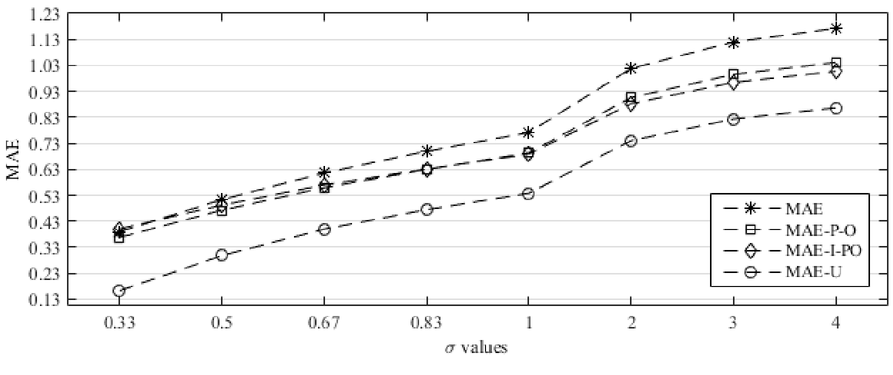
	σ	MAE	Acc	MAE-D	Acc-D
Gaussian Dist.	0.33	0.38972	0.79202	0.37465	0.65482
	0.5	0.51328	0.62423	0.48547	0.53815
	0.67	0.61605	0.52685	0.57572	0.46429
	0.83	0.69924	0.46702	0.64770	0.41875
	1	0.77149	0.41190	0.71173	0.38048
	2	1.01822	0.30839	0.92507	0.30131
	3	1.11984	0.27661	1.01258	0.26899
	4	1.17210	0.26530	1.05913	0.25595
Uniform Dist.	0.33	0.39616	0.69351	0.39567	0.58619
	0.5	0.49549	0.54405	0.48460	0.48554
	0.67	0.60111	0.46750	0.57437	0.42619
	0.83	0.68991	0.41054	0.64987	0.39012
	1	0.77201	0.37030	0.71730	0.36143
	2	1.06181	0.28143	0.95247	0.28012
	3	1.19372	0.25750	1.06445	0.26024
	4	1.26793	0.24268	1.12804	0.24530

Experiment 7- *k*-means clustering-based reconstruction results with combining different auxiliary information. Up to now, the approach, mean of each user's ratings, shows the best results in Experiment 5. However, that approach may be considered as a privacy breach and obtaining the users' mean is thought to be difficult. On the other hand, obtaining the mean of each item may not be a big issue. Taking this approach into consideration, assume that mean of items is known as in Experiment 4. In this experiment, some previous approaches are selected to show how MAE changes. Three approaches, which improve the results alone, are chosen whether using two or more auxiliary information together influence the results. First, an experiment related popular movies and mean of items is conducted. The results are represented by MAE-I_P and Acc-I_P. After the steps of Experiment 1 are performed, popular movies from all movies are removed. Mean of items approach is applied to the rest of the movies. The second experiment is like the previous one except three approaches are used together. In this case, popular movies, the Oscars winner movies and mean of items are used, denoted MAE-I_PO and Acc-I_PO. Popular movies with the Oscars winners are first performed as the Experiment 2, and then mean of items approach is applied to movies, which are neither popular movies nor the Oscars Winners. Table 3.10 represents the different auxiliary information make a significant contribution to the results. For instance, MAE for *k*-means algorithm when σ is 1, and Gaussian distribution is used is 0.77149 without any auxiliary information whereas MAE-I_PO for the same situation using three auxiliary information is 0.68784.

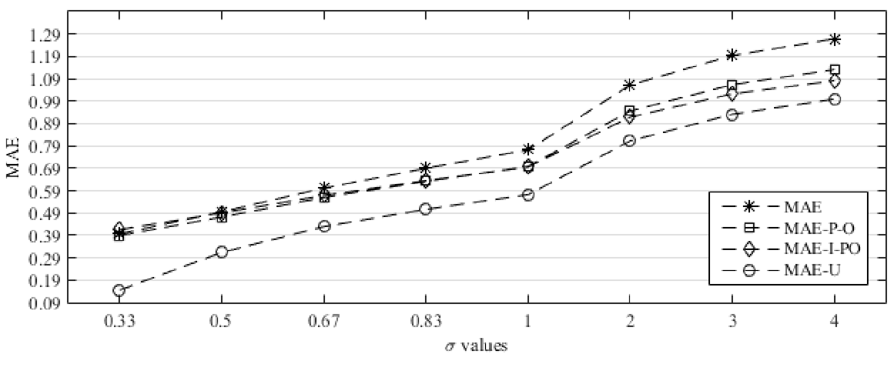
Table 3.10. *k-means clustering-based reconstruction results with combined auxiliary information*

	σ	**MAE**	**Acc**	**MAE-I_P**	**Acc- I_P**	**MAE- I_PO**	**Acc-I_PO**
Gaussian Dist.	0.33	0.38972	0.79202	0.40999	0.65881	0.40076	0.64375
	0.5	0.51328	0.62423	0.50436	0.55750	0.49200	0.55482
	0.67	0.61605	0.52685	0.58271	0.50304	0.56971	0.49190
	0.83	0.69924	0.46702	0.64568	0.45369	0.63092	0.44982
	1	0.77149	0.41190	0.70283	0.42149	0.68784	0.42042
	2	1.01822	0.30839	0.89849	0.32905	0.88230	0.33452
	3	1.11984	0.27661	0.98115	0.30595	0.96439	0.29970
	4	1.17210	0.26530	1.02455	0.29179	1.00677	0.29530
Uniform Dist.	0.33	0.39616	0.69351	0.41839	0.62875	0.41562	0.61470
	0.5	0.49549	0.54405	0.49735	0.53506	0.49078	0.52768
	0.67	0.60111	0.46750	0.57932	0.47119	0.56853	0.46911
	0.83	0.68991	0.41054	0.64843	0.43667	0.63502	0.43548
	1	0.77201	0.37030	0.71173	0.39988	0.69542	0.39512
	2	1.06181	0.28143	0.93801	0.30935	0.91844	0.30851
	3	1.19372	0.25750	1.04183	0.27649	1.02261	0.28548
	4	1.26793	0.24268	1.09981	0.26786	1.08102	0.26887

Until now, different auxiliary information is tested after performing the existing *k*-means clustering based reconstruction. As mentioned before, prior knowledge about users mean approach displays the best results among the others. Both popular movies with the Oscars approach and popular, the Oscars movies, and items mean approach gives favorable results within all experiments. Figure 3.3 gives a picture of three best results amongst all experiments for *k*-means clustering based reconstruction.



a) Gaussian Distribution



b) Uniform Distribution

Figure 3.3. *Comparison of the existing k-means based reconstruction with different auxiliary information*

Experiment 8 – Comparing error rates of true ratings after SVD EM-based reconstruction. The aim of this experiment is to evaluate the error rate between real ratings and the estimated ratings, which are extracted from the SVD-based reconstruction. After applying the SVD-EM algorithm, the estimated z-score values are obtained. zscore-MAE represents the error rate between the estimated z-score values and the original z-score values. However, it is essential to predict the original ratings. Zhang, Ford and Makedon (2006a) only show how to reconstruct the estimated z-score values from the disguised data using SVD-EM algorithm. They assume that the average and the standard deviation for each user are known to calculate the error rate between the original ratings and the estimated ones. P-MAE shows their assumption's results in Table 3.11. However, their assumption is not realistic. Since user averages approach produces promising results in Experiment 5 this information is utilized in this experiment instead of utilizing both user averages and standard deviations. The results in Table 3.11 depict when only users' average is known, denoted as M-MAE. Moreover, an experiment is also conducted when users' average and standard deviation are not known, as in the real world. The proposed method estimates the original ratings from the reconstructed z-score values without information about averages and standard deviations of users by utilizing the rating range of the MovieLens data set. The results are represented as Z-MAE. In this case, the iteration number for the expectation step in the SVD-EM algorithm is 50. The overall averages are displayed in Table 3.11.

Table 3.11. *Error rates of true ratings after SVD-based reconstruction is applied*

	σ	zscore-MAE	P-MAE	M-MAE	Z-MAE
Gaussian Dist.	0.33	0.57250	0.52998	0.55417	0.70699
	0.5	0.59043	0.55085	0.57836	0.74491
	0.67	0.61580	0.58004	0.60406	0.76026
	0.83	0.64384	0.61083	0.62858	0.78444
	1	0.68455	0.65611	0.66291	0.83169
	2	0.97488	0.99561	0.79741	0.98624
	3	1.32010	1.34974	0.86717	1.05861
	4	1.68385	1.72164	0.90654	1.10099
Uniform Dist.	0.33	0.57314	0.53133	0.55666	0.70841
	0.5	0.59081	0.55148	0.57861	0.73308
	0.67	0.61562	0.57938	0.60476	0.76406
	0.83	0.64524	0.61259	0.63204	0.79422
	1	0.68333	0.65510	0.66231	0.82969
	2	0.97973	0.97203	0.80141	0.98475
	3	1.32968	1.33916	0.87639	1.06378
	4	1.70239	1.72443	0.91858	1.10617

As seen from Table 3.11, the proposed method (Z-MAE) beats the approach (P-MAE) for larger σ values although the average and standard deviation of users are not known. Surprisingly, while P-MAE is 1.72164, Z-MAE is 1.10099 when the σ is 4. Of course, knowing the average of users (M-MAE) again gives better results.

Experiment 9- SVD EM-based reconstruction results with different auxiliary information. The experiment is conducted to improve the results of the existing SVD-EM algorithm using auxiliary information like *k*-means clustering algorithm. In the existing SVD-EM algorithm, the average and standard deviation are known publicly and used to derive the original ratings after reconstructing the estimated z-score values. In this experiment, it is tested whether auxiliary information improves the existing work's results. In order to improve the reconstruction results, different public information, which gives better results for *k*-means clustering algorithm, is evaluated. Therefore, the items mean approach is implemented, which is used in Experiment 4, to the results of the existing method. The results are demonstrated as I-MAE in Table 3.12. The second experiment is to apply the approach of popular movies and the Oscar winner movies. Finally, the results represent when all three approaches, which are items mean, popular movies and the Oscar winner movies are used together as in Experiment 7. P_O-MAE and I_PO-MAE denote the result of second and third experiments, respectively. The iteration number for the expectation step in the SVD-EM algorithm is 20 in the experiment.

Table 3.12. *Error rates of true ratings after SVD-based reconstruction is applied based on different auxiliary information*

	σ	**zscore-MAE**	**P-MAE**	**I-MAE**	**P_O-MAE**	**I_PO-MAE**
Gaussian Dist.	0.33	0.59087	0.55206	0.58193	0.59608	0.60725
	0.5	0.60425	0.56673	0.59630	0.60599	0.61785
	0.67	0.62443	0.58950	0.61605	0.62064	0.63301
	0.83	0.64862	0.61673	0.64121	0.63961	0.65222
	1	0.67959	0.65062	0.67027	0.66108	0.67471
	2	0.92377	0.91302	0.89168	0.84463	0.85450
	3	1.21849	1.22178	1.14644	1.08301	1.08453
	4	1.53521	1.55117	1.42705	1.36198	1.35267
Uniform Dist.	0.33	0.59086	0.55169	0.58271	0.59634	0.60735
	0.5	0.60477	0.56779	0.59706	0.60637	0.61838
	0.67	0.62485	0.59023	0.6173	0.62137	0.63371
	0.83	0.64868	0.61664	0.64091	0.63846	0.65137
	1	0.68028	0.65175	0.67008	0.66129	0.67417
	2	0.92636	0.91584	0.89464	0.84745	0.85643
	3	1.2257	1.23023	1.15644	1.09173	1.09272
	4	1.54633	1.56293	1.43683	1.3703	1.36102

Table 3.12 shows that auxiliary information cannot contribute to the results as much as the other experiments for small σ values and the results decline for those values. The main reason is that the effect of randomization is the minimum for small σ values, and knowing the average and standard deviation already provide a huge advantage for small σ values. For example, when σ is 0.5 for the Gaussian distribution, P-MAE is 0.56673 while P_O-MAE is 0.60599 and I_PO-MAE is 0.61785, respectively. However, as the σ values get bigger and the randomization increases, the estimated z-score values by SVD-

EM become less accurate. On the other hand, Table 3.12 shows that the integration of auxiliary information improves the results when the original data is disguised by larger σ values. As seen from Table 3.12, when the σ value greater than or equal to two, auxiliary information makes a significant contribution to the results. For example, when σ is 2 for the Gaussian distribution, P-MAE is 0.91302 while P_O-MAE is 0.84463 and I_PO-MAE is 0.85450, respectively.

Experiment 10- The proposed method with SVD EM-based reconstruction results based on the different auxiliary information. Another experiment is performed to improve the results of the SVD-EM algorithm using auxiliary information when the averages and standard deviations of users are not available. After the estimated z-score values are reconstructed by the SVD-EM algorithm (zscore-MAE), the proposed method, reconstruction from the estimated z-score values, is used to predict the original ratings when averages and standard deviation are not known (Z-MAE). Experiment 10 is the same as Experiment 9 except the proposed method is used. In this experiment, it is tested whether the reconstruction results estimated from the proposed method can be improved by utilizing some of the proposed auxiliary information. Three different auxiliary

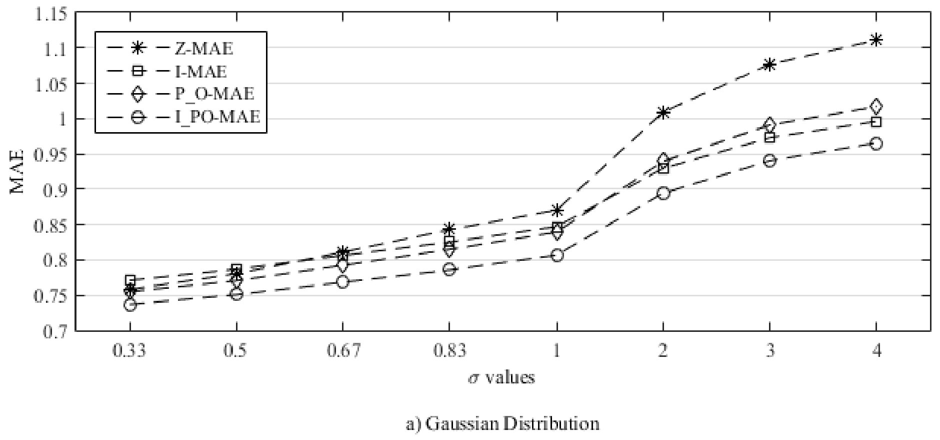
Table 3.13. *Error rates of true ratings after the proposed method is applied to SVD-based reconstruction with different auxiliary information*

	σ	zscore-MAE	Z-MAE	I-MAE	P_O-MAE	I_PO-MAE
Gaussian Dist.	0.33	0.59079	0.75823	0.77110	0.75504	0.73693
	0.5	0.60478	0.78057	0.78693	0.77066	0.75083
	0.67	0.62445	0.81111	0.80642	0.79241	0.76849
	0.83	0.64856	0.84273	0.82528	0.81487	0.78577
	1	0.67956	0.87041	0.84702	0.83927	0.80689
	2	0.92346	1.00934	0.92969	0.93959	0.89456
	3	1.21933	1.07688	0.97282	0.99095	0.94043
	4	1.53616	1.11091	0.99612	1.01671	0.96554
Uniform Dist.	0.33	0.59081	0.75920	0.77145	0.75608	0.73749
	0.5	0.60438	0.78067	0.78658	0.77102	0.75057
	0.67	0.62436	0.81054	0.80609	0.79193	0.76769
	0.83	0.64874	0.83910	0.82659	0.81644	0.78637
	1	0.67880	0.87032	0.84571	0.83683	0.80552
	2	0.92712	1.01249	0.93338	0.94412	0.89826
	3	1.22681	1.08134	0.97612	0.99461	0.94359
	4	1.54663	1.11314	1.00124	1.02107	0.96994

information approaches are run, which demonstrate better results in the experiments of *k*-means clustering-based reconstruction.

As shown in Table 3.13, zscore-MAE demonstrates the error rate between the estimated z-score values and the original z-score values as in the previous experiment. Z-MAE shows the results when the proposed method is applied. In the experiment, estimating the original ratings are more prominent rather than estimating the actual z-score values when user averages and standard deviations are unknown. As seen from Table 3.13, the auxiliary information is valuable to improve the results. When σ values are increased, the effect of auxiliary information to improve the reconstruction results is more obvious. The best result is demonstrated when all three approaches are used together as in the other experiments. For example, when σ is 1 for the Gaussian distribution, Z-MAE is 0.87041 while I_PO-MAE is 0.80689.

Up to now, it is demonstrated how the reconstruction results derived from the SVD-EM algorithm can be enhanced by using various auxiliary information. Figure 3.4 gives



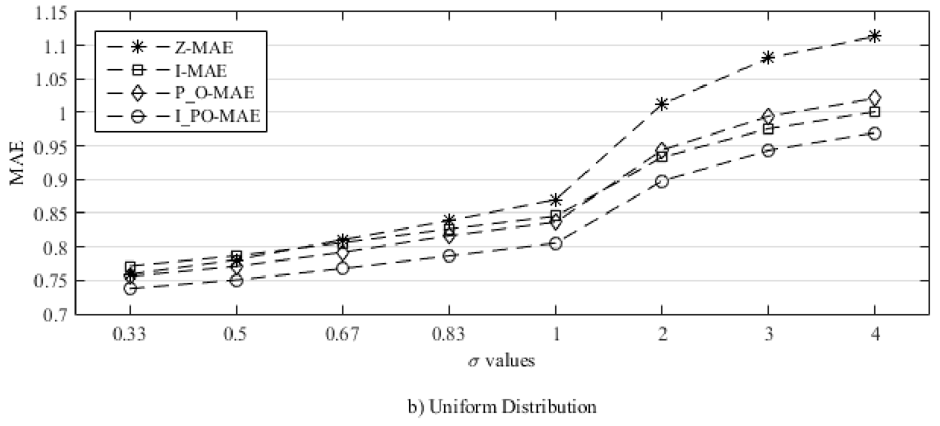


Figure 3.4. *Comparison of estimation of true ratings after the proposed method is applied to SVD-based reconstruction with different auxiliary information*

a picture of the selected auxiliary information results with the proposed method. Figure 3.4 first displays the results of the proposed method (Z-MAE). Then, items' mean approach (I-MAE), popular movies with the Oscars winner movies (P_O-MAE), and the combination of these two approaches (I_PO-MAE) are illustrated, respectively. As seen from Figure 3.4, using more than one approach gives better results as is *k*-means clustering algorithm. When σ gets larger, even one piece of the auxiliary information provides more contribution than the proposed approach. Although items' mean approach is the only auxiliary information and starts with a little worse result as compared with others, it beats the proposed method alone and the proposed method with two pieces of auxiliary information, the popular and Oscars winner movies, for larger σ values. On the other hand, the combination of three pieces of auxiliary information, which are the popular, Oscars winner movies and items' mean, produce the best results for each value of σ.

3.7. Conclusions

In this chapter, it is represented how to derive the original ratings from the disguised z-score values. First, the results of existing methods are demonstrated, which are *k*-means clustering-based and SVD-based reconstruction. Throughout all experiments with regards to these existing methods, the only question is how much contribution to the existing methods can be made by utilizing several types of auxiliary information. To answer this question, different auxiliary information such as popular movies, the Oscars winner movies, unpopular movies, prior knowledge about the mean of each item or each user, and demographic information has been tested to improve the reconstruction results. *k*-means clustering algorithm with auxiliary information produces better results than the existing *k*-means clustering. When three types of auxiliary information, which are popular movies, the Oscars winner movies, and mean of each item, are used together, the results are the best among all experiments for *k*-means clustering algorithm.

In addition, the experiments for SVD-based reconstruction with EM using auxiliary information, which gives better results for *k*-means clustering algorithm, are carried out. *k*-means clustering algorithm beats the SVD-EM reconstruction when the standard deviation is less than two; however, when such values get larger than two, SVD-EM reconstruction starts to beat *k*-means clustering algorithm. However, SVD-based reconstruction method requires that both user and item averages be known, which are very strong assumptions to be true in a real-world scenario. The proposed reconstruction method in this study handles this problem to reconstruct original ratings from estimated z-scores by considering the extreme values of the rating scale. The proposed approach is not only more realistic but also achieves better results for larger standard deviation values.

4. RECONSTRUCTING RATED ITEMS FROM INCONSISTENTLY PERTURBED DATA IN CENTRAL DATA-BASED PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEM

The basic idea behind privacy-preserving collaborative filtering schemes is to prevent data collectors from deriving the actual rating values and rated items. Different data perturbation methods are proposed to protect individual privacy. Due to different privacy concerns, users might disguise their data inconstantly to meet their own privacy concerns. Malicious data collectors might try to derive both original rating values and rated items.

The goal of this chapter is to show how to reconstruct the rated items with the help of auxiliary information when users inconsistently mask their confidential data in privacy-preserving collaborative systems. First of all, the number of the rated items need to be estimated. Then, the ones which are rated should be identified. To do so, existing methods are initially used to eliminate noise from the disguised data. After noise is eliminated, the auxiliary information is utilized to improve the reconstructions. Experiments with a real data set show that the proposed approaches can derive the rated items with decent accuracy in spite of variable data masking.

4.1. Introduction

Randomization is one of the primary methods in PPCF to protect individual privacy. In the first RPT in PPCF (Polat and Du, 2003), users disguise their data by employing same parameters. However, some studies have shown that the randomization scheme may not keep individual privacy safe as desired (Kargupta et al., 2003; Huang, Du and Chen, 2005; Zhang, Ford and Makedon, 2006a). To improve the level of privacy, Polat and Du (2007) introduce a new data disguising method which allows users to disguise their data inconsistently. The first proposed randomization scheme may not sufficiently preserve the individual privacy because this scheme only hides the real values of rated items from the server or CF system. However, the information of whether an item is rated or not is also crucial for users. Users may not want to reveal items they rate. To alleviate privacy concerns, inconsistent data disguising method is proposed (Polat and Du, 2007). According to the proposed scheme, each user disguises his rated items along with some unrated items to protect own original data. Different scenarios are presented how unrated items are filled based on users' demands.

In this chapter, two problems are to be faced to reconstruct the rated items from inconsistently perturbed data. The first problem is to estimate the number of rated items because users fill some unrated items with the original rated items. Two formulas are proposed to estimate the number of the real items from inconsistently perturbed data based on the selected scenario. Then, the rated items are identified using some of the matrix factorization methods. Although matrix factorization methods are usually used for prediction in PPCF, in this case, the effects of the noise data are decreased with the help of matrix factorization methods. Besides, the joint effect of auxiliary information and CF system's properties are investigated in addition to existing noise elimination methods.

4.2. Noise Elimination Methods

The first step to identify rated items is to filter out the noise associated with the original data. Since noise is added to the original z-scores, some of the originally unrated items become rated in the perturbed data. These rating values residing in unrated cells are the noise introduced by the randomization process. Thus, unrated items can be differentiated from rated items after the noise elimination because the estimated value for unrated items approaches to 0. While unrated items lack prior z-scores before the randomization process, rated items have z-scores. Therefore, the estimated cells for the rated items are expected to have relatively greater absolute values than the unrated ones after the noise elimination. As a result, greater absolute values will assist in determining the rated cells.

Four different noise elimination methods are explained briefly in this subsection. The aim of the selected methods is to eliminate noise from the disguised data. Note that random numbers are added to all rated and some unrated items. If the noise is removed, the filtered data can be used to predict the real rated items.

4.2.1. Singular value decomposition

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

$$A_{nxm} = U_{nxn} x S_{nxm} x V^T_{mxm} \tag{4.1}$$

SVD theorem has an interesting and useful property. SVD allows users to transform the data into low-rank (or reduced) matrix. The dimensionality reduction approach in SVD is important for noise reduction. When all singular values are sorted in descending order, small singular values represent noise. If the small singular values are discarded, the top singular values are used to reconstruct the original data. SVD is a conventional method in PPCF. Also, there are studies to reconstruct the original data from the disguised one (Zhang, Ford and Makedon, 2006; Guo, Wu and Li, 2006; 2008).

4.2.2. Discrete cosine transformation

Discrete cosine transformation (DCT) is usually used for Wiener filtering, pattern recognition, video and image compression. The purpose of the DCT is to transform data to a frequency domain so that the redundancy can be removed (Ahmed, Natarajan and Rao, 1974). DCT provides a transformation like SVD and PCA. The basic approach behind the DCT is to transform correlated data into the discrete (uncorrelated) cosine transform coefficients. In other words, the purpose is to define low and high-frequency coefficients because low-frequency coefficients are more significance than high-frequency ones. In the experiment, DCT is used for noise reduction from the disguised data. High-frequency coefficients' values are accepted as noise and ignored when the data is reconstructed using the inverse DCT function. Since the data (m×n) is two-dimensional, there are m×n coefficients, so 2D-DCT is applied in the experiment.

4.2.3. Principal component analysis

PCA has many applications in pattern recognition, image compression, neuroscience, economics, finance, geology, genetics, meteorology, etc. The basic approach is to transform a large number of correlated attributes into a small number of uncorrelated attributes so that predictions or operations in data can be easily done (Huang, Du and Chen, 2006). In order to apply PCA, the covariance matrix must be computed first. Assume that the original data set is A, the covariance matrix is calculated as $A^T A$. The eigenvalues (Λ) and eigenvectors (U and V^T) are calculated from the covariance matrix after computing covariance matrix, PCA is defined as follows:

$$A^T A = U x \Lambda x V^T \tag{4.2}$$

PCA is similar to SVD. The primary approach in PCA is the dimensionality reduction as in SVD. While top singular values are chosen in SVD, top uncorrelated principal components are selected in PCA. To show the relationship between two methods, Eq. 4.1 is substituted for A, in Eq. 4.2, as follows (Johnson, 2010):

$$\begin{aligned} A^T A &= (USV^T)^T (USV^T) \\ A^T A &= VSU^T USV^T \\ A^T A &= VS^2 V^T \end{aligned} \tag{4.3}$$

Eq. 4.3 is like that is shown in Eq. 4.2. Since the singular values can be calculated by taking the square roots of the eigenvalues of matrix $A^T A$, the eigenvalues can be represented as the squares of the singular values of A, in Eq. 4.3.

4.2.4. QR factorization

QR factorization (or QR decomposition) is one of the most important matrix decomposition methods. QR decomposes a matrix A into two matrices called Q and R. $A = QR$, where Q is an orthogonal matrix and R is an upper triangular matrix. The main idea of QR factorization is the calculation of the eigenvalues. There are different methods to compute QR factorization. Two of the most popular methods are Graham-Schmidt process and Householder transformation. The purpose of the QR is similar to SVD or PCA. Briefly, all used data reconstruction methods in this chapter transform the data into a reduced matrix to eliminate noise.

4.3. Filling the Missing Entries

Data reconstruction methods given in this section can be utilized only if data does not contain any unrated items. Thus, all unrated cells should be filled with appropriate default values to obtain complete data. Three methods are available to fill unrated cells. The first method is to fill with zero ratings, as the name implies, all unrated cells are filled with zero. This way of filling unrated cells is convenient for the reconstruction because filled items are already unrated. The second approach is to fill empty cells with corresponding user averages. According to the data disguising method, users convert their ratings into z-score values and add random numbers to the rated and some unrated cells. Hence, the server first estimates the user averages from the disguised data. Bilge and Polat (2010) show how to guess such values from the disguised data to obtain a denser data. The average z-scores for each user can be estimated from the disguised data as follows:

$$\bar{z}'_u = \frac{\sum_{j=1}^{R_u} z'_{uj}}{R_u} = \frac{\sum_{j=1}^{R_u}\left(z_{uj} + r_{uj}\right)}{R_u} = \frac{\sum_{j=1}^{R_u} z_{uj}}{R_u} + \frac{\sum_{j=1}^{R_u} r_{uj}}{R_u} \approx \frac{\sum_{j=1}^{R_u} z_{uj}}{R_u} \tag{4.4}$$

As the data set contains the disguised z-scores instead of the real ratings, the average of z-scores for each user can be calculated roughly using Eq. 4.4. The average of items is also preferred to fill all empty cells as the third approach. In this case, the server tries to guess the item averages from the disguised data. Due to the random data distribution with μ being zero, it is expected that the average of random numbers approximates zero as in the case of the average user estimation. In that case, Eq. 4.5 can be used to estimate the averages of each item's z-scores as follows:

$$\bar{z}'_i = \frac{\sum_{j=1}^{R_i} z'_{ij}}{R_i} = \frac{\sum_{j=1}^{R_i}\left(z_{ij} + r_{ij}\right)}{R_i} = \frac{\sum_{j=1}^{R_i} z_{ij}}{R_i} + \frac{\sum_{j=1}^{R_i} r_{ij}}{R_i} \approx \frac{\sum_{j=1}^{R_i} z_{ij}}{R_i} \tag{4.5}$$

4.4. Estimating Number of Real Ratings from Inconsistently Perturbed Data

It is prominent to compute the number of real rated items from the perturbed data before marking those real rated items. After the random filling process, the number of rated items increases because some unrated items are filled as discussed in the previous part. However, the data holder will not tell whether a rated item in the perturbed data set is indeed rated or unrated in the original data set because each user disguises his ratings and unrated items with random numbers. It is still unknown how many cells need to be picked as rated cells. Hence, the process of identifying rated items in the original data starts with estimating the number of items rated by users. First, the data holder must predict the number of the real ratings in the disguised data before estimating which one is genuinely rated.

Two formulas are defined to estimate the number of the real ratings based on the how β is selected. Users use either the fixed or random β from the range (0, β_{max}] while filling unrated cells. When each user employs the fixed β to fill unrated cells with the random numbers, the server can estimate the number of rated items very close to the original number of rated items. The server knows the β and the total rated items from the disguised data. The number of rated items for each user can be easily estimated using the following equation, Eq. 4.6. *nItems* depicts the number of total items. It is known publicly. *nRated* denotes the total number of the rated items, which can be calculated from the disguised data. β is already known by the server. $nRated_{items}$ depicts the number of true rated items in the original data. $nRated_{items}$ is estimated for each user, as follows.

$$nRated_{items} + (nItems - nRated_{items}) * \beta = nRated \tag{4.6}$$

After rearranging the given formula, Eq. 4.7 is obtained. The result is rounded to find an integer number. Eq. 4.7 shows that the server can estimate the number of rated items for each user although users disguise their rated and some unrated items.

$$nRated_{items} \cong round\left(\frac{nRated - (nItems * \beta)}{1 - \beta}\right) \tag{4.7}$$

Users can also employ different β values (from 0 to β_{max}) to fill unrated cells with random numbers. In this case, Eq. 4.6. and 4.7. should be modified to estimate $nRated_{users}$. Each user determines the value of β independently; therefore, the server does not know the value set by each user. It only knows the range (0, β_{max}], where β can be selected. Since the server does not know individual β values, it is proposed that the server use the expected value of β (β_{exp}). Eq. 4.8. displays how the total number of users rating an item, known by the server, in the perturbed data set by exploting β_{exp}. Notice that β_{exp} is an estimation in this equation. *nUsers* depicts the total number of users in the data set and is known publicly. *nRated* is calculated from the disguised data and depicts how many users vote the item. Real and fake users who vote the item are constituted the value of *nRated*. The value of β_{exp} is calculated as β/2. The purpose is to estimate how many users truly rate the item, $nRated_{users}$.

$$nRated_{users} + (nUsers - nRated_{users}) * \beta_{exp} = nRated \tag{4.8}$$

After rearranging the Eq. 4.8 the following Eq. 4.9 is attained:

$$nRated_{users} \cong round\left(\frac{nRated - \left(nUsers * \beta_{exp}\right)}{1 - \beta_{exp}}\right) \tag{4.9}$$

4.5. Utilizing Auxiliary Information

The concentration in this chapter is to derive the rated items from the disguised data; however, utilizing some auxiliary information can help a malicious server improve the accuracy while recovering real rated items. The fundamental approach, given in the next part, in the dissertation to reconstruct rated items is to benefit from the features of distribution types. Based on the distribution type, some data lie out the range, and those help attackers reconstruct the rated items. Since the data set is a movie-related, another auxiliary information except popular and award-winning movies from Internet Movie Database (www.imdb.com) is the *total number of votes*. This auxiliary information guides attackers to identify the real rated items from the disguised data. Most of the CF system contain user-related information in addition to ratings. The data set used in this dissertation contains user-related demographic information as well and used as auxiliary information. The relationship between user demographic information and items assist a malicious server in reconstructing the rated items.

4.5.1. Features of random data distributions

Uniform and Gaussian distribution have unique characteristics. When Gaussian distribution is used, it is known that 99.7% of data fall within the range of $[\mu - 3\sigma, \mu + 3\sigma]$. On the other hand, if the distribution type is chosen as uniform, all data lie within the range of $[\mu - \sqrt{3}\sigma, \mu + \sqrt{3}\sigma]$. Each user fills selected unrated items with his mean based on the chosen data distribution type. After transforming the original ratings to z-score values, each selected unrated item’s z-score value is calculated as 0. Therefore, if the random numbers are eliminated based on the distribution type, points, which lie out of the range, are considered as rated items. Zhang, Ford and Makedon (2006a) used this idea to identify the originally rated items when data set is complete, in other words all items filled with random numbers. This approach alone is not efficient enough to discover rated items because it fails to notice many rated items as pointed out by the authors. Another problem with this approach is that its reconstruction effect diminishes when σ gets larger. This approach is used together with auxiliary information with main data reconstruction method in the experiments. If values lie out of the predefined range, those are chosen as rated items.

4.5.2. Total number of votes

The focus of this chapter is to discover the rated items in the original data when users perturb their rated and unrated items. The hypothesis is that auxiliary information could be beneficial to improve the reconstruction accuracy when utilized with data reconstruction methods to identify rated items. As stated before, the auxiliary information is collected from the Internet Movie Database (IMDb), which is a famous movie website and has millions of registered users all around the world. The information in IMDb is consistent and reliable. The IMDb offers different statistics about movies based on registered users' opinions. The IMDb serves not only the total number of votes of each movie by registered users but also average ratings, awards, popular and unpopular movies. The total number of votes about movies are considered as a prominent auxiliary information to discover which items might be rated when users disguise rated and some unrated items. Auxiliary information such as popularity, unpopularity or the average rating is not as important as the number of total votes for each movie because the objective is to identify which items might be rated. Such auxiliary information would be more meaningful than the total number of votes if the intention were to derive original rating values. For example, if the movie is rated by a predefined number of registered users, the movie can be chosen as a rated item even if a movie is rated with a lower rating. The lower rating only shows users do not like the movie. The values of ratings are not important in this case. The vital point is to make a distinction between originally rated items and fake items. Therefore, the total number of votes is a more distinctive parameter at this point. To recognize rated items, a threshold value of the total number of votes is defined in the experiments to choose rated items. If the number of total votes of a movie in the IMDb is greater than the threshold, this movie is accepted as rated by the users in the experiments. The purpose of the approach is to make a decision about which items are rated. The main idea is that the more users vote a movie in the IMDb, the likelier that movie is rated by users in the data set no matter what its rating is.

4.5.3. User demographic information

A CF system usually keeps user demographic information such as age, gender, occupation, addresses besides items' properties and ratings. This kind of information can sometimes be related to the ratings. For example, gender can be important demographic information to estimate which items might be enjoyed by users. Female users might be interested in female-specific items although male users might be fascinated by items female users do not like. Likewise, the age range is also prominent demographic information for CF systems. Since the movie-related data set is used in the experiments. It is expected that younger users usually like comedy or action movies while older users enjoy drama movies. The hypothesis is that user demographic information especially age range helps attackers reconstruct the rated items from the rated and fake items in the experiments. The ages are divided into five groups, which are 0-24, 25-34, 35-44, 45-54, and 55+ and a movie genre is defined to each group. The hypothesis is combined with the main method to test whether the approach is suitable to guess the rated items.

4.6. Complexity Analysis

Complexity analysis of the proposed solution is given by laying out the main steps of the reconstruction algorithm. The algorithm includes four main steps:

i. It starts with eliminating noise by widely-used methods (SVD/PCA/QR/2D-DCT). For a matrix of n rows (users) and m columns (items), the computational complexity of a full SVD is $O(m^2n)$. However, when top k singular values are selected instead of all to eliminate the noise, SVD matrix can be computed in $O(nmk)$. Likewise, the computational complexity of QR is approximately same as SVD. On the other hand, PCA requires more time than SVD because PCA first needs to compute covariance matrix taking $O(mn^2)$ and then apply the dimension reduction taking O(mnk). The total computational complexity of PCA is $O(mn^2+mnk)$. The computational complexity of 2D-DCT and inverse 2D-DCT is the same and is $O(m^3 + n^3)$.

ii. In the second step, the number of the real ratings is estimated from the disguised data for each user or item based on β selection ($nRated_{items}$/$nRated_{users}$). Estimating the number of real ratings from the disguised data is explained in detail in section 4.4.

iii. Sorting has been accomplished because it is claimed that larger values demonstrate the real rated items.

iv. In the last step, top $nRated_{items}$/$nRated_{users}$ items/users are selected from the sorted data. There is a small difference between total computational complexities of the proposed method based on β selection. When each user selects β in the same way, estimating the real rated number calculation is performed for each user. All items for each user should be sorted, which takes $O(mlogm)$. When β is selected randomly by each user, estimating the real rated number is performed for each item and the algorithm iterates among items. This time, the ratings of each item are sorted taking *O(nlogn)*. There are *m* items in the matrix, so the total time is *O(mnlogn)*. The complexity analysis is summarized in Table 4.1.

Table 4.1. *The computational complexity of the proposed solution*

The main steps of proposed solution	**Computational complexity**
Eliminating noise (SVD)	*O(nmk)*
# of the real ratings (*t*) are estimated based on *β* selection	*O(nm) or O(mn)*
Sorting the item or users	*O(nmlogm) or O(mnlogn)*
Selecting top *t* items or users from the sorted data	*O(nt) or O(mt)*
Total estimated	*O(nmk +n(m+mlogm+t))* *or* *O(nmk +m(n+nlogn+t))*

Complexity analysis of the proposed method with the auxiliary information should also be considered briefly. The computation complexity of the random data distribution is *O(nm)* because, for each user, all items are checked whether an item lies out of the predefined range or not. Total votes approach is analogous to the main method. In that case, after estimating the number of the real rated items for each user, total votes approach is applied instead of SVD decomposition. The summary of the computational complexity is given in Table 4.2.

Table 4.2. *The computational complexity of used auxiliary information*

The main steps of proposed approach	**Computational complexity**
# of the real ratings (*t*) are estimated	*O(nm)*
Taking IMDb data (*m*) based on rated items (assume *p*)	*O(npm)*
Sorting the items (*p*)	*O(nplogp)*
Selecting top *t* items from the sorted data	*O(nt)*
Total estimated	*O(n(m+pm+plogp+t)*

The computational complexity of the main method given in Table 4.1 and using total votes approach only given in Table 4.2 are similar. When the main method and auxiliary information are used together, their computational time is taken into consideration separately to compute total computation time.

4.7. Experiments

Experiments are conducted to show how much accuracy can be maintained by the proposed approaches to derive actual rated items from the disguised data. In the experiments, the auxiliary information is also utilized with data reconstruction methods to improve the accuracy results. Auxiliary information is a very strong tool to estimate the rated items. In some cases, the contribution of only auxiliary information is equal to the main data reconstruction method (such as the SVD-based method).

4.7.1. Data set and evaluation metric

The standard 100K MovieLens data set is used in the experiments. Precision and recall results of the experiments are measured. Note that precision is the ratio of the number of the correctly reconstructed rated items to the number of marked rated items. The recall is the ratio of the correctly reconstructed rated items to the number of rated items.

4.7.2. Methodology

Each experiment is conducted 100 times to obtain average precision and recall results. To generate random numbers, different σ values are used. 0.33, 0.67, 1, 2, and 3 are chosen as σ the values. Different σ values are used to show how results change when the σ values get larger. Also, experiments are conducted based on different β values. The density of MovieLens is 6%. A priori knowledge about MovieLens is used while choosing how many unrated cells are filled. If larger values are selected, the data set turns into random data set. In that case, predicting the real ratings from large random data might be illogical. Therefore, the value of β is limited to the density of the data set. The density, 6%, is depicted *d*. Different density values both less and greater than *d* are picked. The β values are set *d/4, d/2, d*, and *2d* to test how precision and recall results change based on different density values.

4.7.3. Experimental results

Experiment 1: Since MovieLens is a sparse data set, the first step is to investigate a suitable default value to fill all empty cells. As mentioned before, zero ratings, user average, and item average can be commonly used to obtain a denser data set. Different trials are carried out for each noise elimination method with varying σ and β. A small part of the results for each reconstruction method is shown in Figure 4.1 when the Gaussian distribution is selected. The recall results differ for various σ and β; however, the figure shows similar trends for different σ and β. It is clear in Figure 4.1. that filling unrated items with zero ratings rather than the user or item average gives the best results for different noise elimination methods. Because larger absolute ratings are considered as the rated items assigning zero ratings to all unrated cells is the best way to estimate the rated items from the disguised data. Therefore, one can conclude that filling all unrated cells with zero helps attackers ignore small absolute ratings, which indicate unrated and random ratings. Consequently, the approach is selected as the default value to fill in all unrated cells for the rest of the experiments.

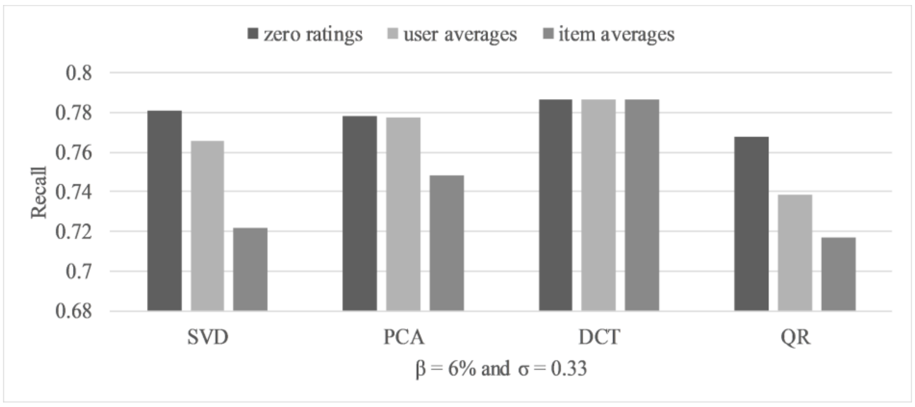


Figure 4.1. *Recall with different default values*

After determining which default value should be used to obtain a denser data set for better reconstruction results, the next objective is to find out the best noise elimination method among four different methods. The hypothesis is that the fake rated items might be considered as the noise. If the noise is eliminated, the rest can indicate the real rated items. However, it is clear that the noisy data cannot be eliminated completely; hence, the data, which is close to zero, is considered as the noise. To test the approach, the value of β is picked 6%. It means that all users randomly select 6% of their unrated items to fill random numbers. To generate random numbers based on the Gaussian distribution in Table 4.3, different σ values are selected. The number of dimensions/principals is set to 10 for SVD, PCA, and QR. A threshold value for 2D-DCT is defined to eliminate high-frequency coefficients, which indicate the noise in the experiment and set values less than the threshold to zero. The selected threshold value is 0.01. Then, data is reconstructed using the inverse 2D-DCT function. After reconstructing data using one of the four methods, the estimation of the real rated items process has begun. Since the number of the real rated items is predicted for each user, the greater ratings are marked as the rated one from the reconstructed data. The purpose is to discard the smaller data because the smaller data is considered as the noise. Table 4.3 shows recall and precision for each method. The best results are given in bold font for convenience for each row of the table. As seen from Table 4.3, SVD is the best method when compared with others for most of the cases. Only DCT can achieve to beat SVD when σ is 0.33. QR is the worst when σ is 0.33 and 0.67. It only manages to outperform DCT when σ is 1, 2 and 3. Although SVD beats the PCA for all σ values, the difference between these two elimination methods is marginal. Precision and recall consistently decline for all methods for increasing values of σ, which is anticipated. The more the noise is eliminated, the more the results are improved. Consequently, the results show that SVD outperforms all other noise elimination method in terms of precision and recall. After testing different methods with different σ values, SVD is selected as the best method among others. For following experiments, SVD is determined as the main method to predict the real rated items within the rated and fake items.

Table 4.3. *Recall and precision results for different reconstruction methods when β = 6%*

		SVD		**PCA**		**DCT**		**QR**	
	σ	**recall**	**precision**	**recall**	**precision**	**recall**	**precision**	**recall**	**precision**
Gaussian Dist. (*σ* is fixed)	0.33	0.78095	0.78507	0.77776	0.78187	**0.78649**	**0.79065**	0.71347	0.71723
	0.67	**0.76435**	**0.76838**	0.76037	0.76439	0.70394	0.70765	0.69299	0.69665
	1	**0.74776**	**0.75171**	0.74437	0.74830	0.66762	0.67115	0.68158	0.68518
	2	**0.72278**	**0.72660**	0.72021	0.72401	0.63353	0.63688	0.66275	0.66625
	3	**0.71603**	**0.71981**	0.71483	0.71861	0.62478	0.62808	0.65814	0.66162

Experiment 2: As mentioned earlier, different data disguising methods are proposed to protect individuals' privacy. Thus, it is first evaluated that all users select *β* values in the same way. In terms of the results of Experiment1, SVD is chosen to estimate the rated items from the disguised data. In this experiment, the results of SVD are investigated in terms of various σ values with different *β* values. Table 4.4 shows that when the σ is 1, how different *β* values affect the estimation of rated items as well as fake items. precR and recR demonstrate the precision and recall for the estimation of rated items, respectively. Likewise, precU and recU display the precision and recall for the estimation of unrated/fake items, respectively. As seen from Table 4.4, there are two different cases based on selecting the σ.

i. The first case is that the server defines σ and the distribution type and let the users know them. Each user employs the fixed σ value and the distribution type. Table

4.4 first shows the results when the uniform distribution is used and then illustrates the results while Gaussian distribution is chosen as the distribution type with the fixed σ. Their results are very close to each other. As the value of β increases, results of precR and recR for estimating the real rated items decreases. The simple reason is that β is associated with the number of unrated items that are filled and a small increase in β means that more unrated items are filled with the random numbers, which causes the reconstruction accuracy to decrease. On the other hand, results of precU and recU of predicting fake items are increased with larger β. The more randomness is added, the more accurate results are for predicting the fake items. It is obvious that recallR/precR and β are inversely correlated. On the contrary, recallU/precU and β are correlated. When β increases, accuracy for the fake items goes up.

Table 4.4. *Recall and precision results for SVD when σ is 1*

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

ii. In the second case (mix distribution), the server only defines σ_{max}, and each user randomly selects own σ. The distribution type is identified randomly. This case performs the best results in Table 4.4. For example, when β is 6%, the results (recallR) for uniform, Gaussian, and mix distribution are 0.74791, 0.74825, and 0.76751, respectively. Considering the results from the first case, the distribution type may not affect the reconstruction because their results are very close to each other. Therefore, the server does not need to know the distribution type. For example, recallR with β being 1.5% is 0.88686 and 0.88726 for uniform and

Gaussian, respectively However, σ values affect accuracy because each user randomly picks own σ values between (0, 1] independently as seen from the mix distribution in Table 4.4. The σ value is fixed at 1 at the beginning of the trials in uniform and Gaussian distribution; however, users randomly select σ values in the second case (mix). Thus, for mix distribution as seen from Table 4.4, not all users select σ with 1 as in the first case (uniform or Gaussian dist.). Some users might select smaller σ values. Thus, how the σ affects the results are examined in Table 4.5 after obtaining the results from Table 4.4. The results show that when the σ increases, accuracy decreases. Since a random σ is chosen by users between (0, σ_{max}] in the mix distribution, the accuracy is better in this case than the cases in uniform and Gaussian distributions where users utilize a fixed value of σ, which equals to σ_{max} in the mix distribution. For example, recallR for σ = 1 is 0.64908 for uniform distribution with β = 12% while recallR with the same β for mix distribution is 0. 67688 using randomly selected σ. Table 4.5 shows the detailed results with different σ values when the β is fixed.

Table 4.5. *Recall and precision results for SVD when β is 6%*

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

Experiment 3: This experiment is conducted to test the auxiliary information, the number of total votes for each item in MovieLens. The public information is retrieved

from the IMDb. In other words, IMDb provides auxiliary information for each item in MovieLens. The aim is to predict rated items from the disguised data set. The claim is that a movie with more number of total votes in the IMDb is more likely to be voted in MovieLens data set regardless of its rating. It means that even a movie with relatively lower rating could achieve to be rated by users in MovieLens. The value of ratings is not considered important while deciding whether it is rated or not. In this experiment, MovieLens's items are sorted in descending order based on public information, the total number of votes. For each user, the number of rated items, $nRated_{items}$, from the disguised data is estimated before picking top *nRateditems* items from the sorted data. As in Experiment 2, the results are first demonstrated in Table 4.6 when the σ is 1 with different β values. After comparing the results of Table 4.6 with Table 4.4, Table 4.6 achieves better results for each β values with the help of auxiliary information. Table 4.4 gives the best result for mix distribution with β being 1.5%. The recallR is 0.89416 in Table 4.4 while the result for the same parameters in Table 4.6 is 0.90496 when auxiliary information is used. The other remarkable fact is that the results are not relevant to distribution type and different σ values. While distribution type is changed in Table 4.6, all results remain almost similar.

Table 4.6. *Recall and precision results based on IMDb data when σ is 1*

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

Likewise, as seen from Figure 4.2, results are not affected by varying values of σ. Difference between them is unnoticeable slight. On the other hand, when β values increase in Table 4.6, recallR and precR decline because of the increased randomness. In other words, the number of fake rated items are increased with larger β values. This situation affects the reconstruction results to estimate correct rated items from the disguised data. As seen from Table 4.6, while recallR for β = 3% is 0.90481, recallR for β = 12% is 0.69066 for uniform distribution. When the same parameters are chosen in Table 4.4, recallR for β = 3% and β = 12% is 0.82689 and 0.64908, respectively. In terms of recallU and precU, the number of fake rated items increase in parallel with increasing β; therefore, the accuracy of truly predicted fake items increases. The results show that auxiliary information is very strong assumption to estimate the real rated items.

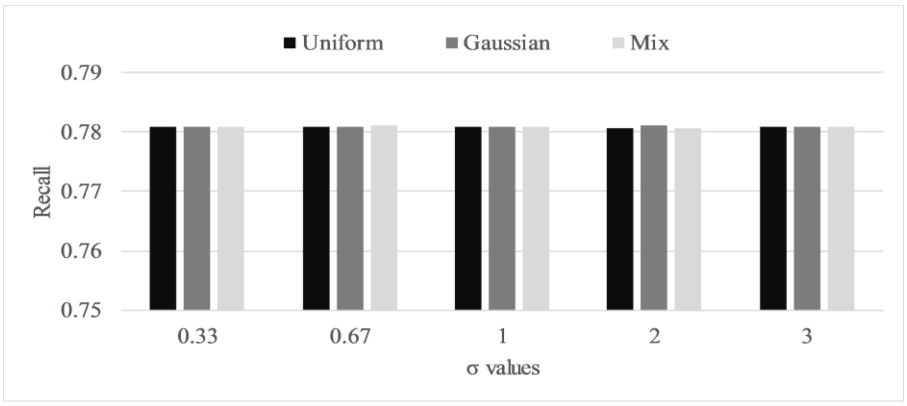


Figure 4.2. *Recall with β = 6% for utilizing IMDb data*

Experiment 4: The distribution type used provides another auxiliary information. Items that fall out of the range based on the distribution are expected to be rated. Precision and recall are evaluation criteria to evaluate how much this approach helps improve the results for each distribution type. Table 4.7 demonstrates the results. As expected, precision is 1 for the uniform distribution for each β value. However, the precision value changes for the Gaussian distribution when the β value goes up. With increasing β, the randomness also increases; thus, some unrated items might be selected as rated items. Consequently, the precision value decreases. Precision is important to understand how much of the items marked rated are indeed rated, but not enough to assess the approach. The recall results of Experiment 4 are very low when compared with Experiment 2 and Experiment 3. The remarkable point in this experiment for Gaussian distribution and the mix is that precision and β are inversely correlated. However, changing β does not affect the recall. The recall stays almost similar for each different β with the same σ. For example, Table 4.7 shows that the recall results in mix distribution are 0.12852 and 0.12868 for β = 1.5% and 12%, respectively.

Table 4.8 demonstrates the results for varying σ values and fixed *β*. Although different *β* values may not affect recall results with the same σ, changing σ values decrease recall results. As σ goes up, the range of random numbers added to the original data increases. As a result, the real rated items, which fall out of the range, become less. As seen from Table 4.8, recall values are 0.60348 for σ being 0.33 and 0.07791 for σ

Table 4.7. *Recall and precision results based on distribution type's feature when σ is 1*

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

Table 4.8. *Recall and precision results based on distribution type's feature when β is 6%*

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

being 3 in the uniform distribution, respectively. As mentioned at the beginning of this experiment, precision results for uniform distribution are always 1 regardless of what σ and β values are.

On the other hand, precision results decrease when the σ values go up for Gaussian and mix distribution. The reason why precision declines is that the randomness goes up when the σ increases. The chance of the rated items falling out of the range decreases due to the randomness. Thus, precision results are better for small σ values compared to the greater ones. For example, in Gaussian distribution, while the precision (precR) is 0.99346 when σ is 0.33, the precision is 0.65177 when σ is 3.

Experiment 5: After demonstrating how recall and precision results change with varying methods and auxiliary information, a new experiment is performed to show joint effects of such approaches. In the previous experiments, the results of each method are illustrated separately. The proposed approach in this experiment is to integrate auxiliary information with noise elimination methods. In Experiment 1, SVD is selected noise elimination method, which produces the best results. In Experiment 4, distribution type's feature is auxiliary information, and precision results are promising; however, recall is worse compared with Experiment 2 and Experiment 3. Therefore, using distribution type's feature alone is not enough to estimate rated items, but it might be helpful as auxiliary information when integrated with other methods. In this experiment, the process of reconstructing rated items is performed in three steps. First, the approach of the distribution type's feature is used as in Experiment 4. Ratings falling out of the range are accepted as the rated items. Second, the approach in Experiment 3 is applied on the reconstructed rated items from the first step. Remember that the approach in Experiment 3 utilizes the number of total votes as auxiliary information, and it demonstrates good results although auxiliary information alone is used to estimate the rated items. In this experiment, some constraints about auxiliary information used in Experiment 3 are defined. If a movie is rated by at least 100,000 users in IMDb, that movie is likely to be rated by MovieLens's users. If a movie is voted by less than 5,000 users in IMDb, that movie is ignored because it is supposed that the movie is unlikely to be rated by MovieLens's users. The discarded movie is never selected as a rated item. Third, SVD in Experiment 2 is applied only if the number of reconstructed rated items from the first and second steps is less than the number of estimated rated items.

As seen from Table 4.9, uniform distribution results are better than others. Compared with the previous experiments, this experiment gives better results for greater β values. The results for small β values are similar. For example, in uniform distribution when β is 12%, recall and precision results are 0.72563 and 0.73120 in this experiment while those results for the same parameters in Experiment 3 are 0.69066 and 0.69415, respectively. In Experiment 2, recall and precision results for the same parameters are 0.64908 and 0.65236. Integrating auxiliary information with SVD produces better results than utilizing these methods of reconstructions alone. The proposed approach using auxiliary information with SVD also gives better results for estimating the fake items, recallU and precU. For example, RecallU for the uniform distribution when β is 12% in Table 4.9, Table 4.6, and Table 4.4 are 0.85006, 0.82892, and 0.80560, respectively.

Table 4.9. *Recall and precision results with σ being 1 when auxiliary information and SVD is used together*

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

Figure 4.3 shows the results of different methods when the uniform distribution is used. Experiment 2, called “SVD”, is compared with Experiment 3, depicted “Total votes-IMDb”, and Experiment 5, defined in the figure as “Bounds + Total votes + SVD”, respectively. The value of β is 6%. In this case, various σ values are tested while fixing β value. The results show that auxiliary information makes a great contribution rather than using only SVD with increasing σ. For small σ values, the contribution is much more than greater values of σ. For greater σ values such as 2 or 3, only using the approach in Experiment 3 is enough rather than using all approaches together. The only reason is that

the effect of distribution type (as seen from Experiment 4) decreases when σ value gets greater.

Figure 4.4 displays similar results with Figure 4.3. The results for mix are slightly worse than the uniform distribution. However, the behavior of figures is the same. When the σ is larger, the gain decreases. Two figures demonstrate that either only auxiliary information or auxiliary information with SVD used to estimate real rated items is much better than only using SVD.

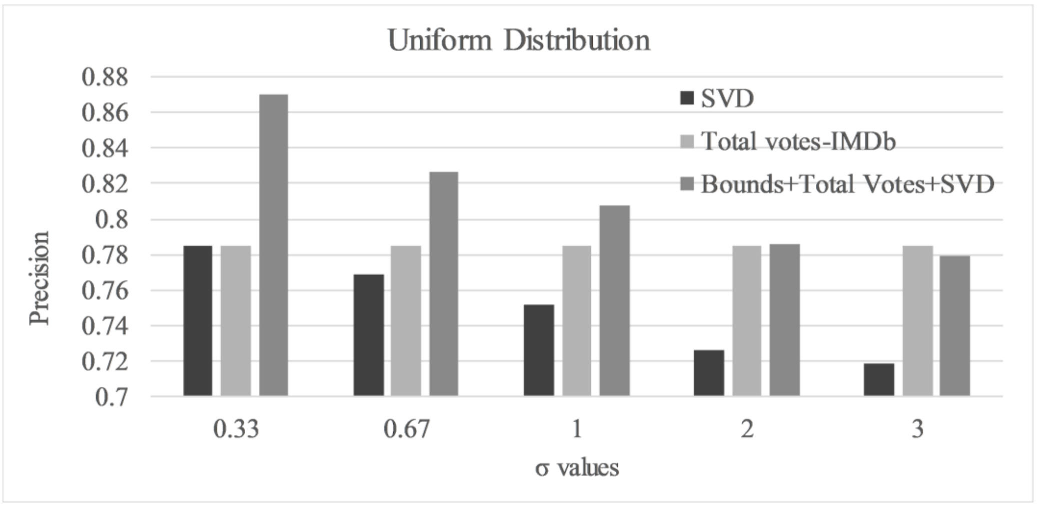


Figure 4.3. *Precision values for SVD, auxiliary information (total votes) and experiment 5 when uniform distribution with β=6% is used*

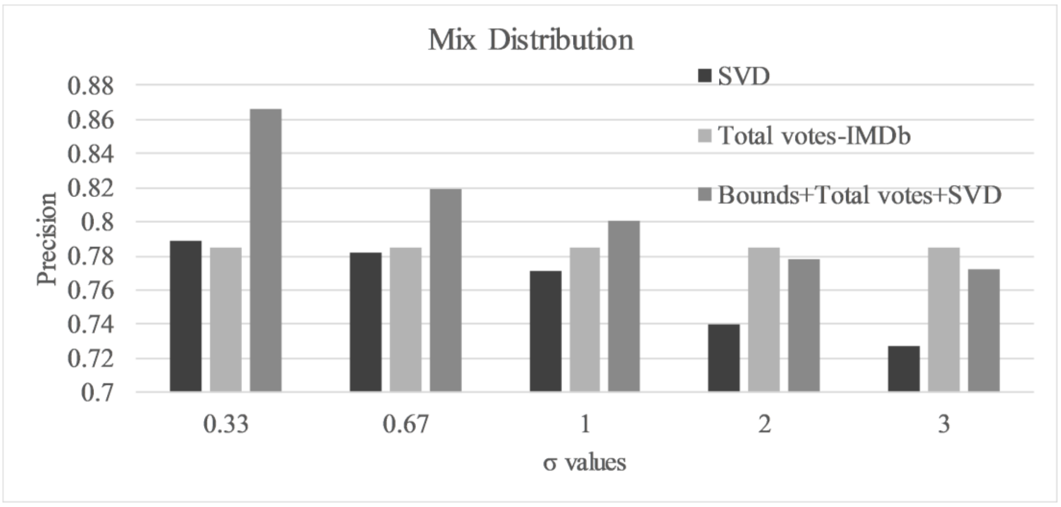


Figure 4.4. *Precision values for SVD, auxiliary information (total votes) and experiment 5 when mix distribution with β=6% is used*

Experiment 6: This experiment is conducted to display how much of real rated items can be reconstructed from the disguised data when each user selects the β value in a different way. In the previous experiments, the server lets users know the fixed β value, and all users perturb their data using the same β value. The other option is to let each user fill unrated items inconsistently because privacy concerns might differ. This experiment lets users pick β values from the range (0, β_{max}]. Reconstruction of the real rated items is more challenging than the previous experiments because each user independently picks the number of unrated items to fill based on selected β. However, each user fills unrated items with the fixed predefined β in the previous experiments. The experiment is similar to Experiment 2. There are two different cases selecting σ values as it is explained in Experiment 2. Results are given in Table 4.10 for varying values of β when σ is 1. Results regardless of the distribution type are very similar. For small β values, the results are promising. As β increases, the results have gradually decreased. As seen from Table 4.10, the distribution type is not a factor affecting the results. Mix distribution is slightly better than the others. This also proves that the distribution type is inert. The only reason why mix distribution is better than the others is that the σ is randomly selected by each user. The upper bound of σ is 1 when mix distribution is employed. Users picks a random value of σ from the range (0, 1]. The probability of selecting σ being 1 for every user is very low in mix distribution as it occurs in the cases of uniform or Gaussian distribution. Note that Experiment 2 shows that σ and reconstruction results are inversely correlated. When the β is 1.5% if this experiment (Experiment 6) is compared with Experiment 2, the best precision for mix distribution is 0.94191 and 0.89960 for Experiment 6 and 2,

Table 4.10. *Recall and precision results based on SVD with σ being 1 when the β is random*

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

respectively. The worst precision for mix distribution is obtained when the β is 12%, which are 0.77173 and 0.68030 for Experiment 6 and 2, respectively. The results show that SVD gives better results with the approach used in this experiment.

Figure 4.5 compares precision results for Experiment 6 and Experiment 2 when the β is set 6% and the σ varies. As explained at the beginning of Experiment 6, the β is selected by each user randomly. As seen from Figure 4.5, SVD gives better results in Experiment 6. The other important result is that when the σ gets larger, precisions are slightly worse although the fall of results in Experiment 2 is much more.

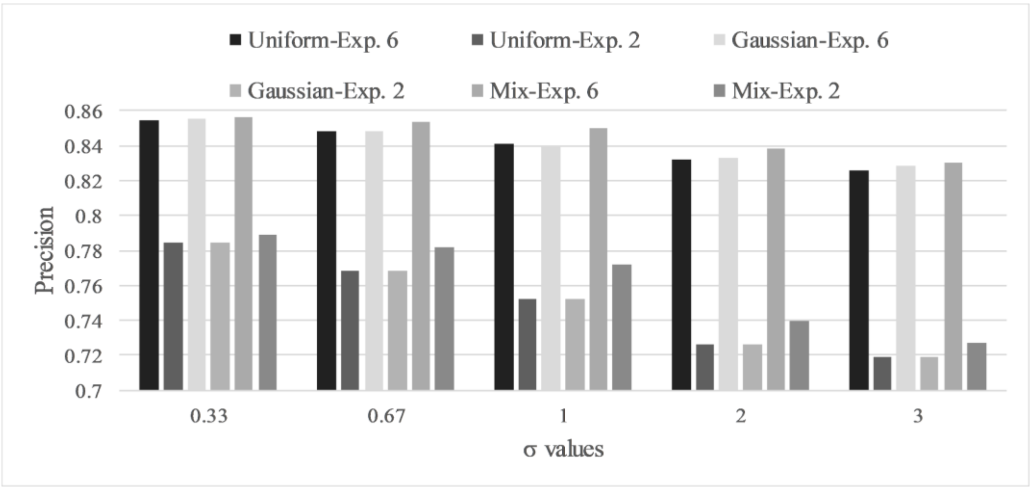


Figure 4.5. *Precision with fixed and random β for SVD*

Experiment 7: Auxiliary information is employed together with SVD to improve the results of Experiment 6. When β is random, Eq.4.9. is used to estimate the number of real rated items, and this equation requires β_{exp}. In this case, the problem is to find out the number of users rating an item. On the other hand, when the β is fixed as in Experiment 3 and Experiment 5, the problem is to find out the number of items rated by a user using the fixed β value. In those experiments, the number of total votes for each item is used to select rated items by users. The same approach cannot be applied in this experiment. The question in this experiment is to find out which users rate the specific item. In this case, user demographic information might help guess items. MovieLens data set contains users' age. Also, each item has movie genre. The hypothesis is that different age range like different movie genres. Assume that the ages up to 25 like comedy movies, ages from 25 to 34 like action and adventure movies, ages 35-44 enjoy Sci-fiction, ages 45-54 like a thriller, and ages 55 and older enjoy drama movies. In the experiment, SVD is first applied as in Experiment 6, and then auxiliary information is performed to improve the reconstruction results. For example, if a movie's genre is a comedy, then that movie is checked whether all users by the ages up to 25 are marked as rated by SVD. If there are users who are not marked as rated by SVD in that range, those users are marked rated and added to the reconstruction results because the user is assumed to rate the item based on his age group. After adding a new user, the last user marked in the SVD is removed from the reconstruction results. As seen from Figure 4.6, the assumption does not hold, as believed. Although some movies might have more than one genres, each movie in the experiment is assumed to have only one genre because SVD gives very good results as seen Experiment 6 compared with previous experiments. SVD has already proved itself to be very useful while reconstructing real rated items; therefore, it is not desirable to complicate the process with different movie genres to modify the promising reconstruction results of SVD.

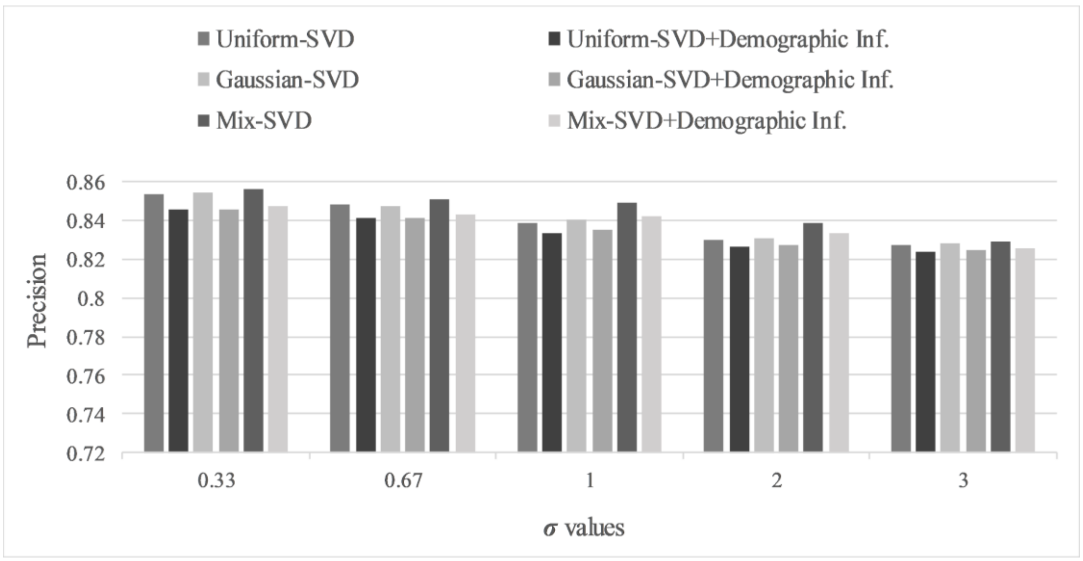


Figure 4.6. *Precision with β = 6% for SVD and demographic information with SVD*

A new experiment is conducted to show how results vary with different β values. As seen from Figure 4.7, for small β values, the proposed approaches utilized demographic information are close to SVD results. For example, when the β value is 3% for Gaussian distribution, the precision for SVD is 0.90102 while the precision for SVD with demographic information is 0.89870. However, the β value gets larger; the gap between the proposed approach and SVD expands as seen from Figure 4.7. For example, when the β value is set 12% for Gaussian distribution, the precision for SVD is 0.75889 while the precision for SVD with demographic information is 0.74693.

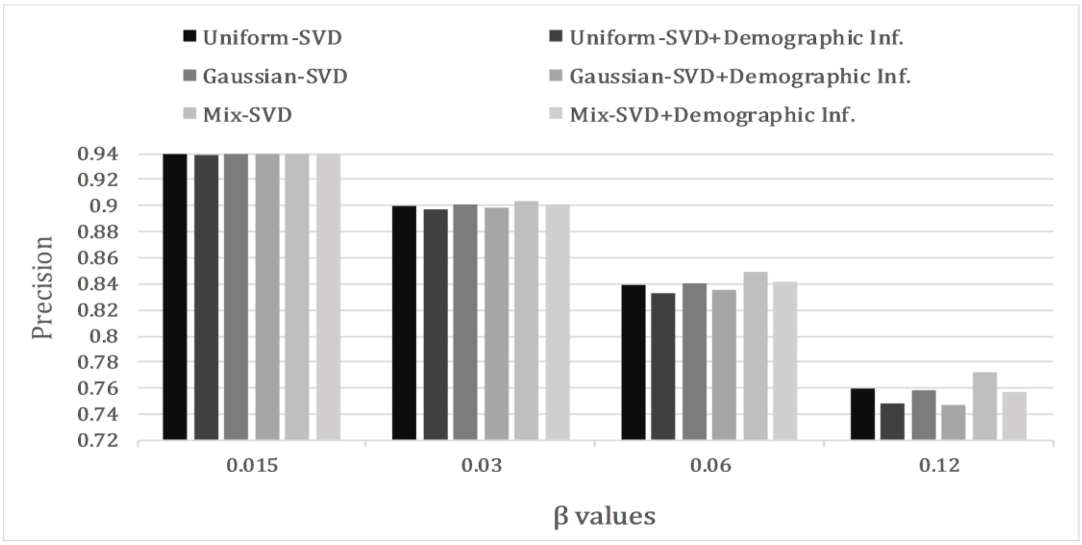


Figure 4.7. *Precision with σ = 1 for SVD and demographic information with SVD*

4.8. Conclusions

Users might have different concerns about privacy; therefore, different data disguising methods are proposed to meet their expectations. In the study targeted in this chapter, researchers propose that users disguise their data inconsistently to preserve their individual privacy. In this chapter, it is hypothesized that data disguising methods may not protect the privacy as much as believed. Even users disguise their private data inconsistently, the attacker server can guess the rated items. Experiments exhibit that it is possible to predict rated items by employing the proposed reconstruction methods with auxiliary information. A common property of the selected data reconstruction methods is to eliminate the noise. These reconstruction methods require that the data be complete; therefore, three different methods of filling unrated items are evaluated. Empirical results show that zero rating approach beats other two, user and item averages. Besides, different noise elimination methods used as data reconstruction methods are evaluated and SVD, one of the selected methods, is chosen as the main method in the experiments due to its success. For σ being 1, SVD has a 65-88% recall results correctly predicting the real rated cells when the fixed β is used by each user in the same way. When each user picks varying β independently, SVD has a 76-94% recall results for choosing the real rated cells. The experiments demonstrate that small σ values provide better results. As β values increase (the randomness increases), the results for predicting the real rated items decrease. On the other hand, the ratio of predicting fake items increases with increasing β. The other important outcome of the experimental results is that precision results are very close to recall results.

It is shown that auxiliary information plays an important role to improve the reconstruction results. The experiments display that some powerful auxiliary information makes a significant contribution to the results while some auxiliary information does not. For example, the contribution of IMDb data (total votes for each item) is tremendous for the reconstruction while the user demographic information approach used in the experiment cannot contribute to the results that much. The distribution type's feature also provides advantages. Although random filling is used to disguise the original data, values, which fall out the range based on the distribution type, are accepted as the rated items. It is demonstrated that precision and recall results are improved using auxiliary information with SVD. For small σ values, the contribution is much more because the effect of the distribution type's feature decreases while increasing σ values. The experiments show that identifying correct and useful auxiliary information is an important issue because all auxiliary information may not help attackers as much as thought.

5. ESTIMATING THE PRIVATE DATA IN DISTRIBUTED DATA-BASED PRIVACY-PRESERVING COLLABORATIVE FILTERING SYSTEM

Up to now, it is assumed that the entire data set is held by a central server. To provide accurate recommendations, CF systems require adequate data. However, companies, especially new ones, may not produce reliable recommendations due to insufficient data in some cases. Data held by a CF system might be horizontally or vertically distributed between two or more companies (parties). Different privacy-preserving collaborative filtering algorithms are proposed to protect parties' privacy when they collaborate with each other. Nevertheless, some type of privacy attacks can be performed against the proposed methods to derive the private data. In this chapter, how much private data can be estimated from the targeted PPCF methods is examined. For this reason, privacy attack scenarios are planned in terms of the number of parties joining a prediction process. Moreover, the contribution of the auxiliary information is studied. Empirical results show that a remarkable amount of private data can be derived based on the data partitioning type between two-parties.

5.1. Introduction

The data in a CF system may be distributed between two or more parties to deal with the problem caused by the sparse data. Different parties can come together and share the data they have to overcome this difficulty. The parties, especially new ones, need to collaborate with each other to offer better recommendations. Thus, the data the parties own is enriched and becomes more inclined to accurate recommendations. The data may be partitioned vertically or horizontally based on the parties' needs. In HPD, two parties will have different users' ratings for the same items. Since each party obtains more user ratings for their items, neighborhood selection and recommendations of CF algorithms will be positively affected by HPD. In the case of VPD, parties share different item sets for the same users. In this case, as the number of items for the same users increases, it can be possible to find more accurate similarities among users. Besides, the data in a CF might be distributed vertically or horizontally among multiple parties. The data different parties hold can be merged to accurate more reliable predictions as in the partitioned data. Although the data may be split between two parties, it may be divided into several parties.

On the other hand, parties must protect their data against each other. They may not be eager to share their data with the others because of privacy, financial and legal concerns. There are different algorithms proposed in distributed data-based PPCF systems to deal with all concerns. The aim of these algorithms is to hide real ratings and rated items of users as well as generate accurate recommendations. The existing algorithms protect the privacy of parties and their confidential data. There may be different attack scenarios to derive the private data, however; existing algorithms have not been analyzed in terms of estimating the private data from the disguised one.

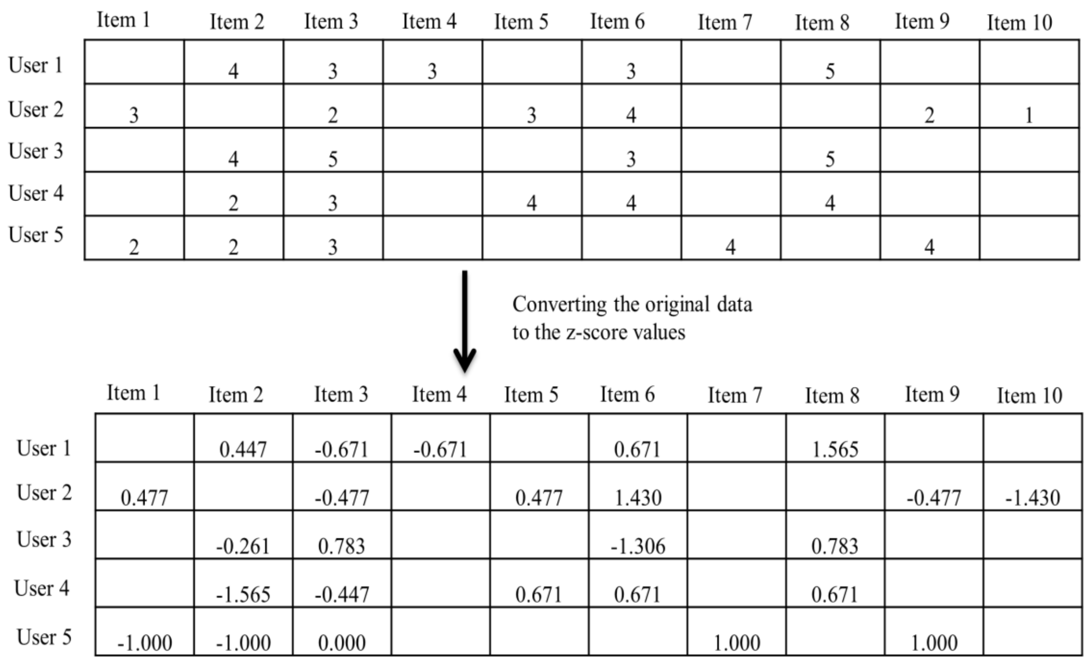
In this chapter, the problem of how much the private data can be derived when the numeric rating-based data is distributed vertically and horizontally among two or more parties is studied. Furthermore, the contribution of the auxiliary information is also tested to improve the accuracy rates. Experiments with the real data set demonstrate that it is likely to estimate a considerable amount of private data based on the partitioning type and the density of the data set.

5.2. Privacy Attack Scenarios on Partitioned Data

When the data in a CF system is partitioned between two parties, possible privacy attack scenarios have been examined. Privacy attack scenarios are divided into two parts according to the data partition. First, privacy attack scenarios on HPD are explained, then when the data is partitioned vertically, the kind of attacks that can be performed are scrutinized.

5.2.1. Privacy attack scenarios on horizontally partitioned data

The most common type of the privacy attack is to act as an active user. The attack of acting as an active user is performed as follows: A party which acts as an active user creates a fake rating vector and sends it to the other party to ask for a prediction for an item, q. The attacker party changes only one item's value in his rating vector and the attacker deduces the real rating value of that item. The purpose is to derive intermediate results from the other party by changing the value of one item at a time. Consequently, the difference between the first result and the result obtained by changing the value of an item in the rating vector ensures the value of that item to be deduced. The attacker party repeats this process until deriving information about all items. In terms of the privacy of the proposed scheme, Polat (2006) proposes to employ PSPCP (as explained in Section 2.2.2.1) to protect parties' privacy against this kind of attack. According to the protocol, the active user's rating vector is changed by the parties by adding fake items into the active user’s rating vector or removing some rated items from his rating vector. For a better understanding of this protocol, the protocol can be explained with an example. Both parties collect rating values of all items from their users and create user-item matrices. Then, these ratings are transformed into z-score values. Assume that party *B* has five users and ten items. After Party *B* receives the ratings from its five users, *B* converts all ratings to z-score values as is seen from Figure 5.1.



	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
User 1		4	3	3		3		5		
User 2	3		2		3	4			2	1
User 3		4	5			3		5		
User 4		2	3		4	4		4		
User 5	2	2	3				4		4	

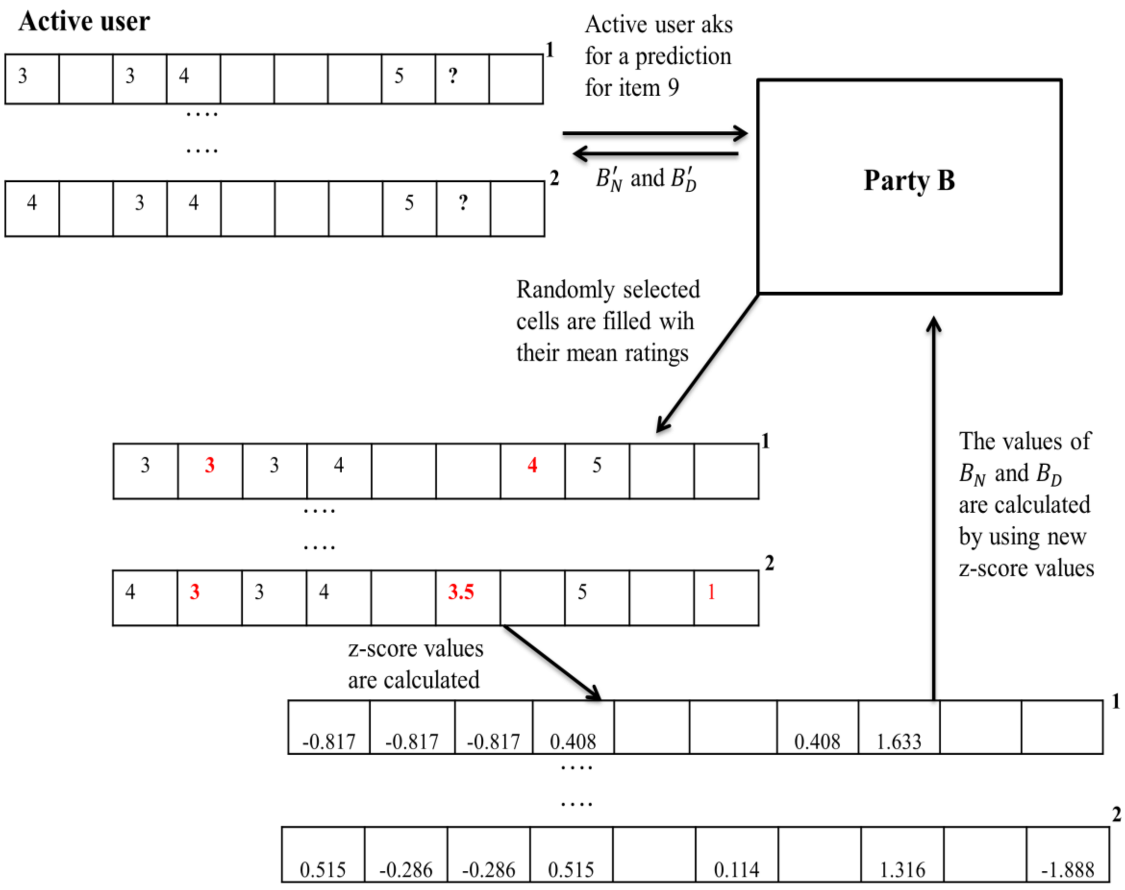
Converting the original data
to the z-score values

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
User 1		0.447	-0.671	-0.671		0.671		1.565		
User 2	0.477		-0.477		0.477	1.430			-0.477	-1.430
User 3		-0.261	0.783			-1.306		0.783		
User 4		-1.565	-0.447		0.671	0.671		0.671		
User 5	-1.000	-1.000	0.000				1.000		1.000	

Figure 5.1. *A sample of transforming the original matrix*

The protocol proposed by Polat (2006) is shown in an example in Figure 5.2. It is assumed that Party *A* wants to derive the data of Party *B*. In this example, it is shown that the active user would like to ask for a prediction for the same item (item 9) twice.At the request of the first prediction, the active user’s rating vector and the requested item (item 9 in the example) are sent to Party *B*. Party *B* selects some of the unrated items of the active user's rating vector randomly. Assume that the second and seventh cells are selected in this example. The selected cells of the active user’s rating vector are then filled with the averages of these items.

In Figure 5.2, the average of the second item is calculated as (4+4+2+2)/4 =3, and the average of the seventh item is 4/1 = 4. After that, *B* converts the active user’s rating vector into z-score values and calculates the values of B'_D and B'_N. Finally, *B* sends these values (B'_D and B'_N) to the active user. The active user requests a prediction for the same item again by simply changing the value of the first item. However, since B chooses different unrated cells every time to fill them with the average of the items (the second, the sixth, and the tenth items), the active user cannot derive a meaningful outcome from these intermediate results (B'_D and B'_N), even if he requests a prediction many times for the same item.



Active user
Active user aks for a prediction for item 9
B'_N and B'_D
Party B
Randomly selected cells are filled wih their mean ratings
The values of B_N and B_D are calculated by using new z-score values
z-score values are calculated

Figure 5.2. *An example of the targeted protocol*

Using the example, it can be demonstrated if the private data can be derived mathematically or not. Assume that Party A acts an active user and wants to derive information from Party B. B calculates B_N and B_D values by utilizing PSPCP and sends them back to A. Party A tries to deduce a result from the intermediate results. This situation is explained briefly in terms of B_D values. Eq. 5.1. shows how B_D is calculated as follows:

$$B_D = z_{a1}\sum_{i=1}^{n_B} z_{i1} + z_{a2}\sum_{i=1}^{n_B} z_{i2} + \cdots + z_{am}\sum_{i=1}^{n_B} z_{im} \tag{5.1}$$

If the PSPCP is not applied, Party *B* only uses the items which the active user voted to compute B_D. The active user obtains the results in Eq. 5.2.

$$B_{D_1} = z_{a1}\sum_{i=1}^{n_B} z_{i1} + z_{a3}\sum_{i=1}^{n_B} z_{i3} + z_{a4}\sum_{i=1}^{n_B} z_{i4} + z_{a8}\sum_{i=1}^{n_B} z_{i8} \tag{5.2}$$

However, since the PSPCP is applied, as shown in Figure 5.2, *B* randomly selects some items, the second and the seventh items in the example, and fills them with the mean of the corresponding item. Consequently, the result derived by the active user is shown in Eq. 5.3.

$$B'_{D_1} = z_{a1}\sum_{i=1}^{n_B} z_{i1} + z_{a2}\sum_{i=1}^{n_B} z_{i2} + z_{a3}\sum_{i=1}^{n_B} z_{i3} + z_{a4}\sum_{i=1}^{n_B} z_{i4} + z_{a7}\sum_{i=1}^{n_B} z_{i7} + z_{a8}\sum_{i=1}^{n_B} z_{i8} \tag{5.3}$$

Even if the active user changes the value of a single item of the same rating vector each time, the PSPCP applied by Polat (2006) prevents a disclosure from the total results because Party *B* selects different unrated cells and fills them with the selected items' mean each time. For example, the active user would like to ask for a prediction for the ninth item twice as seen from Figure 5.2, in that case, since *B* chooses different unrated cells (the second, the sixth, the tenth items) to apply the protocol, the value of B_D is calculated as in Eq. 5.4.

$$\begin{aligned} B'_{D_2} = {} & z_{a1}\sum_{i=1}^{n_B} z_{i1} + z_{a2}\sum_{i=1}^{n_B} z_{i2} + z_{a3}\sum_{i=1}^{n_B} z_{i3} + z_{a4}\sum_{i=1}^{n_B} z_{i4} + z_{a6}\sum_{i=1}^{n_B} z_{i6} + z_{a8}\sum_{i=1}^{n_B} z_{i8} \\ & + z_{a10}\sum_{i=1}^{n_B} z_{i10} \end{aligned} \tag{5.4}$$

The PSPCP ensures that the active user does not estimate anything even if he asks for a prediction for the same target item because *B* randomly selects some items that are not rated by the active user and fills them with the corresponding item's mean at every turn. The protocol protects *B* from the party acting as an active user to estimate information from the intermediate results. Even if the active user sends the same rating vector over and over, *B* chooses different unrated cells every time. Therefore, an inference cannot be made. While the proposed protocol increases privacy, filling the unrated cells may decrease the accuracy of the prediction. Since privacy and accuracy are conflicting goals, it is crucial to scrutinize how much accuracy is sacrificed for the sake of privacy improvements. On the other hand, the originality of the active user's ratings is also significant. When filling unrated cells, the original value of the active user's ratings is distorted, and consequently, accuracy is affected. Assuming that the accuracy rate is prominent, it is anticipated that no randomization has been added to the rating vector to which the active user has sent. In such a case, it is necessary to examine whether the active user can derive private information from the intermediate results. Since the parties use z-score values instead of the original rating values, it is supposed that the active user sends own z-score values to the parties rather than his real rating values. When the active user sends z-score values instead of his real ratings to Party *B*, *B* calculates the final result (B_{D_1}) from Eq. 5.2. using only items that the active user is rated, and sends it back to the active user. Since the active user converts his real values into z-score values, he tries to derive a result from Party *B* only by increasing or decreasing the z-score value of the related item without changing the z-score values of the rest of the items. The active user requests a prediction for the first item again by only changing the z-score value of the first item, he gets the result (B_{D_2}) in Eq. 5.5.

$$B_{D_2} = z_{a1'} \sum_{i=1}^{n_B} z_{i1} + z_{a3} \sum_{i=1}^{n_B} z_{i3} + z_{a4} \sum_{i=1}^{n_B} z_{i4} + z_{a8} \sum_{i=1}^{n_B} z_{i8} \tag{5.5}$$

Looking at the Eq. 5.5, it is seen that the multiplication of the items that the active user has not changed is the same as Eq. 5.2. If Eq. 5.2 is subtracted from Eq. 5.5, the difference can help the active user derive a result about the first item. Using the result of Eq. 5.2 and Eq. 5.5, the following equations, Eq. 5.6, is obtained.

$$\begin{aligned}B_{D_2} - B_{D_1} = \quad & z_{a1'} \sum_{i=1}^{n_B} z_{i1} + z_{a3} \sum_{i=1}^{n_B} z_{i3} + z_{a4} \sum_{i=1}^{n_B} z_{i4} + z_{a8} \sum_{i=1}^{n_B} z_{i8} \\ & - z_{a1} \sum_{i=1}^{n_B} z_{i1} - z_{a3} \sum_{i=1}^{n_B} z_{i3} - z_{a4} \sum_{i=1}^{n_B} z_{i4} - z_{a8} \sum_{i=1}^{n_B} z_{i8} \\ B_{D_2} - B_{D_1} = \; & z_{a1'} \sum_{i=1}^{n_B} z_{i1} - z_{a1} \sum_{i=1}^{n_B} z_{i1} \\ B_{D_2} - B_{D_1} = \; & \sum_{i=1}^{n_B} z_{i1} \left(z_{a1'} - z_{a1}\right)\end{aligned} \tag{5.6}$$

When the result of Eq. 5.6 is evaluated, the active user has the values of $B_{D_2}$and B_{D_1}. Besides, he can easily obtain the value of $\sum_{i=1}^{n_B} z_{i1}$ since the active user knows his z-score values. In other words, the active user may find the total value of the item of the users who rate the first item. However, the active user does not access the information about how many or which users rated the first item from the aggregate value.

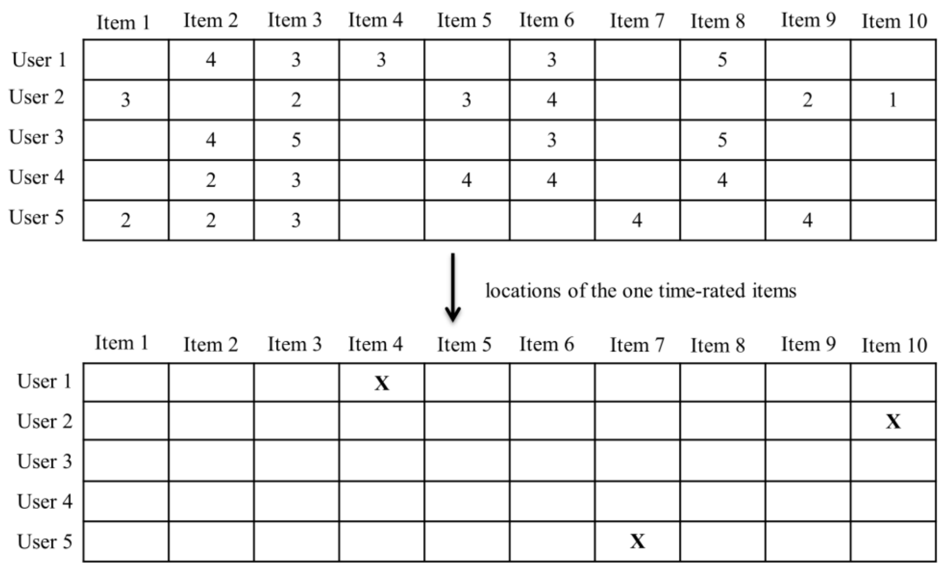
$$\sum_{i=1}^{k} z_{i1} = z_{11} + z_{21} + \cdots + z_{k1} \tag{5.7}$$

In Eq. 5.7, the active user cannot guess the number of k because he does not know the number of users who rate the first item. Even if he estimates the number of k, there is no way to guess which users rate that item. The active user only deduces from the results whether users voted the item or not. If the result is zero, it can be inferred that no user has never voted that item.

Polat and Du (2006) define two conditions for privacy. The first one is to hide the real values of items. When exchanging the data, parties cannot infer the real values from the data they receive. The other condition is to hide which items are rated. Parties cannot guess the list of the rated items. As shown in the previous example, users' real rating values and which items rated are not estimated because of the proposed protocol. If the second privacy condition is violated, and the information of which items are rated by users is assumed as auxiliary information, it is examined whether the private data of the users can be predicted or not.

Some information can be estimated if the locations of the items that are rated by only one user are known as auxiliary information. In Figure 5.3, it assumed that the fourth, seventh and tenth items are rated once, and their locations are known as auxiliary information. Then, the active user derives the value of these items from B_D values. It is then estimated the value of all rated items of these users (the first, the second and the fifth users) using the active user's rating vector. The important part in this attack is to change the value of one item at a time and determine whether that item is rated or unrated and the value of it if rated based on the B_N values.

Likewise, if the locations of the items that are rated by only two users are known as an auxiliary information, the values of these items are estimated using the data obtained from the intermediate results and the data derived before. For example, suppose that the



	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
User 1		4	3	3		3		5		
User 2	3		2		3	4			2	1
User 3		4	5			3		5		
User 4		2	3		4	4		4		
User 5	2	2	3				4		4	

locations of the one time-rated items

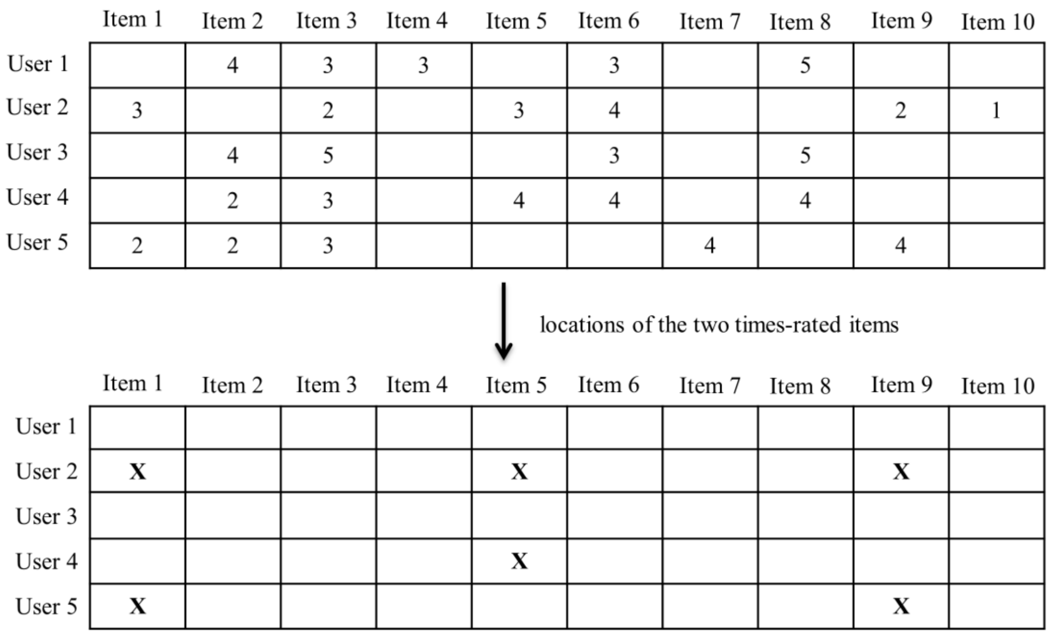
	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
User 1				X						
User 2										X
User 3										
User 4										
User 5							X			

Figure 5.3. *Auxiliary information: locations of the one-time-rated items*

fifth item is rated by two users (the second and fourth users) in Figure 5.4, and the figure depicts the locations of the items which are rated twice. Only the sums of the z-score values of these two users are derived from B_D values. It is impossible to predict the z-score values of these users from the total values. Since there is an equation and two unknowns, there exists infinitively many solutions. However, if the proposed assumption, where the locations of items are known as in Figure 5.3, is used to predict the real z-score values of the rated items, the value of the fifth item rated by the fourth user is easily estimated since the value of the fifth item which is rated by the second user is known. Applying the same privacy attack, if the value of the fifth item rated by the fourth user is estimated, all z-score values of the fourth user are simply derived without any other auxiliary information. As the amount of auxiliary information increases, the information derived also increases.

5.2.2. Privacy attack scenarios on vertically partitioned data

When the data is partitioned vertically between two parties, the active user asks for a prediction for q and sends his rating vector to the party which owns q. One of the parties can act as an active user and seek to derive the data of the other party. Assume that Party B owns q, and Party A acts as an active user. B computes the values of $A'_N + B_N$ and B_D, then sends them back to Party A according to the proposed scheme in Section 2.2.2.2. The final computation is done by Party A. Since Party B wants to protect his data from Party



	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
User 1		4	3	3		3		5		
User 2	3		2		3	4			2	1
User 3		4	5			3		5		
User 4		2	3		4	4		4		
User 5	2	2	3				4		4	

locations of the two times-rated items

	Item 1	Item 2	Item 3	Item 4	Item 5	Item 6	Item 7	Item 8	Item 9	Item 10
User 1										
User 2	**X**				**X**				**X**	
User 3										
User 4					**X**					
User 5	**X**								**X**	

Figure 5.4. *Auxiliary information: locations of the two times-rated items*

A, Party *B* applies the PSPCP as mentioned in HPD. With this protocol, Party *A* never estimates Party *B*'s private data because every time the active user asks for a new prediction, Party *B* adds/removes some randomly items to/from the active user's rating vector before computing the desired values ($A'_N + B'_N$, B'_D). As explained in Section 5.2.1, the addition or subtraction of different items from the active user's rating vector while producing a new prediction in each time ensures that the attacker cannot estimate the private data from the intermediate sums.

When the off-line computation is applied in VPD, two parties need to exchange some data due to owning the half of the items. Since the parties have half of the items, the values of the items in the other half are necessary for online calculations. In online computation process, Party *B* computes $A'_N + B'_N$. To calculate the value of A'_N, Party *B* needs to compute it during the off-line computation and stores into $\sum''A$. However, both the use of permutation function and HE to store the required data off-line prevent to disclose the private data during exchanging the data. Besides, parties add random numbers to the encrypted scalar product results ($\sum'A\ or\ \sum'B$) to ensure their data cannot be derived. While decrypting these results, the other party cannot derive the real values because of the inserted random numbers.

5.3. Privacy Attack Scenarios on Distributed Data

In distributed PPCF systems, parties may become targets of some privacy attacks when they merge their data. The other parties work together to derive the main party's data just as the MP can infer the private data from the other parties. In this section, these types of privacy attack scenarios are examined in the horizontal or vertical division of the data among *z* parties.

5.3.1. Privacy attack scenarios on horizontally distributed data

The attack of acting as an active user does not work on HDD because the active user only sends his data to the MP. Since the MP does not send the active user's data to the other parties, the MP cannot perform an attack by manipulating the active user's data. On the other hand, the MP may infer some information from the other parties' interim results while asking for a calculation for the targeted item. Besides, the MP may collect the interim results for some items from users to produce recommendations to a new active user who will be in the same cluster in the future. Kaleli and Polat (2012b) propose IPDKNN protocol to deal with similar privacy attacks as explained in Section 2.2.3.1. According to the protocol, each party randomly fill some cells of the users who did not vote *q* with non-personalized ratings. Moreover, each party replaces some of the z-score values with zero before calculating the required data. Therefore, each party must calculate the required aggregate data values whenever the active user asks for a prediction even if *q* and the cluster of the active user do not change. The authors ensure that the MP cannot deduce any information from the interim results. Whenever the MP requests aggregate values from the other parties for the same *q*, different interim results are received because of fake users, who did not actually rate *q* and the removal of some z-score values in users' rating vector.

On the other hand, if all parties except the MP coalesce against the MP, the private data may be estimated from the final prediction (Kaleli and Polat, 2012b). The authors argue that such a privacy attack is possible, but they do not elaborate how it can be accomplished, so the details of this attack remain unclear. Therefore, the question is if it is likely to estimate the MP's private data from the final prediction since all parties except the MP merge their aggregate data and work together? To answer this question, the final prediction formula is given in Eq. 5.8.

$$p_{aq} = P + \overline{v_a} \tag{5.8}$$

In fact, it is difficult to obtain information from the value of p_{aq}. Since the MP does not send the active user rating vector to the other parties, the average value of the active user rating vector ($\overline{v_a}$) is not calculated. However, if parties correlate among themselves, and one of the parties acts as an active user, the average value of the active user will be known in advance. In this case, the parties derive the value of P, ($P = p_{aq} - \overline{v_a}$). P is rewritten in Eq. 5.9 in terms of values held by parties. Since one of the parties acts as an active user, the value of P and z-score values of the active user ($\sum_{j=1}^{J} z_{aj}$) are known by parties. Besides, each party has his aggregate values ($\sum_{u=1}^{k_z} z_{uj}$ *and* $\sum_{u=1}^{k_z} z_{uj} v_{duq}$), the only unknown values belong to the MP in Eq. 5.9.

$$\overset{known}{\overset{\frown}{\mathrm{P}}} = \frac{\overbrace{\sum_{j=1}^{J} z_{aj}}^{known} \left[\overbrace{\sum_{u=1}^{k_1} z_{uj} v_{duq}}^{unknown} + \overbrace{\sum_{u=1}^{k_2} z_{uj} v_{duq}}^{known} + \cdots + \overbrace{\sum_{u=1}^{k_z} z_{uj} v_{duq}}^{known}\right]}{\underbrace{\sum_{j=1}^{J} z_{aj}}_{known} \left[\underbrace{\sum_{u=1}^{k_1} z_{uj}}_{unknown} + \underbrace{\sum_{u=1}^{k_2} z_{uj}}_{known} + \cdots + \underbrace{\sum_{u=1}^{k_z} z_{uj}}_{known}\right]} \tag{5.9}$$

Before trying to estimate unknown values in Eq. 5.9, the protocol proposed by Kaleli and Polat (2012b) is shown in an example in Figure 5.5 for a better understanding of it. Figure 5.5 illustrates how the targeted distributed PPCF protocol works briefly. After an active user sends his rating vector and q to the MP, the MP calculates active user's cluster and sends active user's cluster and q to the other parties. Final prediction is performed by the MP.

As seen from Figure 5.5, the active user rated the third, fourth and eighth items and asked for a prediction for the first item. Because z-score values of the rated items of the active user are known by the coalesced parties, P can be rewritten in terms of Figure 5.5 in Eq.5.10., as follows:

$$\mathrm{P} = \frac{z_{a3}\overbrace{\left(\sum_{u=1}^{k_1} z_{u3} v_{duq}\right)}^{n_1} + z_{a4}\overbrace{\left(\sum_{u=1}^{k_1} z_{u4} v_{duq}\right)}^{n_2} + z_{a8}\overbrace{\left(\sum_{u=1}^{k_1} z_{u8} v_{duq}\right)}^{n_3} + N}{z_{a3}\underbrace{\left(\sum_{u=1}^{k_1} z_{u3}\right)}_{d_1} + z_{a4}\underbrace{\left(\sum_{u=1}^{k_1} z_{u4}\right)}_{d_2} + z_{a8}\underbrace{\left(\sum_{u=1}^{k_1} z_{u8}\right)}_{d_3} + D} \tag{5.10}$$

where N is total values of numerator part of P ($\sum_{j=1}^{J} z_{aj} \left[\sum_{u=1}^{k_2} z_{uj} v_{duq}, \ldots, \sum_{u=1}^{k_Z} z_{uj} v_{duq}\right]$) and D is total values of denominator part of P held by the coalesced parties ($\sum_{j=1}^{J} z_{aj} \left[\sum_{u=1}^{k_2} z_{uj}, .., \sum_{u=1}^{k_z} z_{uj}\right]$). d_1, d_2, d_3 and n_1, n_2, n_3illustrate unknown values belonging to the MP.

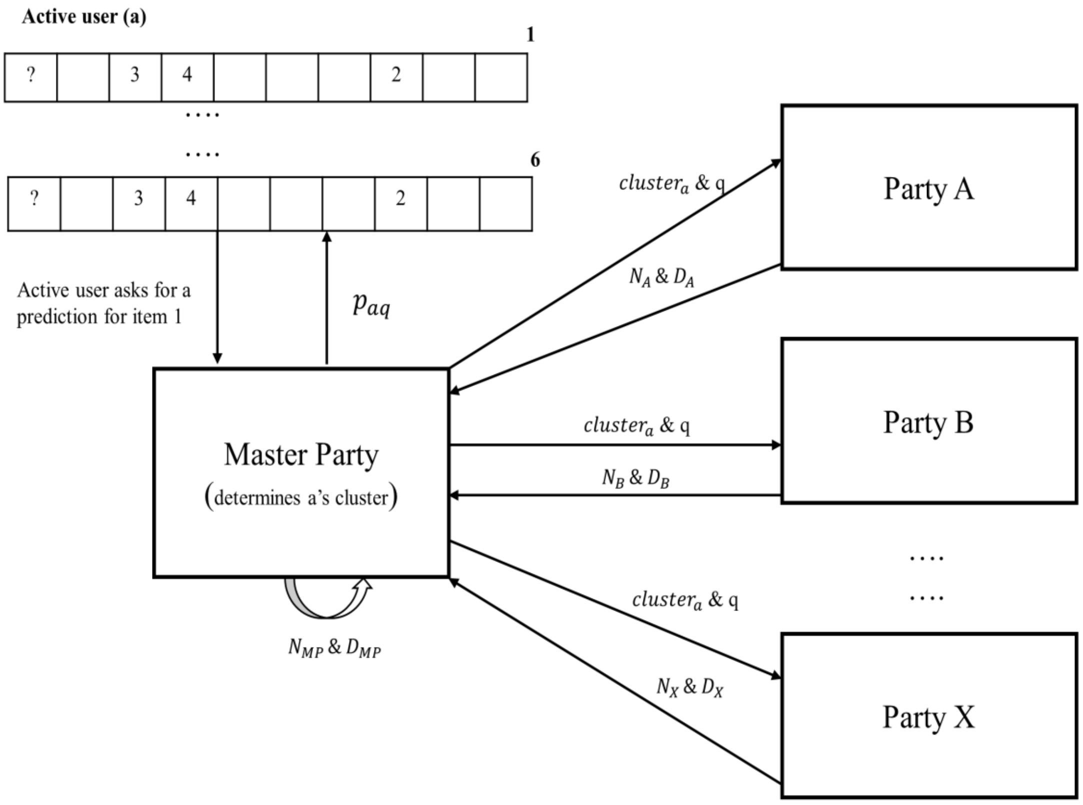


Figure 5.5. *An example of the targeted distributed PPCF protocol*

As seen from Eq. 5.10, there is one equation with six unknown parameters. To solve the problem, six equations are needed according to the active user's rating vector in Figure 5.5. If the active user asks for a prediction for the same item (item 1) six times with the same rating vector, six equations are obtained. Otherwise, if the active user changes any rating in his vector, all z-score values would change in Eq. 5.10, so there would be no way to solve the problem. Therefore, it is assumed that the active user sends the same rating vector in a repeated manner until the private unknowns of the MP is estimated. In each prediction process, although the values of *P*, *N*, and *D* change, there is no problem to derive the private unknowns because the coalesced parties know their disguised aggregate values ($\sum_{u=1}^{k_z} z_{uj} v_{duq}$ and $\sum_{u=1}^{k_z} z_{uj}$). After the active user receives the final prediction results, the partial results of the MP in Figure 5.5 can be derived by solving Eq. 5.11.

$$
\begin{aligned}
P_1 &= \frac{z_{a3}n_1 + z_{a4}n_2 + z_{a8}n_3 + N_1}{z_{a3}d_1 + z_{a4}d_2 + z_{a8}d_3 + D_1} \\
P_2 &= \frac{z_{a3}n_1 + z_{a4}n_2 + z_{a8}n_3 + N_2}{z_{a3}d_1 + z_{a4}d_2 + z_{a8}d_3 + D_2} \\
P_3 &= \frac{z_{a3}n_1 + z_{a4}n_2 + z_{a8}n_3 + N_3}{z_{a3}d_1 + z_{a4}d_2 + z_{a8}d_3 + D_3} \\
P_4 &= \frac{z_{a3}n_1 + z_{a4}n_2 + z_{a8}n_3 + N_4}{z_{a3}d_1 + z_{a4}d_2 + z_{a8}d_3 + D_4} \\
P_5 &= \frac{z_{a3}n_1 + z_{a4}n_2 + z_{a8}n_3 + N_5}{z_{a3}d_1 + z_{a4}d_2 + z_{a8}d_3 + D_5} \\
P_6 &= \frac{z_{a3}n_1 + z_{a4}n_2 + z_{a8}n_3 + N_6}{z_{a3}d_1 + z_{a4}d_2 + z_{a8}d_3 + D_6}
\end{aligned}
\tag{5.11}
$$

In Figure 5.5, the active user has only three rated items in his rating vector in order to show how the unknown values can be estimated by employing the proposed privacy attack. 6% of the data is filled in the real dataset, MovieLens. Therefore, an average active user rating vector is expected to have a density of 6%. As a result, approximately 100 items should be rated in the active user's rating vector. As seen from Eq. 5.10, two unknown parameters emerge for each item rated in the active user's vector. To solve the problem as explained in Eq. 5.11, at least 200 repeated prediction requests are required for 200 equations with 200 unknowns for a query of an active user with 100 rated items. Eq. 5.12 shows a general solution and lists the number of equations needed in terms of the number of rated items in an active user rating vector. Assume that the active user rates *J* items in his rating vector. The active user sends the same rating vector for only one target item to the MP at least *2×J* times. When the complexity of the proposed solution is considered, it seems that it is inapplicable.

$$
\begin{aligned}
P_1 &= \frac{z_{a1}n_1 + z_{a2}n_2 + z_{a3}n_3 + \ldots + z_{aJ}n_J + N_1}{z_{a1}d_1 + z_{a2}d_2 + z_{a3}d_3 + \cdots + z_{aJ}d_J + D_1} \\
P_2 &= \frac{z_{a1}n_1 + z_{a2}n_2 + z_{a3}n_3 + \ldots + z_{aJ}n_J + N_2}{z_{a1}d_1 + z_{a2}d_2 + z_{a3}d_3 + \cdots + z_{aJ}d_J + D_2} \\
&\cdots \\
&\cdots \\
P_{2J-1} &= \frac{z_{a1}n_1 + z_{a2}n_2 + z_{a3}n_3 + \ldots + z_{aJ}n_J + N_{2J-1}}{z_{a1}d_1 + z_{a2}d_2 + z_{a3}d_3 + \cdots + z_{aJ}d_J + D_{2J-1}} \\
P_{2J} &= \frac{z_{a1}n_1 + z_{a2}n_2 + z_{a3}n_3 + \ldots + z_{aJ}n_J + N_{2J}}{z_{a1}d_1 + z_{a2}d_2 + z_{a3}d_3 + \cdots + z_{aJ}d_J + D_{2J}}
\end{aligned}
\tag{5.12}
$$

As seen from Eq. 5.10 and Eq. 5.11, even if the values of unknowns are derived, the coalesced parties can only estimate aggregate values of the MP ($\sum_{u=1}^{k_1} z_{u3}$ or $\sum_{u=1}^{k_1} z_{u3} v_{duq}$). As explained in Eq. 5.7, the coalesced parties do not know how many users rate an item. Even if the parties know the number of users rated that item, they do not know which users rated that item. The proposed approach as in Figure 5.3 in the partitioned data can be used to estimate $\sum_{u=1}^{k_1} z_{u3}$. Assume that the target item on the active user's cluster is rated once and this auxiliary information is known publicly as in Figure 5.3. In such a case, after the equations are solved and unknown values are estimated, their z-score values are derived only if one user votes them. Since only one user rates those items, rated items and their z-score, values are estimated.

5.3.2. Privacy attack scenarios on vertically distributed data

Since the data is vertically distributed among z parties, each party owns a part of items and needs to exchange some required data with each other off-line to improve the performance of online prediction process. Parties can compute their item averages privately because they only use their own ratings while computing item averages. However, each party needs the other parties' data to calculate user averages due to vertical distribution. While exchanging the required data, parties may derive confidential data. For example, if the partial result of a user is zero, it means that the user does not rate any item or if the user only rates one item, the user's real value is derived from the partial data. Kaleli and Polat (2012) offer parties to disguise their data before computing user averages to overcome these shortcomings. Since parties add fake ratings into users' vector, the real values and the number of rated items of users cannot be predictable. Even if someone manages to guess the location of the real rated items, only the aggerate data of the rated items can be obtained. There is no way to derive the real value of any item from the received aggregate data. The parties also change the partial data while computing vector lengths of users. Due to the randomness, the parties fill fake ratings into their original data; no one can derive the confidential data from the received vector lengths.

In order to enhance the performance of online computation, P_N and P_D values (as seen from Eq. 2.8) are stored by all parties off-line. Parties have to exchange the partial results while computing P_N values. Since the MP has only values of q, it has to send these values to the other parties in order that they compute their P_N values. However, the MP encrypts the values of q so that the other parties cannot guess the private data. Parties cannot decrypt the received data because only the MP has the public key. On the other hand, the MP cannot deduce any information from the received permuted data even if it decrypts them. The MP obtains only the sums of each item to compute the value of $\sum_{u\in S} v''_{uj} v_{duq}$. This deduction is useless because the parties use the column permutation function to permute their items. Therefore, it is not likely that the MP can estimate the order of items. Even if the MP manages to estimate the order of the items, it cannot guess which users rate those items due to the row permutation function employed by parties. Parties compute P_D values without exchanging the private data except vector lengths of users. As explained before, it is not possible to derive the private data from the process of vector lengths, no one can deduce information from this computation.

In online computation, an active user sends his data to the MP and asks for a prediction for q. Since only the MP has the active user rating vector, it may manipulate the active user data to derive private data from the other parties. However, each party randomly fills some of the unrated cells whenever the active user asks for a new prediction; the MP cannot guess the faked filled cells. Besides, each party permutes their P_N and P_D values; the MP cannot estimate the private data after decrypting the received permuted values. On the other hand, the other parties can coalesce against the MP as in HDD. Assume that one of the parties is picked as an active user, and the selected party sends fake data to the MP to derive the MP's confidential data. All parties except the MP are assumed to know the active user's data. Although the MP inserts fake ratings into the active user's data to protect the active user's privacy, the coalescing parties have the information about the rated items of the active user. The coalescing parties try to derive the private data of the MP from the final prediction result. Eq. 2.8 can be rewritten with unknown parameters as in 5.13.:

$$\mathrm{P} = \frac{\sum_{j\in J_1} v''_{aj} \overbrace{\left[\sum_{u\in S} v''_{uj} v_{duq}\right]}^{unknown} + \sum_{j\in J_2} v''_{aj} \overbrace{\left[\sum_{u\in S} v''_{uj} v_{duq}\right]}^{known} + \cdots + \sum_{j\in J_z} v''_{aj} \overbrace{\left[\sum_{u\in S} v''_{uj} v_{duq}\right]}^{known}}{\sum_{j\in J_1} v''_{aj} \underbrace{\left[\sum_{u\in S} v''_{uj}\right]}_{unknown} + \sum_{j\in J_2} v''_{aj} \underbrace{\left[\sum_{u\in S} v''_{uj}\right]}_{known} + \cdots + \sum_{j\in J_z} v''_{aj} \underbrace{\left[\sum_{u\in S} v''_{uj}\right]}_{known}} \tag{5.13}$$

As seen from Eq. 5.13, the only unknown values pertain to the MP as in HDD. Like HDD, for each rated items of the active user, two unknown values are obtained. If the active user asks for a prediction for the same target item until the required equations are reached as in Eq. 5.12, unknown values may be estimated. Since items are divided between parties unlike as in HDD, the required equations to solve unknowns are partially smaller than the case in HDD. The coalescing parties can derive only $\sum_{u \in S} v''_{uj} v_{duq}$ and $\sum_{u \in S} v''_{uj}$ aggregate values after solving all equations with unknowns. The correlated parties do not obtain which users really rated q because of DPP. If the parties including the MP do not apply the proposed DPP protocol to disguise their data, the correlated parties can guess the users who rated q from $P_N P$. As mentioned in the steps of $P_N P$ in Section 2.2.3.2, the MP sends each value of q by encrypting to the corresponding party. Even if the correlated parties guess the users who rated q ignoring DPP, they never estimate the real values of users from the aggregate values. Besides, even the value of v''_{uj} would be derived somehow, it is not likely to estimate the real value (v_{uj}) because the correlated parties cannot estimate the item averages of the MP. Parties never exchange their item averages during the prediction process.

5.4. Experiments

How the private data can be derived when the partitioned systems are used is evaluated during the experiments. According to the partitioning type such as vertical or horizontal, the result can be different. Moreover, the auxiliary information is utilized to estimate the private data.

5.4.1. Data set and evaluation metric

The standard 100K MovieLens, a well-known movie data set, is used in the experiments. Accuracy rate is measured, which is the ratio of the number of the correctly estimated ratings to the number of total ratings.

5.4.2. Methodology

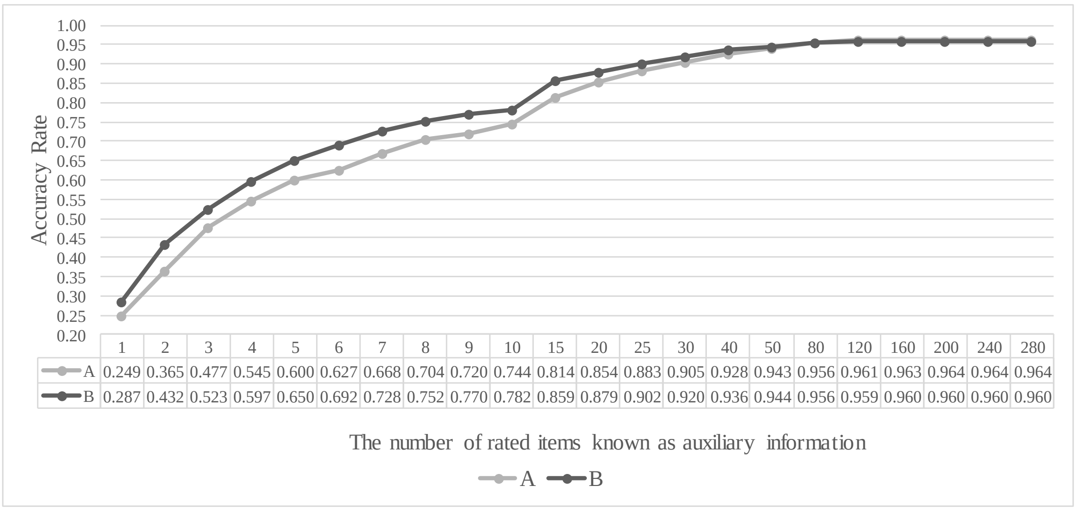
MovieLens data set is vertically and horizontally partitioned between two parties (*A* and *B*). In HPD, Party *A* has 471 users and 1,682 items, although Party *B* has 472 users and 1,682 items. In VPD, both parties contain 943 users and 841 items. The active user's rating vector is formed by the attacking party, which contains randomly 100 items with random ratings' value.

5.4.3. Experimental results

Experiments are divided into two sections. First, it is examined how much data the parties can derive when the data is partitioned horizontally. Then, experiments are conducted to show how the accuracy rates change when the data is partitioned vertically.

Experiment 1: The first experiment evaluates that how much data of Party *B* is estimated by Party *A* when the data is horizontally partitioned. Likewise, Party *B* aims to derive the data of Party *A* while acting as an active user. MovieLens is horizontally partitioned between two parties. One of the parties obtains the upper first half and the other holds the lower second half. While *A* holds 53219 ratings, *B* owns 46781 ratings. Note that the number of users is smaller than the number of items. As seen from the results in Figure 5.6, the accuracy rate of deriving the data of the Party *B* is higher than that of the Party *A* for the small number of rated items' locations known by the malicious party as auxiliary information.

As the number of locations of rated items known as auxiliary information in the experiment increases, the results are getting better and closer, and 96% of the data is estimated correctly for both parties. For example; if *B* acts in a malicious manner and knows the locations of 54 different items that are rated only once in *A*, 13251 ratings are estimated. This means 25% of data is correctly estimated. Likewise, while 165 items of Party *A* are rated once, and their locations are known as an auxiliary information, 25% of all the data of the Party *A*, equal to 13251 ratings, is estimated correctly. In fact, the same result can be derived when the locations of 54 items of Party A are known instead of the locations of 165 items. Once time-rated items are voted by the specific users, so knowing

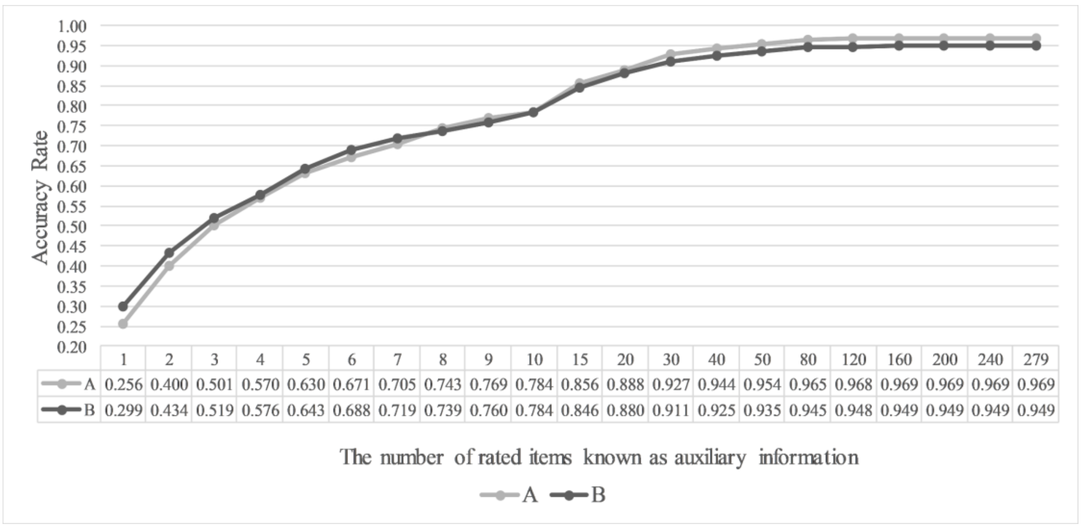


	1	2	3	4	5	6	7	8	9	10	15	20	25	30	40	50	80	120	160	200	240	280
A	0.249	0.365	0.477	0.545	0.600	0.627	0.668	0.704	0.720	0.744	0.814	0.854	0.883	0.905	0.928	0.943	0.956	0.961	0.963	0.964	0.964	0.964
B	0.287	0.432	0.523	0.597	0.650	0.692	0.728	0.752	0.770	0.782	0.859	0.879	0.902	0.920	0.936	0.944	0.956	0.959	0.960	0.960	0.960	0.960

Figure 5.6. *Accuracy on horizontally partitioned data*

a single location of the item of these users is enough to get the whole rating vector of those users. As a result, if the locations of 54 different items of the Party *A* which are rated once are known as an auxiliary information, 25 % of the data of Party *A* is approximately derived.

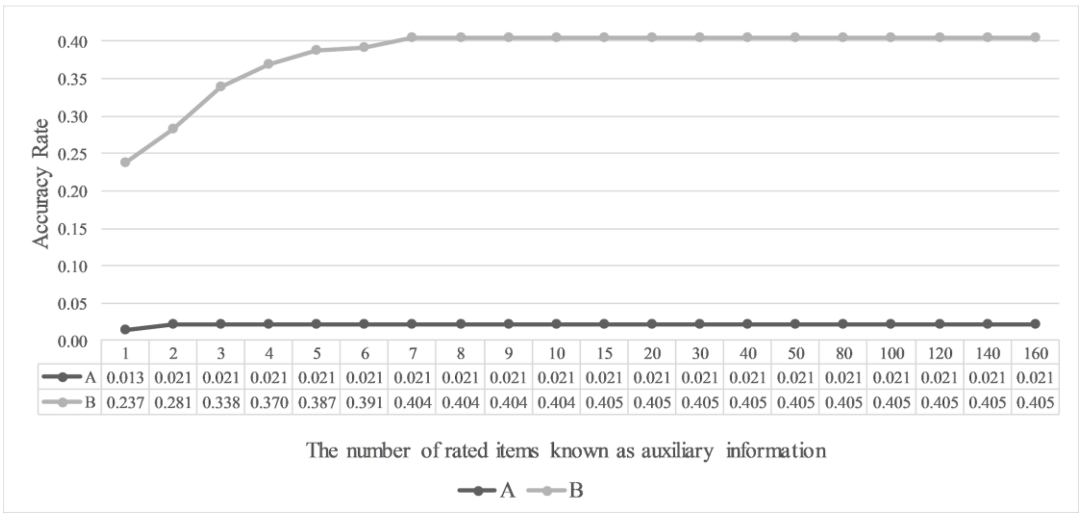
Experiment 2: The experiment analyzes how much private data is derived by the privacy attack when users are partitioned randomly among two parties instead of dividing the data directly into two parts. Both parties have roughly the same number of ratings. It is clear that the results in Figure 5.7 are similar to the results in Figure 5.6. Likewise, the party *B* gives slightly better results for the first values of the auxiliary information as in Experiment 1. The main reason for Figure 5.7 and Figure 5.6 to give the similar results is that the ratings of the parties are close to each other. In other words, since there are numerous items in the data set, the number of items is greater than the number of users in HPD, and it is expected that users may rate different items. During the privacy attack scenario, the proposed approach about the locations of items helps the attacking party derive the data as expected. Knowing the location of an item that is rated once helps the attacking party derive all data of the user even if the locations of the other rated items and their real values are not known. If anyone has such auxiliary information about an item that requires being rated by only a user, other rated items of that user are estimated, and their real values are derived.



	1	2	3	4	5	6	7	8	9	10	15	20	30	40	50	80	120	160	200	240	279
A	0.256	0.400	0.501	0.570	0.630	0.671	0.705	0.743	0.769	0.784	0.856	0.888	0.927	0.944	0.954	0.965	0.968	0.969	0.969	0.969	0.969
B	0.299	0.434	0.519	0.576	0.643	0.688	0.719	0.739	0.760	0.784	0.846	0.880	0.911	0.925	0.935	0.945	0.948	0.949	0.949	0.949	0.949

Figure 5.7. *Accuracy on random horizontally partitioned data*

Experiment 3: The experiment performs to figure out whether the parties can derive the private data from each other while MovieLens is vertically divided into two parties. One of them keeps the first half while the other keeps the second half. A has 86993 ratings although B owns 13007 ratings based on the partition. In the MovieLens data set, there are not many ratings through the last items. For this reason, when the data is partitioned vertically from the midpoint, the more ratings fall on the part of Party *A*. Unlike Experiment 1 and Experiment 2, the number of users is greater than the number of items at each party. As in HPD, the parties act as an active user and try to derive the data of the opposite party from the intermediate sums. As shown in Figure 5.8, even if the auxiliary information is utilized, the success of deriving the data of Party *A* is very low because the data of *A* is very dense, and it is difficult to apply the proposed approach. In other words, the approach does not work here because finding the location of the items rated only once is almost impossible. Since only the aggregated values of items are derived, the real ratings of the users cannot be estimated. On the other hand, as the data of Party *B* is relatively sparse, 40 % of its data can be estimated correctly.

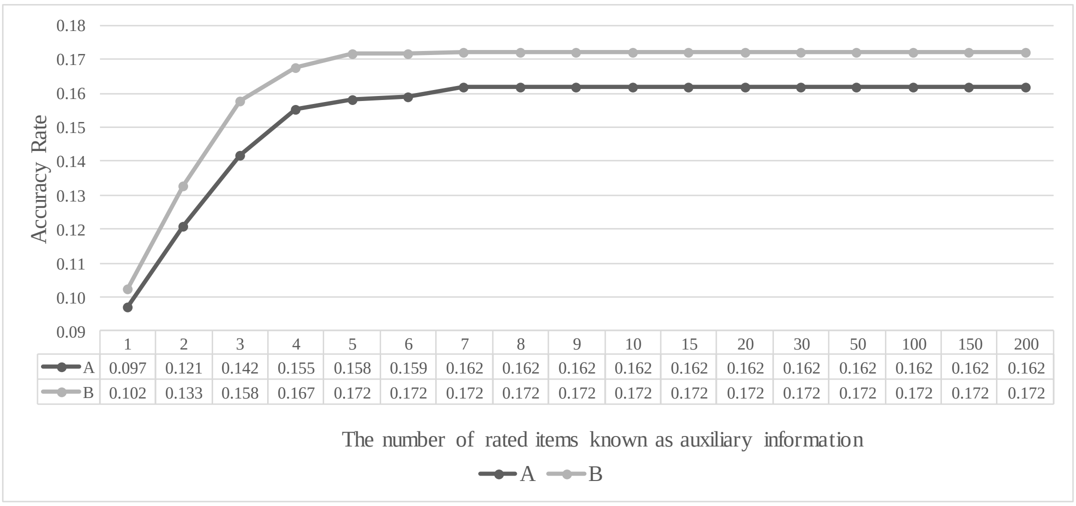


	1	2	3	4	5	6	7	8	9	10	15	20	30	40	50	80	100	120	140	160
A	0.013	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021	0.021
B	0.237	0.281	0.338	0.370	0.387	0.391	0.404	0.404	0.404	0.404	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405	0.405

Figure 5.8. *Accuracy on vertically partitioned data*

Experiment 4: This experiment tests how much private data can be estimated by utilizing the auxiliary information when the data set is randomly vertically divided into two parts. Items are randomly distributed among the parties while users are the same in both parties. Unlike the previous experiment, both parties have roughly a half of data set (50000 ratings). When the data set of the Party *A* is very dense, it has been shown in the previous experiment that the accuracy obtained from the intermediate results even when the auxiliary information is used is very low. When the results of Figure 5.9 are examined, it is seen that the accuracy result that the parties derived from the each other are approximately equal. The density of the data sets is equalized in the random vertical partition and the accuracy rate achieved is around 17%. Party *A* has slightly more ratings than Party *B*; therefore, the accuracy rate of the Party *A* is a little less than that of the Party *B*.

When the results of Figure 5.8 and Figure 5.9 are compared, it is obvious that the sparse data is an important factor for estimating the private data. Considering CF systems, the data is often sparse, so the proposed approach to estimate the private data might play a crucial role. For example, while Party *B* has very sparse data set in Experiment 3, *B* contains a denser one in Experiment 4. Another important point is that the number of users is high while the number of items is relatively low in VPD. It is difficult to find rarely rated items. For this reason, the selected auxiliary information cannot succeed in estimating the confidential data as in Experiment 1 and Experiment 2.



	1	2	3	4	5	6	7	8	9	10	15	20	30	50	100	150	200
A	0.097	0.121	0.142	0.155	0.158	0.159	0.162	0.162	0.162	0.162	0.162	0.162	0.162	0.162	0.162	0.162	0.162
B	0.102	0.133	0.158	0.167	0.172	0.172	0.172	0.172	0.172	0.172	0.172	0.172	0.172	0.172	0.172	0.172	0.172

Figure 5.9. *Accuracy on random vertically partitioned data*

5.5. Conclusions

The methods for deriving the private data may differ according to the partitioning type such as vertical or horizontal. The most common type of privacy attack to derive the private data is to act as an active user. However, analyzes made in partitioned data between two parties show that there is no deduction from the intermediate results. The intermediate results are derived each time as a result of arbitrary addition/subtraction to/from some cells of the active user's rating vector while using the proposed protocol. If the proposed protocol is not applied (to protect the original active user’s rating vector), whether the private data can be derived or not is examined. In this case, the aggregate data is only estimated from the parties. It is impossible to estimate how many users rate the item or which users rate that item. Because of all these reasons, whether the private data can be derived by utilizing auxiliary information is tested. Assume that only the real values of the users are hidden as defined in the privacy definition and the locations of some rated items of users are known as publicly, the private data can be estimated as shown in the experiments. According to the experiment results, the ratio of the estimation of the HPD is greater than the case of VPD. The most important reason for this result is that the number of items is greater than the number of users when the data is partitioned horizontally. The high number of items with fewer users increases the likelihood of different users voting for different items. If it is known that some of the users vote for different items, a large part of the data can be estimated. Thus, the real z-score values are derived by utilizing the auxiliary information. On the other hand, when the data is vertically partitioned, the density of the data set is an important factor in terms of the privacy attack. When the number of items is low, and the number of users who vote for these items is high, the possibility of different users voting for the same items increases. For this reason, even if the locations of items are known, the real z-score values of the users cannot be estimated from the intermediate results.

Data may also be distributed among more than two parties in CF systems. The targeted PPCF protocols are analyzed in case a privacy attack is performed. Analyzes made to derive the private data show that it is complicated and inapplicable when data is distributed either vertically or horizontally among parties. Even the other parties coalesce against the MP; only partial aggregate results can be derived after computing a great number of equations. Therefore, experiments on distributed systems are not performed. Only mathematical analysis has been performed to show whether the private data can be estimated or not.

6. CONCLUSIONS AND FUTURE WORKS

This dissertation scrutinizes the problem of "how much private data can be derived from the disguised numeric data in central or distributed data-based privacy-preserving collaborative filtering systems," and some solutions are proposed by utilizing various methods as well as auxiliary information. In privacy-preserving collaborative filtering systems, it is important to provide reliable and decent predictions while protecting individuals' or data holders' privacy. There are a considerable number of collaborative filtering algorithms proposed in the literature. The aim of all proposed algorithms is to offer more reliable predictions to individuals. However, an attacker or malicious one might want to estimate the confidential data during the prediction process by planning various attack scenarios or employing existing reconstruction methods based on the targeted privacy-preserving collaborative filtering systems.

In order to protect privacy, several data disguising methods such as randomized perturbation techniques, random filling, cryptographic methods, and permutation functions are utilized. However, these data disguising methods may not protect the privacy as much as believed. Throughout the dissertation, the targeted privacy-preserving collaborative filtering schemes in each chapter utilize a different disguising method. The question in each chapter is the same, how much can the private data be estimated from the masked one despite the different data disguising methods?

Privacy-preserving collaborative filtering schemes show differences in terms of the location of data. Although the data is stored by a central server, it can be partitioned or distributed (vertically or horizontally) among companies as well. Therefore, the dissertation is divided into two main parts based on how data is located to derive the private data from the targeted privacy-preserving collaborative filtering schemes. Chapter 2 gives preliminaries and represents how a prediction is produced based on the targeted privacy-preserving collaborative filtering schemes.

In Chapter 3, the dissertation targets central server-based privacy-preserving collaborative filtering schemes when individuals only hide their original values by employing the randomized perturbation technique. Although individuals add random values to their z-score values rather than the original ratings, the server or malicious one may derive the original ratings from the disguised z-score values. The existing study in the literature try to estimate the original ratings from z-score values with strong auxiliary information assumptions such as the averages and standard deviations of users. Therefore, a method called reconstruction from the estimated z-score values is introduced to overcome this problem. Besides, the reconstruction results can be improved by utilizing some domain related auxiliary information. Various auxiliary information is employed to improve the reconstruction results. According to the experiments, the proposed and existing reconstruction methods with auxiliary information enable the server or an attacker to improve the reconstruction accuracy of their estimates of the original ratings from the disguised z-score values.

In Chapter 4, the dissertation targets a central server-based privacy-preserving collaborative filtering scheme; however, in this chapter, individuals mask their original values as well as the location of their rated items by employing the random filling technique. Random filling allows individuals to fill some unrated items with random numbers before disguising their data. In this chapter, which items are actually rated is figured out from the disguised data. The first important step is to determine the number of the rated items from the disguised data including the real and fake items. To overcome this problem, two formulas are introduced in terms of the selected the value of the random filling parameter. The second prominent step is to eliminate the noise, which is added by users to protect their data. Different matrix factorization methods are employed to decrease the effect of noise, and one of them is selected as the main method according to the conducted experiments. Besides, various auxiliary information is introduced to improve the identification accuracy of the rated items. Experiments show that the proposed approaches help the attacker reconstruct the rated items. The other important outcome is that user related auxiliary information makes a great contribution to the results although item related auxiliary information cannot help enhance the results for a movie-related data set used in the experiments. Thus, experiments suggest that selecting appropriate auxiliary information is a prominent issue to enhance the results.

The Chapter 5 of the dissertation targets distributed privacy-preserving collaborative filtering schemes. First, it is examined how much of the private data can be derived when the data is divided into two parties either vertically or horizontally. In the horizontally partitioned data, parties can compute their required data off-line. Since one of the parties can act as an active user to derive the private data from the interim results calculated collaboratively, the other party employs the Private Scalar Product

Computation protocol to protect itself from the attacking party. The Private Scalar Product Computation protocol ensures that the attacking party cannot estimate the private data from the interim results because of the randomness the other party adds into the active user rating vector. On the other hand, the originality of the active user rating vector is also important to produce accurate predictions. Therefore, it is studied whether the private data can be derived unless the protocol is applied. Detailed analysis shows that the attacking party or malicious one cannot estimate the real rated values of users. Only the aggregate values are obtained, or items that are rated by no one are deduced. Because estimating the real rating values and where they are located is shown not to be possible, one of privacy definition is violated to examine whether the real rating values can be estimated. If the locations of some of the rated items of users are known in advance, a large part of the data is estimated as seen from the experiments. On the other hand, when data is partitioned vertically between two parties, parties need to exchange some partial results for providing predictions. To protect data holders' privacy, randomization, cryptographic methods, and permutation functions are employed by the target privacy-preserving collaborative filtering scheme. It is impossible to derive the private data from the cryptographic and permutation methods. Therefore, it is assumed in the experiments that data disguising methods are not applied by the parties. Even if the parties do not apply any the data disguising methods, the attacking party can only derive the aggregate data. This inference is similar to horizontally partitioned data case. The proposed assumption about auxiliary information does not work as much as it does in horizontally partitioned data because of the data partitioning. In addition, the data can be distributed among more than two parties. In horizontally distributed data, parties can compute their required data without needing of other parties' data. Since the master party can store and use the parties' interim data to produce a prediction, some measures are taken by parties to protect themselves against the master party. However, the parties coalesce against the master party to derive its private data. As in the partitioned data, the parties may only estimate the aggregate data of the master party after solving a large number of equations. Since the parties need to exchange their partial results to produce a prediction in vertically distributed data, there are several strict precautions such as randomization, cryptographic and permutation methods that parties apply. Because of these precautions, it is impossible to deduce anything from the partial results in the vertically distributed data.

In order to stress the effect of auxiliary information more, different domains might be selected in the future work. There could be numerous auxiliary information based on the selected domains. The data set in the experiments related to the movies is widely utilized in collaborative filtering environments. Hence, movie-related auxiliary information is utilized throughout the dissertation. Although auxiliary information provides reconstruction enhancements, a different domain still remains an open problem. Besides, the type of auxiliary information may become varied with the advent of social networks. Different social network platforms have opened new avenues to gather great volumes and kinds of information. Thus, the social networks have an enormous potential to find like-minded people based on what they share or write in order to improve reconstructions.

REFERENCES

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

Agrawal, R. and Srikant, R. (2000). Privacy-preserving data mining. *Proceedings of the 19th ACM SIGMOD International Conference on Management of Data*, Dallas, TX, USA: ACM, pp. 439-450

Ahmed, N., Natarajan, T. and Rao, K. R. (1974). Discrete cosine transform. *IEEE transactions on Computers*, 100 (1), 90-93.

Alliance, N. (2011). Ipsos Media CT (2011), Opening our eyes: How film contributes to the culture of the UK. Retrieved from British Film Institute Website: http://old.bfi.org.uk/publications/openingoureyes/downloads/Appendix-2-Results-Tables-Cultural-Contribution-of-Film.pdf

Al-Shamri, M. Y. H. (2016). User profiling approaches for demographic recommender systems. *Knowledge-Based Systems*, 100, 175-187.

Barragáns-Martínez, A. B., Costa-Montenegro, E., Burguillo, J. C., Rey-López, M., Mikic-Fonte, F. A., Peleteiro, A. (2010). A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition. *Information Sciences*, 180 (22), 4290-4311.

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

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

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

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

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

Bogetoft, P., Christensen, D. L., Damgård, I., Geisler, M., Jakobsen, T. P., Krøigaard, M., Nielsen, J.D., Nielsen, J.B., Nielsen, K., Pagter, J., Schwartzbach, M. I., Toft, T. (2009). Secure Multiparty Computation Goes Live. *Financial Cryptography and Data Security: 13th International Conference, FC 2009*, Accra Beach, pp. 325-343.

Burke, R.D. (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. *Proceedings of 2011 IEEE Symposium on Security and Privacy*, Oakland, CA, USA, pp. 231-246.

Campos, L. M., Fernández-Luna, J. M., Huete, J. F., Rueda-Morales, M. A. (2010). Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian Networks. *International Journal of Approximate Reasoning,* 51 (7), 785–799.

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

Christakou, C., Vrettos, S. and Stafylopatis, A. (2007). A hybrid movie recommender system based on neural networks. *International Journal on Artificial Intelligence Tools*, 16 (05), 771-792.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., Sartin, M. (1999). Combining Content-Based and Collaborative Filters in an Online Newspaper. *Proceedings of ACM SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*. Berkeley, California, USA.

Cranor, L. F., Reagle, J. and Ackerman, M. S. (2000). Beyond concern: Understanding net users’ attitudes about online privacy. *The Internet upheaval: Raising questions, seeking answers in communications policy*, Cambridge, MA: MIT Press, pp. 47–70.

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

Demirelli Okkalioglu, B., Koc, M. and Polat, H. (2016). Reconstructing rated items from perturbed data. *Neurocomputing*, 207, 374-386.

Demirelli Okkalioglu, B., Koc, M. and Polat, H. (2017). Deriving private data in partitioned data-based privacy-preserving collaborative filtering systems. *Journal of the Faculty of Engineering and Architecture of Gazi University*, 32 (1), 53-64.

Evfimievski, A., Srikant, R., Agrawal, R., Gehrke, J. (2004). Privacy preserving mining of association rules. *Information Systems*, 29 (4), 343-364.

Gao, L. and Li, C. (2008). Hybrid personalized recommended model based on genetic algorithm. *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing.* Dalian, China, pp. 1-4.

Gogna, A. and Majumdar, A. (2015). Matrix completion incorporating auxiliary information for recommender system design. *Expert Systems with Applications*, 42 (14), 5789-5799.

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.

Grčar, M., Fortuna, B., Mladenič, D., Grobelnik, M. (2006). kNN versus SVM in the collaborative filtering framework. *Data Science and Classification*, 251-260.

Guo, S. and Wu, X. (2006). On the use of spectral filtering for privacy preserving data mining. *In Proceedings of the 2006 ACM symposium on Applied computing*, Dijon, France, pp. 622-626.

Guo, S., Wu, X. and Li, Y. (2006). On the lower bound of reconstruction error for spectral filtering based privacy preserving data mining. *10th European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD 2006*, Berlin, Germany, 520-527.

Guo, S., Wu, X. and Li, Y. (2008). Determining error bounds for spectral filtering based reconstruction methods in privacy preserving data mining. *Knowledge and information systems*, 17 (2), 217-240.

Herlocker, J. L., Konstan, J. A., Borchers, A., Riedl, J. T. (1999), An algorithmic framework for performing collaborative filtering. *Proceedings of the 22nd Annual*

International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, California, USA, pp. 230–237.

Hernando, A., Bobadilla, J. and Ortega, F. (2016). A non negative matrix factorization for collaborative filtering recommender systems based on a Bayesian probabilistic model. *Knowledge-Based Systems*, 97, 188-202.

Hsu, S. H., Wen, M. H., Lin, H. C., Lee, C. C., Lee, C. H. (2007). AIMED-A personalized TV recommendation system. E*uropean Conference on Interactive Television:* 5th European Conference, EuroITV 2007, Amsterdam, The Netherlands, pp. 166-174.

Huang, Z., Du, W. and Chen, B. (2005). Deriving private information from randomized data. *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, *SIGMOD '05*, Baltimore, MD, USA, pp. 37-48.

Jeckmans, A., Tang, Q. and Hartel, P. (2012). Privacy-preserving collaborative filtering based on horizontally partitioned dataset. *2012 International Conference on Collaboration Technologies and Systems (CTS 2012)*, Denver, Colorado, USA, pp. 439-446.

Jiang, M., Cui, P., Chen, X., Wang, F., Zhu, W., Yang, S. (2015). Social recommendation with cross-domain transferable knowledge. *IEEE Transactions on Knowledge and Data Engineering*, 27 (11), 3084-3097.

Johnson, M. L. (2010). Essential Numerical Computer Methods. *Academic Press*.

Kaleli, C. and Polat, H. (2011). Privacy-preserving trust-based recommendations on vertically distributed data. *Proceedings of the 5th IEEE International Conference on Semantic Computing*, Palo Alto, CA, USA, 376-379.

Kaleli, C. and Polat, H. (2012a). SOM-based recommendations with privacy on multi-party vertically distributed data. *Journal of the Operational Research Society*, 63 (6), 826–838.

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

Kargupta, H., Datta, S., Wang, Q., Sivakumar, K. (2003). On the privacy preserving properties of random data perturbation techniques. *Proceedings of Third IEEE International Conference on Data Mining*, Melbourne, FL, USA, pp 99-106.

Kargupta, H., Datta, S., Wang, Q., Sivakumar, K. (2005). Random-data perturbation techniques and privacy preserving data mining. *Knowledge and Information Systems*, 7 (4), 387-414.

Kim, K. J. and Ahn, H. (2008). A recommender system using GA K-means clustering in an online shopping market. *Expert systems with applications*, 34 (2), 1200-1209.

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

Konstas, I., Stathopoulos, V. and Jose, J. M. (2009). On social networks and collaborative recommendation. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval,* Boston, MA, USA, pp. 195-202.

Koren, Y., Bell, R. and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, 42 (8), 30-37.

Li, D., Chen, C., Lv, Q., Shang, L., Zhao, Y., Lu, T., Gu, N. (2016). An algorithm for efficient privacy-preserving item-based collaborative filtering. *Future Generation Computer Systems*, 55, 311-320.

Lindell, Y. and Pinkas, B. (2009). Secure multiparty computation for privacy-preserving data mining. *Journal of Privacy and Confidentiality,* 1 (1), 59-98.

Lucas, J. P., Laurent, A., Moreno, M. N., Teisseire, M. (2012). A fuzzy associative classification approach for recommender systems. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems,* 20 (04), 579-617.

Maneeroj, S. and Takasu, A. (2009). Hybrid recommender system using latent features. *The IEEE 23rd International Conference on Advanced Information Networking and Applications Workshops/Symposia, WAINA 2009,* Bradford, United Kingdom, pp. 661-666.

Martinez, L., Perez, L. G. and Barranco, M. J. (2009). Incomplete preference relations to smooth out the cold-start in collaborative recommender systems. *NAFIPS 2009 - 2009 Annual Meeting of the North American Fuzzy Information Processing Society,* Cincinnati, Ohio, USA, pp. 1-6

Okkalioglu, B. D., Okkalioglu, M., Koc, M., Polat, H. (2015). A survey: deriving private information from perturbed data. *Artificial Intelligence Review*, 44 (4), 547-569.

Okkalioglu, M., Koc, M. and Polat, H. (2015a). On the discovery of fake binary ratings. *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, Salamanca, Spain, pp. 901-907.

Okkalioglu, M., Koc, M. and Polat, H. (2015b). On the privacy of horizontally partitioned binary data-based privacy-preserving collaborative filtering. *Lecture Notes in Computer Science,* 9481, 199-214.

Okkalioglu, M., Koc, M. and Polat, H. (2016). A Privacy Review of Vertically Partitioned Data-based PPCF Schemes. *International Journal of Information Security Science*, 5 (3), 51-68.

Ortega, F., Hernando, A., Bobadilla, J., Kang, J. H. (2016). Recommending items to group of users using Matrix Factorization based Collaborative Filtering. *Information Sciences*, 345, 313-324.

Pazzani, M. J. (1999). A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13 (5), 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. *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, Stanford, CA, USA, pp. 473-480.

Pfitzmann, A. and Hansen, M. (2010). A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. [online] Available: http://dud.inf.tu-dresden.de/Anon_Terminology.shtml.

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

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

Polat, H. and Du, W. (2005b). Privacy-preserving collaborative filtering on vertically partitioned data. *Lecture Notes in Computer Science*, 3721, 651-65.

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

Polat, H. (2006). *Privacy-preserving collaborative filtering*, Ph.D. dissertation, Syracuse University, Computer and Information Science, Syracuse, NY.

Polat, H. and Du, W. (2006). Achieving private recommendations using randomized response techniques. *Proceedings of the 10th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, Singapore, pp. 637-646.

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

Polat, H. and Du, W. (2008). Privacy-preserving top-N recommendation on distributed data. *Journal of the American Society for Information Science and Technology*, 59 (7), 1093-1108.

Polatidis, N., Georgiadis, C. K., Pimenidis, E., Mouratidis, H. (2017). Privacy-preserving collaborative recommendations based on random perturbations. *Expert Systems with Applications*, 71, 18-25.

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

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

Salehi, M., Pourzaferani, M. and Razavi, S. A. (2013). Hybrid attribute-based recommender system for learning material using genetic algorithm and a multidimensional information model. *Egyptian Informatics Journal*, 14 (1), 67-78.

Santos, E. B., Garcia Manzato, M. and Goularte, R. (2014). Evaluating the impact of demographic data on a hybrid recommender model. *IADIS International Journal on WWW/Internet*, 12 (2), 149-167.

Sarwar, B., Karypis, G., Konstan, J., Riedl, J. T. (2000). Analysis of recommendation algorithms for e-commerce. *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. *Proceedings of the 10th International Conference on World Wide Web*, Hong Kong, China, pp. 285–295.

Shinde, S. K. and Kulkarni, U. (2012). Hybrid personalized recommender system using centering-bunching based clustering algorithm. *Expert Systems with Applications*, 39 (1), 1381-1387.

Shmueli, E. and Tassa, T. (2017). Secure Multi-Party Protocols for Item-Based Collaborative Filtering. *Proceedings of the 11th ACM Conference on Recommender Systems*, Como, Italy, pp. 89-97.

Srebro, N. and Jaakkola, T. (2003). Weighted low-rank approximations. *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, Washington, DC, USA, pp. 720-727.

Ungar, L. and Foster, D. (1998). Clustering methods for collaborative filtering. *Proceedings of the Workshop on Recommendation Systems*, Menlo Park, CA, USA, pp. 114-129.

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

Wang, Z., Yu, X., Feng, N., Wang, Z. (2014). An improved collaborative movie recommendation system using computational intelligence. *Journal of Visual Languages & Computing*, 25 (6), 667-675.

Yakut, I. and Polat, H. (2007). Privacy-preserving Eigentaste-based collaborative filtering, *Lecture Notes in Computer Science*, 4752, 169–184.

Yakut, I. and Polat, H. (2010). Privacy-preserving SVD-based collaborative filtering on partitioned data. *International Journal of Information Technology and Decision Making*, 9 (3), 473-502.

Yakut, I. and Polat, H. (2012a). Arbitrarily distributed data-based recommendations with privacy. *Data & Knowledge Engineering*, 72, 239-256.

Yakut, I. and Polat, H. (2012b). Privacy-preserving hybrid collaborative filtering on cross distributed data. *Knowledge and Information Systems*, 30 (2), 405-433.

Yang, W. and Qiao, S. (2010). A novel anonymization algorithm: Privacy protection and knowledge preservation. *Expert Systems with Applications*, 37 (1), 756-766.

Yu, Z., Wang, C., Bu, J., Wang, X., Wu, Y., Chen, C. (2015). Friend recommendation with content spread enhancement in social networks. *Information sciences*, 309, 102-118.

Yuan, T., Cheng, J., Zhang, X., Liu, Q., Lu, H. (2015). How friends affect user behaviors? An exploration of social relation analysis for recommendation. *Knowledge-Based Systems*, 88, 70-84.

Zanker, M. and Jessenitschnig, M. (2009). Case-studies on exploiting explicit customer requirements in recommender systems. *User Modeling and User-Adapted Interaction*, 19 (1), 133-166.

Zhan, J., Hsieh, C. L., Wang, I. C., Hsu, T. S., Liau, C. J., Wang, D. W. (2010). Privacy-preserving collaborative recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40 (4), 472-476.

Zhang, S., Wang, W., Ford, J., Makedon, F., Pearlman, J. (2005). Using singular value decomposition approximation for collaborative filtering. *Seventh IEEE International Conference on E-Commerce Technology (CEC'05)*, Munich, Germany, pp. 257-264.

Zhang, S., Ford, J. and Makedon, F. (2006a). Deriving private information from randomly perturbed ratings. *Proceedings of the 2006 SIAM International Conference on Data Mining*, Bethesda, MD, USA, pp.59-69.

Zhang, S., Ford, J. and Makedon, F. (2006b). A privacy-preserving collaborative filtering scheme with two-way communication. *Proceedings of the 7th ACM conference on Electronic commerce*, Ann Arbor, Michigan, USA, pp. 316-323.

Zhang, J., Zhu, J., Zhang, N. (2014). An Improved Privacy-Preserving Collaborative Filtering Recommendation Algorithm. *Pacific Asia Conference on Information Systems, PACIS,* Chengdu, China.

CIRRUCULUM VITAE

First and Last Name	: Burcu DEMİRELLİ OKKALIOĞLU
Foreign Language	: English
Place and Year of Birth	: Lüleburgaz / 1984
E-Mail	: bdokkalioglu@anadolu.edu.tr

Education

- The University of Texas at San Antonio, College of Sciences, Department of Computer Science, Master of Science, 2012
- Pamukkale University, Faculty of Engineering, Department of Computer Engineering, 2007

Professional

- Yalova University, Research Assistant, Faculty of Engineering, Department of Computer Engineering, 2012-
- Pamukkale University, IT Department, 2007 – 2009.

Honors

- 2009 – 2012, Scholarship from Ministry of National Education to Study Abroad

Publications

- **Demirelli Okkalioglu, B.,** Koc, M., and Polat, H. (2017). Deriving private data in partitioned data-based privacy-preserving collaborative filtering systems. *Journal Of The Faculty Of Engineering and Architecture Of Gazi University,* 32 (1), 53-64.
- **Demirelli Okkalioglu, B.**, Koc, M. and Polat, H. (2016). Reconstructing rated items from perturbed data. *Neurocomputing*, 207, 374-386.

- **Demirelli Okkalioglu, B.**, Okkalioglu, M., Koc, M. and H Polat (2015). A survey: deriving private information from perturbed data. *Artificial Intelligence Review* 44 (4), 547-569.
- Wichmann, A., **Demirelli Okkalioglu, B.** and Korkmaz, T. (2014). The integration of mobile (tele) robotics and wireless sensor networks: A survey. *Computer Communications*, 51, 21-35.