University of Alberta

Library Release Form

Name of Author: Stanley Robson de Medeiros Oliveira

Title of Thesis: Data Transformation For Privacy-Preserving Data Mining

Degree: Doctor of Philosophy

Year this Degree Granted: 2005

Stanley Robson de Medeiros Oliveira
Embrapa Informática Agropecuária
Av. André Tosello, 209
Caixa Postal 6041 - Barão Geraldo
13083-886 - Campinas, SP, Brasil

Date: ____________________

“The fear of the Lord is the beginning of knowledge”
Proverbs 1:7 (The Bible)

University of Alberta

Data Transformation For Privacy-Preserving Data Mining

by

Stanley Robson de Medeiros Oliveira

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

Department of Computing Science

Edmonton, Alberta
Spring 2005

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Data Transformation For Privacy-Preserving Data Mining** submitted by Stanley Robson de Medeiros Oliveira in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

Dr. Osmar R. Zaïane
Supervisor

Dr. Jörg Sander

Dr. Russell Greiner

Dr. Gerald Häubl

Dr. Stan Matwin
External Examiner

Date:

To my parents

Severino Luiz de Oliveira
and
Maria Salete de Medeiros Oliveira

Abstract

The sharing of data is often beneficial in data mining applications. It has been proven useful to support both decision-making processes and to promote social goals. However, the sharing of data has also raised a number of ethical issues. Some such issues include those of privacy, data security, and intellectual property rights.

In this thesis, we focus primarily on privacy issues in data mining, notably when data are shared before mining. Specifically, we consider some scenarios in which applications of association rule mining and data clustering require privacy safeguards. Addressing privacy preservation in such scenarios is complex. One must not only meet privacy requirements but also guarantee valid data mining results. This status indicates the pressing need for rethinking mechanisms to enforce privacy safeguards without losing the benefit of mining. These mechanisms can lead to new privacy control methods to convert a database into a new one in such a way as to preserve the main features of the original database for mining.

In particular, we address the problem of transforming a database to be shared into a new one that conceals private information while preserving the general patterns and trends from the original database. To address this challenging problem, we propose a unified framework for privacy-preserving data mining that ensures that the mining process will not violate privacy up to a certain degree of security. The framework encompasses a family of privacy-preserving data transformation methods, a library of algorithms, retrieval facilities to speed up the transformation process, and a set of metrics to evaluate the effectiveness of the proposed algorithms, in terms of information loss, and to quantify how much private information has been disclosed.

Our investigation concludes that privacy-preserving data mining is to some extent possible. We demonstrate empirically and theoretically the practicality and feasibility of achieving privacy preservation in data mining. Our experiments reveal that our framework is effective, meets privacy requirements, and guarantees valid data mining results while protecting sensitive information (e.g., sensitive knowledge and individuals' privacy).

Acknowledgements

This is one of the best moments in my doctoral program - to publicly acknowledge those who have contributed, in many different ways, to make my success a part of their own. Although I recognize this is a tough task, I will try my best.

First of all, with deepest thanks, I would like to praise my God for he has given me the desire of my heart - to achieve my PhD degree. I have dreamed with this moment for more than 10 years. Now the Lord has made my dream come true blessing the work of my hands.

I am blessed to have my lovely wife, Deise, with me in the last steps of my PhD. We got married just six months ago. When steam was running low, she inspired me with encouraging words and supported me by unceasing prayers. It was under her loving care that I made it through. She made the last steps of my PhD enjoyable and stress free.

I consider myself blessed and lucky to have been born into a truly and lovely family. No matter what the circumstances are, my parents, brothers and sisters are always available and standing by me to provide moral support and encouragement. My parents taught me to be determined, to work hard, to be kind, and to be committed to excellence. My gratitude and appreciation also goes to my parents-in-law for their continuous support, assistance, and kind attention. My whole family encouraged me with blessing words. When they did not have words to say, they supported me in prayers. I hope I am making them proud of my achievements.

I am most fortunate to have had Osmar Zaïane as my supervisor. His supervision style brings the best out of his students. I wish to express my deep gratitude for his guidance and continuous encouragement through these four years, especially at the end of this endeavor. He seamlessly encouraged me to be pro-active in learning. I also learned a lot from his experience. I believe he is building a rewarding and prosperous career.

My gratitude and appreciation to my examiners Dr. Stan Matwin, Dr. Gerald Häubl, Dr. Jörg Sander, and Dr. Russell Greiner for the time and effort invested on my thesis. They carefully read my thesis and provided many valuable comments and suggestions. Special thanks to Dr. Jörg Sander for fruitful discussions we had on the issues addressed in Chapter 6 of this thesis. Thanks are also due to Dr. Robert Holte for his helpful suggestions and feedback on my thesis.

My sincere thanks to Mario Nascimento for inviting me and encouraging me to pursue my doctoral program at the University of Alberta. I am indebted to him for his continuous support, especially in the first year of my PhD program.

Thanks are due to Dave Clyburn, from the Academic Support Centre of the University of Alberta, for diligently reading my thesis and providing nice suggestions to improve the readability of this manuscript.

I had the privilege of visiting the Sabanci University in Istanbul, Turkey. Dr. Yücel Saygin invited me to work for the CODMINE project during the Summer 2003. I had the opportunity to share and implement some of my ideas to the improvement of that project. I am thankful to Yücel for his support and hospitality. I greatly benefited from this collaboration.

I would like to thank Dr. Vladimir G. Troitsky for diligently checking the soundness of my mathematical proofs in Chapter 6 of this thesis.

Sincere thanks to my friend Tibério Caetano for being so available to help and encourage me. I saw Tibério stopping his work, many times, to discuss issues related to my thesis. He read my whole thesis and made constructive and useful comments and suggestions. I really appreciate his time, effort, and feedback on my thesis and, above all, I appreciate his friendship and kind attention.

A conversation with Reza Sherkat provided initial impetus for my solution based on random projection to address privacy preservation in clustering. Reza was always willing to discuss technical issues and provide useful suggestions.

My gratitude also to Edith Drummond who from the very beginning has provided immeasurable assistance to me in completing my studies and research at the Department of Computing Science, at this university. Special thanks are also extended to Frances Moore for her administrative assistance and support during my PhD program.

Acknowledgment is due to Embrapa (Brazilian Agricultural Research Corporation) for the four-year leave of absence to improve my education pursuing my PhD. Special thanks to my friends and workmates Kleber Sampaio de Souza and Carla Macário for kindly looking after my affairs in Campinas, SP, during my absence.

I also acknowledge the Brazilian funding agency CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) of Ministry for Science and Technology of Brazil, for the indispensable financial support under the grant 200077/00-7.

I have very much enjoyed being a member of the Database Research Group at the University of Alberta. I am grateful to the professors and other fellow students - Alex Coman, Luiza Antonie, Reza Sherkat, Leila Homaeian, Gabriela Moise, Fan Deng, Samer Nassar, Chi-Hoon Lee, Veena Sridhar, Jeffery Antoniuk, Marianne Morris, Yi Li, Andrew Foss, Silvio Cazella, and Daniel Mallet.

Contents

1 Introduction 1

1.1 Motivation 1

1.2 Privacy Preservation: Problem Definition 3

1.2.1 Privacy-Preserving Association Rule Mining 3

1.2.2 Privacy-Preserving Clustering 4

1.2.3 Knowledge Protection Versus Privacy Preservation 4

1.3 Thesis Statement 5

1.4 Thesis Contributions 6

1.5 Research Methodology 7

1.6 Organization of the Dissertation 8

2 Basic Concepts 11

2.1 Knowledge Discovery in Databases 11

2.1.1 The KDD Process 11

2.1.2 A Glance at Data Mining Tasks 13

2.1.3 Data Mining versus Statistical Methods 15

2.1.4 Data Stores for Mining 16

2.2 The Basics of Clustering Analysis 18

2.2.1 The Major Distance-Based Clustering Methods 18

2.2.2 Data Matrix 19

2.2.3 Dissimilarity Matrix 20

2.3 The Basics of Association Rule Mining 22

2.3.1 The Support-Confidence Framework 22

2.3.2 Interestingness Measures 22

2.3.3 Sensitive Rules and Sensitive Transactions 23

2.3.4 The Process of Protecting Sensitive Knowledge 24

2.4 The Basics of Dimensionality Reduction 25

2.4.1 Methods for Dimensionality Reduction 26

2.4.2 Random Projection 27

3 Toward Foundations of Privacy-Preserving Data Mining 29

3.1 The Different Meanings of Privacy 30

3.1.1 Problems in Defining Privacy 30

3.1.2 Privacy Violation in Data Mining 31

3.2 The Basis of Privacy-Preserving Data Mining 31

3.2.1 PPDM Landmarks 31

3.2.2 Defining Privacy Preservation in Data Mining 32

3.2.3 Characterizing Scenarios in PPDM 33

3.2.4 Model of Data Miners 34

3.3 Designing Principles and Policies For PPDM 35

3.3.1 The OECD Privacy Guidelines 35

3.3.2 The implications of the OECD Privacy Guidelines in PPDM 36

3.3.3 Adopting PPDM Policies from the OECD Privacy Guidelines 38
3.4 Requirements for Technical Solutions 38
3.4.1 Requirements for the development of technical solutions 38
3.4.2 Requirements to guide the deployment of technical solutions 39
3.5 Summary 40

4 Privacy-Preserving Data Mining: A Literature Review 42
4.1 Data Partitioning Techniques 43
4.1.1 Cryptography-Based Techniques 43
4.1.2 Generative-Based Techniques 45
4.2 Data Modification Techniques 46
4.2.1 Noise Addition Techniques 46
4.2.2 Space Transformation Techniques 50
4.3 Data Restriction Techniques 52
4.3.1 Blocking-Based Techniques 52
4.3.2 Sanitization-Based Techniques 53
4.4 Data Ownership Techniques 54
4.5 Summary 55

5 Methods for Privacy-Preserving Association Rule Mining 57
5.1 Motivation for Privacy-Preserving Association Rule Mining 58
5.2 The Framework for Privacy-Preserving Association Rule Mining 59
5.2.1 The Inverted File 60
5.2.2 The Library of Sanitizing Algorithms 60
5.2.3 The Set of Metrics 61
5.3 Data Sharing-Based Sanitizing Algorithms 64
5.3.1 Heuristic 1: Sanitization Based on the Degree of Sensitive Transactions 64
5.3.2 The Round Robin Algorithm 66
5.3.3 The Random Algorithm 67
5.3.4 The Item Grouping Algorithm 69
5.3.5 Heuristic 2: Sanitization Based on the Size of Sensitive Transactions . 72
5.3.6 The Sliding Window Algorithm 74
5.4 Pattern Sharing-Based Sanitizing Algorithms 76
5.4.1 Heuristic 3: Rule Sanitization With Blocked Inference Channels 77
5.4.2 Inference Channels in Pattern Sharing-Based Algorithms 78
5.4.3 The Downright Sanitizing Algorithm 79
5.5 Summary 81

6 Methods for Privacy-Preserving Data Clustering 83
6.1 Motivation for Privacy-Preserving Clustering 84
6.2 Addressing Privacy-Preserving Clustering 85
6.2.1 PPC over Centralized Data 85
6.2.2 PPC over Vertically Partitioned Data 85
6.2.3 The Communication Protocol 86
6.3 Taxonomy of PPC Data Transformations 87
6.4 The Object Similarity-Based Representation 88
6.4.1 General Assumptions 89
6.4.2 PPC over Centralized Data 89
6.4.3 How Secure is the OSBR? 90
6.4.4 The Complexity of the OSBR 91
6.4.5 The Advantages and Disadvantages of OSBR 92
6.5 The Dimensionality Reduction Transformation 92
6.5.1 General Assumptions 93

6.5.2 PPC over Centralized Data 93
6.5.3 PPC over Vertically Partitioned Data 95
6.5.4 How Secure is the DRBT? 96
6.5.5 The Accuracy of the DRBT 96
6.5.6 The Complexity of the DRBT 98
6.6 Summary 98

7 Results and Evaluation **100**
7.1 Datasets 100
7.2 Evaluation of the Data Sharing-Based Algorithms 102
7.2.1 Sanitizing Algorithms 102
7.2.2 Methodology 103
7.2.3 Evaluating the Window Size for SWA 105
7.2.4 Measuring Effectiveness 105
7.2.5 Special Cases of Data Sanitization 108
7.2.6 CPU Time for the Sanitization Process 109
7.2.7 Discussion on the Data Sharing-Based Algorithms 111
7.3 Evaluation of the Pattern Sharing-Based Algorithms 111
7.3.1 Sanitizing Algorithms 112
7.3.2 Methodology 112
7.3.3 Measuring Effectiveness 113
7.3.4 CPU Time for the Sanitization Process 115
7.3.5 Discussion on the Pattern Sharing-Based Algorithms 116
7.4 Evaluation of the Dimensionality Reduction-Based Transformation 116
7.4.1 Methodology 117
7.4.2 Measuring the Effectiveness of the DRBT over Centralized Data . . . 118
7.4.3 Measuring the Effectiveness of the DRBT over Vertically Partitioned Data 120
7.4.4 Discussion on the DRBT When Addressing PPC 122
7.5 Summary 122

8 Conclusions and Future Work **124**
8.1 Summary 124
8.2 Contributions 125
8.3 Future Research 127
8.3.1 Challenges Left to Explore 127
8.3.2 Future Research Trends 128

Bibliography **130**

A Complexity of the Sanitizing Algorithms **139**
A.1 Analysis of the Data Sharing-Based Algorithms 139
A.1.1 The Running Time of the Round Robin Algorithm 139
A.1.2 The Running Time of the Item Grouping Algorithm 140
A.1.3 The Running Time of the Sliding Window Algorithm 140
A.2 Analysis of the Pattern Sharing-Based Algorithm 141
A.2.1 The Running Time of the Downright Sanitizing Algorithm 141

B Results of the Sanitization in Real Datasets **143**
B.1 Sensitive Rules 143
B.1.1 The BMS-Web-View-1 Dataset 143
B.1.2 The Retail Dataset 144
B.1.3 The Reuters Dataset 145
B.1.4 The Kosarak Dataset 146
B.2 Evaluation of Window Size for SWA 147

B.3 Results of Misses Cost on the Datasets 148
B.3.1 Condition C1 (A set of 6 Sensitive Rules) 148
B.3.2 Condition C2 (Varying the Number of Sensitive Rules) 149
B.3.3 Condition C3 (Varying the Minimum Support Threshold) 153
B.4 Results of the Difference Between the Original and the Sanitized Datasets . . 157
B.4.1 Condition C1 (A set of 6 Sensitive Rules) 157
B.4.2 Condition C2 (Varying the Number of Sensitive Rules) 158
B.4.3 Condition C3 (Varying the Minimum Support Threshold) 162
B.4.4 Effect of ψ on MC and HF (Rules in Scenario S3) 163
B.5 CPU Time - The Kosarak Dataset 164
B.6 Results of Side Effect Factor on the Datasets 165

C Results of the Effectiveness of the DRBT 166
C.1 Results of the Stress Function Applied to the Datasets 166
C.2 Results of F-measure for the Clusters Mined from the Transformed Datasets . 167

List of Figures

1.1 An example of a database transformed before the mining phase 3
1.2 A schematic view of the framework for privacy preservation 7

2.1 An overview of the steps comprising the KDD process 12
2.2 Locations, items, and sales represented as fragments of relations. 17
2.3 (a): An example multi-dimensional view of relational data; (b): A sample transactional database for a hiking equipment store. 17
2.4 Major steps of the process of protecting sensitive knowledge. 24

4.1 A taxonomy of PPDM techniques 43

5.1 The sketch of the framework for privacy-preserving association rule mining. . 59
5.2 (a) A sample transactional database. (b) The corresponding inverted file. . 60
5.3 A taxonomy of sanitizing algorithms. 61
5.4 Data sharing-based sanitization problems. 62
5.5 Pattern sharing-based sanitization problems. 64
5.6 The sanitized databases using: (a) the Round Robin algorithm; (b) the Random algorithm; (c) the Item Grouping algorithm. 72
5.7 (a): A copy of the sample transactional database in Figure 5.2(a); (b): An example of partial sanitization; (c): An example of full sanitization. 74
5.8 (a) A transactional database. (b) The corresponding frequent itemset graph. 77
5.9 (a) An example of forward-inference. (b) An example of backward-inference. 79
5.10 An example of a frequent itemset graphs before and after sanitization. 81

6.1 A taxonomy of PPC data transformations. 87
6.2 An example of Caetano's lemma in $\Re^2$ [22]. 91

7.1 Results of CPU time for the sanitization process. 110
7.2 Results of CPU time for the sanitization process. 115
7.3 (a) The error produced on the dataset *Pumsb* ($d_o = 74$); (b) The error produced on the dataset *Chess* ($d_o = 37$). 119
7.4 The error produced on the dataset *Pumsb* over vertically partitioned data. . . 121

List of Tables

6.1 Thread of selecting the attributes on the $party_k$ side. 86
6.2 Thread of selecting the objects on the $party_k$ side. 87
6.3 A cardiac arrhythmia database. 90
6.4 The relative error that the distances in 6-3 space suffer from, on the average. 94
6.5 Disguised dataset D' using RP_1 and RP_2. 95
6.6 The number of points in cluster c_i that falls in cluster c'_j in the transformed dataset. 97

7.1 A summary of the datasets used in our experiments 102
7.2 Parameters used for mining the four datasets 104
7.3 Summary of the best algorithms in terms of misses cost under condition C1. . 106
7.4 Summary of the best algorithms for misses cost under condition C2. 106
7.5 Summary of the best algorithms for misses cost under condition C3. 107
7.6 Summary of the best algorithms for dif(D, D') under conditions C1 and C3. . 107
7.7 An example of different thresholds for the sensitive rules in scenario S3. . . . 108
7.8 Effect of ψ on misses cost (MC) and hiding failure (HF). 109
7.9 Summary of the best algorithms in terms of misses. 113
7.10 Summary of IGA and DSA in terms of misses cost varying the number of rules.114
7.11 Summary of the best algorithms in terms of side effect factor. 114
7.12 Average of F-measure (10 trials) for the Accidents dataset ($d_o = 18, d_r = 12$). 119
7.13 Average of F-measure (10 trials) for the Iris dataset ($d_o = 5, d_r = 3$). 120
7.14 An example of partitioning for the Pumsb dataset. 120
7.15 Average of F-measure (10 trials) for the Pumsb dataset over vertically partitioned data. 121

B.1 A set of 6 sensitive rules mutually exclusive 143
B.2 A set of 6 sensitive rules selected randomly 143
B.3 A set of 6 sensitive rules with high support 143
B.4 A set of 6 sensitive rules with low support 144
B.5 A set of 6 sensitive rules mutually exclusive 144
B.6 A set of 6 sensitive rules selected randomly 144
B.7 A set of 6 sensitive rules with high support 144
B.8 A set of 6 sensitive rules with low support 145
B.9 A set of 6 sensitive rules mutually exclusive 145
B.10 A set of 6 sensitive rules selected randomly 145
B.11 A set of 6 sensitive rules with high support 145
B.12 A set of 6 sensitive rules with low support 146
B.13 A set of 6 sensitive rules mutually exclusive 146
B.14 A set of 6 sensitive rules selected randomly 146
B.15 A set of 6 sensitive rules with high support 146
B.16 A set of 6 sensitive rules with low support 147
B.17 Effect of window size on the difference between D and D', misses cost and hiding failure. 147

B.18 Results of misses cost on the dataset Kosarak for condition C1. 148
B.19 Results of misses cost on the dataset Retail for condition C1. 148
B.20 Results of misses cost on the dataset Reuters for condition C1. 148
B.21 Results of misses cost on the dataset BMS-1 for condition C1. 148
B.22 Results of misses cost on the Kosarark dataset for condition C2 (S1) 149
B.23 Results of misses cost on the Retail dataset for condition C2 (S1) 149
B.24 Results of misses cost on the Reuters dataset for condition C2 (S1) 149
B.25 Results of misses cost on the BMS-1 dataset for condition C2 (S1) 149
B.26 Results of misses cost on the Kosarark dataset for condition C2 (S2) 150
B.27 Results of misses cost on the Retail dataset for condition C2 (S2) 150
B.28 Results of misses cost on the Reuters dataset for condition C2 (S2) 150
B.29 Results of misses cost on the BMS-1 dataset for condition C2 (S2) 150
B.30 Results of misses cost on the Kosarak dataset for condition C2 (S3) 151
B.31 Results of misses cost on the Retail dataset for condition C2 (S3) 151
B.32 Results of misses cost on the Reuters dataset for condition C2 (S3) 151
B.33 Results of misses cost on the BMS-1 dataset for condition C2 (S3) 151
B.34 Results of misses cost on the Kosarak dataset for condition C2 (S4) 152
B.35 Results of misses cost on the Retail dataset for condition C2 (S4) 152
B.36 Results of misses cost on the Reuters dataset for condition C2 (S4) 152
B.37 Results of misses cost on the BMS-1 dataset for condition C2 (S4) 152
B.38 Effect of σ on misses cost in the Kosarak dataset for condition C3 (S1) 153
B.39 Effect of σ on misses cost in the Retail dataset for condition C3 (S1) 153
B.40 Effect of σ on misses cost in the Reuters dataset for condition C3 (S1) 153
B.41 Effect of σ on misses cost in the BMS-1 dataset for condition C3 (S1) 153
B.42 Effect of σ on misses cost in the Kosarak dataset for condition C3 (S2) 154
B.43 Effect of σ on misses cost in the Retail dataset for condition C3 (S2) 154
B.44 Effect of σ on misses cost in the Reuters dataset for condition C3 (S2) 154
B.45 Effect of σ on misses cost in the BMS-1 dataset for condition C3 (S2) 154
B.46 Effect of σ on misses cost in the Kosarak dataset for condition C3 (S3) 155
B.47 Effect of σ on misses cost in the Retail dataset for condition C3 (S3) 155
B.48 Effect of σ on misses cost in the Reuters dataset for condition C3 (S3) 155
B.49 Effect of σ on misses cost in the BMS-1 dataset for condition C3 (S3) 155
B.50 Effect of σ on misses cost in the Kosarak dataset for condition C3 (S4) 156
B.51 Effect of σ on misses cost in the Retail dataset for condition C3 (S4) 156
B.52 Effect of σ on misses cost in the Reuters dataset for condition C3 (S4) 156
B.53 Effect of σ on misses cost in the BMS-1 dataset for condition C3 (S4) 156
B.54 Difference(D, D') for condition C1. 157
B.55 Difference(D, D') for condition C1. 157
B.56 Difference(D, D') for condition C1. 157
B.57 Difference(D, D') for condition C1. 157
B.58 Difference(D, D') on the Kosarark dataset for condition C2 (S1) 158
B.59 Difference(D, D') on the Retail dataset for condition C2 (S1) 158
B.60 Difference(D, D') on the Reuters dataset for condition C2 (S1) 158
B.61 Difference(D, D') on the BMS-1 dataset for condition C2 (S1) 158
B.62 Difference(D, D') on the Kosarark dataset for condition C2 (S2) 159
B.63 Difference(D, D') on the Retail dataset for condition C2 (S2) 159
B.64 Difference(D, D') on the Reuters dataset for condition C2 (S2) 159
B.65 Difference(D, D') on the BMS-1 dataset for condition C2 (S2) 159
B.66 Difference(D, D') on the Kosarak dataset for condition C2 (S3) 160
B.67 Difference(D, D') on the Retail dataset for condition C2 (S3) 160
B.68 Difference(D, D') on the Reuters dataset for condition C2 (S3) 160
B.69 Difference(D, D') on the BMS-1 dataset for condition C2 (S3) 160
B.70 Difference(D, D') on the Kosarak dataset for condition C2 (S4) 161
B.71 Difference(D, D') on the Retail dataset for condition C2 (S4) 161

B.72 Difference(D, D') on the Reuters dataset for condition C2 (S4) 161
B.73 Difference(D, D') on the BMS-1 dataset for condition C2 (S4) 161
B.74 Difference(D, D') on the Kosarak dataset for condition C3 162
B.75 Difference(D, D') on the Retail dataset for condition C3 162
B.76 Difference(D, D') on the Reuters dataset for condition C3 162
B.77 Difference(D, D') on the BMS-1 dataset for condition C3 162
B.78 Effect of ψ on misses cost and hiding failure in the Kosarak dataset 163
B.79 Effect of ψ on misses cost and hiding failure in the Retail dataset 163
B.80 Effect of ψ on misses cost and hiding failure in the Reuters dataset 163
B.81 Effect of ψ on misses cost and hiding failure in the BMS-1 dataset 163
B.82 CPU time for sanitization varying the number of sensitive rules (msec.) . . . 164
B.83 CPU time for sanitization varying the dataset size (msec.) 164
B.84 Results of side effect factor on the dataset Kosarak. 165
B.85 Results of side effect factor on the dataset Retail. 165
B.86 Results of side effect factor on the dataset Reuters. 165
B.87 Results of side effect factor on the dataset BMS-1. 165

C.1 The error produced on the *Chess* dataset ($d_o = 37$). 166
C.2 The error produced on the *Mushroom* dataset ($d_o = 23$). 166
C.3 The error produced on the *Pumsb* dataset ($d_o = 74$). 166
C.4 The error produced on the *Connect* dataset ($d_o = 43$). 166
C.5 The error produced on the *Accidents* dataset ($d_o = 18$). 166
C.6 Average of F-measure (10 trials) for the Chess dataset ($d_o = 37, d_r = 25$). . . 167
C.7 Average of F-measure (10 trials) for the Mushroom dataset ($d_o = 23, d_r = 15$).167
C.8 Average of F-measure (10 trials) for the Pumsb dataset ($d_o = 74, d_r = 38$). . 167
C.9 Average of F-measure (10 trials) for the Connect dataset ($d_o = 43, d_r = 28$). . 167

Acronym List

AP: Artifactual Patterns.

DRBT: Dimensionality Reduction-Based Transformation.

DSA: Downright Sanitizing Algorithm.

HF: Hiding Failure.

IGA: Item Grouping Algorithm.

MC: Misses Cost.

OSBR: Object Similarity-Based Representation.

P3P: Platform for Privacy Preferences.

PPARM: Privacy-Preserving Association Rule Mining.

PPC: Privacy-Preserving Clustering.

PPDM: Privacy-Preserving Data Mining.

PPDT: Privacy-Preserving Data Transformation.

RA: Random Algorithm.

RF: Recovery Factor.

RRA: Round Robin Algorithm.

SEF: Side Effect Factor.

SWA: Sliding Window Algorithm.

Glossary

Association Rules: describe interesting relationships between items grouped together in a sufficient number of examples.

Collective Privacy: is concerned with the protection of *sensitive knowledge* representing the activities of a group. The sensitive knowledge must be protected due to strategic or competitive reasons by the caretaker or the owner of the collected data .

Clustering: is concerned with grouping objects into clusters (groups) of similar objects. The goal is to achieve high similarity between objects within individual clusters (interclass similarity) and low similarity between objects that belong to different clusters (intraclass similarity).

Confidence: in *association rules*, the confidence of a rule $X \rightarrow Y$ is defined as the ratio of transactions containing X that also contain Y over the transactions containing X.

Data Matrix: is a matrix in which the rows are points (vectors) in a multi-dimensional space describing objects. Each dimension represents a distinct attribute of the objects. Data matrices are widely used in *clustering.*

Data Sanitization: is the process of hiding *sensitive rules* in *transactional databases.* The sanitization is achieved by modifying some transactions. In some cases, a number of items are deleted from a group of transactions (*sensitive transactions*) with the purpose of hiding the sensitive rules derived from those transactions. In doing so, the support of such sensitive rules are decreased below a certain *disclosure threshold* defined by the data owner. Another way to hide sensitive rules is to add new items to some transactions to alter (decrease) the confidence of sensitive rules. For instance, in a rule $X \rightarrow Y$, if the items are added to the antecedent part X of this rule in transactions that support X and not Y, then the confidence of such a rule is decreased.

Data Sharing-Based Algorithms: are a class of sanitizing algorithms in which the sanitization process acts on the data to remove or hide the group of *sensitive association rules.* After sanitizing a database, the released database is shared for *association rule* mining.

Dimensionality Reduction: is the process of mapping d-dimensional objects onto k-dimensional objects, where $k \ll d$. The goal of dimensionality reduction methods is to map each object to a point in a k-dimensional space minimizing the relative error that the distances in k-d space suffer from, on the average.

Disclosure Threshold (ψ): In our framework, the process of hiding some *sensitive rules* satisfies a *disclosure threshold* ψ controlled by the database owner. This threshold basically expresses how relaxed the privacy preserving mechanisms should be. When $\psi = 0\%$, no sensitive association rules are allowed to be discovered from the sanitized database. When $\psi = 100\%$, there are no restrictions on the sensitive association rules. The advantage of having this threshold is that it enables a compromise to be found between hiding association rules while missing non-sensitive ones.

Dissimilarity Matrix: stores a collection of proximities that are available for all pairs of objects. This matrix is often represented by an $m \times m$ table, where m is the number of objects, and each element $d(i, j)$ represents the difference or dissimilarity between objects i and j.

Euclidean Distance: is the most popular distance measure used to calculate the dissimilarity between objects i and j in a d-dimensional space.

Individual Privacy: is concerned with the protection of personally identifiable information. In general, information is considered personally identifiable if it can be linked, directly or indirectly, to an individual person.

Inference Channels: occur when someone mines a sanitized set of rules and, based on non-sensitive rules, deduces one or more *sensitive rules* that are not supposed to be discovered.

Pattern Sharing-Based Algorithms: are a class of sanitizing algorithms in which the sanitization acts on the rules mined from a database instead of the data itself. The sanitization removes not only all sensitive patterns but also blocks other patterns that could be used to infer the sensitive hidden ones.

Platform for Privacy Preferences (P3P): a project developed by the World Wide Web Consortium, which is emerging as an industry standard providing a simple, automated way for users to gain more control over the use of personal information on Web sites they visit.

Privacy-Preserving Data Mining: encompasses the dual goal of meeting privacy requirements and providing valid data mining results.

Random Projection: is one of the methods designed for dimensionality reduction. A random projection from d dimensions to k dimensions is a linear transformation represented by a $d \times k$ matrix R, which is generated by first setting each entry of the matrix to a value drawn from an i.i.d. $N(0,1)$ distribution and then normalizing the columns to unit length. Given a d-dimensional dataset represented as an $n \times d$ matrix D, the mapping $D \times R$ results in a reduced-dimension dataset D'.

Sensitive Knowledge: is described as the knowledge that can provide competitive advantage in the business world.

Sensitive Rules: are a special group of association rules which represent the *sensitive knowledge* mined from databases.

Sensitive Transactions: are defined as a set of transactions which participate in the generation of the *sensitive rules*.

Support: in *association rules*, the support of an itemset is defined as the ratio of the transactions containing the itemset over all the transactions.

Transactional Database: is a relation consisting of transactions in which each transaction t is characterized by a unique transaction identifier number (TID) and a list of items making up the transaction. Transactional databases are widely used in association rule mining.

Victim Item: is defined as a candidate item that should be eliminated from the *sensitive transactions* in *data sanitization*. Removing this item from sensitive transactions, one or more *sensitive rules* will be hidden in a *transactional database*.

Chapter 1

Introduction

The beginning is the half of every action.
– Greek Proverb

If you understand the beginning well, the end will not trouble you.
– Ashanti Proverb

1.1 Motivation

Recent developments in information technology have made possible the collection and analysis of millions of transactions containing personal data. These data include shopping habits, criminal records, medical histories, and credit records, among others [18]. This progress in the storage and analysis of data has led individuals and organizations to face the challenge of turning such data into useful information and knowledge.

Data mining is a promising approach to meet this challenging requirement. The area of data mining, also called Knowledge Discovery in Databases (KDD), has received special attention since the 1990s. This new research area has emerged as a means of extracting hidden patterns or previously unknown implicit information from large repositories of data [53]. The fascination with the promise of analysis of large volumes of data has led to an increasing number of successful applications of data mining in recent years. Undoubtedly, these applications are very useful in many areas such as marketing, business, medical analysis, and other applications in which pattern discovery is paramount for strategic decision making.

Despite its benefits in various areas, the use of data mining techniques can also result in new threats to privacy and information security. The problem is not data mining itself, but the way data mining is done [81]. As Vaidya & Clifton [140] state, "Data mining results rarely violate privacy, as they generally reveal high-level knowledge rather than disclosing instances of data." However, the concern among privacy advocates is well founded, as bringing data together to support data mining projects makes misuse easier. Thus, in the absence of adequate safeguards, the use of data mining can jeopardize the privacy and autonomy of individuals.

More serious is the privacy invasion occasioned by secondary usage of data when individuals are unaware of "behind the scenes" use of data mining techniques [74]. As an example, Culnan [36] made a particular study of secondary information use, which she defined as "the use of personal information for other purposes subsequent to the original transaction between an individual and an organization when the information was collected." The key finding of this study was that concern over secondary use was correlated with the level of control the individual has over the secondary use.

Even though many nations have developed privacy protection laws and regulations to guard against private use of personal information, the existing laws and their conceptual foundations have become outdated because of changes in technology [91, 100, 46, 32]. As a result, these personal data reside on thousands of file servers, largely beyond the control of existing privacy laws, leading to potential privacy invasion on a scale never before possible.

Complex issues, such as those involved in privacy-preserving data mining (PPDM), cannot simply be addressed by restricting data collection or even by restricting the secondary use of information technology [7, 18, 101]. Moreover, there is no exact solution that resolves privacy preservation in data mining. An approximate solution could be sufficient, depending on the application since the appropriate level of privacy can be interpreted in different contexts [28, 27]. In some applications (e.g., association rules, classification, or clustering), an appropriate balance between a need for privacy and knowledge discovery should be found.

Preserving privacy when data are shared for mining is a challenging problem. The traditional methods in database security, such as access control and authentication [23, 58, 125] that have been adopted to successfully manage the access to data present some limitations in the context of data mining. While access control and authentication protections can safeguard against direct disclosures, they do not address disclosures based on inferences that can be drawn from released data [134, 52, 147]. Preventing this type of inference detection is beyond the reach of the existing methods [7, 101]. Therefore, the work presented in this thesis lies outside of traditional work on database security.

Clearly, privacy issues pose new challenges for novel uses of data mining technology [100, 88, 98]. These technical challenges indicate a pressing need to rethink mechanisms to address some issues of privacy and accuracy when data are either shared or exchanged before mining. Such mechanisms can lead to new privacy control methods to convert a database into a new one that conceals private information while preserving the general patterns and trends from the original database.

1.2 Privacy Preservation: Problem Definition

In this thesis, we address the problem of transforming a database into a new one that conceals sensitive information while preserving the general patterns and trends from the original database. The sensitive information is not limited to personal data, but may reflect customers' purchasing behaviour, financial, medical, and insurance liability information and sensitive patterns.

The transformation applied to the database occurs before the sharing of data for mining, as can be seen in Figure 1.1. We focus primarily on privacy preserving data mining, notably in the context of the mining tasks: a) *association rules* which describe interesting relationships among items grouped together in a sufficient number of examples; and b) *clustering* which is concerned with grouping objects into classes of similar objects.

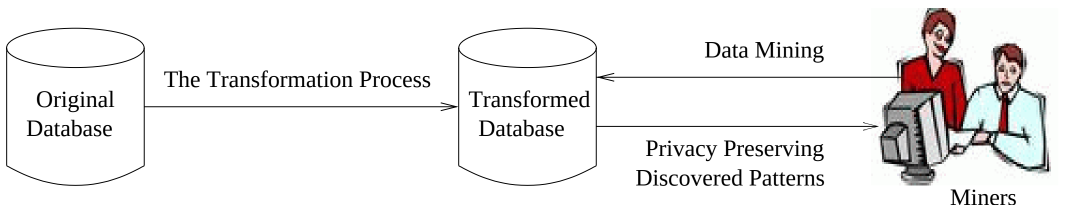

Original Database
The Transformation Process
Transformed Database
Data Mining
Privacy Preserving Discovered Patterns
Miners

Figure 1.1: An example of a database transformed before the mining phase

We will approach the problem of transforming a database, before the sharing of data for mining, by first dividing it into two sub-problems: privacy-preserving association rule mining and privacy-preserving clustering.

1.2.1 Privacy-Preserving Association Rule Mining

In the context of privacy-preserving association rule mining, we do not address privacy of individuals. Rather, we address the problem of protecting sensitive knowledge mined from databases. The sensitive knowledge is represented by a special group of association rules called sensitive association rules. These rules are paramount for strategic decision and must remain private (i.e., the rules are private to the company or organization owning the data).

The problem of protecting sensitive knowledge in transactional databases, draws the following assumptions:

- Data owners have to know in advance some knowledge (rules) that they want to protect. Such rules are fundamental in decision making, so they must not be discovered.
- The individual data values (e.g. a specific item) are not restricted. Rather, some aggregates and relationships must be protected. This approach works in the opposite way to the idea behind statistical databases [23, 3, 38] which prevents against discovering individual tuples.

The problem of protecting privacy in association rule mining can be stated as follows: If D is the source database of transactions and R is a set of relevant association rules that could be mined from D, the goal is to transform D into a database D' so that the most association rules in R can still be mined from D' while others, representing sensitive knowledge, are hidden. In this case, D' becomes the released database.

1.2.2 Privacy-Preserving Clustering

The goal of privacy-preserving clustering is to protect the underlying attribute values of objects subjected to clustering analysis. Unlike privacy-preserving association rule mining that aims at protecting sensitive knowledge in databases, privacy-preserving clustering focuses on protecting the privacy of individuals. In this context, the data are assumed to be a matrix $D_{m \times n}$, where each of the m rows represents an object, and each object contains values for each of the n attributes. The matrix $D_{m \times n}$ may contain binary, categorical, or numerical attributes.

We assume that the attribute values associated with an object are private and must be protected. After transformation, the attribute values of an object in D would look very different from the original. Therefore, miners would rely on the transformed data to build valid results, i.e., clusters.

The problem of privacy preservation in clustering can be stated as follows: Let D be a relational database and C a set of clusters generated from D. The goal is to transform D into D' so that the following restrictions hold:

- A transformation $\mathfrak{T}$ when applied to D must preserve the privacy of individual records, so that the released database D' conceals the values of confidential attributes, such as salary, disease diagnosis, credit rating, and others.

- The similarity between objects in D' must be the same as that one in D, or just slightly altered by the transformation process. Although the transformed database D' looks very different from D, the clusters in D and D' should be as close as possible since the distances between objects are preserved or marginally changed.

1.2.3 Knowledge Protection Versus Privacy Preservation

Protecting sensitive information in the context of our research encompasses two important goals: *knowledge protection* and *privacy preservation*. The former is related to privacy-preserving association rule mining, while the latter refers to privacy-preserving clustering.

An interesting aspect between knowledge protection and privacy preservation is that they have a common characteristic. For instance, in knowledge protection, an organization is the owner of the data so it must protect the sensitive knowledge discovered from such data, while in privacy preservation individuals are the owner of their personal information. On the other hand, knowledge protection and privacy preservation also have a unique characteristic. Privacy preservation is related to the protection of explicit data (e.g., salary), while knowledge protection is concerned with the protection of implicit data, i.e., patterns discovered from the data.

One limitation with the approach of knowledge protection is that the sensitive knowledge should be known in advance by the data owners. In this case, data owners have to mine their databases and use interestingness measures (e.g., support and confidence) with the purpose of finding the valuable patterns, i.e, the sensitive knowledge. Subsequently, data owners hide the sensitive knowledge by using the algorithms introduced in Chapter 5. The released database is then shared for mining.

Another limitation of the approach of knowledge protection is that we do not focus on protecting against correlations between variables, such as salary and age. Rather, we protect specific binary rules (e.g., $X \rightarrow Y$), where X and Y represent items purchased in a store or attributes with specific values. Again, these rules are private to the company or organization owning the data and must be protected since they can provide competitive advantage in the business world.

1.3 Thesis Statement

In this work, we investigate the feasibility of achieving PPDM by data transformation. The central thesis statement of this research is presented as follows:

> *Privacy preservation in data mining, by data transformation, is to some extent possible.*

This research demonstrates empirically and theoretically the practicality and feasibility of achieving PPDM. In particular, it is shown that a balance between privacy preservation and knowledge discovery can be accomplished when addressing knowledge protection in association rule mining and privacy preservation in clustering.

Four major issues are addressed to support the central thesis statement of this research, as follows:

- It is possible to transform a database by protecting the attribute values of objects subjected to clustering and get valid clustering results, i.e., the clusters generated in the transformed database are very similar to those mined from the original database.
- It is possible to protect sensitive knowledge discovered from databases without losing the benefit of mining the transformed database.

- It is possible to quantify the disclosure of sensitive knowledge discovered from a transformed database.
- It is possible to measure the information loss in a transformed database available for association rule mining.

1.4 Thesis Contributions

The major contributions of this thesis can be summarized as follows:

1. **Toward foundations of PPDM:** We put forward foundations and standardization issues in PPDM. In particular, we describe the problems we face in defining what information is private in data mining, and discuss how privacy can be violated in data mining. We describe the basis of PPDM including the historical roots, the definition of privacy preservation in data mining, and models of data miners in PPDM. We then analyze the implications of standard privacy principles in knowledge discovery and suggest some policies for PPDM based on these privacy principles. Subsequently, we suggest some desirable privacy requirements that are related to industrial initiatives. These requirements are essential for the development and deployment of technical solutions and will allow vendors and developers to make solid advances in the future of PPDM.

2. **A taxonomy of PPDM techniques:** We surveyed the existing PPDM techniques in the literature, and we propose a taxonomy including such techniques, which is described in Chapter 4.

3. **A family of privacy-preserving methods:** Addressing privacy preservation in data mining requires different kinds of data transformation since the mining tasks are versatile. We propose a family of privacy-preserving data transformation (PPDT) methods for protecting privacy before data are shared for association rule mining and clustering. These methods are described in Chapter 5 and Chapter 6, respectively.

4. **A library of algorithms:** To enforce knowledge protection in association rule mining, we propose a library of algorithms. Such algorithms are designed taking into account heuristics for our PPDT methods presented in Chapter 5.

5. **Retrieval facilities:** These retrieval facilities are used specifically when some algorithms are applied to protect sensitive knowledge in association rule mining. As mentioned in Section 2.1.2, pattern discovery may require various scans over a transactional database. To speed the process of hiding sensitive knowledge in transactional databases, our framework is built on an index. As a consequence, our algorithms

require only two scans to protect sensitive knowledge regardless the number of association rules to be hidden: one scan to build an inverted index, and the other scan to hide the sensitive rules. Other techniques require multiple scans [12, 37, 126].

6. A set of metrics: Since there is no exact solution to address privacy preservation in data mining, we need to be able to measure how much sensitive information is disclosed and verify the usefulness of the data after the transformation process. To evaluate our method for association rule mining, we propose a set of metrics to measure not only how much sensitive knowledge has been disclosed, but also to measure the effectiveness of the proposed algorithms in terms of information loss and in terms of non-sensitive rules removed as a side effect of the transformation process.

Our contributions mentioned previously are the major parts of a privacy preservation framework depicted in Figure 1.2.

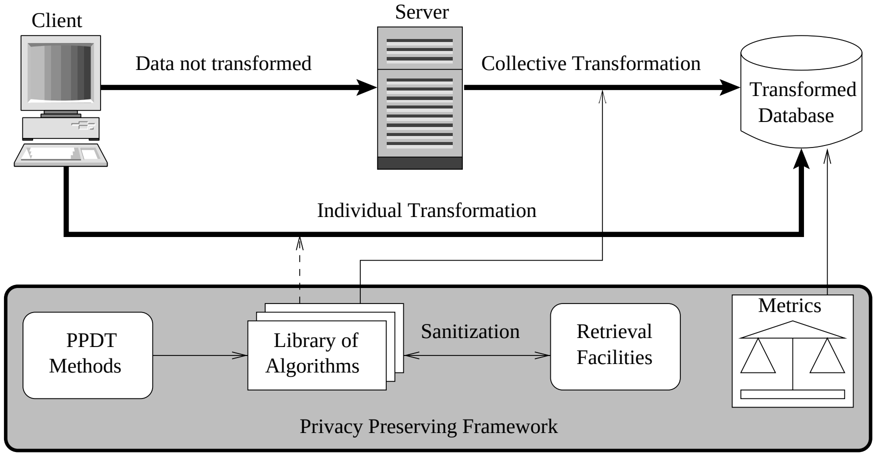


Figure 1.2: A schematic view of the framework for privacy preservation

1.5 Research Methodology

This research was conducted in a phased approach, as follows:

Evaluation of prior work: Research in the area of PPDM is still at a very preliminary stage. We have surveyed efforts to address PPDM and investigated the existing techniques in the literature. The existing solutions for PPDM are reviewed in Chapter 4.

Conception and design of new PPDT Methods: We designed new methods to address the problem of PPDM. These methods cover the main aspects of clustering and association rule mining. Apart from the methods, we also designed some metrics to quantify how much private information is disclosed and to measure the impact of the PPDT methods on the data and on valid data mining results.

Implementation and test: We implemented and tested both existing and new PPDT methods. We applied our methods to real datasets to test their effectiveness and to study their performance.

Evaluation: We specified an evaluation framework to compare the PPDT methods. This framework, implemented in a common platform, was used to determine the appropriate technique for each type of application. The impact of a PPDT method on a data mining task was evaluated by measuring the result of the task with and without the transformation. For newly designed PPDT methods that found equivalent or similar counterparts in the literature, we compared with the published results. We evaluated the effectiveness and scalability of our algorithms as well. We also used our metrics to quantify the privacy level provided by the PPDT methods, to measure information loss and the quality of the data mining results in the transformed datasets.

Dissemination: We disseminated the results of this research through submission of papers to peer reviewed conferences [102, 101, 103, 104, 105, 109, 108, 106, 107]. The review process and discussions at conferences were essential for further improvement of our PPDT methods.

1.6 Organization of the Dissertation

The rest of this dissertation is organized as follows:

- In Chapter 2, we present the fundamentals of knowledge discovery in databases (KDD). We also discuss the main differences between data mining and statistical methods, and review the most common data stores for mining. Then, we highlight the basics of clustering analysis and association rule mining since our PPDT methods are designed to protect privacy when these mining tasks are applied to a transformed database. For clustering, we briefly review the major distance-based clustering methods and the concepts of data matrix and dissimilarity matrix. For association rules, we concentrate on the support-confidence framework and some interestingness measures. In addition, we provide the definitions of sensitive association rules and sensitive transactions. Then, we describe the process of protecting sensitive knowledge in transactional databases. Finally, we review the basics of dimensionality reduction that are the basis of our proposed solution for privacy-preserving clustering in Chapter 6.

- In Chapter 3, we take some steps toward the foundations of PPDM. In particular, we discuss the problems in defining privacy and how privacy can be violated in data mining. Then, we describe the historical roots and the basis of PPDM including: a) the PPDM landmarks that characterize the progress and success of this new research area; b) the definition of privacy preservation in data mining; c) a number of challenging scenarios in PPDM; and d) models of data miners in PPDM. Subsequently, we analyze the implications of the Organization for Economic Cooperation and Development (OECD) data privacy principles in knowledge discovery. As a consequence, we suggest some policies for PPDM based on the OECD privacy guidelines. Moreover, we suggest some privacy requirements for the development and for the deployment of technical solutions.

- In Chapter 4, we review the state-of-the-art in PPDM research. We describe the main idea behind the existing PPDM techniques in the literature. Then, we classify the existing techniques into four major categories: data partitioning, data modification, data restriction, and data ownership.

- In Chapter 5, we introduce the data sanitization method that hides sensitive association rules by reducing either the support or the confidence of these rules. The protection of sensitive rules is achieved by modifying some transactions. In some cases, a number of items are deleted from a group of transactions with the purpose of hiding the sensitive rules derived from those transactions. To accomplish that, we introduce a set of algorithms for protecting sensitive rules. Those algorithms are classified into two groups: *Data-Sharing approach* and *Pattern-Sharing approach*. In the former, the sanitization acts on the data to remove or hide the group of sensitive association rules that contain sensitive knowledge. In the latter, the sanitizing algorithm acts on the rules mined from a database, instead of the data itself. We also introduce a taxonomy covering these two categories of sanitizing algorithms.

- In Chapter 6, we introduce some PPDM methods for privacy preserving data clustering. In particular, we show that the dual-goal of achieving privacy and accuracy can be accomplished by the idea of dissimilarity between objects but at a high communication cost. We refer to this solution as Object Similarity-Based Representation (OSBR). In order to alleviate the communication cost introduced by OSBR, we show that a trade-off between privacy and accuracy can be accomplished by using the intuition behind dimensionality reduction, notably random projection. We refer to the latter solution as Dimensionality Reduction-Based Transformation (DRBT). This latter solution aims at finding a trade-off between privacy, communication cost, and the quality of the clusters mined from the transformed database.

- In Chapter 7, we validate our framework by using a broad set of experiments. First, we describe the methodology adopted to compare our algorithms with the similar counterparts in the literature. We also describe the real datasets used in our experiments. Then we study the effectiveness and the scalability of our sanitizing algorithms. In the context of clustering, we study the trade-off between privacy, communication cost, and the quality of the clusters mined from the transformed database. We conclude this chapter with a discussion of the main results of our performance evaluation.

- Chapter 8 concludes this dissertation with a brief summary of the work presented, discusses the main results achieved in this research, draws some conclusions and points to future work directions as a continuation of this research.
- The complexity analyzes of our sanitizing algorithms are presented in Appendix A.
- The results of the performance evaluation of our sanitizing algorithms are available at Appendix B.
- The results of the performance evaluation of our dimensionality reduction-based transformation (for privacy-preserving clustering) are available at Appendix C.

Chapter 2

Basic Concepts

Just definitions either prevent or put an end to a dispute.

– Nathaniel Emmons

In this Chapter, we review the basic concepts that are necessary to understand the issues addressed in this research. We give special attention to association rules and clustering since these mining tasks are the focus of this research. Section 2.1 describes the foundations of knowledge discovery in databases (KDD). We outline the KDD process and the main data mining tasks. A brief discussion about the differences between data mining and statistical methods is also included. Then we review a number of different data stores on which mining can be performed. Section 2.2 reviews the preliminaries of clustering. We briefly review the major distance-based clustering methods and the data structures for clustering, such as data matrix and dissimilarity matrix. Section 2.3 provides the basics of association rule mining. In particular, we define the support-confidence framework and review some interestingness measures. In addition, we provide the definitions of sensitive association rules and sensitive transactions. Subsequently, we describe the process of protecting sensitive knowledge in transactional databases. Finally, in Section 2.4, we review the basics of dimensionality reduction.

2.1 Knowledge Discovery in Databases

2.1.1 The KDD Process

Knowledge discovery in databases, also called the KDD process, is a non-trivial process of discovering useful knowledge from data [55, 65]. This process consists of several steps, as can be seen in Figure 2.1. In this process, *knowledge* simply refers to information that is relevant and actionable represented by patterns or models. A *pattern* describes relationships among the facts in a subset of the given data, while a model is a characterization of the global dataset. In this context, discovering knowledge means finding patterns, fitting a model to data, or even any general high-level description of a set of data.

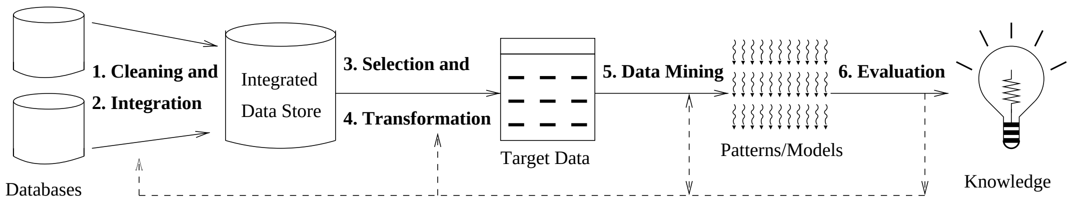


Figure 2.1: An overview of the steps comprising the KDD process

The major steps in the KDD process can briefly be defined as follows:

1. **Data Cleaning:** In the real world, data are often noisy or incomplete, and unless this is understood and corrected, it is likely that many interesting patterns will be missed and the reliability of detected patterns will be low. Data cleaning routines act on the data by filling in missing values, smoothing noisy data, identifying and removing outliers, and resolving inconsistencies.

2. **Data Integration:** The way that data are merged from multiple data stores may need to be transformed into forms appropriate for mining. In this step, data from multiple sources (with heterogeneous data) are combined into a coherent data store, as in data warehousing. These sources include multiple DBMSs, data cubes, or even flat files.

3. **Data Selection:** In this step, the goal is to identify the data that are relevant to the analysis task and retrieve them from the database. Only the selected data are subject to the mining process.

4. **Data Transformation:** In this step, the data are transformed or consolidated into forms appropriate for mining. This is achieved by performing summary, aggregation, generalization, or normalization operations.

5. **Data Mining:** This is the central activity in the KDD process. It is concerned with the extracting of implicit, previously unknown, and potentially useful patterns from the data. To do so, computational techniques are applied to produce a particular enumeration of patterns (or models) from the data.

6. **Pattern Evaluation:** In this step, interestingness[1] measures are applied with the purpose of searching for valuable patterns. This may be accomplished by using interestingness thresholds to filter out discovered patterns. Also, correlation analyses may be applied to evaluate the importance of the discovered information.

[1]These functions are used to separate uninteresting patterns from knowledge. Interestingness measures for associations rules include support and confidence.

After the pattern evaluation phase, visualization and knowledge representation techniques can be used to present the mined knowledge to the users. However, as can be seen in Figure 2.1, the KDD process may iterate many times over previous steps and the process usually requires a great deal of experimentation.

2.1.2 A Glance at Data Mining Tasks

In this section, we briefly describe the main idea behind the primary mining tasks [25, 41, 55, 65, 26]. As pointed out by Chen et al. [25], data mining tasks can be classified according to the following criteria: (a) the kinds of databases to work on, (b) the knowledge to be mined (e.g., association rules), and (c) the techniques to be utilized (e.g., data-driven or query-driven). In this thesis, we focus on the second category (the knowledge to be mined).

Summarization: Also called characterization, summarization refers to general characteristics or features of a target class of data. Sometimes, the goal is to simply extract compact patterns that describe a subset of data. The data corresponding to the user-specified class or subset are collected by a database query. For example, to analyze the characteristics of certain products whose sales increased by 15% in the last year, the data related to such products can be collected by executing an SQL query. In general, the summary data are typically generated using basic statistics or by aggregation in OLAP (e.g., roll-up operation) and can be presented in various graphical forms, such as pie charts or bar charts.

Predictive Modeling: The goal of this mining task is to predict some attributes in a database based on other attributes. The target attribute is called *class*, i.e., the dependent variable in statistics terminology. If the attribute being predicted is a numeric variable (e.g., salary), then the prediction problem is a regression one. In contrast, if the class label attribute is categorical, the task at hand is called classification. In both cases, a set of data is taken as input, and a model (a pattern or a set of patterns) is generated. This model can be used to predict values of the class for new data. For instance, given a dataset, only a part of it is typically used to generate a predictive model. This part is referred to as the training dataset. The individual tuples making up the training dataset are referred to as training samples and are randomly selected from the sample population. Since the class label of each training sample is provided, this step is known as supervised learning. The remaining part, which is called the testing set, is reserved for evaluating the predictive performance of the learned model. The testing set is used to estimate the performance of the model on new (unseen data), i.e., to estimate the validity of the patterns on new data. Classification and regression have numerous applications including credit approval, medical diagnosis, and selective marketing.

Clustering: Also known as segmentation, clustering is concerned with grouping objects into classes of similar objects. Given a dataset, the task of clustering is to partition the data into new classes (clusters). The goal is to achieve high similarity between objects within individual clusters (interclass similarity) and low similarity between objects that belong to different clusters (intraclass similarity). Unlike predictive modeling that analyzes class-labeled data objects, clustering analyzes data objects without consulting a known class label. For this reason, clustering is also known as unsupervised learning. Clustering plays an outstanding role in data mining applications such as scientific data exploration, marketing, medical diagnostics, and computational biology. In Section 2.2, we discuss data structures (e.g., data matrix and dissimilarity matrix) for clustering, which are relevant to the PPDM methods we propose in this thesis.

Association Rules: Association analysis is the discovery of association rules describing interesting relationships among items grouped together in a sufficient number of examples. Market basket analysis has been a strong motivation for the development of association rule mining. The task of finding association rules is typically performed in two steps. First, all frequent itemsets are found, where an itemset is said to be frequent if it appears in at least a given percentage σ (called support) of all transactions. Then, associations rules are found of the form $X \Rightarrow Y$, where X and Y are frequent itemsets. Strong association rules are derived from frequent itemsets and constrained by a minimum confidence φ, i.e., the percentage of transactions containing X that also contain Y. In Section 2.3, we formally define the support-confidence framework for association rules and review other interestingness measures besides support and confidence. We also describe the process of protecting sensitive knowledge (sensitive association rules) in transactional databases.

Other Data Mining Tasks: The above data mining tasks have received most attention within the data mining field. Algorithms for performing such tasks are typically included in data mining tools [148]. Apart from these data mining tasks, Han and Kamber [65] include other tasks, such as evolution analysis and outlier detection. We briefly describe them as follows: (a) *Evolution analysis* describes and models regularities or trends whose behaviour changes over time. It includes change and deviation detection [55], which focuses on discovering the most significant changes in the data from previously measured or normative values; (b) *Outlier detection* is concerned with finding data objects that do not fit the general behaviour or model of data. Such data objects are called outliers. Outliers can be of interest in fraud detection, for example. They can be found by looking for objects that are a substantial distance away from any of the clusters in the data or show large differences from the average characteristics of objects in a group.

2.1.3 Data Mining versus Statistical Methods

Although statistical techniques are fundamental to data mining, such techniques alone may not be sufficient to address some of the more challenging issues in data mining, especially those arising from massive datasets [66]. When large or very large datasets are taken into account, it is not straightforward to know the facts about the data. Doing so requires sophisticated search and examination methods to make features understandable which would be readily apparent in small datasets. Indeed, the most fundamental difference between classical statistical applications and data mining concerns the size of the dataset [54, 66].

To a conventional statistician, a large dataset may contain a few hundred data points. However, from data mining's viewpoint, many millions or even billions of data points are not unexpected. Such large databases are common in several real world applications today. For instance, the American retailer Wal-Mart makes over 20 million transactions daily, and it constructed an 11 terabyte database of customer transactions in 1998 [113]. Another example is reported in [54], in which the Sloan Digital Sky Survey created a raw observational dataset of 40 terabytes. With datasets of this size come problems beyond those traditionally considered by statisticians.

To deal with massive datasets, miners could apply some techniques such as *sampling*, *segmentation*, or *summarization* [66, 26]. Sampling can be used as a data reduction technique since it allows a large dataset to be represented by a much smaller random sample of the data. In a sense, whether sampling is used or not distinguishes statistical analysis from data mining. While statistical analysis is applied to a set of sample data for analysis, in data mining it is preferable that the entire set of data is used for exploration. The reason is that since data mining is a process seeking the unexpected, it does not try to answer questions that were specified before data were collected. On one hand, sampling may be ideally suited to the purposes for which the data were collected but not adequate for data mining uses.

Segmentation helps to keep the size of the data under control. In this case, the data are divided into segments, and each segment is analyzed separately. The larger the dataset, the richer its patterns content. However, if someone analyzes a large dataset, patterns from different data segments begin to dilute each other and the number of useful patterns decreases.

Apart from sampling and segmentation, summarization may be used to reduce data sizes. In summarization, individual data elements are replaced by the summary data. For instance, individual sales from each store are now replaced by the total sales from all stores. However, summarization can also cause problems since the summarization of the same dataset with two samplings may produce the same result. In addition, the summarization of the same dataset with two methods may produce two different results. Therefore, summarization should be used with caution.

Clearly, various problems arise from the difficulties of accessing very large datasets. The most serious is the fact that datasets are constantly evolving. Thus, distributed or evolving data can multiply the size of the dataset, changing the nature of the problems requiring solution. While the size of a dataset may lead to difficulties, so also may other properties not often found in standard statistical applications.

Another fundamental difference between data mining and statistics is that data mining relies on exploratory analysis, while statistics is concerned with confirmatory analysis. In confirmatory analysis, one has a hypothesis and either confirms or refutes it. However, the bottleneck for confirmatory analysis is the shortage of hypotheses on the part of the analyst [66]. In exploratory analysis, interesting patterns are discovered without the user thinking of the relevant questions first. In large databases, there are so many patterns that the user can never practically think of the right questions to ask.

2.1.4 Data Stores for Mining

Data mining should be applicable to any kind of information repository [65]. The most common types of data repositories include flat files (e.g., transactional databases), relational databases, and data warehouses.

Usually, applications in the real world contain various types of entities involved in multiple tables in a database management system (DBMS) [42]. A DBMS consists of a collection of interrelated data, known as a database, and a set of software programs to manage and access the data. The software programs contain mechanisms for the definition of database structures: (a) for data storage; (b) for concurrent, shared, or distributed data access; and (c) for ensuring the consistency and security of the information stored [117]. A relational database is a collection of tables, each of which is assigned a unique name. Each table consists of a set of attributes (columns or fields) and usually stores a large set of tuples (records or rows). Each tuple in a relational table represents an object identified by a unique key and described by a set of attribute values.

In many cases, data from relational databases is augmented with summary information covering a long period of time. Such data are consolidated in much larger repositories called warehouses [117]. A data warehouse is an integrated repository derived from multiple source (operational and legacy) databases which in principle can provide views[2] of the data that are not practical for other kinds of databases [65, 26]. A data warehouse is usually modeled by a multidimensional database structure in which each dimension corresponds to

[2]A view is a table whose rows are not explicitly stored in the database, i.e., it temporarily relates information from many tables together into a virtual relation.

an attribute or a set of attributes in relational tables, with each cell storing the value of some aggregate measure. For instance, a supermarket chain may want to compare sales trends across regions at the level of products, broken down by months, and by class of store within a region. Such views are often precomputed and stored in special data stores, which are called multi-dimensional databases, that provide a multi-dimensional front-end to the underlying relational database. Figure 2.2 shows an example of locations, items, and sales represented as fragments of relations, while Figure 2.3(a) shows an example of a multi-dimensional view for a hiking equipment store.

locid	city	state	country
1	Jasper	AB	Canada
2	Boston	MA	USA
3	Toronto	ON	Canada

Locations

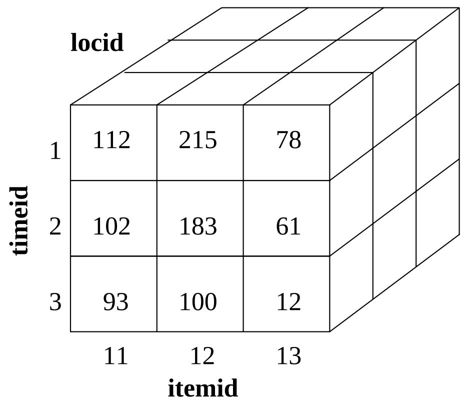
itemid	iname	category	price
11	Boots	Hiking	125
12	Sunglasses	Skiing	120
13	Sunscreen	Body care	20

Items

itemid	timeid	locid	sales
11	1	1	112
11	2	1	102
11	3	1	93
12	1	1	215
12	2	1	183
12	3	1	100
13	1	1	78
13	2	1	61
13	3	1	12

Sales

Figure 2.2: Locations, items, and sales represented as fragments of relations.



(A)

TID	List of Items
100	Ski pants, T−shirt, Gloves
200	Hiking boots, Sunglasses
300	Jackets, Sunglasses
400	Jackets, T−shirt, Gloves
500	T−shirt, Hiking boots, Gloves, Jackets
600	Sunglasses, T−shirt, Gloves
700	Sunglasses

(B)

Figure 2.3: (a): An example multi-dimensional view of relational data; (b): A sample transactional database for a hiking equipment store.

Because of the integration of multiple sources of data, data warehouses are capable of handling some problems associated with data integration [53, 117]. These problems include dealing with multiple formats, multiple database management systems, distributed databases, unifying data representation, and data cleaning.

Once the data are transformed and pre-processed into forms appropriate for mining, the pattern discovery can be done directly either from relational databases or data warehouses. However, the input of a data mining algorithm is most commonly a single flat table comprising a number of variables or attributes (columns), and records or objects (rows). In general, each row represents an object and columns represent properties of objects. Such a table represents the matrix form that has been traditionally used in statistics. One particular example of a single flat file is a transactional database, which is widely used in association rule mining. A transactional database is a relation consisting of transactions in which each transaction t is characterized by an ordered pair, defined as $t = \langle \mathit{TID},\ \mathit{list\_of\_elements} \rangle$, where *TID* is a unique transaction identifier number and *list_of_elements* is a list of items making up the transaction [26]. For instance, in market basket analysis, a transactional database is composed of business transactions in which the list of elements represents items purchased in a store. In general, transactional databases require the transformation of relational data into a single table. Miners are then able to look for patterns in this table. Figure 2.3(b) shows an example of a transactional database for a store that sells hiking equipment.

2.2 The Basics of Clustering Analysis

2.2.1 The Major Distance-Based Clustering Methods

There exist a large number of clustering algorithms in the literature. The choice of clustering algorithm depends on the type of data available and on the particular purpose and application [44]. In this thesis, we focus on methods for protecting privacy when distance-based clustering algorithms are used to mine a transformed database.

We review the major distance-based clustering methods as follows. More details regarding clustering methods can be found in [152].

Partitioning methods: given a database of n objects, a partitioning method first creates an initial set of k partitions, where the parameter k is the number of partitions to construct, and uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another. In the end, the partitioning method constructs k partitions of the data, where each partition represents a cluster and $k \leq n$. The most classical and popular partitioning methods are k-means [93] and k-medoid [86], where each cluster is represented by the gravity centre of the cluster in k-means or by the central objects of the cluster in k-medoid.

Hierarchical methods : these methods create a hierarchical decomposition of the given set of data objects. The goal is to produce a hierarchical series of nested clusters, ranging from clusters of individual points at the bottom to an all-inclusive cluster at the top. A diagram called a dendogram graphically represents this hierarchy. These methods can be classified as being either agglomerative (bottom-up) or divisive (top-down) based on how the hierarchy decomposition is formed. To compensate for the rigidity of merge or split, the quality of hierarchical agglomeration can be improved by analyzing object linkages at each hierarchical partitioning, such as in CURE [64] and Chameleon [84], or integrating other clustering techniques, such as iterative relocation as in BIRCH [154].

Density-based methods : these methods cluster objects based on the notion of density. The main idea behind these methods is to continue growing the given cluster as long as the density, i.e., the number of data points in the neighborhood exceeds some threshold. In other words, for each data point within a given cluster, the neighborhood of a given radius has to contain a minimum number of points. Examples of these methods are DBSCAN that grows clusters according to the density of neighborhood objects [43], and DENCLUE which grows clusters according to some density function [65].

Grid-based methods : these methods quantize the object space into a finite number of cells that form a grid structure. The main advantage of this approach is its fast processing time, which is typically independent of the number of data objects and dependent only on the number of cells in each dimension in the quantized space. STING [143] is a typical example of grid-based method that relies on statistical information stored in grid cells. CLIQUE [5] and WaveCluster [131] are two clustering algorithms that are both grid-based and density-based.

2.2.2 Data Matrix

Objects (e.g., individuals, patterns, events) are usually represented as points (vectors) in a multi-dimensional space. Each dimension represents a distinct attribute describing the object. Thus, objects are represented as an $m \times n$ matrix D, where there are m rows, one for each object, and n columns, one for each attribute. This matrix may contain binary, categorical, or numerical attributes. It is referred to as a data matrix, represented as follows:

$$D = \begin{bmatrix} a_{11} & \dots & a_{1k} & \dots & a_{1n} \\ a_{21} & \dots & a_{2k} & \dots & a_{2n} \\ \vdots & & \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mk} & \dots & a_{mn} \end{bmatrix} \tag{2.1}$$

The attributes in a data matrix are sometimes transformed before being used. The main reason is that different attributes may be measured on different scales (e.g., centimeters and kilograms). When the range of values differs widely from attribute to attribute, attributes with large range can influence the results of the cluster analysis. For this reason, it is common to standardize the data so that all attributes are on the same scale.

There are many methods for data normalization [65]. We review only two of them in this section: *min-max normalization* and *z-score normalization.*

Min-max normalization performs a linear transformation on the original data. Each attribute is normalized by scaling its values so that they fall within a small specific range, such as 0.0 and 1.0. Min-max normalization maps a value v of an attribute A to v' as follows:

$$v' = \frac{v - min_A}{max_A - min_A} \times (new\_max_A - new\_min_A) + new\_min_A \tag{2.2}$$

where min_A and max_A represent the minimum and maximum values of an attribute A, respectively, while new_min_A and new_max_A are the new range in which the normalized data will fall.

When the actual minimum and maximum of an attribute are unknown, or when there are outliers that dominate the min-max normalization, z-score normalization (also called zero-mean normalization) should be used. In z-score normalization, the values for an attribute A are normalized based on the mean and the standard deviation of A. A value v is mapped to v' as follows:

$$v' = \frac{v - \overline{A}}{\sigma_A} \tag{2.3}$$

where $\overline{A}$ and σ_A are the mean and the standard deviation of the attribute A, respectively.

2.2.3 Dissimilarity Matrix

A dissimilarity matrix stores a collection of proximities that are available for all pairs of objects. This matrix is often represented by an $m \times m$ table. In (2.4), we can see the dissimilarity matrix D_M corresponding to the data matrix D in (2.1), where each element $d(i,j)$ represents the difference or dissimilarity between objects i and j.

$$D_M = \begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(m,1) & d(m,2) & \dots & \dots & 0 \end{bmatrix} \tag{2.4}$$

In general, $d(i,j)$ is a nonnegative number that is close to zero when the objects i and j are very similar to each other, and becomes larger the more they differ.

To calculate the dissimilarity between objects i and j one could use either the distance measure in Equation (2.5) or in Equation (2.6), or others, where $i = (x_{i1}, x_{i2}, ..., x_{in})$ and $j = (x_{j1}, x_{j2}, ..., x_{jn})$ are n-dimensional data objects.

$$d(i,j) = \left[\ \textstyle\sum_{k=1}^{n} |x_{ik} - x_{jk}|^2 \ \right]^{1/2} \tag{2.5}$$

$$d(i,j) = \sum_{k=1}^{n} |x_{ik} - x_{jk}| \tag{2.6}$$

The metric in Equation (2.5) is the most popular distance measure called Euclidean distance, while the metric in Equation (2.6) is known as Manhattan or city block distance. Both Euclidean distance and Manhattan distance satisfy the following constraints:

- $d(i,j) \geq 0$: distance is a nonnegative number.
- $d(i,i) = 0$: the distance of an object to itself.
- $d(i,j) = d(j,i)$: distance is a symmetric function.
- $d(i,j) \leq d(i,k) + d(k,j)$: distance satisfies the triangular inequality.

In case of binary variables (attributes), one can compute the dissimilarity between objects i and j by using the Jaccard coefficient [145, 65]. The Jaccard coefficient is defined as:

$$d(i,j) = \frac{r+s}{q+r+s} \tag{2.7}$$

where r is the number of variables that equal 1 for object i but that are 0 for object j; s is the number of variables that equal 0 for object i but equal 1 for object j, and q is the number of variables that equal 1 for both objects i and j. This metric assumes that variables are asymmetric, i.e., the outcomes of the states are not equally important, such as positive and negative outcomes of a disease test.

Nominal variables can be encoded either by asymmetric binary variables or by mapping them to a numerical domain. However, if a dataset contains mixed variables, a more preferable approach is to process all variable types together performing a single cluster analysis. Combining the different variables into a single dissimilarity matrix brings all of the meaningful variables onto a common scale of the interval [0.0, 1.0]. For a dataset containing p variables of mixed types, the dissimilarity $d(i,j)$ between objects i and j is defined as [65]:

$$d(i,j) = \frac{\sum_{f=1}^{p} \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^{p} \delta_{ij}^{(f)}} \tag{2.8}$$

where the indicator $\delta_{ij}^{(f)} = 0$ if either: (a) x_{if} or x_{jf} is missing; or (b) $x_{if} = x_{jf} = 0$ and variable f is asymmetric binary; otherwise $\delta_{ij}^{(f)} = 1$. The contribution of variable f to the dissimilarity between i and j, $d_{ij}^{(f)}$, is computed dependent on its type:

- If f is binary or nominal: $d_{ij}^{(f)} = 0$ if $x_{if} = x_{jf}$; otherwise $d_{ij}^{(f)} = 1$.
- If f is interval-based: $d_{ij}^{(f)} = \frac{|x_{if} - x_{jf}|}{max_h x_{hf} - min_h x_{hf}}$, where h runs over all non-missing objects for variable f.

One advantage of using Equation (2.8) is that the dissimilarities between objects can be computed even when the variables describing the objects are of different types. Moreover, the dissimilarities are already normalized.

2.3 The Basics of Association Rule Mining

2.3.1 The Support-Confidence Framework

One of the most studied problems in data mining is the process of discovering association rules from large databases. Most of the existing algorithms for association rules rely on the support-confidence framework introduced in [6, 8].

Formally, association rules are defined as follows: Let $I = \{i_1,...,i_n\}$ be a set of literals, called items. Let D be a database of transactions, where each transaction t is an itemset such that $t \subseteq I$. A unique identifier, called *TID*, is associated with each transaction. A transaction t supports X, a set of items in I, if $X \subset t$. An association rule is an implication of the form $X \Rightarrow Y$, where $X \subset I$, $Y \subset I$ and $X \cap Y = \emptyset$. Thus, we say that a rule $X \Rightarrow Y$ holds in the database D with *confidence* φ if $\frac{|X \cup Y|}{|X|} \geq \varphi$, where $|A|$ is the number of occurrences of the set of items A in the set of transactions D. Similarly, we say that a rule $X \Rightarrow Y$ holds in the database D with *support* σ if $\frac{|X \cup Y|}{N} \geq \sigma$, where N is the number of transactions in D.

While the support is a measure of the frequency of a rule, the confidence is a measure of the strength of the relation between sets of items. A survey of algorithms for association rules can be found in [71].

2.3.2 Interestingness Measures

Interesting rules are defined as rules describing surprising uncommon situations [65]. Support and confidence, reviewed in the previous section, are the most basic measures of rule interestingness. However, in many cases support and confidence are not sufficient. Many other measures have been proposed in the literature [135] for ranking association patterns according to their degree of interestingness. We describe some of those measures as follows:

Lift: also known as interest [20] and strength [39], *lift* is defined as $lift(X \Rightarrow Y) = support(X \cup Y)/(support(X) \times support(Y))$. Lift only measures co-occurrence not implication, in that it is completely symmetric. Based on statistical independence, if lift is equal to 1 then the condition X and the conclusion Y are independent. If lift is greater than 1 then the condition is associated with the conclusion. If lift of a rule is between 0 and 1 then the condition is negatively associated with the conclusion.

Coverage: describes the importance of dependency [132]. This measure is defined as $coverage(X \Rightarrow Y) = support(X \cup Y)/support(Y)$. A rule has coverage c if $c\%$ of all transactions that contain Y also contain X. Its values belong to the interval [0, 1]. Note that the formulas of confidence and coverage are very similar.

Conviction: this measure was derived from the implication definition [20]. Logically, $X \Rightarrow Y$ can be rewritten as $\sim(A \wedge \sim B)$. This measure shows the level of dependence between A and $\sim B$. After some transformation, conviction is defined as $conviction(X \Rightarrow Y) = support(X) \times (1 - support(Y))/(support(X) - support(X \cup Y))$. Conviction values are in the interval [0, $+\infty$]. If conviction is equal to 1, the antecedent and the consequent are independent. Conviction is different from confidence because it does not suffer from the the same problem of producing misleading rules.

Piatetsky-Shapiro: this measures was introduced in [111], and it is defined as $PS(X \Rightarrow Y) = support(X \cup Y) - support(X) \times support(Y)$. Absolute value of this measure shows dependence between antecedent and consequent. Statistically independence occurs at $PS = 0$. The values for this measure fall in the interval [-0.25, 0.25].

Other Measures: other measures to determine the interestingness of association patterns can be found in [135]. The work presents a comparative study of interestingness measures for ranking association patterns. The key finding of this study was that there is no measure that is consistently better than the others in all cases. However, there are situations in which many of these measures are highly correlated with each other (e.g., when support-based pruning is used). For example, measures such as Laplace, Jaccard, Piatetsky-Shapiro, coverage, confidence, and Cosine IS behave similarly in the region of low support values, which typically occurs in large databases.

2.3.3 Sensitive Rules and Sensitive Transactions

Protecting sensitive knowledge in transactional databases is the task of hiding a group of association rules which contains sensitive knowledge. We refer to these rules as sensitive association rules and define them as follows:

Definition 1 (Sensitive Association Rules) *Let D be a transactional database, R be a set of all association rules that can be mined from D based on a minimum support σ, and $Rules_H$ be a set of decision support rules that need to be hidden according to some security policies. A set of association rules, denoted by S_R, is said to be sensitive iff $S_R \subset R$ and S_R would derive the set $Rules_H$. $\tilde{}S_R$ is the set of non-sensitive association rules such that $\tilde{}S_R \cup S_R = R$.*

A group of sensitive association rules is mined from a database D based on a special group of transactions. We refer to these transactions as sensitive transactions and define them as follows:

Definition 2 (Sensitive Transactions) *Let T be a set of all transactions in a transactional database D and S_R be a set of sensitive association rules mined from D. A set of transactions is said to be sensitive, denoted S_T, if $S_T \subset T$ and $\forall\ t \in S_T$, $\exists\ sr \in S_R$ such that items(sr) $\subseteq t$.*

2.3.4 The Process of Protecting Sensitive Knowledge

The process of protecting sensitive knowledge in transactional databases is composed of two major steps: *identifier suppression* and *sanitization*, as can be seen in Figure 2.4.

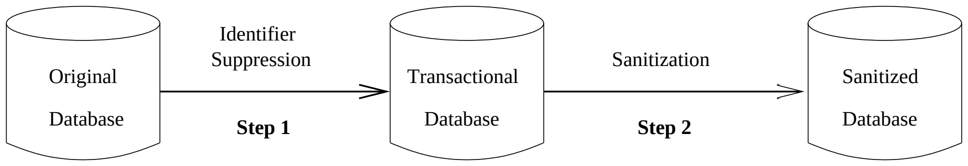


Figure 2.4: Major steps of the process of protecting sensitive knowledge.

Step 1: Identifier Suppression

The first step of the sanitization process refers to the suppression of identifiers (e.g., IDs, names, etc) from the data to be shared. The procedure of removing identifiers allows database owners to disclose purchasing behavior of customers without disclosing their identities [87]. To accomplish that, database owners must transform the data into forms appropriate for mining.

After removing identifiers, the selected data which are subjected to mining, can be stored in a single table, also called a transactional database. Figure 2.3B shows an example of a transactional database for a hiking equipment store. As can be seen in that Figure, the transactional database does not contain personal information, but only costumers' buying activities.

Although the deletion of identifiers from the data is useful to protect personal information, we do not argue that this procedure ensures full privacy at all. In many cases, it is very difficult to extract the specific identity of one or more costumers from a transactional database, even combining the transactions with other data. However, a specific transaction may contain some items that can be linked with other datasets to re-identify an individual or and entity [124, 123, 134].

Once the data is transformed into a transactional database, the process of hiding sensitive rules from this transactional database is the next step to be pursued.

Step 2: Sanitization

After removing the identifiers from the data, the goal now is to efficiently hide sensitive knowledge represented by sensitive rules.

In most cases, the notion of sensitive knowledge may not be known in advance. That is why the process of identifying sensitive knowledge requires human evaluation of the intermediate results before the sharing of data for mining. In this context, sensitive knowledge is represented by a special group of rules referred to as sensitive association rules.

An efficient way to hide sensitive rules is by transforming a transactional database into a new one that conceals the sensitive rules while preserving most of the non-sensitive ones. The released database is called a sanitized database. To accomplish that, the sanitization process acts on the data modifying some transactions. In some cases, a number of items are deleted from a group of transactions (sensitive transactions) with the purpose of hiding the sensitive rules derived from those transactions. In doing so, the support of such sensitive rules are decreased below a certain disclosure threshold denoted by ψ. Another way to hide sensitive rules is to add new items to some transactions to alter (decrease) the confidence of sensitive rules. For instance, in a rule $X \rightarrow Y$, if the items are added to the antecedent part X of this rule in transactions that support X and not Y, then the confidence of such a rule is decreased. Clearly, the sanitization process slightly modifies some data, but this is perfectly acceptable in some real applications [12, 37, 126].

Although the sanitization process is performed to hide sensitive rules only, the side effect of this process also hides some non-sensitive ones. By deleting some items in a group of transactions, the support or even the confidence of non-sensitive rules are also decreased. Therefore, sanitizing algorithms must focus on hiding sensitive rules and, at the same time, reducing the side effect on the non-sensitive rules as much as possible.

2.4 The Basics of Dimensionality Reduction

In many applications of data mining, the high dimensionality of the data restricts the choice of data processing methods. Examples of such applications include market basket data, text classification, and clustering. In these cases, the dimensionality is large due to either a wealth of alternative products, a large vocabulary, or an expressive number of attributes to be analyzed in Euclidean space, respectively.

When data vectors are defined in a high-dimensional space, it is computationally intractable to use data analysis or pattern recognition algorithms which repeatedly compute similarities or distances in the original data space. It is therefore necessary to reduce the dimensionality before, for instance, clustering the data [85, 57].

The goal of the methods designed for dimensionality reduction is to map d-dimensional objects into k-dimensional objects, where $k \ll d$ [89]. These methods map each object to a point in a k-dimensional space minimizing the stress function:

$$stress^2 = (\sum_{i,j} (\hat{d}_{ij} - d_{ij})^2)/(\sum_{i,j} {d_{ij}}^2) \tag{2.9}$$

where d_{ij} is the dissimilarity measure between objects i and j in a d-dimensional space, and $\hat{d}_{ij}$ is the dissimilarity measure between objects i and j in a k-dimensional space. The function *stress* gives the relative error that the distances in k-d space suffer from, on the average.

2.4.1 Methods for Dimensionality Reduction

There exists a number of methods for reducing the dimensionality of data, ranging from different feature extraction methods to multidimensional scaling. The feature extraction methods are often performed according to the nature of the data, and therefore they are not generally applicable in all data mining tasks [85]. The multidimensional scaling (MDS) methods, on the other hand, have been used in several diverse fields (e.g, social sciences, psychology, market research, and physics) to analyze subjective evaluations of pairwise similarities of entities [151].

Another alternative for dimensionality reduction is to project the data onto a lower-dimensional orthogonal subspace that captures as much of the variation of the data as possible. The best and most widely way to do so is Principal Component Analysis [59]. Principal component analysis (PCA) involves a mathematical procedure that transforms a number of (possibly) correlated variables into a smaller number of uncorrelated variables called principal components. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible. Unfortunately, PCA is quite expensive to compute for high-dimensional datasets.

Although the above methods have been widely used in data analysis and compression, these methods are computationally costly and if the dimensionality of the original data points is very high it is infeasible to apply these methods to dimensionality reduction.

Random projection (RP) has recently emerged as a powerful method for dimensionality reduction. The accuracy obtained after the dimensionality has been reduced using random projection is almost as good as the original accuracy [85, 1, 16]. The key idea of random projection arises from the Johnson-Lindenstrauss lemma [75]: "if points in a vector space are projected onto a randomly selected subspace of suitably high dimension, then the distances between the points are approximately preserved."

Lemma 1 *([75]). Given $\epsilon > 0$ and an integer n, let k be a positive integer such that $k \geq k_0 = O(\epsilon^{-2} log\, n)$. For every set P of n points in $\Re^d$ there exists $f : \Re^d \to \Re^k$ such that for all $u, v \in P$*

$$(1-\epsilon) \parallel u-v \parallel^2 \leq \parallel f(u)-f(v) \parallel^2 \leq (1+\epsilon) \parallel u-v \parallel^2.$$

The classic result of Johnson and Lindenstrauss [75] asserts that any set of n points in d-dimensional Euclidean space can be embedded into k-dimensional space, where k is logarithmic in n and independent of d.

We will refer to the Johnson and Lindenstrauss' lemma as JL-lemma. In the last few years, this lemma has been useful in solving a variety of problems. The rationale is that by providing a low dimensional representation of the data, JL-lemma speeds up certain algorithms drastically, in particular algorithms whose run-time depends exponentially on the dimension of the working space. At the same time, the provided guarantee regarding pairwise distances often allows one to establish that the solution found by working in the low dimensional space is a good approximation to the solution found by working in the high dimensional space. We give a few example as follows. In the work presented in [110], Papadimitriou et al. proved that embedding a pointset A into a low-dimensional space can significantly speed up the computation of a low rank approximation to A without significantly affecting its quality. In [67], Indyk and Motwani showed that JL-lemma is useful in solving the ϵ-approximate nearest neighbor problem (in a pointset P) in which one is to answer queries such as "Given an arbitrary point x, find a point $y \in P$, such that for every point $z \in P$, $\parallel x-z \parallel \geq \parallel (1-\epsilon) \parallel x-y \parallel$." In a different approach, Schulman [130] used JL-lemma as part of an approximation algorithm for the version of clustering in which one seeks to minimize the sum of the squares of intracluster distances.

In this research, we focus on random projection for privacy-preserving clustering. Our motivation for exploring random projection is based on the following aspects. First, it is a general data reduction technique. In contrast to the other methods, such as PCA, RP does not use any defined interestingness criterion to optimize the projection. Second, random projection has shown to have promising theoretical properties for high dimensional data clustering [57, 16]. Third, despite its computational simplicity, random projection does not introduce a significant distortion in the data. Finally, the dimensions found by random projection are not a subset of the original dimensions, which is relevant for privacy preservation. We provide the background of random projection in the next section.

2.4.2 Random Projection

A random projection from d dimensions to k dimensions is a linear transformation represented by a $d \times k$ matrix R, which is generated by first setting each entry of the matrix to a value drawn from an i.i.d. $\sim N(0,1)$ distribution and then normalizing the columns to unit length. Given a d-dimensional dataset represented as an $n \times d$ matrix D, the mapping $D \times R$ results in a reduced-dimension dataset D', i.e.,

$$D'_{n\times k} = D_{n\times d} R_{d\times k} \tag{2.10}$$

Random projection is computationally very simple. Given the random matrix R and projecting the $n \times d$ matrix D into k dimensions is of the order $O(ndk)$, and if the matrix D is sparse with about c nonzero entries per column, the complexity is of the order $O(cnk)$ [110].

After applying random projection to a dataset, the distance between two d-dimensional vectors i and j is approximated by the scaled Euclidean distance of these vectors in the reduced space as follows:

$$\sqrt{d/k} \parallel R_i - R_j \parallel \tag{2.11}$$

where d is the original and k the reduced dimensionality of the dataset. The scaling term $\sqrt{d/k}$ takes into account the decrease in the dimensionality of the data.

To satisfy the JL-Lemma, the random matrix R must hold the follow constraints:

- The columns of the random matrix R are composed of orthonormal vectors, i.e, they have unit length and are orthogonal.
- The elements r_{ij} of R have zero mean and unit variance.

Clearly, the choice of the random matrix R is one of the key points of interest. The elements r_{ij} of R are often Gaussian distributed, but this need not to be the case. Achlioptas [1] showed that the Gaussian distribution can be replaced by a much simpler distribution, as follows:

$$r_{ij} = \sqrt{3} \times \begin{cases} +1 & \textit{with probability} \quad 1/6 \\ 0 & \textit{with probability} \quad 2/3 \\ -1 & \textit{with probability} \quad 1/6 \end{cases} \tag{2.12}$$

In fact, practically all zero mean, unit variance distributions of r_{ij} would give a mapping that still satisfies the Johnson-Lindenstrauss lemma. Achlioptas' result means further computational savings in database applications since the computations can be performed using integer arithmetics.

Chapter 3

Toward Foundations of Privacy-Preserving Data Mining

If you have built castles in the air, your work need not be lost;
that is where they should be. Now put the foundations under them.
– Henry David Thoreau (1817 - 1862)

Privacy-preserving data mining (PPDM) is one of the newest trends in privacy and security research. It is driven by one of the major policy issues of the information era - *the right to privacy*. Although this research field is very new, we have already seen great interests in it: a) the recent proliferation of PPDM techniques is evident; b) the interest from academia and industry has grown quickly; and c) separate workshops and conferences devoted to this topic have emerged in the last few years.

In this chapter, we take some steps toward foundations for further research in PPDM, as follows. In Section 3.1, we discuss the different meanings of privacy and how privacy can be violated in data mining. The historical roots and the basis of PPDM are introduced in Section 3.2. In Section 3.3, we analyze the implications of the Organization for Economic Cooperation and Development (OECD) data privacy principles concerning knowledge discovery. We then show that some PPDM principles can be adopted from the OECD privacy guidelines. We also suggest some policies for PPDM based on instruments accepted worldwide. In Section 3.4, we suggest some desirable requirements for the development and for deployment of technical solutions. These requirements are related to industrial initiatives.

A preliminary version of this chapter appeared in a paper presented in the 3nd Workshop on Data Mining Standards (DM-SSP 2004), in conjunction with KDD 2004 [108].

3.1 The Different Meanings of Privacy

3.1.1 Problems in Defining Privacy

Analyzing what right to privacy means is a fraut with problems, such as the exact definition of privacy, whether it constitutes a fundamental right, and whether people are and/or should be concerned with it. Several definitions of privacy have been given, and they vary according to context, culture, and environment. For instance, in an 1890 paper [144], Warren & Brandeis defined privacy as “the right to be alone.” Later, in a paper published in 1967 [146], Westin defined privacy as “the desire of people to choose freely under what circumstances and to what extent they will expose themselves, their attitude, and their behavior to others”. In [129], Schoeman defined privacy as “the right to determine what (personal) information is communicated to others” or “the control an individual has over information about himself or herself.” More recently, Garfinkel [60] stated that “privacy is about self-possession, autonomy, and integrity.” On the other hand, Rosenberg argues that privacy may not be a right after all but a taste [121]: “If privacy is in the end a matter of individual taste, then seeking a moral foundation for it – beyond its role in making social institutions possible that we happen to prize – will be no more fruitful than seeking a moral foundation for the taste for truffles.”

The above definitions suggest that, in general, privacy is viewed as a social and cultural concept. However, with the ubiquity of computers and the emergence of the Web, privacy has also become a digital problem [119]. With the Web revolution and the emergence of data mining, privacy concerns have posed technical challenges fundamentally different from those that occurred before the information era. In the information technology era, privacy refers to the right of users to conceal their personal information and have some degree of control over the use of any personal information disclosed to others [32, 2, 73].

Clearly, the concept of privacy is often more complex than realized. In particular, in data mining, the definition of privacy preservation is still unclear, and there is very little literature related to this topic. A notable exception is the work presented in [29], in which PPDM is defined as “getting valid data mining results without learning the underlying data values.” However, at this point, each existing PPDM technique has its own privacy definition. Our primary concern about PPDM is that mining algorithms are analyzed for the side effects they incur in data privacy. Therefore, our definition for PPDM is close to those definitions in [129, 29] – *PPDM encompasses the dual goal of meeting privacy requirements and providing valid data mining results.* Our definition emphasizes the dilemma of balancing privacy preservation and knowledge disclosure.

3.1.2 Privacy Violation in Data Mining

Understanding privacy in data mining requires understanding how privacy can be violated and the possible means for preventing privacy violation. In general, one major factor contributes to privacy violation in data mining: *the misuse of data*.

Users' privacy can be violated in different ways and with different intentions. Although data mining can be extremely valuable in many applications (e.g., business, medical analysis, etc), it can also, in the absence of adequate safeguards, violate informational privacy. Privacy can be violated if personal data are used for other purposes subsequent to the original transaction between an individual and an organization when the information was collected.

One of the sources of privacy violation is called data magnets [119]. Data magnets are techniques and tools used to collect personal data. Examples of data magnets include explicitly collecting information through on-line registration, identifying users through IP addresses, software downloads that require registration, and indirectly collecting information for secondary usage. In many cases, users may or may not be aware that information is being collected or do not know how that information is collected [36, 91]. Worse is the privacy invasion occasioned by secondary usage of data when individuals are unaware of "behind the scenes" uses of data mining techniques [74]. In particular, collected personal data can be used for secondary usage largely beyond the users' control and privacy laws. This scenario has led to an uncontrollable privacy violation not because of data mining itself, but fundamentally because of the misuse of data.

3.2 The Basis of Privacy-Preserving Data Mining

3.2.1 PPDM Landmarks

The debate on PPDM has received special attention as data mining has been widely adopted by public and private organizations. We have witnessed three major landmarks that characterize the progress and success of this new research area: *the conceptive landmark*, *the deployment landmark*, and *the prospective landmark*. We describe these landmarks as follows:

The Conceptive landmark characterizes the period in which central figures in the community, such as O'Leary [99, 100], Fayyad, Piatetsky-Shapiro and Smith [55, 112], and others [88, 31], investigated the success of knowledge discovery and some of the important areas where it can conflict with privacy concerns. The key finding was that knowledge discovery can open new threats to informational privacy and information security if not done or used properly. They highlighted the success of knowledge discovery and some of important areas where it could conflict with privacy concerns. Since then, the debate on PPDM has gained momentum.

The Deployment landmark is the current period in which an increasing number of PPDM techniques have been developed and have been published in refereed conferences. The information available today is spread over countless papers and conference proceedings[1]. The results achieved in the last years are promising and suggest that PPDM will achieve the goals that have been set for it.

The Prospective landmark is a new period in which directed efforts toward standardization occur. At this stage, there is no consent about what privacy preservation means in data mining. In addition, there is no consensus on privacy principles, policies, and requirements as a foundation for the development and deployment of new PPDM techniques. The excessive number of techniques is leading to confusion among developers, practitioners, and others interested in this technology. One of the most important challenges in PPDM now is to establish the groundwork for further research and development in this area.

3.2.2 Defining Privacy Preservation in Data Mining

In general, privacy preservation occurs in two major dimensions: users' personal information and information concerning their collective activity. We refer to the former as *individual privacy preservation* and the latter as *collective privacy preservation*, which is related to corporate privacy in [29].

Individual privacy preservation: The primary goal of data privacy is the protection of personally identifiable information. In general, information is considered personally identifiable if it can be linked, directly or indirectly, to an individual person. Thus, when personal data are subjected to mining, the attribute values associated with individuals are private and must be protected from disclosure. Miners are then able to learn from global models rather than from the characteristics of a particular individual.

Collective privacy preservation: Protecting personal data may not be enough. Sometimes, we may need to protect against learning sensitive knowledge representing the activities of a group. We refer to the protection of sensitive knowledge as collective privacy preservation. The goal here is quite similar to that one for statistical databases, in which security control mechanisms provide aggregate information about groups (population) and, at the same time, should prevent disclosure of confidential information about individuals. However, unlike as is the case for statistical databases, another objective of collective privacy preservation is to preserve (hide) strategic patterns that are paramount for strategic decisions, rather than minimizing the distortion of all statistics (e.g., bias and precision). In other words, the goal here is not only to protect personally identifiable information but also some patterns and trends that are not supposed to be discovered.

[1]The Privacy-Preserving Data Mining Site: http://www.cs.ualberta.ca/~oliveira/psdm/psdm_index.html

In the case of collective privacy preservation, organizations have to cope with some interesting conflicts. For instance, when personal information undergoes analysis, processes that produce new facts about users' shopping patterns, hobbies, or preferences, these facts could be used in recommender systems to predict or affect their future shopping patterns. In general, this scenario is beneficial to both users and organizations. However, when organizations share data in a collaborative project, the goal is not only to protect personally identifiable information but also to protect some strategic patterns. In the business world, such patterns are described as the knowledge that can provide competitive advantages, and therefore must be protected [138]. More challenging is to protect the knowledge discovered from confidential information (e.g., medical, financial, and crime information). The absence of privacy safeguards can equally compromise individuals' privacy. While violation of individual privacy is clear, violation of collective privacy can lead to violation of individual's privacy.

3.2.3 Characterizing Scenarios in PPDM

Before we describe the general parameters for characterizing scenarios in PPDM, let us consider two real-life motivating examples where PPDM poses different constraints:

Scenario 1: A hospital shares some data for research purposes (e.g., concerning a group of patients who have a similar disease). The hospital's security administrator may suppress some identifiers (e.g., name, address, phone number, etc) from patient records to meet privacy requirements. However, the released data may not be fully protected. A patient record may contain other information that can be linked with other datasets to re-identify individuals or entities [123]. How can we identify groups of patients with a similar disease without revealing the values of the attributes associated with them?

Scenario 2: Two or more companies have a very large dataset of records on their customers' buying activities. These companies decide to cooperatively conduct association rule mining on their datasets for their mutual benefit since this collaboration brings them an advantage over other competitors. However, some of these companies may not want to share some strategic patterns hidden within their own data (also called sensitive association rules) with the other parties. They would like to transform their data in such a way that these sensitive association rules cannot be discovered but others can be. Is it possible for these companies to benefit from such collaboration by sharing their data while preserving some sensitive association rules?

Note that the above scenarios describe different privacy preservation problems. Each scenario poses a set of challenges. For instance, scenario 1 is a typical example of individual's privacy preservation, while scenario 2 refers to collective privacy preservation.

How can we characterize scenarios in PPDM? One alternative is to describe them in terms of general parameters. In [30], some parameters are suggested as follows:

Outcome: Refers to the desired data mining results. For instance, someone may look for association rules identifying relationships among attributes, or relationships among customers' buying behaviors as in scenario 2, or may even want to cluster data as in scenario 1.

Data Distribution: How are the data available for mining: are they centralized or distributed across many sites? In the case of data distributed throughout many sites, are the entities described with the same schema in all sites (horizontal partitions), or do different sites contain different attributes for one entity (vertical partitions)?

Privacy Preservation: What are the privacy preservation requirements? If the concern is solely that values associated with an individual entity not be released (e.g., personal information), techniques must focus on protecting such information. In other cases, the notion of what constitutes "sensitive knowledge" may not be known in advance. This would lead to human evaluation of the intermediate results before making the data available for mining.

3.2.4 Model of Data Miners

In this thesis, we assume that there are two types of data miners. The first type refers to the conscious data miners. These miners always act legally in that they perform regular data mining tasks and would never intentionally breach the privacy of the data. On the other hand, malicious data miners would purposely breach the privacy in the data being mined. Malicious data miners come in many forms. We focus on a particular sub-class of malicious miners. That is, malicious data miners follow standards but are curious [61, 62]: they follow proper protocols and standard procedures, but they may perform some analysis (i.e., they are curious) to discover private information.

This kind of curious (nevertheless malicious) behavior is most common and has been widely adopted as an adversary model in the literature [153]. This is because, in reality, a workable system must benefit both the conscious and the malicious data miners. For example, in an online bookstore a data miner may use the association rules of purchase records to make recommendations to its customers (data providers). In addition, this data miner can also perform some analysis to discover private information. This data miner, as a long-term agent, requires large numbers of data providers to collaborate with. In other words, even a malicious data miner desires to build a reputation for trustworthiness. Thus, honest but curious behavior is an appropriate choice for many malicious data miners.

3.3 Designing Principles and Policies For PPDM

In this section, we highlight the privacy guidelines established by the Organization for Economic Cooperation and Development (OECD)[2], and analyze the implications of those guidelines in the context of PPDM.

3.3.1 The OECD Privacy Guidelines

World-wide, privacy legislation, policies, guidelines, and codes of conduct have been derived from the set of principles established in 1980 by the OECD. They represent the primary components for the protection of privacy and personal data, comprising a commonly understood reference point. A number of countries have adopted these principles as statutory law, in whole or in part. The OECD Privacy Guidelines outline the following basic principles:

1. **Collection Limitation Principle:** There should be limits to the collection of personal data and any such data should be obtained by lawful and fair means and, where appropriate, with the knowledge or consent of the data subject (consumer).

2. **Data Quality Principle:** Personal data should be relevant to the purposes for which they are to be used, and, to the extent necessary for those purposes, should be accurate, complete and up-to-date.

3. **Purpose Specification Principle:** The purposes for which personal data are collected should be specified not later than at the time of data collection and the subsequent use limited to the fulfillment of those purposes or such others as are not incompatible with those purposes and as are specified on each occasion of change of purpose.

4. **Use Limitation Principle:** Personal data should not be disclosed, made available or otherwise used for purposes other than those specified in accordance with [the Purpose Specification Principle] except: (a) with the consent of the data subject; or (b) by the authority of law.

5. **Security Safeguards Principle:** Personal data should be protected by reasonable security safeguards against such risks as loss or unauthorized access, destruction, use, modification, or disclosure of data.

6. **Openness Principle:** There should be a general policy of openness about developments, practices, and policies with respect to personal data. Means should be readily available for establishing the existence and nature of personal data, and the main purposes of their use, as well as the identity and usual residence of the data controller (e.g., a public or a private organization).

[2]Privacy Online: OECD Guidance on Policy and Practice: http://www.oecd.org/dataoecd/33/43/2096272.pdf

7. **Individual Participation Principle:** An individual should have the right: a) to obtain from a data controller, or otherwise, confirmation of whether or not the data controller has data relating to him; b) to have communicated to him, data relating to him within a reasonable time; at a charge, if any, that is not excessive; in a reasonable manner; and in a form that is readily intelligible to him; c) to be given reasons if a request made under subparagraphs (a) and (b) is denied, and to be able to challenge such denial; and d) to challenge data relating to him and, if the challenge is successful to have the data erased, rectified, completed, or amended.

8. **Accountability Principle:** A data controller should be accountable for complying with measures which give effect to the principles stated above.

3.3.2 The implications of the OECD Privacy Guidelines in PPDM

We now analyze the implications of the OECD principles in PPDM. Then we suggest which principles should be considered absolute principles in PPDM.

1. **Collection Limitation Principle:** This principle states that some very sensitive data should not be held at all. Collection limitation is too general in the data mining context incurring in two grave consequences: a) the notion of "very sensitive" is sometimes unclear and may differ from country to country, leading to vague definitions; b) limiting the collection of data may make the data useless for knowledge discovery. Thus, this principle seems to be unenforceable in PPDM.

2. **Data Quality Principle:** This principle is related to the pre-processing stage in data mining in which data cleaning routines are applied to resolve inaccuracy and inconsistencies. Somehow, this principle is relevant in the pre-processing stage of knowledge discovery. However, most PPDM techniques assume that the data are already in an appropriate form to mine.

3. **Purpose Specification Principle:** This principle is the fundamental basis of privacy. Individuals should be informed of the purposes for which the information collected about them will be used, and the information must be used solely for that purpose. In other words, restraint should be exercised when personal data are collected. This principle is extremely relevant in PPDM.

4. **Use Limitation Principle:** This principle is closely related to the purpose specification principle. Use limitation is perhaps the most difficult principle to address in PPDM. This principle states that the purpose specified to the data subject (consumer) at the time of the collection restricts the use of the information collected, unless the data subject has provided consent for additional uses. This principle is also fundamental in PPDM.

5. **Security Safeguards Principle:** This principle is basically irrelevant in the case of data privacy, but relevant for database security. Security safeguards principle is typically concerned with keeping sensitive information (e.g., personal data) out of the hands of unauthorized users, which ensures that the data is not modified by users who do not have permission to do so. This principle is unenforceable in the context of PPDM.

6. **Openness Principle:** This principle, also called transparency, states that people have the right to know what data about them have been collected, who has access to the data, and how the data are being used. In other words, people must be aware of the conditions under which their information is being kept and used. However, data mining is not an open and transparent activity requiring analysts to inform individuals about particular derived knowledge, which may inhibit the use of data. This principle is equally important in PPDM.

7. **Individual Participation Principle:** This principle suggests that data subjects should be able to challenge the existence of information gained through data mining applications. Since knowledge discovery is not openly apparent to data subjects, the data subjects are not aware of knowledge discoveries related to them. While debatably collected individual information could belong to individuals, one can argue that collective information mined from databases belongs to organizations that hold such databases. In this case, the implications of this principle for PPDM should be carefully weighed; otherwise, it could be too rigid in PPDM applications.

8. **Accountability Principle:** This principle states that data controllers should inform data subjects of the use and findings from knowledge discovery. In addition, data controllers should inform individuals about the policies regarding knowledge discovery activities, including the consequences of inappropriate use. Some countries (e.g., the UK, Japan, Canada) that have adopted the OECD privacy principles do not consider this principle since it is not limited in scope, area, or application. Thus, the accountability principle is too general for PPDM.

Our analysis above suggests that the OECD privacy principles can be categorized into three groups according to their influence on the context of PPDM: *Group 1* is composed of those principles that should be considered as absolute principles in PPDM, such as Purpose Specification, Use Limitation, and Openness. *Group 2* consists of some principles that somehow impact PPDM applications, and their full implications should be understood and carefully weighed depending on the context. The principles that fall into this category are Data Quality and Individual Participation. *Group 3* encompasses some principles that are too general or unenforceable in PPDM. This group includes Collection Limitation, Security

Safeguards, and Accountability. Clearly, the principles categorized in groups 1 and 2 are relevant in the context of PPDM and are fundamental for further research, development, and deployment of PPDM techniques.

3.3.3 Adopting PPDM Policies from the OECD Privacy Guidelines

One fundamental point to be considered when designing some privacy policies is that too many restrictions could seriously hinder the normal functioning of business and governmental organizations. The worst thing is that restrictions, if not carefully weighed, could make PPDM results useless.

Given these facts, we suggest some policies for PPDM based on the OECD privacy principles. We try to find a good compromise between privacy requirements and knowledge discovery. We describe the policies as follows:

1. **Awareness Policy:** When a data controller collects personally identifiable information, the data controller shall express why the data are collected and whether such data will be used for knowledge discovery.

2. **Limit Retention Policy:** A data controller shall take all reasonable steps to keep only personal information collected that is accurate, complete, and up to date. In the case of personal information that is no longer useful, it shall be removed and not subjected to analysis to avoid unnecessary risks, such as wrong decision making that may incur liability.

3. **Forthcoming Policy:** Policies regarding collecting, processing, and analyzing that produce new knowledge about individuals shall be communicated to those about whom the knowledge discovered pertains, in particular when the discovered knowledge is to be disclosed or shared.

4. **Disclosure Policy :** Data controllers shall only disclose discovered knowledge about an individual for purposes for which the individual consents and the knowledge discovered about individuals shall never be disclosed inadvertently or without consent.

3.4 Requirements for Technical Solutions

3.4.1 Requirements for the development of technical solutions

Ideally, a technical solution for a PPDM scenario would enable us to enforce privacy safeguards and to control the sharing and use of personal data. However, such a solution raises some crucial questions:

- What levels of effectiveness are in fact technologically possible and what corresponding regulatory measures are needed to achieve these levels?

- What degrees of privacy and anonymity must be sacrificed to achieve valid data mining results?

These questions cannot have "yes-no" answers, but involve a range of technological possibilities and social choices. The worst response to such questions is to ignore them completely and not pursue the means by which we can eventually provide informed answers.

Technology alone cannot address all of the concerns surrounding PPDM scenarios [7]. The above questions can be to some extent addressed if we provide some key requirements to guide the development of technical solutions.

The following key words are used to specify the extent to which an item is a requirement for the development of technical solutions to address PPDM:

- **Must:** this word means that the item is an absolute requirement;
- **Should:** this word means that there may exist valid reasons not to treat this item as a requirement, but the full implications should be understood and the case carefully weighed before discarding this item.

Independence: A promising solution for the problem of PPDM, for any specific data mining task (e.g., association rules, clustering, classification), *should* be independent of the mining task algorithm.

Accuracy: When it is possible, an effective solution *should* do better than a trade-off between privacy and accuracy on the disclosure of data mining results. Sometimes a trade-off *must* be found as in scenario 2 in Section 3.2.3.

Privacy Level: This is also a fundamental requirement in PPDM. A technical solution *must* ensure that the mining process does not violate privacy up to a certain degree of security.

Attribute Heterogeneity: A technical solution for PPDM *should* handle heterogeneous attributes (e.g., categorical and numerical).

Versatility: A versatile solution to address the problem of PPDM *should* be applicable to different kinds of information repository, i.e., the data could be centralized, or even distributed horizontally or vertically.

Communication Cost: When addressing data distributed across many sites, a technical solution *should* consider carefully issues of communication cost.

3.4.2 Requirements to guide the deployment of technical solutions

Information technology vendors in the near future will offer a variety of products which claim to help protect privacy in data mining. How can we evaluate and decide whether what is being offered is useful? The nonexistence of proper instruments to evaluate the usefulness and feasibility of a solution to address a PPDM scenario challenge us to identify the following requirements:

Privacy Identification: We should identify what information is private. Is the technical solution aiming at protecting individual privacy or collective privacy?

Privacy Standards: Does the technical solution comply with international instruments that state and enforce rules (e.g., principles and/or policies) for use of automated processing of private information?

Privacy Safeguards: Is it possible to record what has been done with private information and be transparent with individuals about whom the private information pertains?

Disclosure Limitation: Are there metrics to measure how much private information is disclosed? Since privacy has many meanings depending on the context, we may require a set of metrics to do so. What is most important is that we need to measure not only how much private information is disclosed, but also measure the impact of a technical solution on the data and on valid mining results.

Update Match: When a new technical solution is launched, two aspects should be considered: a) the solution should comply with existing privacy principles and policies; b) in case of modifications to privacy principles and/or policies that guide the development of technical solutions, any release should consider these new modifications.

3.5 Summary

In this chapter, we have laid down some concepts toward foundations of PPDM. Although our work described here is preliminary and conceptual in nature, we argue that it is a vital prerequisite for standardization in PPDM. The absence of consensus on defining privacy preservation, policies and requirements for the development of new PPDM techniques is leading to confusion among developers, practitioners, and others interested in this technology.

Our primary goal here is to conceive a common framework for PPDM. Such a framework is composed of definitions, principles, policies, and requirements. The advantages of a framework of this nature are: (a) it will avoid confusing developers, practitioners, and many others interested in PPDM; (b) it will inhibit inconsistent efforts in different ways, and will enable vendors and developers to make solid advances in the future in the PPDM area.

Our contributions, in this chapter, can be summarized as follows: (1) we discussed the problems in defining privacy and how privacy can be violated in data mining. Then, we described the historical roots and the basis of PPDM including the PPDM landmarks, the definition of privacy preservation in data mining, and some models of data miners in PPDM; (2) we analyzed the implications of the Organization for Economic Cooperation and Development (OECD) data privacy principles in knowledge discovery. We showed that some PPDM principles can be inherited from the OECD privacy guidelines; (3) we suggested some policies for PPDM based on the OECD privacy guidelines; and (4) we suggested some requirements for the development and for guiding the deployment of technical solutions. These requirements are related to industrial initiatives.

Chapter 4

Privacy-Preserving Data Mining: A Literature Review

Revolution is not a onetime event.
– Audre Lorde

Ideas are the factors that lift civilization. They create revolutions.
There is more dynamite in an idea than in many bombs.
– Bishop Vincent

Issues concerning privacy preservation in data mining have emerged globally. A number of successful techniques have been proposed to obtain valid data mining results while maintaining privacy safeguards. In this chapter, we review the existing PPDM techniques in the literature. We classify these techniques into four major categories: data partitioning, data modification, data restriction, and data ownership, as can be seen in Figure 4.1. Data partitioning techniques are designed to address some scenarios in which the data available for mining are partitioned across multiple sites (parties). Instead of sharing the original data among the parties, only the data mining results are known to every party. We review these techniques in Section 4.1. When the source data are to be shared or exchanged, data modification and data restriction techniques are needed. Data modification techniques convert an original database into a new one that is subjected to mining. We review such techniques in Section 4.2. The data restriction-based techniques are reviewed in Section 4.3. To preserve private information in a released database, these techniques suppress some information from the data before mining occurs. The fourth category of PPDM techniques is called Data Ownership. These techniques aim at protecting the ownership of data by people about whom the data were collected. When sharing confidential data, these techniques can also be used to ensure that no one can read confidential data except the receiver(s) that are authorized to do so. We review such techniques in Section 4.4.

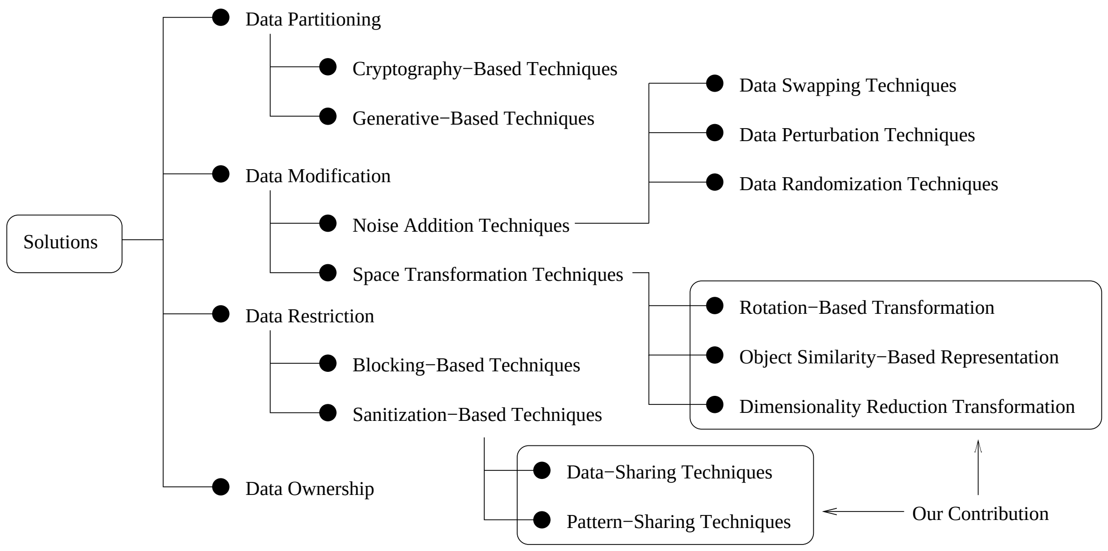

Solutions
Data Partitioning
Cryptography−Based Techniques
Generative−Based Techniques
Data Modification
Noise Addition Techniques
Data Swapping Techniques
Data Perturbation Techniques
Data Randomization Techniques
Space Transformation Techniques
Rotation−Based Transformation
Object Similarity−Based Representation
Dimensionality Reduction Transformation
Data Restriction
Blocking−Based Techniques
Sanitization−Based Techniques
Data Ownership
Data−Sharing Techniques
Pattern−Sharing Techniques
Our Contribution

Figure 4.1: A taxonomy of PPDM techniques

4.1 Data Partitioning Techniques

Data partitioning techniques have been applied to some scenarios in which the databases available for mining are distributed across a number of sites, with each site willing to share only data mining results, not the source data. In these cases, the data are distributed either horizontally or vertically [30]. In a horizontal partition, different entities are described with the same schema in all partitions, while in a vertical partition the attributes of the same entities are split across the partitions. The existing solutions can be classified into *Cryptography-Based Techniques* and *Generative-Based Techniques*.

4.1.1 Cryptography-Based Techniques

In the context of PPDM over distributed data, cryptography-based techniques have been developed to solve problems of the following nature: two or more parties want to conduct a computation based on their private inputs. The issue here is how to conduct such a computation so that no party knows anything except its own input and the results. This problem is referred to as the secure multi-party computation problem [62, 40, 114].

Generally speaking, secure multi-party computation is the branch of cryptography that deals with the realization of distributed tasks in a secure manner; in this case, the definition of security can have different flavours, such as preserving the privacy of the data or protecting the computation against malicious attacks [63]. Typically, secure multi-party computation consists of computing some function $f(x, y)$, where input x is in the hands of one participant and input y is in the hands of the other. For the computation to be secure, no more information is revealed to a participant than can be inferred from that participant's input and the output of the function itself (the final results).

The idea behind secure multi-party computation was introduced in [150]. The paper introduces a technique that enables the implementation of any probabilistic computation between two participants in a secure manner. Later on, this technique was generalized to the setting of multiple participants [15, 24]. However, the concept of PPDM by using secure multi-party computation was introduced in [92]. In this model, two parties owning confidential databases (e.g. confidential patient records) wish to run a data mining algorithm on the union of their databases without revealing any unnecessary information. In particular, this paper focuses on the problem of decision tree learning and uses ID3 [116], a popular and widely used algorithm for this problem. The training set is distributed between two parties. This approach treats PPDM as a special case of secure multi-party computation, and not only aims at preserving individual privacy but also tries to preserve leakage of any information other than the final result. The solution is efficient for data partition applications and demands slow overhead of communication and reasonable bandwidth.

The solutions presented in [80, 139] aim at mining globally valid results from distributed data without revealing information that compromises the privacy of the individual sources. In particular, the work in [80] addresses secure mining of association rules over horizontally partitioned data. This approach considers the discovery of associations in transactions that are split across sites, without revealing the contents of individual transactions. In this model, the data available in all parties have the same schema, and it is assumed that three or more parties are involved to minimize the leakage of information. The solution is based on secure multi-party computation to minimize the information shared, while adding overhead to the mining task. On the other hand, the work in [139], addresses the problem of association rule mining in which transactions are distributed across sources. Each site holds some attributes of each transaction, and the sites wish to collaborate to identify globally valid association rules. In this model, two parties are involved, one party being designated as the primary, which is the initiator of the protocol. The other party is the responder. There is a join key present in both databases. The goal is to find association rules involving attributes other than the join key.

In the context of privacy-preserving data clustering, the first solution using secure multi-party computation was introduced in [140]. Specifically, a method for k-means clustering was proposed when different sites contain different attributes for a common set of entities. Each site has information for all the entities for a specific subset of attributes. In this model, it is assumed that the existence of an entity in a particular site's database may be revealed (e.g., because of join operations with other parties). However, the values associated with an entity are private. In this solution, each site learns the cluster of each entity, but learns nothing about the attributes of an entity at other sites. This work ensures reasonable privacy while limiting communication cost.

Regarding privacy preservation in classification, one solution was proposed in [82] based on a Naïve Bayes classifier. Naïve Bayes is based on a Bayesian formulation of the classification problem which uses the simplifying assumption of attribute independence. This approach assumes that the data available for mining are horizontally partitioned, i.e., all parties involved collect the same set of information about different entities. Parties want to improve classification accuracy as much as possible by leveraging other parties' data. They do not want to reveal their own instances or the instance to be classified. Thus, the parties decide on a model. The model's parameters are generated jointly from the local data. Classification is performed individually without involving the other parties. Thus, the parties decide on sharing the model, but not the training set nor the instance to be classified.

More recently, a new approach was introduced in [21] to address privacy preservation in classification. An algorithm is proposed to protect data before a data mining process takes place. The algorithm encrypts not only the attribute values but also the attribute labels. The algorithm is reversible, thus allowing the results of the models to be translated back to the readable form, but only by the database owner. Although the data are encrypted before the data mining process, the data remain unchanged (not-distorted), and the statistics inside the data remain the same. In this way, the modeling algorithm performs equally well on the protected as on the non-protected data. This approach is especially useful when the knowledge discovery process is outsourced.

It should be pointed out that although universal and general, secure multi-party computation can be very inefficient and heavy in terms of communication complexity when the inputs are large and when the function to compute is relatively complicated to describe [94, 128].

4.1.2 Generative-Based Techniques

Generative-based techniques are designed to perform distributed mining tasks. In this approach, each party shares just a small portion of its local model that is used to construct the global model. The existing solutions are built over horizontally partitioned data.

The solution presented in [141] addresses privacy-preserving frequent itemsets in distributed databases. Each site S_i $(3 \leq i \leq n)$ sends its frequent itemsets to a combiner that finds the globally frequent itemsets based on the local models. Each site uses another representation of the itemsets (ABC becomes, for instance, 0-14-28) in a way such that the combiner is not able to identify the itemsets. It is assumed that all sites use the same codification. After combining the locally frequent itemsets, the combiner sends the upper bound for the globally frequent itemsets to all sites, and each site is able to restore the original itemsets' codification. At this point, each site knows only the information concerning its frequent itemsets and the upper bound of the globally frequent itemsets. Site S_1 then generates a random number for each of its itemsets. This number is then added to the support count of each itemset, and the perturbed support counts are sent to site S_2. The algorithm continues in the same way as before up to the last iteration. After receiving the values of the total local counts, site S_n requests from site S_1 the values of random numbers and their respective itemsets. Site S_n simply decrements each global support count by the respective number and checks which itemsets are locally frequent. It is shown that the global model generated is accurate and the communication cost requires only one round of message passing around the sites and one reduction operation to aggregate the final results.

The solution in [95] addresses privacy-preserving distributed clustering using generative models. This solution relies on Expectation Maximization (EM) based algorithms. These algorithms are guaranteed to asymptotically converge to a global model that is locally optimal as the sample size used to obtain the global model goes to infinity. The intuition behind this approach is that, rather than sharing parts of the original data or perturbed data, the parameters of suitable generative models are built at each local site. Such parameters are then transmitted to a central location. The best representative of all data is a certain "mean" model. It was empirically shown that such a model can be approximated by generating artificial samples from the underlying distributions using Markov Chain Monte Carlo techniques. This approach achieves high quality distributed clustering with acceptable privacy loss and low communication cost. This framework also encompasses a measure for quantifying privacy based on ideas from information theory.

4.2 Data Modification Techniques

These techniques modify the original values of a database that needs to be shared, and in doing so, privacy preservation is ensured. The transformed database is made available for mining and must meet privacy requirements without losing the benefit of mining. In general, data modification techniques aim at finding an appropriate balance between privacy preservation and knowledge disclosure. Methods for data modification include *noise addition techniques* and *space transformation techniques*.

4.2.1 Noise Addition Techniques

In statistical databases, noise addition techniques are used to protect individuals' privacy, but at the expense of allowing partial disclosure, providing information with less statistical quality, and introducing biases into query responses [137]. In data mining, the major requirement of a security control mechanism (in addition to protect the privacy) is not to ensure precise and bias-free statistics but rather to preserve the high-level descriptions of knowledge discovered from large databases [18, 45]. Thus, the idea behind noise addition techniques for PPDM is that some noise (e.g., information not present in a particular tuple or transaction) is added to the original data to prevent the identification of confidential information relating to a particular individual. In other cases, noise is added to confidential attributes by randomly shuffling the attribute values to prevent the discovery of some patterns that are not supposed to be discovered. We categorize noise addition techniques into three groups: (1) data swapping techniques; (2) data perturbation techniques; and (3) data randomization techniques.

Data swapping techniques replace the original database with a new one that has the same probability distribution. Such techniques are suitable for privacy protection in knowledge discovery. The idea behind data swapping is that it interchanges the values in the records of the database in such a way that statistics about groups (e.g., frequencies, averages, etc) are preserved.

The method proposed in [45] was designed for privacy preservation in classification. In this approach, a new training set, which is released to miners, is a perturbed version of the original training set. A data owner first builds a local decision tree over true data and then swaps values amongst records in a leaf node of the tree to generate randomized training data. The swapping is performed over the confidential attribute (class label) rather than other attributes in the dataset. As the class is typically a categorical attribute containing just two different values, the swapping is performed by changing the class in a small number of records. This is achieved by randomly shuffling the values of the class in the heterogeneous leaves. It has also been shown that is possible to balance statistical precision against the security level by choosing to perform the swapping in the internal nodes rather than in the leaves of the decision tree, i.e., the closer to the root, the higher the security but the lower the precision.

The work presented in [45] was extended in [69, 68]. The proposed method adds noise to datasets used for building decision trees. The method was evaluated taking into account the noise added to the class label and the noise added to the other attributes in a dataset. The authors measured the data quality by the similarity between the tree produced from the original data and a tree produced from the perturbed data. It was experimentally shown that the decision trees built on the perturbed data are very similar to the decision trees built on the original data.

Data perturbation techniques distort the data to protect individuals' privacy by introducing an error (noise) to the original data. The noise is used to generate the new (distorted) database which is subjected to mining. Miners should be able to obtain valid results (e.g., patterns and trends) from the distorted data. As opposed to statistical data analysis, miners do not aim at obtaining a definite, unbiased statistical test that answers with a probabilistic degree of confidence whether the data fit a preconceived statistical model. Data mining is not about hypothesis testing but about the generation of plausible hypotheses [66, 45].

The work presented in [9] addresses privacy preservation in classification by using data perturbation. The proposed solution aims at building a decision-tree classifier from training data in which the values of individual records have been perturbed by adding random values from a probability distribution. The resulting data records look very different from the original records, and the distribution of data values is also very different from the original distribution. While it is not possible to accurately estimate original values in individual data records, the authors proposed a novel reconstruction procedure to accurately estimate the distribution of original data values. By using these reconstructed distributions, one is able to build classifiers whose accuracy is comparable to the accuracy of classifiers built with the original data. The distribution reconstruction process naturally leads to some loss of information, but it can be acceptable in many practical situations.

A new algorithm for distribution reconstruction was introduced in [4]. This algorithm is more effective than that one proposed in [9], in terms of information loss. More specifically, the new algorithm is based on Expectation Maximization (EM) algorithms. It converges to the maximum likelihood estimate of the original distribution based on the perturbed data. When a large amount of data is available, the EM algorithm provides robust estimates of the original distribution. It was shown that the EM algorithm was in fact identical to the Bayesian reconstruction proposed in [9], except for the approximation partitioning values into intervals. Furthermore, the work in [4] introduces two new metrics, namely privacy loss and information loss to capture the amount of data in an individual record leaked to the data mining algorithm and the fidelity of the estimate respectively.

As previously mentioned, the distribution reconstruction obtained by using an EM algorithm was greatly improved in [4]. However, it it shown in [149] that the number of computations at each iteration is proportional to the size of the dataset and the number of intervals used in the estimate. Thus, two ways to reduce such great amounts of computation were proposed in [149]. In the first approach, the problem is studied from a signal processing viewpoint, and algorithms are proposed to reduce the computation in the original protocol of perturbation. In particular, a Fourier series-based method is presented to compute, in one step, a good initial estimate of the distribution to reduce the number of iterations. In the second approach, a scheme for data perturbation is presented by modifying the protocol of data perturbation proposed in [4]. Unlike EM algorithms, this scheme estimates the unknown distribution in one step, and it is very simple to implement. This approach also achieves significant improvements over the previous ones [9, 4] in terms of the small privacy loss and the high fidelity in the estimate of the distribution.

A different line of work is investigated in [83], in which the authors question the utility of the random value perturbation techniques in privacy preservation. In particular, this paper considers a class of techniques for PPDM involving the random perturbation of the data while preserving the underlying probabilistic properties. It explores the random value perturbation-based approach presented in [9]. This approach tries to preserve data privacy by adding random noise, while making sure that the random noise still preserves the "signal" from the data so that the patterns can still be accurately estimated. In [83] it is shown that in many cases, the original data can be closely estimated from the perturbed data using a spectral filter that exploits some theoretical properties of random matrices. The paper presents the theoretical foundation and provides experimental results to support this claim.

Data randomization techniques allow one to discover the general patterns in a database with error bound, while protecting individual values. Like data swapping and data perturbation techniques, randomization techniques are designed to find a good compromise between privacy protection and knowledge discovery.

A framework for mining association rules from transactions consisting of categorical items was proposed in [50, 48]. In this approach, the data are randomized to preserve the privacy of individual transactions. The idea behind this approach is that some items in each transaction are replaced by new items not originally present in this transaction. In doing so, some true information is taken away and some false information is introduced to obtain a reasonable privacy protection. In general, this strategy is feasible to recover association rules that are less frequent than they are originally, and preserve privacy using a straightforward uniform randomization. However, this technique introduces some false drops and may lead a data analyst to find association rules that are not supposed to exist.

The approach proposed in [50] is susceptible to privacy breaches. Intuitively, a privacy breach with respect to some property P occurs when, for some possible outcome of randomization, the posterior probability P is higher than a given threshold called the privacy breach level. The work presented in [49] introduces new formulations of privacy breaches and a methodology for limiting them. This approach addresses the problem of large transactions resulting from randomization from the perspective of reducing storage and communication costs. Specifically, a new compression technique for reducing the effective size of the randomized database was introduced.

A new data randomization technique has been applied to Boolean association rules [120]. Again, the idea is to modify data values such that reconstruction of the values for any individual transaction is difficult, but the rules learned from the distorted data are still valid. One interesting feature of this work is its flexible definition of privacy. For instance, the ability to correctly guess a value of '1' from the distorted data can be considered a greater threat to privacy than correctly learning a '0'. This scheme is based on probabilistic distortion of personal data, which is composed of a privacy metric and an analytical formula. Although this framework provides a high degree of privacy to the user and retains a high level of accuracy in the mining results, mining the distorted database can be, apart from being error-prone, significantly more expensive in terms of both time and space as compared to mining the original database. In [10], some issues of efficiency in privacy-preserving association rule mining are considered. The goal here is to determine how to identify parameter settings for the perturbation method that simultaneously ensure acceptable levels of privacy, accuracy, and efficiency in the mining process. To accomplish this, the authors derive a formula for estimating settings of the distortion parameters and present an optimization process that is applied to the reconstruction process so that the runtime efficiency is well within an order of magnitude of that for undistorted database mining.

Considering that randomization is economical and an efficient approach for PPDM, some effort has been made to optimize the trade-off between knowledge discovery and the protection of individuals' privacy. A general framework was proposed for randomization using mixture of models [156]. In this work, the authors argue that the existing theory and methods for mixture of models can be employed to facilitate the construction of optimal randomization for PPDM. In this framework, the impact of randomization is quantified by performance degradation and mutual information loss, while privacy and privacy loss are quantified by interval-based metrics. In the same direction, the work in [153] introduces a new randomization scheme for privacy-preserving association rule mining based on singular value decomposition. In comparison with the previous approaches, this new scheme introduces a two-way communication mechanism between the data miner and data providers with little overhead. In particular, the data miner sends a perturbation guidance to the data providers. The data providers then distort the data transactions to be transmitted to the miner. As a result, this scheme is able to identify association rules more precisely than the previous approaches and, at the same time, reaches a reasonable level of privacy.

4.2.2 Space Transformation Techniques

Space transformation techniques are specifically designed to address privacy-preserving clustering. These techniques aim at protecting the underlying data values subjected to clustering without jeopardizing the similarity between objects under analysis. Thus, a space transformation technique must not only meet privacy requirements but also guarantee valid clustering results.

A hybrid geometric data transformation method was proposed in [104] to meet privacy requirements as well as guarantee valid clustering results. This method distorts numerical attributes by translations, scalings, and rotations or even by the combination of these geometric transformations. The viability of using either a specific or the combination of all transformations (hybrid) for privacy preserving clustering was extensively studied. The key finding was that by transforming a data matrix by rotations only, one would attain both accuracy and a reasonable level of privacy. In contrast, transformation by translations are feasible if one is interested in accuracy since distortion by translations does not provide any level of privacy. The experiments also revealed that the scaling, hybrid (all transformations), and the Additive Data Perturbation (ADP) method, the latter of which is widely used in statistical databases, offer some level of privacy, but they do not preserve the distances between data points after the transformation process. Therefore, they are not recommended for privacy-preserving clustering because they are non-isometric transformations. As a consequence, they jeopardize the similarity between data points compromising the clustering results.

A more accurate investigation on privacy-preserving clustering using geometric transformation is presented in [106]. In particular, it is shown that distorting attribute pairs in a database by using only rotations is a promising approach. In this work, a spatial data transformation method is introduced for privacy-preserving clustering, called Rotation-Based Transformation (RBT). The method is designed to protect the underlying attribute values subjected to clustering without jeopardizing the similarity between data objects under analysis. RBT can be seen as a technique that is similar to obfuscation since the transformation process makes the original data difficult to perceive or understand, and preserves all the information for clustering analysis.

Two new space transformation techniques were introduced in [107], called *object similarity-based representation* and *dimensionality reduction-based transformation*. The former relies on the idea behind the similarity between objects, i.e., a data owner could share some data for clustering analysis by simply computing the dissimilarity matrix (matrix of distances) between the objects and then sharing such a matrix with a third party. Many clustering algorithms in the literature operate on a dissimilarity matrix [65]. This solution is simple to be implemented but requires a high communication cost since its complexity is of the order $O(m^2)$, where m is the number of objects under analysis. In addition, this solution is sometimes restrictive when an adversary has external knowledge of the original data. For instance, when two or more parties share data for clustering, if one party knows all the coordinates of a few points (the attribute values of a few objects), the dissimilarity matrix may disclose the original dataset.

The latter solution (dimensionality reduction-based transformation) can be used to address privacy-preserving clustering when the attributes of objects are available either in a central repository or split across many sites. We refer to the former approach as privacy-preserving clustering over centralized data, and the latter as privacy-preserving clustering over partitioned data. By reducing the dimensionality of a dataset to a sufficiently small value, one can find a trade-off between privacy and accuracy. Once the dimensionality of a database is reduced, the released database preserves (or slightly modifies) the distances between data points. In tandem with the benefit of preserving the similarity between points, this solution protects individuals' privacy since the underlying data values of the objects subjected to clustering are completely different from the original ones.

4.3 Data Restriction Techniques

Data restriction techniques focus on limiting the access to mining results through either the generalization or suppression of information (e.g., items in transactions or attributes in relations), or even by blocking the access to some patterns that are not supposed to be discovered. Such techniques can be divided into two groups: *Blocking-based techniques* and *Sanitization-based techniques.*

4.3.1 Blocking-Based Techniques

Blocking-based techniques aim at hiding some sensitive information when data are shared for mining. The private information includes sensitive association rules and classification rules that must remain private. Before releasing data for mining, data owners must consider how much information can be inferred or calculated from large databases and must look for ways to minimize the leakage of such information. In general, blocking-based techniques are feasible to recover patterns that are less frequent than they are originally since sensitive information is either suppressed or replaced with unknowns to preserve privacy.

The work presented in [76, 77] investigates the need for developing secure policies to minimize or eliminate the threat introduced by classification algorithms in the context of relational databases. It is shown that from unclassified data, one is able to infer confidential information that is not supposed to be disclosed. Two algorithms are proposed to strategically suppress some items in some attributes of a relational database with the purpose of limiting the disclosure of confidential information. This work was later extended in [78] for privacy-preserving association rule mining. A new methodology was proposed to hide knowledge in relational databases and to control the unauthorized disclosure of directed and indirected inferences.

Another blocking-based technique was introduced in [126, 127] for association rules. A set of algorithms was designed to hide sensitive information (sensitive rules) by replacing certain attributes of data items with a mark "?" (unknown), instead of deleting such items. In doing so, this solution obscures sensitive information, representing some sensitive association rules, and protects miners from learning "false" rules. This approach imposes some changes to the definition of support and confidence of an association rule. In this regard, the minimum support and minimum confidence will be altered into a minimum support interval and a minimum confidence interval correspondingly. As long as the support and/or the confidence of a sensitive rule lies below the middle of these two ranges of values, then it is expected that the confidentiality of data is not violated.

4.3.2 Sanitization-Based Techniques

Unlike blocking-based techniques that hide sensitive information by limiting or replacing some items or attribute values with unknowns, sanitization-based techniques hide sensitive information by strategically suppressing some items in transactional databases, or even by generalizing information to preserve privacy in classification. These techniques can be categorized into two major groups: *data-sharing techniques* and *pattern-sharing techniques*. In the former, the sanitization process acts on the data to remove or hide the group of sensitive association rules that contain sensitive knowledge. To do so, a small number of transactions that contain the sensitive rules have to be modified by deleting one or more items from them or even adding some noise, i.e., new items not originally present in such transactions. In the latter case, the sanitizing algorithm acts on the rules mined from a database, instead of on the data itself. The algorithm removes all sensitive rules before the sharing process.

The idea behind data-sharing techniques was first introduced in [12]. The authors considered the problem of limiting disclosure of sensitive rules, aiming at selectively hiding some frequent itemsets from large databases with as little impact on other non-sensitive frequent itemsets as possible. Specifically, the authors dealt with the problem of modifying a given database so that the support of a given set of sensitive rules, mined from the database, decreases below the minimum support value. The authors focused on the theoretical approach and showed that the optimal sanitization is an NP-hard problem.

In [37], the authors investigated confidentiality issues related to a broad category of association rules and proposed some algorithms to preserve the privacy of such rules above a given privacy threshold. Although these algorithms ensure privacy preservation, they are CPU-intensive since they require multiple scans over a transactional database. In addition, such algorithms, in some way, modify true data values and relationships by turning some items from 0 to 1 in some transactions.

A unified framework for protecting sensitive association rules was introduced in [102]. This framework combines techniques for efficiently hiding sensitive patterns: a transaction retrieval engine relying on an inverted file and Boolean queries; a set of algorithms to "sanitize" a database; and a set of metrics to quantify the disclosure of information and to evaluate the impact of the sanitization process on the released database. The sanitizing algorithms require two scans regardless of the database's size and the number of sensitive patterns that must be protected. The first scan is required to build an index (an inverted file) for speeding up the sanitization process, while the second scan is used to sanitize the original database. This work was extended in [103], in which two new sanitizing algorithms were introduced to balance privacy and knowledge disclosure. One algorithm hides sensitive association rules by removing some items randomly, and the other hides sensitive rules by removing some items in a round robin fashion.

A new, efficient sanitizing algorithm, called the Sliding Window Algorithm (SWA), was proposed in [105]. SWA requires only one pass over a transactional database regardless of the database's size and the number of sensitive association rules that must be protected. In particular, SWA scans a group of K transactions, one window at a time, and then sanitizes the set of sensitive rules that can be mined from these K transactions. This algorithm improves the balance between protection of sensitive knowledge and pattern discovery, and it is useful for sanitizing large transactional databases. Another advantage of SWA is the fact that it does not introduce false drops (rules that do not exist in the original database) to the data. However, an extra cost is incurred because some rules are removed inadvertently.

Regarding pattern-sharing techniques, the only known approach that falls into this category was introduced in [109]. This framework addresses the sharing of association rules between two or more parties. In this problem, one party may decide to disclose only part of the knowledge and conceal strategic patterns, which we call sensitive rules. These sensitive rules must be protected before sharing since they are paramount for strategic decisions and need to remain private. The proposed framework is composed of a sanitizing algorithm for protecting sensitive knowledge before sharing association rules, and a set of metrics to evaluate attacks against sensitive knowledge and the impact of the sanitization on the released association rules (non-sensitive ones). The algorithm blocks some inference channels to ensure that an adversary cannot reconstruct sensitive rules from the non-sensitive ones. It is shown that this algorithm reduces drastically the side effect factor, i.e., the percentage of non-sensitive rules accidentally removed during the sanitization process.

In the context of predictive modeling, a framework was proposed in [70] for preserving the anonymity of individuals or entities when data are shared or made public. To do so, this solution transforms an original database into a new one by using generalization and suppression to satisfy some privacy constraints. In particular, this work investigates the privacy transformations for building classification and regression models. This work also introduces some metrics to quantify information loss in the transformed database. The data transformation problem is solved by using a genetic algorithm framework to optimize the appropriate metric. This work considers the trade-off between privacy and information loss.

4.4 Data Ownership Techniques

Data ownership techniques can be applied to two different scenarios: (a) to protect the ownership of data by people about whom the data were collected; and (b) to identify the entity that receives confidential data when such data are shared or exchanged.

In the context of data mining, the first effort toward a technical solution guaranteeing privacy of data owners was introduced in [56]. The idea behind this approach is that a data owner may prevent the data from being used for some purposes and allow them to be used for other purposes. To accomplish that, this solution is based on encoding permissions on the use of data as theorems about programs that process and mine the data. Theorem proving techniques are then used to guarantee that these programs comply with the permissions. The proposed technique provides a tool for verifiable implementation of the Use Limitation Principle, one of the most known data privacy principles. The authors also discuss a mechanism by which this technique could be introduced in industrial practice.

A different approach was introduced in [97] for sharing of confidential data. When sharing or exchanging confidential data, this approach ensures that no one can read confidential data except the receiver(s). It can be used in different scenarios, such as statistical or research purposes, data mining, and on-line business-to-business (B2B) interactions. This framework is composed of a fingerprint, encoder, decoder, and a detection algorithm. The main idea behind fingerprinting is that when data are shared, each copy of the data has a hidden mark that identifies not only the owner of the data but also the entity that receives the particular copy. This approach is complementary to the work proposed in [56].

4.5 Summary

The proliferation of PPDM techniques is evident. In this chapter, we reviewed the existing PPDM techniques present in the literature. We classified such techniques into four major categories: data partitioning, data modification, data restriction, and data ownership. We introduced a taxonomy of PPDM techniques covering these four major categories.

Data partitioning techniques address scenarios in which the data available for mining are partitioned across multiple sites. The existing solutions can be classified into cryptography-based and generative-based techniques. Regarding data modification techniques, these techniques convert an original database into a new one that is subjected to mining. The released database balances privacy preservation and knowledge discovery. Data modification techniques can be classified into two groups: noise addition and space transformation techniques. While noise addition techniques were designed to address privacy preservation in association rule mining and classification, space transformation techniques are specifically designed for privacy preservation in clustering. The third category of PPDM techniques that we introduced in this chapter is called data restriction. This category includes blocking-based and sanitization-based techniques. The goal of such techniques is to limit the access to knowledge mined from databases through either the generalization or suppression of information. In general, blocking-based techniques hide sensitive information by limiting or replacing some items or attribute values with unknowns, while sanitization-based techniques hide sensitive information by suppressing some items in transactional databases, or even by generalizing some information to preserve privacy in classification. The fourth category of PPDM techniques is called Data Ownership. These techniques implement a mechanism enforcing data ownership by the individuals to whom the data belongs. When sharing confidential data, these techniques can also be used to ensure that no one can read confidential data except the receiver(s) that are authorized to do so.

Our contributions, in this chapter, include the space transformation techniques which were designed to address privacy-preserving clustering, and some of the sanitization-based techniques which were designed to address privacy-preserving association rule mining. These techniques are described in more detail in Chapters 5 and 6.

Chapter 5

Methods for Privacy-Preserving Association Rule Mining

The secret to creativity is knowing how to hide your sources.
– Albert Einstein

It is common sense to take a method and try it.
If it fails, admit it frankly and try another. But above all, try something.
– Franklin D. Roosevelt

The sharing of association rules is often beneficial in industry, but requires privacy safeguards. One may decide to disclose only part of the knowledge mined from databases, and protect sensitive knowledge represented by sensitive rules. These sensitive rules must remain private since they are essential for strategic decisions. Some companies prefer to share their data for collaboration, while others prefer to share only the patterns discovered from their data. Our algorithms presented in this chapter take into account these two important aspects, i.e., the sharing of data and the sharing of patterns. The process of protecting sensitive rules in transactional databases is called data sanitization (see Section 2.3.4 for further information).

This chapter is organized as follows: In Section 5.1, we describe some scenarios that demonstrate the need for techniques to protect collective privacy (e.g., sensitive knowledge) in association rule mining. We introduce our framework for protecting sensitive knowledge in transactional databases in Section 5.2. This framework is composed of a retrieval facility (e.g., inverted index), a set of algorithms to "sanitize" a database, and a set of metrics to measure how much private information is disclosed as well as the impact of the sanitizing algorithms on valid mining results. In Section 5.3, we introduce our data sharing-based sanitizing algorithms in which the sanitization process acts on the data to remove or hide the group of sensitive association rules. After sanitizing a database, the released database is shared for association rule mining. A different approach to hide sensitive knowledge is introduced in Section 5.4, called pattern sharing-based. In this approach, the sanitizing algorithm acts on the rules mined from a database instead of the data itself. Rather than sharing the data, data owners may prefer to mine their own data and share some discovered patterns. In this case, the sanitization removes not only all sensitive patterns but also blocks other patterns that could be used to infer the sensitive hidden ones.

5.1 Motivation for Privacy-Preserving Association Rule Mining

Today, collaboration has become prevalent in the competitive commercial world since it brings mutual benefits [118]. Such collaboration may occur between competitors or companies that have conflicts of interest. However, collaborators are aware that they are provided with an advantage over other competitors.

Association rule mining creates assets that collaborating companies can leverage to expand their businesses, improve profitability, reduce costs, and support marketing more effectively. In tandem with these benefits, association rule mining can also, in the absence of adequate safeguards, open new threats to both individual and collective privacy. Let us consider some examples in which privacy-preserving association rule mining really matters.

- Suppose we have a server and many clients, with each client having a set of sold items (e.g., books, movies, etc). The clients want the server to gather statistical information about associations among items in order to provide recommendations to the clients. However, the clients do not want the server to be able to derive some sensitive association rules. In this context, the clients represent companies and the server hosts a recommendation system for an e-commerce application. In the absence of ratings, which are used in collaborative filtering for automatic recommendation building, association rules can be effectively used to build models for on-line recommendations. When a client sends its frequent itemsets to the server, this client sanitizes some sensitive itemsets according to some specific policies. The sensitive itemsets contain sensitive knowledge that can provide a competitive advantage. The server then gathers statistical information from the sanitized itemsets and recovers from them the actual associations. Is it possible for these companies to benefit from such collaboration by sharing association rules while preserving some sensitive rules?

- Two companies have a very large dataset of records of their customers' buying activities. These companies decide to cooperatively conduct association rule mining on their datasets for their mutual benefit since this collaboration brings them an advantage over other competitors. However, these companies may not want to share some strategic patterns hidden within their own data with the other party. They would like to transform their data in such a way that these sensitive associations rules cannot be discovered. Is it possible for these companies to benefit from such collaboration by sharing their data while preserving some sensitive association rules?

- Let us consider the case in which one supplier offers products in reduced prices to some consumers and, in turn, this supplier receives permission to access the database of the consumers' customer purchases. The threat becomes real whenever the supplier is allowed to derive sensitive association rules that are not even known to the database owners (consumers). In this case, the consumers benefit from reduced prices, whereas the supplier is provided with enough information to predict inventory needs and negotiate other products to obtain a better deal for his consumers. This implies that the competitors of this supplier start losing business. How can the consumers protect some sensitive association rules of customer purchases, while allowing the supplier to mine other useful association rules?

To address the above scenarios, we propose a framework to protect sensitive knowledge in transactional databases. This framework is introduced in the next section.

5.2 The Framework for Privacy-Preserving Association Rule Mining

In this section, we introduce the framework to address privacy preservation in association rule mining. As depicted in Figure 5.1, the framework encompasses an inverted file to speed up the sanitization process, a library of sanitizing algorithms used for hiding sensitive association rules from the database, and a set of metrics to quantify not only how much private information is disclosed, but also the impact of the sanitizing algorithms on the transformed database and on valid mining results.

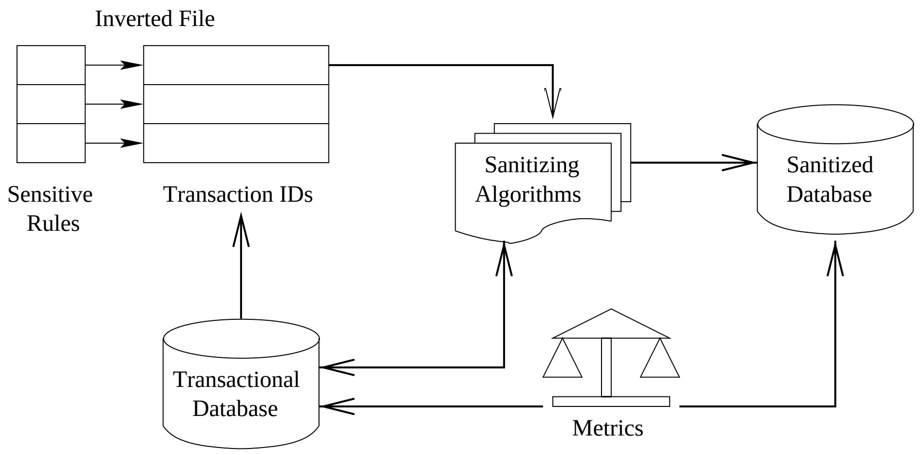


Figure 5.1: The sketch of the framework for privacy-preserving association rule mining.

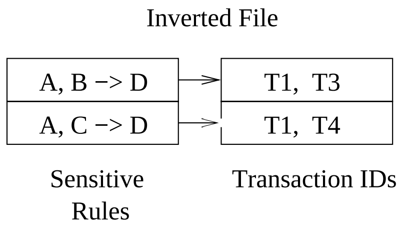
5.2.1 The Inverted File

Sanitizing a transactional database consists of identifying the sensitive transactions and adjusting them. To speed up this process, we scan a transactional database only once and, at the same time, we build our retrieval facility (inverted file) [14]. The inverted file's vocabulary is composed of all the sensitive rules to be hidden, and for each sensitive rule there is a corresponding list of transaction IDs in which the rule is present.

Figure 5.2(b) shows an example of an inverted file corresponding to the sample transactional database shown in Figure 5.2(a). For this example, we assume that the sensitive rules are A,B $\rightarrow$ D and A,C $\rightarrow$ D.

TID	**Items**
T1	A B C D
T2	A B C
T3	A B D
T4	A C D
T5	A B C
T6	B D

(A)



(B)

Figure 5.2: (a) A sample transactional database. (b) The corresponding inverted file.

Note that once the inverted file is built, a data owner will sanitize only the sensitive transactions whose IDs are stored in the inverted file. Knowing the sensitive transactions prevents a data owner from performing multiple scans in the transactional database. Consequently, the CPU time for the sanitization process is optimized. Apart from optimizing the CPU time, the inverted file provides other advantages, as follows:

- The information kept in main memory is greatly reduced since only the sensitive rules are stored in memory. The occurrences (transaction IDs) can be stored on disk when not fitted in main memory.

- Our algorithms require at most two scans regardless of the number of sensitive rules to be hidden: one scan to build the inverted file, and the other to sanitize the sensitive transactions. The previous methods require as many scans as there are rules to hide.

5.2.2 The Library of Sanitizing Algorithms

In our framework, the sanitizing algorithms modify some transactions to hide sensitive rules based on a disclosure threshold ψ controlled by the database owner. This threshold indirectly controls the balance between knowledge disclosure and knowledge protection by controlling the proportion of transactions to be sanitized. For instance, if $\psi = 50\%$ then half of the sensitive transactions will be sanitized, when $\psi = 0\%$ all the sensitive transaction will be sanitized, and when $\psi = 100\%$ no sensitive transaction will be sanitized. In other words, ψ represents the ratio of sensitive transactions that should be left untouched. The advantage of this threshold is that it enables a compromise between hiding association rules while missing non-sensitive ones, and finding all non-sensitive association rules but uncovering sensitive ones.

As can be seen in Figure 5.1, the sanitizing algorithms are applied to the original database to produce the sanitized one. We classify our algorithms into two major groups: *data sharing-based algorithms* and *pattern sharing-based algorithms*, as can be seen in Figure 5.3.

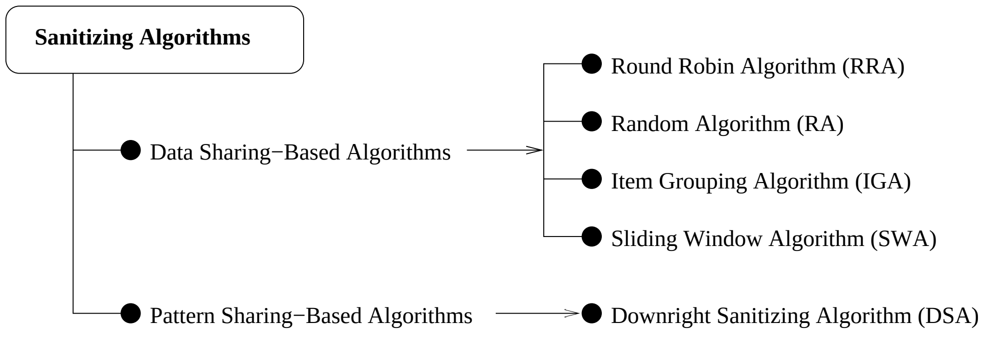

Sanitizing Algorithms
Round Robin Algorithm (RRA)
Random Algorithm (RA)
Data Sharing−Based Algorithms
Item Grouping Algorithm (IGA)
Sliding Window Algorithm (SWA)
Pattern Sharing−Based Algorithms
Downright Sanitizing Algorithm (DSA)

Figure 5.3: A taxonomy of sanitizing algorithms.

In the former, the sanitization process acts on the data to remove or hide the group of sensitive association rules representing the sensitive knowledge. To accomplish this, a small number of transactions that participate in the generation of the sensitive rules have to be modified by deleting one or more items from them. In doing so, the algorithms hide sensitive rules by reducing either their support or confidence below a privacy threshold (disclosure threshold). In the latter, the sanitizing algorithm acts on the rules mined from a database, instead of the data itself. The algorithm removes all sensitive rules before the sharing process. In Section 5.3, we introduce our data sharing-based sanitizing algorithms, and in Section 5.4 we present our pattern sharing-based sanitizing algorithms.

5.2.3 The Set of Metrics

In this section, we introduce the set of metrics to quantify not only how much sensitive knowledge has been disclosed, but also to measure the effectiveness of the proposed algorithms in terms of information loss and in terms of non-sensitive rules removed as a side effect of the transformation process. We classify these metrics into two major groups: *Data sharing-based metrics* and *Pattern sharing-based metrics*.

a) Data sharing-based metrics are related to the problems illustrated in Figure 5.4. This figure shows the relationship between the set R of all association rules in the database D, the sensitive rules S_R, the non-sensitive association rules $\sim S_R$, as well as the set R' of

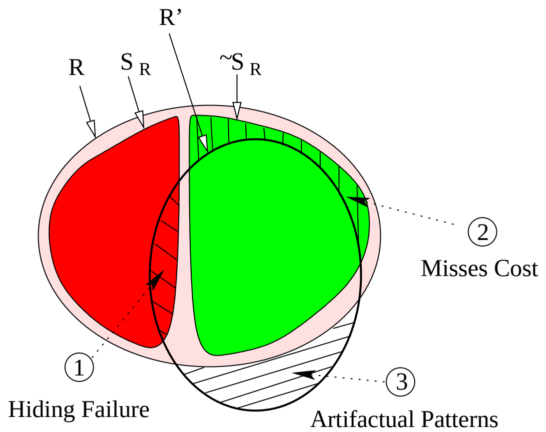


Figure 5.4: Data sharing-based sanitization problems.

rules discovered from the sanitized database D'. The circles with the numbers 1, 2, and 3 are potential problems that respectively represent the sensitive association rules that were failed to be hidden, the legitimate rules accidentally missed, and the artificial association rules created by the sanitization process.

Problem 1 occurs when some sensitive association rules are discovered in the sanitized database. We call this problem ***Hiding Failure (HF)***, and it is measured in terms of the percentage of sensitive association rules that are discovered from D'. Ideally, the hiding failure should be 0%. The hiding failure is measured as follows:

$$HF = \frac{\#\ S_R(D')}{\#\ S_R(D)} \tag{5.1}$$

where $\#\ S_R(X)$ denotes the number of sensitive association rules discovered from the database X.

Problem 2 occurs when some legitimate association rules are hidden as a side effect of the sanitization process. This happens when some non-sensitive association rules lose support in the database due to the sanitization process. We call this problem ***Misses Cost (MC)***, and it is measured in terms of the percentage of legitimate association rules that are not discovered from D'. In the best case, this should also be 0%. The misses cost is calculated as follows:

$$MC = \frac{\#\ {\sim}S_R(D) - \#\ {\sim}S_R(D')}{\#\ {\sim}S_R(D)} \tag{5.2}$$

where $\#\ {\sim}S_R(X)$ denotes the number of non-sensitive association rules discovered from the database X.

Notice that there is a compromise between the misses cost and the hiding failure. The more sensitive rules we hide, the more non-sensitive rules we miss. This is basically the justification for our disclosure threshold ψ, which with tuning, allows us to find the balance between privacy and disclosure of information whenever the application permits it.

Problem 3 occurs when some artificial association rules are generated from D' as a product of the sanitization process. We call this problem ***Artifactual Patterns (AP)***, and it is measured in terms of the percentage of the discovered association rules that are artifacts, i.e., rules that are not present in the original database. Artifacts are generated when new items are added to some transactions to alter (decrease) the confidence of sensitive rules. For instance, in a rule $X \rightarrow Y$, if the items are added to the antecedent part X of this rule in transactions that support X and not Y, then the confidence of such a rule is decreased. Artifactual patterns are measured as follows:

$$AP = \frac{|R'| - |R \cap R'|}{|R'|} \tag{5.3}$$

where $|X|$ denotes the cardinality of X.

We could measure the dissimilarity between the original and sanitized databases by computing the difference between their sizes in bytes. However, we believe that this dissimilarity should be measured by comparing their contents instead of their sizes. Comparing their contents is more intuitive and gauges more accurately the modifications made to the transactions in the database.

To measure the dissimilarity between the original and the sanitized datasets, we could simply compare the difference in their histograms. In this case, the horizontal axis of a histogram contains all items in the dataset, while the vertical axis corresponds to their frequencies. The sum of the frequencies of all items gives the total of the histogram. So the dissimilarity between D and D' is given by:

$$Dif(D, D') = \frac{1}{\sum_{i=1}^{n} f_D(i)} \times \sum_{i=1}^{n} [f_D(i) - f_{D'}(i)] \tag{5.4}$$

where $f_X(i)$ represents the frequency of the i-th item in the dataset X, and n is the number of distinct items in the original dataset.

b) Pattern sharing-based metrics: are related to the problems illustrated in Figure 5.5. Problem 1 conveys the non-sensitive rules ($\sim S_R$) that are removed as a side effect of the sanitization process (R_{SE}). We refer to this problem as *side effect*. It is related to the misses cost problem in data sanitization (Data sharing-based metrics). Problem 2 occurs when using some non-sensitive rules, an adversary may recover some sensitive ones by inference channels. We refer to such a problem as *recovery factor*.

Side Effect Factor (SEF) measures the number of non-sensitive association rules that are removed as a side effect of the sanitization process. The measure is calculated as follows:

$$SEF = \frac{(|R| - (|R'| + |S_R|))}{(|R| - |S_R|)} \tag{5.5}$$

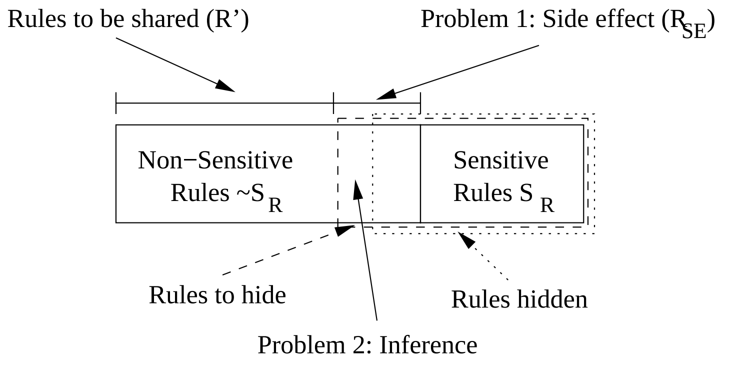


Figure 5.5: Pattern sharing-based sanitization problems.

where R, R', and S_R represent the set of rules mined from a database, the set of sanitized rules, and the set of sensitive rules, respectively, and $|S|$ is the size of the set S.

Recovery Factor (RF) expresses the possibility of an adversary recovering a sensitive rule based on non-sensitive ones. The recovery factor of one pattern takes into account the existence of its subsets. The rationale behind the idea is that all nonempty subsets of a frequent itemset must be frequent. Thus, if we recover all subsets of a sensitive itemset (rule), we say that the recovery factor for such an itemset is possible, and thus we assign it the value 1. However, the recovery factor is never certain, i.e., an adversary may not learn an itemset even with its subsets. On the other hand, when not all subsets of an itemset are present, the recovery of the itemset is improbable, thus we assign value 0 to the recovery factor.

In the pattern sharing-based approach, the set of sanitized rules to be shared (R') is defined as $R' = R - (S_R + R_{SE})$, where R is the set of all rules mined from a database, S_R is the set of sensitive rules, and R_{SE} is the set of rules removed as a side effect of the sanitization process.

5.3 Data Sharing-Based Sanitizing Algorithms

In this section, we describe two heuristics to hide sensitive rules in transactional databases. We then introduce our data sharing-based algorithms that rely on these heuristics.

5.3.1 Heuristic 1: Sanitization Based on the Degree of Sensitive Transactions

As mentioned in Chapter 4.3.2, the optimal sanitization is an NP-hard problem [12]. To alleviate the complexity of the optimal sanitization, we could use some heuristics. An heuristic does not guarantee the optimal solution, but usually finds a solution close to the best one in a faster response time [34].

Our first heuristic for data sanitization is based on the fact that, in many cases, a sensitive transaction (see Section 2.3.3) participates in the generation of one or more sensitive association rule to be hidden. We refer to the number of sensitive rules supported by a sensitive transaction as the *degree of a sensitive transaction*, defined as:

Definition 3 (Degree of a Sensitive Transaction) *Let D be a transactional database and S_T a set of all sensitive transactions in D. The degree of a sensitive transaction t, denoted by degree(t), such that $t \in S_T$, is defined as the number of sensitive association rules that can be found in t.*

Our sanitizing algorithms (Round Robin, Random, and Item Grouping), presented in the next sections, act on the original database taking into account the degree of sensitive transactions. For instance, given the number of sensitive transactions to alter, based on ψ, our algorithms select for each sensitive rule the sensitive transactions whose degree is sorted in descending order. The rationale is that by sanitizing the sensitive transactions that share a common item with more than one sensitive rule, the hiding strategy of such rules is optimized and, consequently, the impact of the sanitization on the discovery of the legitimate association rules is minimized.

All our data sharing-based algorithms, which rely on Heuristic 1, have essentially four major steps:

- *Step 1:* Scan a database and identify the sensitive transactions for each sensitive association rule. This step is accomplished when the inverted file is built;
- *Step 2:* Based on the disclosure threshold ψ, calculate for each sensitive association rule the number of sensitive transactions that should be sanitized and mark them. Most importantly, the sensitive transactions are selected based on their degree (descending order);
- *Step 3:* For each sensitive association rule, identify a candidate item that should be eliminated from the sensitive transactions. This candidate item is called the *victim item*;
- *Step 4:* Scan the database again, identify the sensitive transactions marked to be sanitized and remove the victim items from them.

Most of our sanitizing algorithms mainly differ in step 2 where the sensitive transactions to be sanitized are selected, and in step 3 in the way they identify a victim item to be removed from the sensitive transactions for each sensitive rule. Steps 1 and 4 remain essentially the same for all approaches. In general, the inputs for these algorithms are a transactional database D, a set of sensitive association rules S_R, and a disclosure threshold ψ controlled by the database owner, while the output is the sanitized database D'.

5.3.2 The Round Robin Algorithm

The main idea behind the round robin algorithm, denoted by RRA, is that rather than selecting a unique victim item per given sensitive association rule, we select different victim items in turns starting from the first item, then the second and so on in each sensitive transaction. The process starts again at the first item of the sensitive rule as a victim item each time the last item is reached. The rationale behind this selection is that removing one item at a time from the sensitive transactions would minimize the number of non-sensitive rules removed as the side effect of the sanitization process, since this strategy tries to balance the decreasing of the support of the items in sensitive association rules. The sketch of the round robin algorithm is given as follows:

Algorithm 1: Round_Robin_Algorithm

```
input  : $D$, $S_R$, $\psi$
output: $D'$
begin
    // Step 1: Identifying sensitive transactions and building index $T$
    foreach transaction $t \in D$ do
        Sort the items in $t$ is alphabetic order;
        foreach sensitive association rule $sr_i \in S_R$ do
            if $items(sr_i) \subseteq t$ then
                $T[sr_i].tid\_list \leftarrow T[sr_i].tid\_list \cup TID\_of(t)$;
            end
        end
    end
    // Step 2: Selecting the number of sensitive transactions
    foreach sensitive association rule $sr_i \in S_R$ do
        Sort the vector $T[sr_i].tid\_list$ in descending order of degree;
        $NumbTrans_{sr_i} \leftarrow |T[sr_i]| \times (1 - \psi)$;
        // $|T[sr_i]|$ is the number of sensitive transactions for $sr_i$
    end
    // Step 3: Identifying victim items for each sensitive transaction
    foreach sensitive association rule $sr_i \in S_R$ do
        for $j$ = 1 to $NumbTrans_{sr_i}$ do
            $ChosenItem \leftarrow item_v$ such that $item_v \in sr_i$ and if there are $k$ items in $sr_i$,
                        the $ith$ item is assigned to $item_v$ mod $k$ in round robin fashion
            $Victims[T[sr_i, j]].item\_list \leftarrow Victims[T[sr_i, j]].item\_list \cup ChosenItem$;
        end
    end
    // Step 4: $D' \leftarrow D$
    Sort the vector $Victims$ in ascending order of $t_{ID}$;
    $j \leftarrow 1$;
    foreach transaction $t \in D$ do
        if $t_{ID} == Victims[j].t_{ID}$ then
            $t \leftarrow (t - Victims[j].item\_list)$;
            $j \leftarrow j + 1$;
        end
    end
end
```

In the first step, the algorithm builds an inverted index (represented by the vector T), based on the transactions in D, in one scan. The vocabulary of the inverted index contains all the sensitive rules, and for each sensitive rule there is a corresponding list of transaction IDs in which the rule is present. Before building the inverted index, the items in each transaction are first sorted in alphabetic order. Then for each association rule $sr_i \in S_R$, the algorithm verifies if such rule is present in the current transaction t. To do so, the algorithm performs binary searches, in the worst case on all items of the rule sr_i in the transaction t, to make sure that this rule is present in t. If all the items in sr_i are found in t, the transaction t is sensitive, and this transaction ID is added to the occurrences of sr_i (line 7).

In step 2, the algorithm sorts the sensitive transactions associated with all the sensitive rules in descending order of t_{ID} (line 13). This is the basis of our Heuristic 1. Then in line 14, the number of sensitive transactions to be sanitized, in each sensitive rule sr_i, is selected based on the disclosure threshold ψ.

In step 3, the structure $Victim$ is composed of two fields: t_{ID} and $item\_list$ the latter which corresponds to a list of victim items that are marked to be removed from the corresponding transaction whose ID is t_{ID}. In line 20, the victim item of the current sensitive rule sr_i is chosen. In line 22, this chosen item is then added to the list of victim items in $Victim$, which contains the items marked to be sanitized.

In the last step, first the vector $Victim$ is sorted in ascending order of t_{ID} (line 26). Then the algorithm scans the database again (for the second and last time) in the loop from line 28 to line 33. If the current transaction (t_{ID}) is selected to be sanitized, the victim items corresponding to this transaction t are removed from it. In our implementation, transactions that do not need sanitization are directly copied from D to D'.

Theorem 1 *The running time of the Round Robin algorithm is $O(n_1 \times N \times log\ N)$, in the worst case, where n_1 is the number of sensitive rules and N is the number of transactions in the database.*

The proof of Theorem 1 is given in Appendix A.1.1.

5.3.3 The Random Algorithm

The intuition behind the Random Algorithm, denoted by RA, is to select as a victim item, for a given sensitive association rule sr_i, one item of such rule randomly. The selected items for each rule sr_i are removed, one at a time, from the sensitive transactions associated with sr_i. Like the Round Robin algorithm, the rationale behind this selection is that removing different items from the sensitive transactions would slightly decrease the support of non-sensitive association rules that would be available for being mined in the sanitized database.

The selection of the sensitive transactions to sanitize is simply based on their degree. We evaluated the sanitization through the Random Algorithm by selecting sensitive transactions sorted in ascending and descending order. The approach based on descending order, in general, yielded the best results. This is why we have adopted such an approach for our algorithm. The sketch of the Random Algorithm is given as follows:

Algorithm 2: Random_Algorithm

```
input  : $D$, $S_R$, $\psi$
output: $D'$
1  begin
2      // Step 1:  Identifying sensitive transactions and building index $T$
3      foreach transaction $t \in D$ do
4          Sort the items in $t$ is alphabetic order;
5          foreach sensitive association rule $sr_i \in S_R$ do
6              if items($sr_i$) $\subseteq t$ then
7                  $T[sr_i].tid\_list \leftarrow T[sr_i].tid\_list \cup TID\_of(t)$;
8              end
9          end
10     end
11     // Step 2:  Selecting the number of sensitive transactions
12     foreach sensitive association rule $sr_i \in S_R$ do
13         Sort the vector $T[sr_i].tid\_list$ in descending order of degree;
14         $NumbTrans_{sr_i} \leftarrow |T[sr_i]| \times (1 - \psi)$;
15         // $|T[sr_i]|$ is the number of sensitive transactions for $sr_i$
16     end
17     // Step 3:  Identifying victim items for each sensitive transaction
18     foreach sensitive association rule $sr_i \in S_R$ do
19         for $j$ = 1 to $NumbTrans_{sr_i}$ do
20             $ChosenItem \leftarrow item_v$ such that $item_v \in sr_i$ and if there are $k$ items in $sr_i$,
21                             the item assigned to $item_v$ is random(k);
22             $Victims[T[sr_i, j]].item\_list \leftarrow Victims[T[sr_i, j]].item\_list \cup ChosenItem$;
23         end
24     end
25     // Step 4:  $D' \leftarrow D$
26     Sort the vector $Victims$ in ascending order of $t_{ID}$;
27     $j \leftarrow 1$;
28     foreach transaction $t \in D$ do
29         if $t_{ID} == Victims[j].t_{ID}$ then
30             $t \leftarrow (t - Victims[j].item\_list)$;
31             $j \leftarrow j + 1$;
32         end
33     end
34 end
```

The four steps of this algorithm correspond to those in the Round Robin algorithm. The only difference is that the Random algorithm selects the victim item randomly (line 20), while the Round Robin algorithm selects the victim item taking turns. The complexities of the Round Robin and the Random algorithms are the same.

5.3.4 The Item Grouping Algorithm

The main idea behind the Item Grouping Algorithm, denoted by IGA, is to group sensitive rules in groups of rules sharing the same itemsets. If two sensitive rules intersect, by sanitizing the sensitive transactions containing both sensitive rules, one would take care of hiding these two sensitive rules at once and consequently reduce the impact on the released database. However, clustering the sensitive rules based on the intersections between items in rules leads to groups that overlap since the intersection of itemsets is not transitive. By computing the overlap between clusters and thus isolating the groups, we can use a representative of the itemset linking the sensitive rules in the same group as a victim item for all rules in the group. By removing the victim item from the sensitive transactions related to the rules in the group, all sensitive rules in the group will be hidden in one step. This again would minimize the impact on the database and reduce the potential accidental hiding of legitimate rules.

Like Round Robin and Random algorithms, the Item Grouping algorithm builds an inverted index, based on the transactions in D, in one scan. The vocabulary of the inverted index contains all the sensitive rules, and for each sensitive rule there is a corresponding list of transaction IDs in which the rule is present. From lines 7 to 11, the IGA builds the inverted index, and in lines 4 and 5, the IGA computes the frequencies of all items in the database D. These frequencies (support) are used for computing the victim items in step 3.

Steps 2 and 4 are exactly the same as those in the Round Robin and Random algorithms. The main difference is in step 3, in the way that the IGA selects the victim items.

The goal of step 3 is to identify a victim item per sensitive rule. The victim item in one rule sr_i is fixed and must be removed from all the sensitive transactions associated with this rule sr_i. The selection of the victim item is done by first clustering sensitive rules in a set of overlapping groups GP (step 3.1), such that all sensitive rules in the same group G share the same items. Then the groups of sensitive rules are sorted in descending order of shared items (step 3.2). The shared items are the class label of the groups. For example, the patterns "ABC" and "ABD" would be in the same group labeled either A or B depending on support of A and B (step 3.3). However, "ABC" could also be in another group if there was one where sensitive rules shared "C." From line 26 to 32, the IGA identifies such overlap between groups and eliminates it by favoring larger groups or groups with a class label with lower support in the database.

Again, the rationale behind the victim selection in IGA is that since the victim item now represents a set of sensitive rules (from the same group), sanitizing a sensitive transaction will allow many sensitive rules to be taken care of at once per sanitized transaction. This strategy greatly reduces the side effect on the non-sensitive rules mined from the sanitized database. The sketch of the Item Grouping algorithm is given as follows:

Algorithm 3: Item_Grouping_Algorithm

```
input  : $D$, $S_R$, $\psi$
output: $D'$
begin
    // Step 1: Identifying sensitive transactions and building index $T$
    foreach transaction $t \in D$ do
        for $k$ = 1 to size($t$) do
            $Sup(item_k, D) \leftarrow Sup(item_k, D) + 1$; //Update support of each $item_k$ in $t$;
        Sort the items in $t$ in alphabetic order;
        foreach sensitive association rule $sr_i \in S_R$ do
            if items($sr_i$) $\subseteq t$ then
                $T[sr_i].tid\_list \leftarrow T[sr_i].tid\_list \cup TID\_of(t)$;
            end
        end
    end
    // Step 2: Selecting the number of sensitive transactions
    foreach sensitive association rule $sr_i \in S_R$ do
        Sort the vector $T[sr_i].tid\_list$ in descending order of degree;
        $NumbTrans_{sr_i} \leftarrow |T[sr_i]| \times (1 - \psi)$;
        // $|T[sr_i]|$ is the number of sensitive transactions for $sr_i$
    end
    // Step 3: Identifying victim items for each sensitive transaction
    3.1 Group sensitive rules in a set of groups $GP$ such that $\forall$ $G \in GP$,
         $\forall$ $sr_i, sr_j \in G$, $sr_i$ and $sr_j$ share the same itemset $I$. Give the class label
         $\alpha$ to $G$ such that $\alpha \in I$ and $\forall \beta \in I$, sup($\alpha$, $D$) $\leq$ sup($\beta$, $D$);
    3.2 Order the groups in $GP$ by size in terms of number of sensitive rules
         in the group;
    // Compare groups pairwise $G_i$ and $G_j$ starting with the largest
    3.3 forall $sr_k \in G_i \cap G_j$ do
        if size($G_i$) $\neq$ size($G_j$) then
            remove $sr_k$ from smallest($G_i, G_j$);
        else
            remove $sr_k$ from group with class label $\alpha$ such that sup($\alpha$, $D$) $\geq$ sup($\beta$, $D$)
            and $\alpha, \beta$ are class labels of either $G_i$ or $G_j$;
        end
    end
    3.4 foreach sensitive association rule $sr_i \in S_R$ do
        for $j$ = 1 to $NumbTrans_{sr_i}$ do
            $ChosenItem \leftarrow \alpha$ such that $\alpha$ is the class label of $G$ and $sr_i \in G$;
            $Victims[T[sr_i, j]].item\_list \leftarrow Victims[T[sr_i, j]].item\_list \cup ChosenItem$;
        end

    end
    // Step 4: $D' \leftarrow D$
    Sort the vector $Victims$ in ascending order of $t_{ID}$;
    $j \leftarrow 1$;
    foreach transaction $t \in D$ do
        if $t_{ID} == Victims[j].t_{ID}$ then
            $t \leftarrow (t - Victims[j].item\_list)$;
            $j \leftarrow j + 1$;
        end
    end
end
```

Theorem 2 *The running time of the Item Grouping algorithm is* $O(n_1 \times N \times log\ N)$*, where* n_1 *is the number of sensitive rules and* N *is the number of transactions in the database.*

The proof of Theorem 2 is given in Appendix A.1.2.

To illustrate how our presented algorithms work, let us consider the sample transactional database in Figure 5.2(a). Suppose that we have a set of sensitive association rules $S_R = \{$A,B→D; A,C→D$\}$. This example yields the following results:

- *Step 1*: The algorithms scan the database to identify the sensitive transactions. For this example, the sensitive transactions S_T containing the sensitive association rules are {T1, T3, T4}. The degrees of the transactions T1, T3 and T4 are 2, 1 and 1 respectively. In particular, the rule A,B→D can be mined from the transactions T1 and T3 and the rule A,C→D can be mined from T1 and T4.

- *Step 2*: Suppose that we set the disclosure threshold ψ to 50%. Then the algorithms sort the sensitive transactions in descending order of degree. The algorithms sanitize half of the sensitive transactions for each sensitive rule. In this case, only the transaction T1 will be sanitized.

- *Step 3*: In this step, the victim items are selected. Note that the three algorithms employ different strategies for this selection. The Round Robin algorithm selects the victim items for each rule taking turns. The item A is selected for both rules minimizing the impact on the database. The Random algorithm selects one item for each rule randomly. Let us assume that the item A was selected for the first rule and the item C was selected for the second rule. The Item Grouping Algorithm clusters sensitive rules that share a common item. Both rules share the items A and D. In this case, only one item is selected, say the item D. By removing the item D from T1 the sensitive rules will be hidden from T1 in one step and the disclosure threshold will be satisfied.

- *Step 4*: The algorithms perform the sanitization taking into account the victim items selected in the previous step. The sanitized databases using the algorithms Round Robin, Random, and Item Grouping, respectively, are showed in Figure 5.6.

An important observation here is that any association rule that contains a sensitive association rule is also sensitive. Hence, if A,B→D is a sensitive association rule, any association rule derived from the itemset ABCD will also be sensitive since it contains ABD. This is because if ABCD is discovered to be a frequent itemset, it is straightforward to conclude that ABD is also frequent, which should not be disclosed. In other words, any superset containing ABD should not be allowed to be frequent.

TID	Items
T1	B C D
T2	A B C
T3	A B D
T4	A C D
T5	A B C
T6	B D

(a)

TID	Items
T1	B D
T2	A B C
T3	A B D
T4	A C D
T5	A B C
T6	B D

(b)

TID	Items
T1	A B C
T2	A B C
T3	A B D
T4	A C D
T5	A B C
T6	B D

(c)

Figure 5.6: The sanitized databases using: (a) the Round Robin algorithm; (b) the Random algorithm; (c) the Item Grouping algorithm.

5.3.5 Heuristic 2: Sanitization Based on the Size of Sensitive Transactions

In this section, we introduce the second heuristic to hide sensitive knowledge in transactional databases. The idea behind this heuristic is to sanitize the sensitive transactions with the shortest sizes. The rationale is that by removing items from shortest transactions we would minimize the impact on the sanitized database since the shortest transactions have fewer combinations of association rules. As a consequence, we would reduce the side effect of the sanitization process on non-sensitive rules.

Our heuristic approach has essentially four steps as follows:

Step 1: Distinguishing the sensitive transactions from the non-sensitives ones. For each transaction read from a database D, we identify whether this transaction is involved in the generation of any sensitive association rule. If not, the transaction is copied directly to the sanitized database D'. Otherwise, this transaction is sensitive and must be sanitized.

Step 2: Selecting the victim item. In this step, we first compute the frequencies of all items in the sensitive association rules presented in the current sensitive transaction. The item with the highest frequency is the victim item since it is shared by a group of sensitive rules. If a sensitive rule shares no item with the other sensitive ones, the frequencies of its items are the same (freq $= 1$). In this case, the victim item for this particular sensitive rule is selected randomly. The rationale behind this selection is that removing different items from the sensitive transactions would slightly minimize the support of the legitimate association rules that would be available for being mined in the sanitized database D'.

Step 3: Computing the number of sensitive transactions to be sanitized. Given the disclosure threshold, ψ, set by the database owner, we compute the number of transactions to be sanitized. Every sensitive rule will have a list of sensitive transaction IDs associated with it. In this step, we sort the sensitive transactions computed previously for each sensitive rule. The sensitive transactions are sorted in ascending order of size. Thus, we start sanitizing the shortest transactions.

Step 4: Sanitizing a sensitive transaction. Given that the victim items for all sensitive association rules were selected in step 2, they can now be removed from the sensitive transactions. Every sensitive rule now has a list of sensitive transaction IDs with their respective selected victim item. Every time we remove a victim item from a sensitive transaction, we perform a look ahead procedure to verify whether that transaction has been selected as a sensitive transaction for other sensitive rules. If so, and the victim item we just removed from the current transaction is also part of this other sensitive rule, we remove that transaction from the list of transaction IDs marked in the other rules. In doing so, the transaction will be sanitized and then copied to the sanitized database D'. This look-ahead procedure is done only when the disclosure threshold is 0%. This is because the look-ahead improves the misses cost but could significantly degrade the hiding failure. When $\psi = 0$, there is no hiding failure (i.e., all sensitive rules are hidden) and thus there is no degradation possible but an improvement in the misses cost.

To illustrate how our heuristic works, let us consider the sample transactional database in Figure 5.2(a). Suppose that we have a set of sensitive association rules $S_R = \{$A,B$\rightarrow$D; A,C$\rightarrow$D$\}$ and we set the disclosure threshold $\psi = 50\%$. This example yields the following results:

Step 1: The sensitive transactions are identified. In this case, the sensitive transactions of A,B$\rightarrow$D and A,C$\rightarrow$D are {T1, T3} and {T1, T4} respectively.

Step 2: After identifying the sensitive transactions, we select the victim items. For example, the victim item in the transaction T1 could be either A or D since these items are shared by the sensitive rules and consequently their frequencies are equal to 2. However, the victim item for the sensitive rule A,B$\rightarrow$D in T3 is selected randomly because the items A, B, and D have frequencies equal to 1. Let us assume that the victim item selected is B. Similarly, the victim item for the sensitive rule A,C$\rightarrow$D, in transaction T4, is selected randomly, say, the item A.

Step 3: In this step, we compute the number of sensitive transactions to be sanitize, for each sensitive rule. This computation is based on the disclosure threshold ψ. We selected $\psi = 50\%$ for both rules. However, we could set a particular disclosure threshold for each sensitive rule. We also sorted the sensitive transactions in ascending order of size before performing the sanitization in the next step.

Step 4: We then sanitize the transactions for each sensitive rule. Half of the transactions for each sensitive rule will be intact since the disclosure threshold $\psi = 50\%$. We start by sanitizing the shortest transactions (sorted in the previous step). Thus, transactions T3 and T4 are sanitized. The released database is depicted in Figure 5.7(b). Note that the sensitive rules are present in the sanitized database, but with lower support. This is an example of partial sanitization. The database owner could also set the disclosure threshold $\psi = 0\%$. In this case, we have a full sanitization since the sensitive rules will no longer be discovered. In this example, we assume that the victim item in transaction T1 is D since this item is shared by both sensitive rules. Figure 5.7(c) shows the database after a full sanitization. As we can see, the database owner can tune the disclosure threshold to find a balance between protecting sensitive association rules by data sanitization and providing information for mining.

TID	Items
T1	A B C D
T2	A B C
T3	A B D
T4	A C D
T5	A B C
T6	B D

(A)

TID	Items
T1	A B C D
T2	A B C
T3	A D
T4	C D
T5	A B C
T6	B D

(B)

TID	Items
T1	A B C
T2	A B C
T3	A D
T4	C D
T5	A B C
T6	B D

(C)

Figure 5.7: (a): A copy of the sample transactional database in Figure 5.2(a); (b): An example of partial sanitization; (c): An example of full sanitization.

In the next section, we introduce one algorithm based on the Heuristic 2, called the Sliding Window Algorithm (SWA). The SWA has an advantage over the previous algorithms - it allows a database owner to set a specific disclosure threshold for each sensitive rule. This specific disclosure threshold works as a weight. In many cases, some rules are more important than others. Thus giving different disclosure thresholds to different rules is reasonable and may reflect the need in the real world.

5.3.6 The Sliding Window Algorithm

In this section, we introduce the Sliding Window Algorithm (SWA) that is based on Heuristic 2. The intuition behind this algorithm is that the SWA scans a group of K transactions (window size) at a time. SWA then sanitizes the set of sensitive transactions, denoted by S_T, considering a disclosure threshold ψ defined by a database owner. All the steps in Heuristic 2 are applied to every group of K transactions read from the original database D.

Unlike the previous sanitizing algorithms that have a unique disclosure threshold for all sensitive rules, the SWA has a disclosure threshold assigned to each sensitive association rule. We refer to the set of mappings of a sensitive association rule into its corresponding disclosure threshold as the set of mining permissions, denoted by M_P, in which each mining permission mp is characterized by an ordered pair, defined as $mp = < sr_i, \psi_i >$, where $\forall i$ $sr_i \in S_R$ and $\psi_i \in [0 \ldots 1]$. The sketch of the Sliding Window algorithm is given as follows:

Algorithm 4: Sliding_Window_Algorithm

```
input  : $D$, $M_P$, $K$
output: $D'$
begin
    foreach $K$ transactions in $D$ do
        // Step 1:  Identifying sensitive transactions & building index $T$
        foreach transaction $t \in K$ do
            Sort the items in $t$ in alphabetic order;
            foreach sensitive association rule $sr_i \in M_P$ do
                if $items(sr_i) \subseteq t$ then
                    $T[sr_i].tid\_list \leftarrow T[sr_i].tid\_list \cup TID\_of(t)$;    //$t$ is sensitive
                    $T[sr_i].size\_list \leftarrow T[sr_i].size\_list \cup size(t)$;
                    $freq[item_j[\leftarrow freq[item_j] + 1$;
                    $v\_transac \leftarrow v\_transac \cup t$;   //Sensitive transactions in memory
                end
            end
            // Step 2:  Identifying the victim items
            if $t$ is sensitive then
                Sort vector $freq$ in descending order;
                foreach sensitive association rule $sr_i \in M_P$ do
                    Select $item_v$ such that $item_v \in sr_i$ and $\forall$ $item_k \in sr_i$,
                            $freq[item_v] \geq freq[item_k]$;
                    if $freq[item_v] > 1$ then
                        $T[sr_i].victim \leftarrow T[sr_i].victim \cup item_v$;
                    else
                        $T[sr_i].victim \leftarrow T[sr_i].victim \cup RandomItem(sr_i)$;
                    end
                end
            end
        end
    end
    // Step 3:  Selecting the number of sensitive transactions
    foreach sensitive association rule $sr_i \in M_P$ do
        $NumTrans_{sr_i} \leftarrow |T[sr_i]| \times (1 - \psi_i)$;
        Sort the vector $T$ in ascending order of size;
    end
    // Step 4:  $D' \leftarrow D$
    foreach sensitive association rule $sr_i \in M_P$ do
        for $j$ = 1 to $NumbTrans_{sr_i}$ do
            $remove(v\_transac[T[sr_i].tid\_list[j], T[sr_i].victim[j]])$;
            if $\psi_i = 0$ then
                do $look\_ahead(sr_i, T[sr_i].tid\_list[j], T[sr_i].victim[j])$;
            end
        end
    end
end
```

The inputs for the Sliding Window algorithm are a transactional database D, a set of mining permissions M_P, and the window size K. The output is the sanitized database D'.

The SWA has essentially four steps. In the first, the algorithm scans K transactions and stores some information in the data structure T. This data structure contains: 1) a list of sensitive transactions IDs for each sensitive rule; 2) a list with the size of the corresponding sensitive transactions; and 3) another list with the victim item for each corresponding sensitive transaction. A transaction t is sensitive if it contains all items of at least one sensitive rule. The SWA also computes the frequencies of the items of the sensitive rules that are present in each sensitive transaction. This computation will support the selection of the victim items in the next step. In line 11, the vector *v_transac* stores the sensitive transactions in main memory.

In step 2, the vector with the frequencies, computed in the previous step, is sorted in descending order. Subsequently, the victim item is selected for each sensitive transaction. The item with the highest frequency is the victim item and must be marked to be removed from the transaction. If the frequencies of the items is equal to 1, any item from a sensitive association rule can be the victim item. In this case, we select the victim item randomly.

In step 3, the number of sensitive transactions for each sensitive rule is selected. Line 31 shows that ψ_i is used to compute the number $NumTrans_{sr_i}$ of transactions to sanitize. The SWA then sorts the list of sensitive transactions for each sensitive rule in ascending order of size. This sort is the basis of our Heuristic 2.

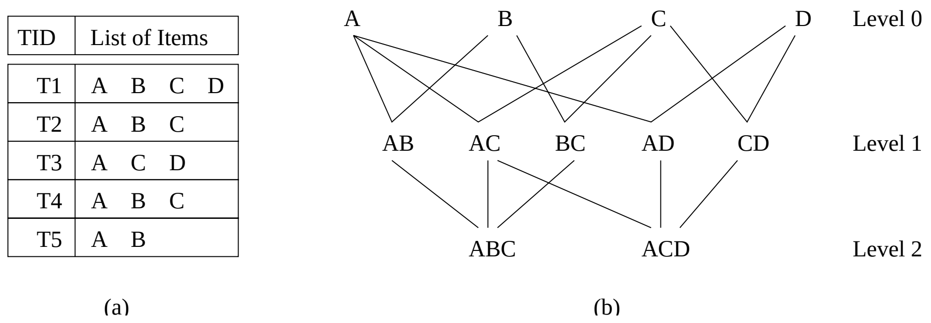
In the last step, the sensitive transactions are sanitized in the loop from line 35 to 42. If the disclosure threshold is 0 (i.e., all sensitive rules need to be hidden), we do a look ahead in the mining permissions (M_P) to check whether a sensitive transaction need not be sanitized more than once. This is to improve the misses cost. The function *look_ahead*() looks in M_P from sr_i onward to determine whether a given transaction t is selected as a sensitive transaction for another sensitive rule r. If this is the case and, $T[sr_i].tid\_list[j]$ and $T[sr_i].victim[j]$ are part of the sensitive rule r, the transaction t is removed from that list since it has already just been sanitized.

Theorem 3 *The running time of the SWA is $O(n_1 \times N \times log\ K)$ when $\psi \neq 0$ and $O(n_1^2 \times N \times K)$ when $\psi = 0$, where n_1 is the initial number of sensitive rules in the database D, K is the window size chosen, and N is the number of transactions in D.*

The proof of Theorem 3 is given in Appendix A.1.3.

5.4 Pattern Sharing-Based Sanitizing Algorithms

The sanitizing algorithms introduced in Section 5.3 are designed to protect sensitive knowledge before the sharing of data for association rule mining. The heuristic behind those



TID	List of Items
T1	A B C D
T2	A B C
T3	A C D
T4	A B C
T5	A B

Figure 5.8: (a) A transactional database. (b) The corresponding frequent itemset graph.

algorithms relies on transforming a transactional database to be shared in such a way that the sensitive rules can no longer be discovered or can only be discovered with lower support and confidence. In this section, we introduce a new heuristic for protecting sensitive knowledge. Rather than sanitizing a database before the mining phase, the new heuristic sanitizes sensitive rules from a set of rules mined from a database, while blocking some inference channels. The sanitization applied to a set of sensitive rules must not leave a trace that could be exploited by an adversary to recover sensitive rules from non-sensitive ones.

5.4.1 Heuristic 3: Rule Sanitization With Blocked Inference Channels

Before introducing our new heuristic, we briefly review some terminology from graph theory. In particular, we represent the itemsets in a database as a directed graph. We refer to such a graph as a *frequent itemset graph* and define it as follows:

Definition 4 (Frequent Itemset Graph) *A frequent itemset graph, denoted by $G = (C, E)$, is a directed graph which consists of a nonempty set of frequent itemsets C, a set of edges E that are ordered pairings of the elements of C, such that $\forall u, v \in C$ there is an edge from u to v if $u \cap v = u$ and $|v| - |u| = 1$, where $|x|$ is the size of itemset x.*

Figure 5.8(b) shows a frequent itemset graph for the sample transactional database depicted in Figure 5.8(a). In this example, the minimum support threshold σ is set to 2. As can be seen in Figure 5.8(b), in a frequent itemset graph G, there is an order for each itemset. We refer to such an ordering as the *itemset level* and define it as follows:

Definition 5 (The Itemset Level) *Let $G = (C, E)$ be a frequent itemset graph. The level of an itemset u, such that $u \in C$, is the length of the path connecting an 1-itemset to u.*

Based on Definition 5, we define the level of a frequent itemset graph G as follows:

Definition 6 (Frequent Itemset Graph Level) *Let $G = (C, E)$ be a frequent itemset graph. The level of G is the length of the maximum path connecting an 1-itemset u to any other itemset v, such that $u, v \in C$, and $u \subset v$.*

In general, the discovery of itemsets in G is the result of top-down traversal of G constrained by a minimum support threshold σ. The discovery process employs an iterative approach in which k-itemsets are used to explore $(k+1)$-itemsets.

Our heuristic approach for rule sanitization has essentially three steps, as follows. These steps are applied after the mining phase, i.e., we assume that the frequent itemset graph G is built. The set of all itemsets that can be mined from G, based on a minimum support threshold σ, is denoted by C.

Step1: Identifying the sensitive itemsets. For each sensitive rule $sr_i \in S_R$, convert it into a sensitive itemset $c_i \in C$.

Step2: Selecting subsets to sanitize. In this step, for each itemset c_i to be sanitized, we compute its item pairs from level 1 in G, subsets of c_i. If none of them is marked, we randomly select one of them and mark it to be removed.

Step3: Sanitizing the set of supersets of marked pairs in level 1. The sanitization of sensitive itemsets is simply the removal of the set of supersets of all itemsets in level 1 of G that are marked for removal. This process blocks possibilities of inferring sensitive rules. We refer to these possibilities as inference channels that we describe in the next section.

5.4.2 Inference Channels in Pattern Sharing-Based Algorithms

In pattern sharing-based algorithms, an inference channel occurs when someone mines a sanitized set of rules and, based on non-sensitive rules, deduces one or more sensitive rules that are not supposed to be discovered. We have identified some inferences against sanitized rules, as follows:

Forward-Inference Channel: Let us consider the frequent itemset graph in Figure 5.9(a). Suppose we want to sanitize the sensitive rules derived from the itemset ABC. The naïve approach is simply to remove the itemset ABC. However, if AB, AC, and BC are frequent, a miner could deduce that ABC is frequent. A database owner must assume that an adversary can use any inference channel to learn something more than just the permitted association rules. We refer to this inference as a forward-inference channel. To handle this inference channel, we must also remove at least one subset of ABC (randomly) in level 1 of the frequent itemset graph. This complementary sanitization is necessary. In the case of a deeper graph, the removal is done recursively up to level 1. Thus, the items in level 0 of the frequent itemset graph are not shared with a second party. We could also remove subsets of ABC recursively up to level 0. In this case, the balance between knowledge protection and knowledge discovery should

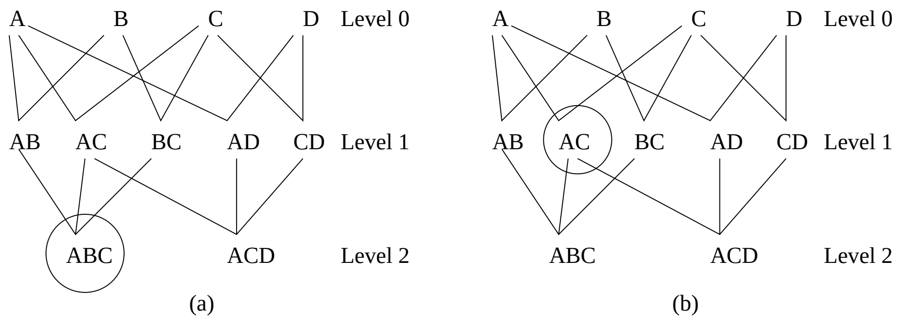


Figure 5.9: (a) An example of forward-inference. (b) An example of backward-inference.

be considered in the released set of rules, since more frequent patterns are lost by the sanitization process.

Backward-Inference Channel: Another type of inference occurs when we sanitize a non-terminal itemset. Based on Figure 5.9(b), suppose we want to sanitize any rule derived from the itemset AC. If we simply remove AC, it is straightforward to infer the rules mined from AC, since both ABC and ACD are frequent. We refer to this inference as a backward-inference channel. To block this inference channel, we must remove any superset that contains AC. In this particular case, ABC and ACD must be removed as well. This kind of inference clearly shows that rule sanitization is not a simple filter after the mining phase to weed out or hide the sensitive rules. Trimming some rules out does not ensure full protection. Some inference channels must be blocked as well.

5.4.3 The Downright Sanitizing Algorithm

The idea behind the Downright Sanitizing Algorithm, denoted by DSA, is to sanitize some sensitive rules while blocking inference channels as well. To block inference channels, the DSA removes at least one subset of each sensitive itemset in the level 1 of the frequent itemset graph. The removal is done recursively up to level 1. The DSA starts removing from level 1 because we assume that the association rules recovered from the sanitized itemsets (shared itemsets) have at least 2 items. A data owner could also set DSA to start removing from level 0, but this option would decrease the usability of the shared knowledge since more itemsets would be removed, increasing the side effect factor and misses cost. Thus, the items in level 0 of the frequent itemset graph are not shared at all. In doing so, we reduce the inference channels and minimize the side effect on non-sensitive rules mined from the sanitized frequent itemset graph.

The inputs for the DSA are the frequent itemset graph G and the set of sensitive rules S_R to be sanitized. The output is the sanitized frequent itemset graph G'. The sketch of the Downright Sanitizing algorithm is given as follows:

Algorithm 5: Downright_Sanitizing_Algorithm

```
input  : $G$, $S_R$
output: $G'$
begin
    // Step 1:  Identifying the sensitive itemsets
    foreach sensitive association rule $sr_i \in S_R$ do
        $c_i \leftarrow sr_i$;   //Convert each $sr_i$ into a frequent itemset $c_i$
    end
    // Step 2:  Selecting subsets to sanitize
    foreach $c_i$ in the level $k$ of $G$, where $k \geq 1$ do
        Pairs($c_i$);   //Compute all the item pairs of $c_i$
        if (Pairs($c_i$) $\cap$ $MarkedPair$ = $\emptyset$) then
            $p_i \leftarrow$ random(Pairs($c_i$));   //Select randomly a pair $p_i \in c_i$;
            $MarkedPair \leftarrow MarkedPair \cup p_i$;   //Update the list $MarkedPair$
        end
    end
    // Step 3:  Sanitizing the set of supersets of marked pairs
    //          in level 1 ($R' \leftarrow R$)
    foreach itemset $c_j \in G$ do
        Sort the items in $c_j$ in alphabetic order;
    end
    foreach itemset $c_j \in G$ do
        if $\exists$ a marked pair $p$, such that $p \in MarkedPair$ and $p \subset c_j$ then
            Remove($c_j$) from $R'$;   //$c_j$ belongs to the set of supersets of $p$;
        end
    end
end
```

To illustrate how the DSA works, let us consider the frequent itemset graph depicted in Figure 5.8(b). This frequent itemset graph corresponds to the sample transactional database in Figure 5.8(a), with minimum support threshold $\sigma = 2$. Now suppose that we are sanitizing the sensitive rule A,B→C before sharing the frequent itemset graph. This example yields the following results:

Step 1: Each sensitive rule is converted into its corresponding frequent itemset. In this case, the rule A,B→C is converted into the itemset ABC.

Step 2: In this step, the subsets for each rule are selected. In general, the subsets are selected from the level 1 in the frequent itemset graph. For this example, considering that there is no subset marked, we select one of the subsets of ABC randomly. If we had subsets marked previously, the algorithm would select one already marked to optimize the sanitization process. Let us assume that we selected the subset AB randomly. Then the subset AB was added to the list $MarkedPair$, which contains all the marked pairs to be sanitized.

Step 3: In this step, the sanitization takes place. The algorithm removes all supersets of each pair $p \in MarkedPair$. In this case, all the supersets of AB will be removed.

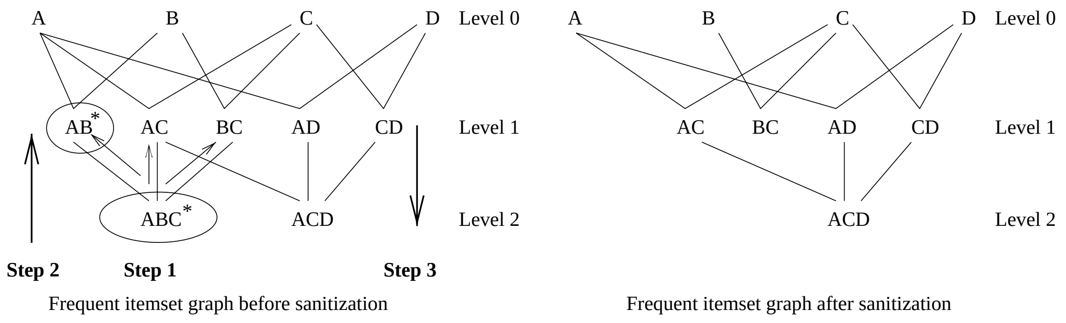


Figure 5.10: An example of a frequent itemset graphs before and after sanitization.

Figure 5.10 shows the frequent itemset graphs before and after the rule sanitization. The sanitized frequent itemset graph is shared for association rule mining.

Theorem 4 *The running time of the Downright Sanitizing Algorithm is* $O(n \times (k^2 + m \times \log k))$*, where* n *is the number of sensitive rules to be sanitized,* m *is the number of itemsets in a frequent itemsets graph* G*, and* k *the maximum number of items in a frequent itemset in* G*.*

The proof of Theorem 4 is given in Appendix A.2.1.

5.5 Summary

In this chapter, we have introduced three heuristics to hide sensitive association rules by reducing either the support or the confidence of these rules. The protection of sensitive rules is achieved by modifying some transactions. In some cases, a number of items are deleted from a group of transactions with the purpose of hiding the sensitive rules mined from those transactions. To accomplish that, we proposed a unified framework for privacy-preserving association rule mining, which is the major contribution of this chapter. This framework encompasses: a) an inverted index to speed up the sanitization process; b) a library of sanitizing algorithms used for hiding sensitive association rules from the database; and c) a set of metrics to quantify not only how much private information is disclosed, but also the impact of the sanitizing algorithms on the transformed database and on valid mining results.

To speed the process of hiding sensitive rules in transactional databases, our framework is built on an index. As a result, the sanitizing algorithms require only two scans to protect sensitive rules regardless of the number of association rules to be hidden: one scan to build an inverted index, and the other scan to hide the sensitive rules. Other techniques proposed in the literature require multiple scans [12, 37, 126, 142].

The sanitizing algorithms are classified into two major groups: *Data-Sharing approach* and *Pattern-Sharing approach*. In the former, the sanitization acts on the data to hide the group of sensitive association rules that contain sensitive knowledge. In the latter, the sanitizing algorithm acts on the rules mined from a database, instead of the data itself. We also introduced a taxonomy covering these two categories of sanitizing algorithms.

The set of metrics was designed to quantify not only how much sensitive knowledge has been disclosed, but also to measure the effectiveness of the sanitizing algorithms in terms of information loss and in terms of non-sensitive rules removed as a side effect of the transformation process. The proposed metrics are classified into two major groups: *Data sharing-based metrics* and *Pattern sharing-based metrics*.

It is important to note that our sanitization method is robust in the sense that there is no de-sanitization possible. The alterations to the original database are not saved anywhere since the owner of the database still keeps an original copy of the database intact while distributing the sanitized database for mining. Moreover, there is no encryption involved. There is no possible way to reproduce the original database from the sanitized one.

In Section 7.2, we will present our results of the performance evaluation for our data sharing-based algorithms, while in Section 7.3 we will present the results for our pattern sharing-based algorithm (DSA). Our performance evaluation suggests guidance on under which conditions one can use a specific sanitizing algorithm to balance privacy and knowledge discovery.

Chapter 6

Methods for Privacy-Preserving Data Clustering

You know my methods, Watson. Apply them.
– Arthur Conan Doyle, "The Memoirs of Sherlock Holmes"

Where does a wise man hide a leaf? In the forest.
But what does he do if there is no forest? ... He grows a forest to hide it in.
– G. K. Chesterton, "The Sign of the Broken Sword"

Preserving the privacy of individuals when data are shared for clustering is a complex problem. The challenge is how to protect the underlying data values subjected to clustering without jeopardizing the similarity between objects under analysis. To address this problem, data owners must not only meet privacy requirements but also guarantee valid clustering results. In this chapter, we introduce some methods for privacy-preserving clustering (PPC). Our experiments demonstrate that these methods protect individual privacy by disguising the underlying attribute values while providing useful data for clustering analysis.

This chapter is organized as follows: In Section 6.1, we describe a number of realistic scenarios in which privacy issues in clustering really matter. In Section 6.4, we show that the dual-goal of achieving privacy and accuracy can be accomplished by the idea of dissimilarity between objects but at a high communication cost. We refer to this solution as Object Similarity-Based Representation (OSBR). In order to alleviate the communication cost introduced by OSBR, we show that a trade-off between privacy and accuracy can be achieved by using the intuition behind dimensionality reduction, notably random projection [16, 1]. We refer to the latter solution as Dimensionality Reduction-Based Transformation (DRBT). The use of random projection in the context of PPC is investigated in Section 2.4. Our solution for PPC based on random projection aims at finding a trade-off between privacy and accuracy.

6.1 Motivation for Privacy-Preserving Clustering

Achieving privacy when sharing data for clustering poses new challenges for novel uses of data mining technology. Each application poses a new set of challenges. Let us consider some real-life motivating examples in which the sharing of data poses different constraints.

- Let us consider a challenging problem described in [140]. A law enforcement agency wants to cluster individuals based on their financial transactions and study the differences between the clusters and known money laundering operations. Knowing the differences and similarities between normal individuals and known money launderers would enable better direction of investigations. An individual's financial transactions may be divided between banks, credit card companies, tax collection agencies, etc, i.e., a transaction is composed of different attributes split into these parties. Each of these parties has effective controls governing release of information. How can the law enforcement agency learn about the clusters without learning anything about the attribute values of these parties? Privacy safeguards must be provided to avoid giving access to an individual's entire financial history.

- Two organizations, an Internet marketing company and an on-line retail company, have datasets with different attributes for a common set of individuals. These organizations decide to share their data for clustering to find the optimal customer targets so as to maximize return on investments. How can these organizations learn about their clusters using each other's data without learning anything about the attribute values of each other?

- Small companies have recognized the value in data, especially with the introduction of the knowledge discovery process. However, small companies do not have enough (if any) expertise for doing data analysis, although they have good domain knowledge and understand their data. They have two choices: not mining the data at all, which is not a good option due to competitive reasons, or doing it with help from outside. Outsourcing the data mining process poses a potential security threat to data. A small enterprise could hire the service of a company specialized in data mining, i.e., the data mining would be outsourced. How can this enterprise transform its data before the outsourced data mining occurs without putting in jeopardy the analysis itself?

- Suppose that a hospital shares some data for research purposes (e.g., to group patients who have a similar disease). The hospital's security administrator may suppress some identifiers (e.g., name, address, phone number, etc) from patient records to meet privacy requirements. However, the released data may not be fully protected. A patient record may contain other information that can be linked with other datasets to re-identify individuals or entities [124, 123, 134, 133]. How can we identify groups of patients with a similar disease without revealing the values of the attributes associated with them?

Note that the above scenarios describe two different problems of PPC. We refer to the first two examples as *PPC over vertically partitioned data*, and the last two scenarios as *PPC over centralized data*. In this chapter, we address the problem of PPC over centralized data. However, we show that one of our proposed methods (dimensionality reduction-based transformation) can also be used to address PPC over vertically partitioned data with little overhead in communication cost.

6.2 Addressing Privacy-Preserving Clustering

We will approach the problem of PPC by first dividing it into two sub-problems: PPC over centralized data and PPC over vertically partitioned data. In the centralized data approach, different entities are described with the same schema in a unique centralized data repository, while in a vertical partition, the attributes of the same entities are split across the partitions. We do not address the case of horizontally partitioned data.

6.2.1 PPC over Centralized Data

In this scenario, two parties, **A** and **B**, are involved, party **A** owning a dataset D and party **B** wanting to mine it for clustering. The dataset is assumed to be a data matrix $D_{m \times n}$, where each of the m rows represents an entity or object, and each entity contains values for each of the n attributes.

Before sharing the dataset D with party **B**, party **A** must transform D to preserve the privacy of individual data records. However, the transformation applied to D must not jeopardize the similarity between objects. Our first two real-life motivating examples are particular cases of PPC over centralized data. The problem associated with PPC over centralized data was stated in Section 1.2.2.

6.2.2 PPC over Vertically Partitioned Data

Consider a scenario wherein k parties, such that $k \geq 2$, have different attributes for a common set of objects, as mentioned in the last two real-life examples. In these scenarios, the goal is to do a join over the k parties and cluster the common objects. The data matrix for this case is given as follows:

$$\begin{array}{c} \vdash \text{ Party 1 } \dashv\vdash \text{ Party 2 } \dashv\vdash \quad \ldots \quad \dashv\vdash \text{ Party } k \dashv \\ \left[\begin{array}{cccc} a_{11}\ldots a_{1i} & a_{1i+1}\ldots a_{1j} & & a_{1p+1}\ldots a_{1n} \\ \vdots & \vdots & \ldots & \vdots \\ a_{m1}\ldots a_{mi} & a_{mi+1}\ldots a_{mj} & & a_{mp+1}\ldots a_{mn} \end{array}\right] \end{array} \tag{6.1}$$

Note that, after doing a join over the k parties, the problem of PPC over vertically partitioned data becomes a problem of PPC over centralized data. For simplicity, we do not consider communication cost here since this issue is addressed later.

In our model for PPC over vertically partitioned data, the parties are asymmetric, i.e., one of the parties is the central one which is in charge of merging the data and finding the clusters in the merged data. After finding the clusters, the central party would share the clustering results with the other parties. This central party could be any of the k parties. The challenge here is how to move the data from each party to a central party concealing the values of the attributes of each party. However, before moving the data to a central party, each party must transform its data to protect the privacy of the attribute values. We assume that the existence of an object (ID) should be revealed for the purpose of the join operation, but the values of the associated attributes are private.

A different approach is addressed in [140] in which all the parties participate of the computation to find the clusters. In this approach, each party learns only its input and the clustering results. This solution is based on secure multi-party computation. We refer to this approach as the symmetric model, i.e., there is no privilege among the parties.

6.2.3 The Communication Protocol

To address the problem of PPC over vertically partitioned data, we need to design a communication protocol. This protocol is used between two parties: the first party is the central one and the other represents any of the $k-1$ parties, assuming that we have k parties. We refer to the central party as $party_c$ and any of the other parties as $party_k$. There are two threads on the $party_k$ side, one for selecting the attributes to be shared, as can be seen in Table 6.1, and the other for selecting the objects before the sharing data, as can be seen in Table 6.2.

Steps to select the attributes for clustering on the $party_k$ side:
1. Negotiate the attributes for clustering before the sharing of data.
2. Wait for the list of attributes available in $party_c$.
3. Upon receiving the list of attributes from $party_c$:
a) Select the attributes of the objects to be shared.

Table 6.1: Thread of selecting the attributes on the $party_k$ side.

Steps to select the list of objects on the $party_k$ side:
1. Negotiate the list of m objects before the sharing of data.
2. Wait for the list of m object IDs.
3. Upon receiving the list of m object IDs from $party_c$:
a) Select the m objects to be shared;
b) Transform the attribute values of the m objects;
c) Send the transformed m objects to $party_c$.

Table 6.2: Thread of selecting the objects on the $party_k$ side.

6.3 Taxonomy of PPC Data Transformations

In this section, we introduce a taxonomy including our methods for PPC, as can be seen in Figure 6.1. The taxonomy basically encompasses three major categories: *Attribute Value Masking*, *Pairwise Object Similarity*, and *Attribute Reduction*. These categories are described as follows:

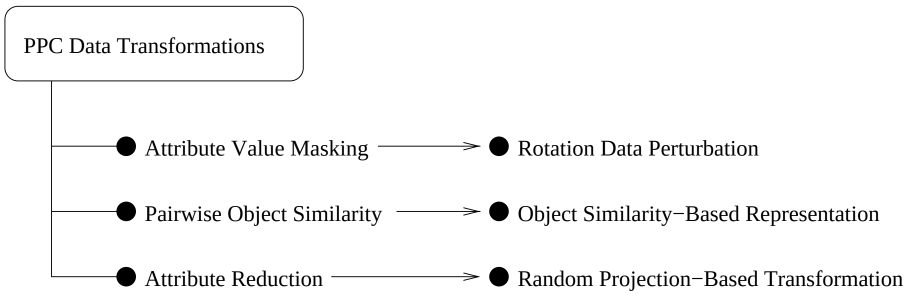

PPC Data Transformations
Attribute Value Masking
Rotation Data Perturbation
Pairwise Object Similarity
Object Similarity−Based Representation
Attribute Reduction
Random Projection−Based Transformation

Figure 6.1: A taxonomy of PPC data transformations.

Attribute Value Masking : This data transformation makes the original attribute values difficult to perceive or understand and preserves all the information for clustering analysis. Our data transformation that falls into this category is called Rotation-Based Transformation (RBT) [106]. The idea behind this technique is that the attributes of a database are split into pairwise attributes selected randomly. One attribute can be selected and rotated more than once, and the angle θ between an attribute pair is also selected randomly. RBT can be seen as a technique on the border with obfuscation. Obfuscation techniques aim at making information highly illegible without actually changing its inner meaning [33]. In other words, using RBT the original data are masked so that the transformed data capture all the information for clustering analysis while protecting the underlying data values. One interesting application of RBT is privacy preservation of health data [11].

Pairwise Object Similarity : This technique is a data matrix representation in which a data owner shares the distance of the data objects instead of the location of the data points. This technique relies on the idea of the similarity between objects, i.e., a data owner shares some data for clustering analysis by simply computing the dissimilarity matrix (matrix of distances) between the objects and then sharing such a matrix with a third party [107]. This solution is simple to implement and addresses PPC over centralized data. One of the most important advantages of this solution is that it can be applied to either categorical, binary, numerical attributes, or even a combination of these attributes. On the other hand, this solution can sometimes be restrictive since it requires a high communication cost. Our solution that lies in this category is called Object Similarity-Based Representation (OSBR), described in Section 6.4.

Attribute Reduction : In this approach, the attributes of a database are reduced to a smaller number. The small number of attributes is not a subset of the original attributes since the transformation disguises the original attribute values by projecting them onto a random space. Our data transformation that lies in this category is called Dimensionality Reduction-Based Transformation (DRBT) [107]. This data transformation can be applied to both PPC over centralized data and PPC over vertically partitioned data. The idea behind this data transformation is that by reducing the dimensionality of a database to a sufficiently small value, one can find a trade-off between privacy and accuracy. Once the dimensionality of a database is reduced, the released database preserves (or slightly modifies) the distances between data points. In tandem with the benefit of preserving the similarity between points, this solution protects individuals' privacy since the underlying data values of the objects subjected to clustering are completely different from the original ones.

In this chapter, we concentrate on the last two solutions for PPC: Object Similarity-Based Representation (OSBR) described in Section 6.4, and Dimensionality Reduction-Based Transformation (DRBT), described in Section 6.5.

6.4 The Object Similarity-Based Representation

One alternative to achieve PPC is to share the distance between data points without revealing the location of such data points. This can be accomplished by a simple and effective data representation. In the context of this thesis, we refer to such a representation as the Object Similarity-Based Representation (OSBR). The OSBR is indeed the dissimilarity matrix between a set of points, as defined in Section 2.2.3.

In this section, we investigate the feasibility of this solution for PPC over centralized and vertically partitioned data. In particular, we study the complexity of the OSBR in terms of both space requirements and communication cost. In addition, we study the security offered by the OSBR and the evidence for attacks against this solution.

6.4.1 General Assumptions

The solution to the problem of PPC, based on the similarity between objects, relies on the following assumptions:

- The data matrix D subjected to clustering could contain either binary, numerical, or categorical attributes, or even a combination of these attributes.
- An object (ID) should be replaced by a fictitious identifier.

6.4.2 PPC over Centralized Data

To address PPC over centralized data, the OSBR performs three major steps before sharing the data for clustering, as follows:

- *Step 1 - Suppressing identifiers*: Attributes that are not subjected to clustering (e.g., address, phone number, etc) are suppressed.
- *Step 2 - Normalizing numerical attributes*: If all the attributes subjected to clustering are numerical, they should be normalized. Normalization helps prevent attributes with large ranges (e.g., salary) from outweighing attributes with smaller ranges (e.g., age). If the dataset contains mixed variables, there is no need for normalization. The distances between objects are normalized when computing the dissimilarity matrix using Equation (2.8).
- *Step 3 - Computing the dissimilarity matrix*: In the last step, the pairwise distances between objects are computed. Euclidean distance is widely used for numerical attributes and the Jaccard coefficient for binary attributes. If the attributes are mixed types, Equation (2.8) can be used.

To illustrate how this solution works, let us consider the sample relational database in Table 6.3. This sample contains real data from the Cardiac Arrhythmia Database available at the UCI Repository of Machine Learning Databases [17]. The attributes for this example are: *age*, *weight*, *h_rate* (number of heart beats per minute), *int_def* (number of intrinsic deflections), *QRS* (average of QRS duration in msec.), and *PR_int* (average duration between onset of P and Q waves in msec.).

Now suppose these data are made available for research purposes. One may be interested in clustering patients with similar characteristics to give a specific treatment to each group. Our goal here is to protect the underlying attribute values and, at the same time, guarantee accurate clustering results. Following the three steps of the OSBR, the dissimilarity matrix D_M corresponding to the data matrix in Table 6.3, using the Euclidean distance in Equation (2.5), is given as follows:

ID	age	weight	h_rate	int_def	QRS	PR_int
123	75	80	63	32	91	193
342	56	64	53	24	81	174
254	40	52	70	24	77	129
446	28	58	76	40	83	251
286	44	90	68	44	109	128

Table 6.3: A cardiac arrhythmia database.

$$D_M = \begin{bmatrix} 0 & & & & \\ 2.2436 & 0 & & & \\ 3.3489 & 2.4776 & 0 & & \\ 3.6903 & 3.8844 & 3.1767 & 0 & \\ 3.0203 & 4.0828 & 4.1303 & 3.9955 & 0 \end{bmatrix} \tag{6.2}$$

The dissimilarity matrix is the dataset shared for clustering. Many clustering algorithms in the literature operate on a dissimilarity matrix [65]. In Section 6.4.3, we show that a dissimilarity matrix is no longer invertible, as long as the data analysts have no extra knowledge concerning the original data. This situation generally occurs when the attribute values are not split across many parties.

6.4.3 How Secure is the OSBR?

Now we move on to show that sharing a dissimilarity matrix is a secure procedure, as long as the original data points are not revealed to the other parties. Our goal here is to show that given the distance between two d-dimensional vectors, one cannot determine the coordinates of these two vectors.

Lemma 2 *Let $DM_{m \times m}$ be a dissimilarity matrix, where m is the number of objects. It is impossible to determine the coordinates of the two objects by knowing only the distance between them.*

Proof: Let i and j be any two vectors in a d-dimensional space and let r be the distance between these vectors. For any given distance r, there exist infinitely many pairs of vectors i and j such that $d(i, j) = r$. In fact, for every vector i there exists a vector j such that $d(i, j) = r$. Therefore, the coordinates of i can be chosen completely arbitrarily, and there is no way to deduce the coordinates of i from r. □

Even when sufficient care is taken, a solution that adheres to OSBR can be still vulnerable to partial disclosure. For instance, suppose that a user who has access to a dissimilarity matrix, shared by a data owner, knows all the attribute values of one particular object o_i. In this case, partial disclosure may occur.

Lemma 3 *Knowing the coordinates of a particular object i and the distance r between i and any other object j, it is possible to estimate the attribute values of j.*

Proof: Let i and j be any two vectors in a d-dimensional space and let r be the distance between these vectors. If all the coordinates of i are known, then every coordinate of j cannot differ from the corresponding coordinate of i by more than r since j will lie on the sphere of radius r. In this case, one can make some estimates of the coordinates of j, as long as some attribute ranges are known. $\square$

In [22], Caetano shows that if an adversary knows the dissimilarity matrix of a set of points and the coordinates of $d + 1$ points, where d is the number of dimensions of the data points, it is possible to disclose the entire dataset. This result holds if and only if the $d + 1$ points do not lie in a $(d - 1)$-dimensional vector subspace.

To illustrate Caetano's lemma, let us consider a particular case in $\Re^2$, as can be seen in Figure 6.2. In this example, suppose that three points $(d + 1)$ objects $(d = 2)$ and their distances are known. If the center of the dashed circle does not lie in the same straight line, the $(d-1)$-dimensional vector subspace, defined by the centers of the other two circles, the intersection set has at most one point (the one pointed to by the arrow). Thus, if one adversary has the distances of other p points to these three points in Figure 6.2, s/he can determine the coordinates of the p points.

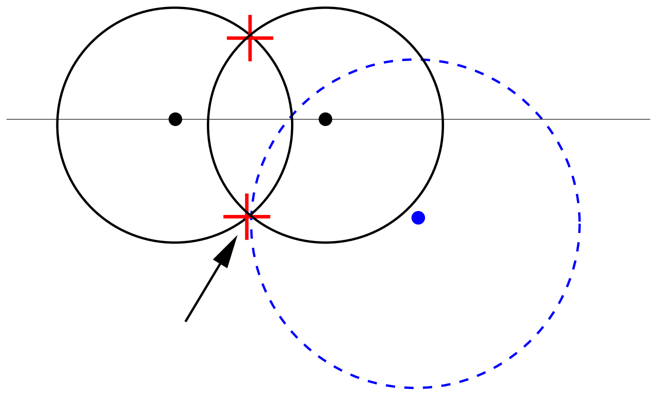


Figure 6.2: An example of Caetano's lemma in $\Re^2$ [22].

Lemma 3 and Caetano's lemma suggest that the OSBR can sometimes be ineffective to address PPC over vertically partitioned data. The main reason is that knowing the attribute join (the object's ID used for the join purpose) and some data points (objects' attribute values), one may disclose the entire dissimilarity matrix. Therefore, we suggest that the OSBR should be used to address PPC over centralized data since an adversary does not have external knowledge to disclose the attribute values of the original dataset.

6.4.4 The Complexity of the OSBR

Here we analyze the space requirements and communication costs for the OSBR. A dissimilarity matrix is an $m \times m$ table, where m is the number of objects under analysis. When

m grows, which is not unexpected in data mining applications, this solution becomes too expensive, especially in terms of both space requirements and communication cost.

For instance, for a data matrix where m = 1,000,000 objects, the number of pairwise distances (elements of the matrix) is a combination of m elements taken two at a time, i.e., $C_{m,2} = (m \times (m-1))/2 = 5\text{e}{+}11$. Thus, the space requirements of the OSBR is of the order $O(m^2)$. However, only half of the dissimilarity matrix is transmitted from one party to another since distance is a symmetric function. Thus, the communication cost of the OSBR takes $O(((m \times (m-1))/2) \times l)$, where l represents the size (in bits) of one element of the $m \times m$ dissimilarity matrix.

6.4.5 The Advantages and Disadvantages of OSBR

We have shown that it is possible to address PPC over centralized data based on the concept of the dissimilarity matrix. The main advantages of this solution are:

- it is independent of the distance-based clustering method.
- it preserves the privacy of the attribute values subjected to clustering.
- it is accurate and very simple to implement.
- it can be applied to datasets containing binary, numerical, or categorical attributes, or even a combination of these attributes.

On the other hand, in the context of PPC over vertically partitioned data, the OSBR present two limitations, as follows:

- Lemma 3 shows the restriction of the OSBR when an adversary has external knowledge of the original data. When two or more parties share data for clustering, if one party knows all the coordinates of a few points, the dissimilarity matrix may disclose the original dataset.
- The significant communication cost of the OSBR indicates that this solution is not attractive for PPC over vertically partitioned data. In addition, in vertically partitioned data, IDs need to be shared among the parties, which would lead to the problem illustrated in Lemma 3.

The above limitations motivate our next solution based on the idea of dimensionality reduction.

6.5 The Dimensionality Reduction Transformation

When two or more parties share data for clustering, privacy preservation and the communication cost are key requirements. On the one hand, the sharing of data requires privacy safeguards. On the other hand, sharing a large amount of data for mining can be, in many cases, prohibitive.

In this section, we show that the dual-goal of achieving privacy preservation and a reduced communication cost in PPC can be accomplished by dimensionality reduction. By reducing the dimensionality of a dataset to a sufficiently small value, one can find a trade-off between privacy and accuracy. In particular, random project (see the basics of random projection in Section 2.4, page 25) can fulfill this dual-goal. We refer to this solution as the Dimensionality Reduction-Based Transformation (DRBT).

Dimensionality reduction techniques have been studied in the context of pattern recognition [59], information retrieval [16, 51, 72], and data mining [57, 51]. To our best knowledge, dimensionality reduction has not been used in the context of data privacy in any detail.

6.5.1 General Assumptions

The solution to the problem of PPC based on random projection draws the following assumptions:

- The data matrix D subjected to clustering contains only numerical attributes that must be transformed to protect individuals' data values before the data sharing for clustering occurs.
- In PPC over centralized data, the existence of an object (ID) should be replaced by a fictitious identifier. In PPC over vertically partitioned data, the IDs of the objects are used for the join purposes between the parties involved in the solution.
- The transformation (random projection) applied to the original data might slightly modify the distance between data points. Such a transformation justifies the trade-off between privacy and accuracy.

One interesting characteristic of the solution based on random projection is that, once the dimensionality of a database is reduced, the attribute names in the released database are irrelevant. In other words, the released database preserves, in general, the similarity between the objects, but the underlying data values are completely different from the original ones. We refer to the released database as a *disguised database*, which is shared for clustering.

6.5.2 PPC over Centralized Data

To address PPC over centralized data, the DRBT performs three major steps before sharing the data for clustering:

- *Step 1 - Suppressing identifiers*: Attributes that are not subjected to clustering (e.g., address, phone number, etc.) are suppressed.

- *Step 2 - Reducing the dimension of the original dataset*: After pre-processing the data according to *Step 1*, an original dataset D is then transformed into the disguised dataset D' using random projection.

- *Step 3 - Computing the stress function*: This function is used to determine whether the accuracy of the transformed dataset is marginally modified, which guarantees the usefulness of the data for clustering.

To illustrate how this solution works, let us consider the sample relational database in Table 6.3. We are going to reduce the dimension of this dataset from 6 to 3, one at a time, and compute the error (stress function). To reduce the dimension of this dataset, we apply Equation (2.10). In this example, the original dataset corresponds to the matrix D. We compute a random matrix R_1 by setting each entry of the matrix to a value drawn from an independent and identically distributed (i.i.d.) $N(0,1)$ distribution and then normalizing the columns to unit length. We also compute a random matrix R_2 where each element r_{ij} is computed using Equation (2.12). We transform D into D' using both R_1 and R_2. The random transformation RP_1 refers to the random projection using R_1, and RP_2 refers to the random projection using R_2.

The relative error that the distances in 6-3 space suffer from, on the average, is computed using Equation (2.9). Table 6.4 shows the values of the error using RP_1 and RP_2. In this Table, k represents the number of dimensions in the disguised database D'.

Transformation	k = 6	k = 5	k = 4	k = 3
RP_1	0.0000	0.0223	0.0490	0.2425
RP_2	0.0000	0.0281	0.0375	0.1120

Table 6.4: The relative error that the distances in 6-3 space suffer from, on the average.

In this case, we have reduced the dimension of D from 6 to 3, i.e, the transformed dataset has only 50% of the dimensions in the original dataset. Note that the error is relatively small for both RP_1 and RP_2, especially for RP_2. However, this error is minimized when the random projection is applied to high dimensional datasets, as can be seen in Figure 7.3, in Section 7.4.2.

After applying random projection to a dataset, the attribute values of the transformed dataset are completely disguised to preserve the privacy of individuals. Table 6.5 shows the attribute values of the transformed database with 3 dimensions, using both RP_1 and RP_2. In this table, we have the attributes labeled *Att1*, *Att2*, and *Att3* since we do not know the labels for the disguised dataset. Using random projection, one cannot select the attributes to be reduced beforehand. The attributes are reduced randomly. More formally, $\forall i$ if $Attr_i \in D'$, then $Attr_i \notin D$.

ID	D' using RP_1			D' using RP_2		
	Att1	Att2	Att3	Att1	Att2	Att3
123	-50.40	17.33	12.31	-55.50	-95.26	-107.96
342	-37.08	6.27	12.22	-51.00	-84.29	-83.13
254	-55.86	20.69	-0.66	-65.50	-70.43	-66.97
446	-37.61	-31.66	-17.58	-85.50	-140.87	-72.74
286	-62.72	37.64	18.16	-88.50	-50.22	-102.76

Table 6.5: Disguised dataset D' using RP_1 and RP_2.

As can be seen in Table 6.5, the attribute values are entirely different from those in Table 6.3.

6.5.3 PPC over Vertically Partitioned Data

The solution for PPC over vertically partitioned data is a generalization of the solution for PPC over centralized data. In particular, if we have k parties involved in this case, each party must apply the random projection over its dataset and then send the reduced data matrix to a central party. Note that any of the k parties can be the central one. We show in Section 6.5.6 that DRBT greatly alleviates the communication cost when compared with the communication cost in OSBR.

When k parties ($k \geq 2$) share some data for PPC over vertically partitioned data, these parties must satisfy the following constraints:

- *Agreement*: The k parties must follow the communication protocol described in Section 6.2.3.

- *Mutual exclusivity*: We assume that the attribute split across the k parties are mutually exclusive. More formally, if $A(D_1), A(D_2)..., A(D_k)$ are a set of attributes of the k parties, $\forall i \neq j\ A(D_i) \cap A(D_j) = \emptyset$. The only exception is that IDs are shared for the join purpose.

The solution based on random projection for PPC over vertically partitioned data is performed as follows:

- *Step 1 - Individual transformation*: If k parties, $k \geq 2$, share their data in a collaborative project for clustering, each party k_i must transform its data according to the steps in Section 6.5.2.

- *Step 2 - Data exchanging or sharing*: Once the data are disguised by using random projection, the k parties are able to exchange the data among themselves. However, one party could be the central one to aggregate and cluster the data.

- *Step 3 - Sharing clustering results*: After the data have been aggregated and mined in a central party k_i, the results could be shared with the other parties.

6.5.4 How Secure is the DRBT?

In the previous sections, we showed that transforming a database using random projection is a promising solution for PPC over centralized data and consequently for PPC over vertically partitioned data since the similarities between objects are marginally changed. Now we show that random projection also has promising theoretical properties for privacy preservation. In particular, we demonstrate that a random projection from d dimensions to k, where $k \ll d$, is a non-invertible transformation.

Lemma 4 *A random projection from d dimensions to k dimensions, where $k \ll d$, is a non-invertible linear transformation.*

Proof: A classic result from Linear Algebra asserts that there is no invertible linear transformation between Euclidean spaces of different dimensions [13]. Thus, if there is an invertible linear transformations from $\Re^m$ to $\Re^n$, then the constraint $m = n$ must hold. A random projection is a linear transformation from $\Re^d$ to $\Re^k$, where $k \ll d$. Hence, a random projection from d dimensions to k dimensions is a non-invertible linear transformation. $\square$

When a set of points in a high dimensional space are mapped onto a lower dimensional space by random projection, the coordinates of the points in the low space are completely disguised. In other words, there is no means to reconstruct the coordinates of the points in the original space based on the coordinates of the points in the low dimensional space. Therefore, a data owner would not give away the original points. The only useful information preserved in the lower dimensional space are the distances between the points but with a relatively small error that is acceptable for practical applications. That is the basis of privacy preservation of the DRBT.

6.5.5 The Accuracy of the DRBT

In Section 2.4, we pointed out that random projection has emerged as a powerful method for dimensionality reduction. Theoretical results indicate that the method preserves distances quite nicely. Some effort has been done to determine the proper number of dimensions for a random projection to effectively preserve distances. However, to our knowledge it is still an open question how to choose the dimensionality for a random projection in order to preserve separation among clusters in general clustering applications.

In the context of PPC, our goal is to evaluate the quality of data for general clustering applications after dimensionality reduction. In particular, the results presented in [130, 57] suggest that random projection is promising for clustering.

	c'_1	c'_2	...	c'_k
c_1	$freq_{1,1}$	$freq_{1,2}$	...	$freq_{1,k}$
c_2	$freq_{2,1}$	$freq_{2,2}$	...	$freq_{2,k}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
c_k	$freq_{k,1}$	$freq_{k,2}$	...	$freq_{k,k}$

Table 6.6: The number of points in cluster c_i that falls in cluster c'_j in the transformed dataset.

When using random projection, a perfect reproduction of the Euclidean distances may not be the best possible result. The clusters in the transformed datasets should be equal to those in the original database. However, this is not always the case, and we have some potential problems after dimensionality reduction: a) a noise data point ends up clustered; b) a point from a cluster becomes a noise point; and c) a point from a cluster migrates to a different cluster. In this research, we focus primarily on partitioning methods. In particular, we use K-Means [93], one the most used clustering algorithms. Since K-means is sensitive to noise points and clusters all the points in a dataset, we have to deal with the third problem mentioned above (a point from a cluster migrates to a different cluster).

Our evaluation approach focuses on the overall quality of generated clusters after dimensionality reduction. We compare how closely each cluster in the transformed data matches its corresponding cluster in the original dataset. To do so, we first identify the matching of clusters by computing the matrix of frequencies showed in Table 6.6. We refer to such a matrix as the clustering membership matrix (CMM), where the rows represent the clusters in the original dataset, the columns represent the clusters in the transformed dataset, and $freq_{i,j}$ is the number of points in cluster c_i that falls in cluster c'_j in the transformed dataset.

After computing the frequencies $freq_{i,j}$, we scan the clustering membership matrix calculating precision, recall, and F-measure for each cluster c'_j with respect to c_i in the original dataset [90]. These formulas are given by the following equations:

$$Precision\ (P) = \frac{freq_{i,j}}{|c'_i|} \tag{6.3}$$

$$Recall\ (R) = \frac{freq_{i,j}}{|c_i|} \tag{6.4}$$

$$F - measure\ (F) = \frac{2 \times P \times R}{(P + R)} \tag{6.5}$$

where $|X|$ is the number of points in the cluster X.

For each cluster c_i, we first find a cluster c'_j that has the highest F-measure among all the c'_l, $1 \leq l \leq k$. Let $F(c_i)$ be the highest F-measure for cluster c_i, we denote the overall F-measure as the weighted average of $F(c_i)$, $1 \leq i \leq k$, as follows:

$$Overall\ F - measure\ (OF) = \frac{\sum_{i=1}^{k} |c_i| \times F(c_i)}{\sum_{i=1}^{k} |c_i|} \tag{6.6}$$

In section 7.4, we present our performance evaluation results for clustering based on Equation (6.6).

6.5.6 The Complexity of the DRBT

One of the major benefits of a solution that adheres to the DRBT is the communication cost to send a disguised dataset from one party to a central one. In general, a disguised data matrix is of size $m \times k$, where m is the number of objects and k is the number of attributes (dimensions). The complexity of DRBT is of the order $O(m \times k)$, however $k \ll m$.

To quantify the communication cost of one solution, we consider the number of bits or words required to transmit a dataset from one party to a central or third party. Using DRBT, the bit communication cost to transmit a dataset from one party to another is $O(mlk)$, where l represents the size (in bits) of one element of the $m \times k$ disguised data matrix.

Recall that the communication cost in the OSBR is much more expensive, i.e., $O(((m \times (m-1))/2) \times l)$, where l represents the size (in bits) of one element of the $m \times m$ dissimilarity matrix. Clearly, the solution that adheres to the DRBT is much more effective for addressing the problem of PPC over vertically partitioned data.

6.6 Summary

In this chapter, we have shown that transforming data to address PPC is to some extent possible. In particular, we showed that the challenging goal of achieving full privacy and accuracy can be accomplished by the idea of dissimilarity between objects, but at a high communication cost. We referred to this solution as the Object Similarity-Based Representation (OSBR). One major advantage of the OSBR is that it can be applied to datasets containing either binary, numerical, or categorical attributes, or even a combination of these attributes. On the other hand, we showed that the OSBR is ineffective to address PPC over vertically partitioned data when an adversary has external knowledge of some attributes subjected to clustering. As a result, the OSBR is more effective to address PPC over centralized data.

In order to alleviate the communication cost introduced by the OSBR, we showed that a trade-off between privacy, accuracy, and communication cost can be achieved by using the intuition behind random projection. We referred to this solution as the Dimensionality Reduction-Based Transformation (DRBT). This solution is promising to either PPC over centralized or vertically partitioned data since it greatly alleviates communication costs while preserving the accuracy of reduced data as well as the accuracy of the original data.

The highlights of our approaches are as follows: a) they are independent of distance-based clustering algorithms; b) they have a sound mathematical foundation; and c) they do not require CPU-intensive operations.

In Section 7.4, we will present our results of the performance evaluation for our solution for PPC based on dimensionality reduction. We show that the DRBT maintains the usefulness of the data and provides acceptable values in practice to address privacy concerns in clustering.

Chapter 7

Results and Evaluation

Your true value depends entirely on what you are compared with.
– Bob Wells

In this section, we empirically validate our privacy-preserving data transformation (PPDT) methods for privacy-preserving association rule mining (PPARM) and privacy-preserving clustering (PPC). We start by describing the real datasets used in our experiments in Section 7.1. In Section 7.2, we present the results for the performance evaluation of our data sharing-based algorithms. We conduct two series of experiments, one to study the effectiveness of our sanitizing algorithms and the other to evaluate their scalability. In Section 7.3, we validate our pattern sharing-based algorithm, the Downright Sanitizing algorithm (DSA). To do so, we also conduct two series of experiments to study the effectiveness and the scalability of the DSA. For both categories of algorithms (data sharing-based and pattern sharing-based), the performance evaluation offers guidance on under which conditions one can use a specific sanitizing algorithm to balance privacy and knowledge discovery. In Section 7.4, we evaluate the dimensionality reduction-based transformation (DRBT) proposed to address PPC. Our experiments demonstrate that the DRBT protects individuals' privacy by disguising the underlying attribute values of a dataset while providing useful data for clustering. Finally, in Section 7.5 we report the main results observed and the conclusions drawn from our experiments.

7.1 Datasets

We validated our methods for privacy-preserving clustering and privacy-preserving association rule mining using nine real datasets. These datasets are described as follows:

1. **BMS-Web-View-1 (BMS-1):** This dataset contains click-stream data from the web site of a (now defunct) legwear and legcare retailer. The dataset contains 59,602 transactions with 497 distinct items, and each customer purchasing has 2.51 items on

average. BMS-Web-View-1 [155] has been placed in the public domain of the company Blue Martini Software.

2. **Retail:** This dataset contains the (anonymized) retail market basket data from an anonymous Belgian retail supermarket store [19]. The data were collected over three non-consecutive periods. The first period ran from mid December 1999 to mid January 2000. The second period ran from 2000 to the beginning of June 2000. The third period ran from the end of August 2000 to the end of November 2000. The total number of receipts collected was 88,162 with 16,470 distinct items, and each customer purchasing has 10.3 items on average.

3. **Accidents:** This dataset concerning traffic accidents was obtained from the National Institute of Statistics (NIS) for the region of Flanders in Belgium. The transactions are traffic accident forms filled out by police officers for each traffic accident that occurred involving injures or deaths on a public road in Belgium. There are 340,183 traffic accident records included in the dataset, with 468 distinct items, and each record has 33.81 attribute values (items) on average.

4. **Kosarak:** This dataset contains (anonymized) click-stream data from a Hungarian online news portal. Kosarak[1] has 990,007 transactions with 41,270 distinct items, and each customer purchasing has 8.1 items on average.

5. **Reuters:** The Reuters-21578 text collection is composed of 7,774 transactions with 26,639 distinct items and 46.81 items on average per transaction. The Reuters dataset is available at the UCI Repository of Machine Learning Databases [17].

6. **Mushroom:** This dataset is available at the UCI Repository of Machine Learning Databases [17]. Mushroom contains records drawn from The Audubon Society Field Guide to North American Mushrooms. There are 8,124 records with 119 distinct items (data values) and 23 numerical attributes.

7. **Chess:** The format for instances in this database is a sequence of 37 attribute values. Each instance is a board-descriptions of a chess endgame. The first 36 attributes describe the board. The last (37th) attribute is the classification: ”win” or ”nowin”. Chess is available at the UCI Repository of Machine Learning Databases [17] and contains 3,196 records. There is no missing value in this dataset.

8. **Connect:** This database contains all legal 8-ply positions in the game of connect-4 in which neither player has won yet, and in which the next move is not forced. Connect

[1]This dataset was provided to us by Ferenc Bodon. A copy of the dataset is placed in the Frequent Itemset Mining Implementations Repository: http://fimi.cs.helsinki.fi/data/

is composed of 67,557 records and 43 attributes without missing values. This dataset is also available at the UCI Repository of Machine Learning Databases [17].

9. **Pumsb:** The Pumsb dataset contains census data for population and housing. This dataset is available at http://www.almaden.ibm.com/software/quest. There are 49,046 records with 2,113 different data values (distinct items), and 74 attribute values without missing values.

Table 7.1 shows the summary of the datasets used in our experiments. The columns represent, respectively, the database name, the total number of records, the number of distinct items, the average number of items per record (transaction), the size of the shortest record, and the size of largest record.

Dataset	#records	# items	Avg. Length	Shortest Record	Largest Record
BMS-Web-View-1	59,602	497	2.51	1	145
Retail	88,162	16,470	10.30	1	76
Accidents	340,183	468	33.81	18	51
Kosarak	990,573	41,270	8.10	1	1065
Reuters	7,774	26,639	46.81	1	427
Mushroom	8,124	119	23	23	23
Chess	3,196	75	37	37	37
Connect	67,557	129	43	43	43
Pumbs	49,046	2,113	74	74	74

Table 7.1: A summary of the datasets used in our experiments

7.2 Evaluation of the Data Sharing-Based Algorithms

In this section, we present the results of our performance evaluation. We study the effectiveness and scalability of our data sharing-based algorithms and the similar counterpart in the literature.

7.2.1 Sanitizing Algorithms

The sanitizing algorithms, under analysis in this section, are those that rely on the Heuristics 1 and 2 described in Section 5.3. These algorithms are described as follows:

- The *Item Grouping Algorithm (IGA)* groups sensitive association rules in clusters of rules sharing the same itemsets. If two or more sensitive rules intersect, by sanitizing the shared item of these sensitive rules, one would take care of hiding such sensitive rules in one step;

- The *Round Robin Algorithm (RRA)* selects different victim items in turns starting from the first item, then the second, and so over the set of sensitive transactions. The process starts again at the first item of the sensitive rule as a victim item each time the last item is reached.

- The *Random Algorithm (RA)* selects a victim item for a given sensitive rule sr_i randomly. For each sensitive transaction associated with sr_i, RA randomly selects a victim item.

- The *Sliding Window Algorithm (SWA)* scans one group of K transactions at a time and sanitizes the sensitive rules present in such transactions based on a set of disclosure thresholds defined by a database owner. There is a disclosure threshold assigned to each sensitive association rule.

- *Algo2a* is a similar counterpart sanitizing algorithm which hides sensitive rules by reducing support [37]. The algorithm GIH, designed by Saygin et al. [126], is similar to Algo2a. The basic difference is that in Algo2a, some items are removed from sensitive transactions, while in GIH a mark "?" (unknowns) is placed instead of item deletions. To our best knowledge there is no other similar sanitizing algorithm in the literature. The algorithms published in [142] are an extension of the algorithms published in [37, 126].

A different line of sanitizing algorithms was introduced in [142] in which the sensitive rules are hidden by reducing their confidence below a privacy threshold. Our sanitization process focuses primarily on hiding rules by reducing their support. The main reason is that most of the interestingness measures (e.g., confidence, lift, coverage, conviction, etc.) for association rules rely on support [135], as discussed in Section 2.3.2.

7.2.2 Methodology

We performed two series of experiments. The first series was performed to evaluate the effectiveness of our sanitizing algorithms, and the second to measure their efficiency and scalability. One question that we wanted to answer was:

> *Under which conditions can one use a specific sanitizing algorithm to balance privacy and knowledge discovery?*

We purposely selected the sensitive rules to be sanitized based on four different scenarios, as follows:

- **S1**: The sensitive rules selected contain only items that are mutually exclusive. In other words, there is no intersection of items over all the sensitive rules. The purpose of this scenario is to unfavor the algorithms IGA and SWA, both of which take advantage of rule overlaps.

- **S2**: In this scenario, the sensitive rules were selected randomly.
- **S3**: Only sensitive rules with very high support were selected. Sanitizing such rules would maximize the differential between an original dataset and its corresponding sanitized dataset.
- **S4**: Only sensitive rules with low support were selected. Sanitizing such rules would minimize the differential between an original dataset and its corresponding sanitized dataset.

Our comparison study was carried out through the following steps:

- *Step 1*: we selected the datasets BMS-1, Retail, Kosarak, and Reuters. The first three datasets are specific for association rule mining, and the last one contains long transactions, on average, with high frequency items.
- *Step 2*: we ran an association rule mining algorithm with a low minimum support threshold σ to capture as many association rules as possible. Subsequently, we selected the sensitive rules to be sanitized based on the four scenarios described above.
- *Step 3*: we compared the sanitizing algorithms described in Section 7.2.1 against each other and with respect to the following benchmark: the results of association rules mined in the original (D) and sanitized (D') datasets. We used our metrics described in Section 5.2.3 to measure information loss (misses cost, and the difference between D and D'), disclosure of private information (hiding failure), and fraction of artifactual rules created by the sanitization process.

All the experiments were conducted on a PC (AMD 3200/2200) with 1.2 GB of RAM running a Linux operating system. In our experiments, we selected four sets of sensitive rules for each dataset based on the scenarios described above (S1 - S4). These rules, for each individual dataset, can be found in Appendix B.1. Table 7.2 shows the parameters we used to mine the datasets before the selection of the sensitive rules.

Dataset	Support (%)	Confidence (%)	No. Rules	Max. Size
BMS-1	0.1	60	25,391	7 items
Retail	0.1	60	7,319	6 items
Reuters	5.5	60	16,559	10 items
Kosarak	0.2	60	349,554	13 items

Table 7.2: Parameters used for mining the four datasets

7.2.3 Evaluating the Window Size for SWA

We evaluated the effect of the window size, for the SWA algorithm, with respect to the difference between an original dataset D and its corresponding sanitized dataset D', misses cost, and hiding failure. To do so, we varied the K (window size) from 500 to 100,000 transactions with the disclosure threshold $\psi = 25\%$. Table B.17 in Appendix B.2 shows that for up to 5,000 transactions, the difference between D and D' and misses cost improve slightly for the Reuters dataset. Similarly, these metrics improve after 40,000 transactions for the datasets Kosarak, Retail, and BMS-1. The results reveal that a window size representing 64.31% of the size of the Reuters dataset suffices to stabilize the misses cost and hiding failure, while a window size representing 4.04%, 45.37%, and 67.11% is necessary to stabilize the same measures in the datasets Kosarak, Retail, and BMS-1, respectively.

In this example, we intentionally selected a set of 6 sensitive association rules with high support (scenario S3) to accentuate the differential between the sizes of the original database and the sanitized database and thus to better illustrate the effect of window size on the difference between D and D', misses cost, and hiding failure.

Note that the distribution of the data affects the values for misses cost and hiding failure. To obtain the best results for misses cost and hiding failure, hereafter we set the window size K to 50,000 in our experiments.

7.2.4 Measuring Effectiveness

The effectiveness of the sanitizing algorithms is measured in terms of the number of sensitive association rules effectively hidden, as well as the proportion of non-sensitive rules accidentally hidden due to the sanitization process.

We studied the effectiveness of the sanitizing algorithms based on three major conditions, as follows:

- *C1*: we set the disclosure threshold ψ to 0% and fixed the minimum support threshold σ, the minimum confidence threshold φ, and the number of sensitive rules to hide.
- *C2*: we fixed the parameters as we did in condition *C1* but varied the number of sensitive rules to hide.
- *C3*: we set the disclosure threshold ψ to 0%, fixed the minimum confidence threshold φ and the number of sensitive rules to hide, and varied the minimum support threshold σ for each dataset.

Note that in all the three conditions above, we purposely set the disclosure threshold ψ to 0%. In this particular case, no sensitive rule is allowed to be mined from the sanitized dataset. Later (in special cases section), we will show that a database owner could also slide the disclosure threshold ($\psi > 0$) to allow a balance between knowledge discovery and privacy protection in the sanitized database.

Table 7.3 shows a summary of the best sanitizing algorithms, in terms of misses cost, under condition C1. For further information on the values of misses cost (the lower the better) under condition C1, we refer the reader to Appendix B.3.1. The algorithm IGA yielded the best results in almost all the cases. The exceptions are the scenarios S2, S3, and S4 of the dataset Retail that contains sensitive rules with high support items. In this case, the algorithms SWA and RA benefit from the selection of the victim items, a choice which varies in each sensitive transaction, alleviating the impact on the sanitized dataset. As a result, the values for misses cost are slightly minimized.

Dataset	$\psi = 0\%$, 6 sensitive rules			
	S1	S2	S3	S4
Kosarak	IGA	IGA	IGA	IGA
Retail	IGA	SWA	RA	RA
Reuters	IGA	IGA	IGA	IGA
BMS-1	IGA	IGA	IGA	IGA

Table 7.3: Summary of the best algorithms in terms of misses cost under condition C1.

We also observed the same behaviour of the sanitizing algorithms when analyzing misses cost under conditions C2 and C3. Table 7.4 provides a summary of the best sanitizing algorithms in terms of misses cost under condition C2. The results confirm the same finding we reported previously for the algorithms SWA and RA for the dataset Retail. The only exceptions here are the scenario S2 for the datasets Retail and BMS-1. In the dataset Retail, IGA yielded the best results of misses cost with up to 2 sensitive rules being sanitized. As the number of sensitive rules increased, the SWA yielded the best values of misses cost. The same behavior can be observed in the dataset BMS-1, i.e., Algo2a yielded the best values for misses cost with up to 3 rules being sanitized. When the number of rules to be sanitized increases, IGA yields the best results for misses cost. The values for misses cost under condition C2 can be found in Appendix B.3.2.

Dataset	$\psi = 0\%$, varying the no. of rules			
	S1	S2	S3	S4
Kosarak	IGA	IGA	IGA	IGA
Retail	IGA	IGA/SWA	RA	RA
Reuters	IGA	IGA	IGA	IGA
BMS-1	IGA	Algo2a/IGA	IGA	IGA

Table 7.4: Summary of the best algorithms for misses cost under condition C2.

Table 7.5 shows the results of misses cost under condition C3. As we can see, the same trends were observed, except in scenario S4 for the dataset Retail in which the algorithm

RRA yielded better results than those in the RA. However, the results for misses cost in the RRA are slightly better, as can be seen in Appendix B.3.3.

Dataset	$\psi = 0\%$, varying the values of σ			
	S1	S2	S3	S4
Kosarak	IGA	IGA	IGA	IGA
Retail	IGA	SWA	RA	RRA
Reuters	IGA	IGA	IGA	IGA
BMS-1	IGA	IGA	IGA	IGA

Table 7.5: Summary of the best algorithms for misses cost under condition C3.

We also investigated the differential between the initial size of the four datasets and the size of the sanitized datasets under the three conditions described above. Under condition C1 and C3, the algorithm SWA yielded results slightly better than those in the other algorithms, as can be seen in Table 7.6. Details about these results can be found in Appendix B.4.1 and Appendix B.4.3.

Dataset	$\psi = 0\%$, 6 sensitive rules			
	S1	S2	S3	S4
Kosarak	SWA	SWA	SWA	SWA
Retail	SWA	SWA	SWA	SWA
Reuters	SWA	SWA	SWA	SWA
BMS-1	SWA	SWA	SWA	SWA

Table 7.6: Summary of the best algorithms for dif(D, D') under conditions C1 and C3.

Under condition C2, when we increased the number of rules to be sanitized, the algorithms RA, RRA, and SWA yielded the best results for the differential between the original and the sanitized datasets, i.e., dif(D, D'). These results are available at Appendix B.4.2.

Based on the results for dif(D, D'), a natural question arises: *how can SWA, RA, and RRA get the best results for dif(D, D') and not for misses cost?* The main reason is that the victim items in these three algorithms are dynamic, i.e., a new victim item is selected for each sensitive transaction to be sanitized. This approach reduces support of every item in a sensitive rule (one item is selected for each sensitive transaction) regardless of whether an item has high or low support. Reducing items with high support would prune the candidate generation of discovered rules in the sanitized dataset, compromising the values of misses cost. On the contrary, the victim item selected by the IGA, for a sensitive rule, is fixed for all sensitive transactions. Moreover, the IGA always selects the item with lower support for each rule, which greatly improves the values of misses cost.

Regarding the third performance measure, artifactual patterns, one may claim that when we decrease the frequencies of some items, the relative frequencies in the database may be modified by the sanitization process, and new rules may emerge. However, in our experiments, the problem artifactual pattern AP was always 0% with all algorithms regardless of the values of ψ. Our sanitization, indeed, does not remove any transaction. The same results could be observed for the algorithm Algo2a. On the other hand, some of the sanitizing algorithms introduced in [142] present the case in which artifactual patterns appear (i.e., $AP > 0$), since the sensitive rules are hidden by reducing their confidence below a privacy threshold. To do so, some items are added to transactions that participate in the generation of the antecedent part X, but not the consequent part Y of a rule, where the rule is the form $X \rightarrow Y$. Adding items to some transactions results in the generation of new association rules that are not supposed to exist in the original database.

7.2.5 Special Cases of Data Sanitization

There are two special cases of data sanitization regarding the data sharing-based algorithms validated in the previous section. The first case occurs only for the algorithm SWA, i.e., this algorithm has an advantage over the counterpart algorithms. The advantage is that SWA allows a database owner to set a specific disclosure threshold for each sensitive rule. In our previous examples, we set the disclosure thresholds of all the sensitive rules with a unique value ($\psi = 0\%$). This specific disclosure threshold works as a weight. In many cases, some rules are more important than others. Thus, giving different disclosure thresholds to different rules is reasonable and may reflect real-world needs. For instance, let us consider the set of sensitive rules in scenario S3. Now we set the window size of SWA to 100,000 transactions (K = 100,000) and give different disclosure thresholds for each set of 6 rules in the four datasets, as follows: {[rule 1, 30%], [rule 2, 25%], [rule 3, 15%], [rule 4, 45%], [rule 5, 15%], and [rule 6, 20%]}, where for each ordered pair [rule i, ψ_i], rule i represents a sensitive rule in each dataset, and ψ_i is the corresponding disclosure threshold. In this example, we obtained the following results for misses cost, hiding failure, and dif(D, D') as shown in Table 7.7.

Metric	Kosarak	Retail	Reuters	BMS-1
MC	37.22	31.07	46.48	8.68
HF	5.57	7.45	0.01	21.84
Dif(D, D')	1.68	1.24	0.63	0.70

Table 7.7: An example of different thresholds for the sensitive rules in scenario S3.

The second special case of data sanitization occurs when data owners slide the disclosure threshold ($\psi > 0$) to allow miners to find a balance between knowledge discovery and privacy. This scenario is reasonable because here we are not disclosing personal information but special association rules that are strategic in decision making. Therefore, making a trade-off between privacy and data for mining can be done as long as an application permits it.

While the algorithm Algo2a hides rules by reducing their absolute support below a privacy threshold controlled by the database owner, our proposed algorithms hide rules based on a disclosure threshold ψ. Table 7.8 shows the effect of ψ on misses cost and hiding failure for the set of sensitive rules (scenario S3) in the Kosarak dataset. We varied ψ from 0 to 25%. The results for the other datasets can be found in Appendix B.4.4. Since Algo2a does not allow the input of a disclosure threshold, it is not compared with our algorithms.

Algorithm	$\psi = 0\%$		$\psi = 5\%$		$\psi = 10\%$		$\psi = 15\%$		$\psi = 25\%$	
	MC	HF	MC	HF	MC	HF	MC	HF	MC	HF
IGA	62.11	0.00	61.85	0.00	61.66	0.08	61.38	0.08	60.33	0.24
R. Robin	74.42	0.00	73.42	0.00	72.32	0.00	70.94	0.00	67,70	0.12
Random	74.37	0.00	73.32	0.00	72.36	0.00	70.87	0.00	67.73	0.00
SWA	72.70	0.00	67.03	0.00	59.56	0.75	53.06	3.81	39.87	17.83

Table 7.8: Effect of ψ on misses cost (MC) and hiding failure (HF).

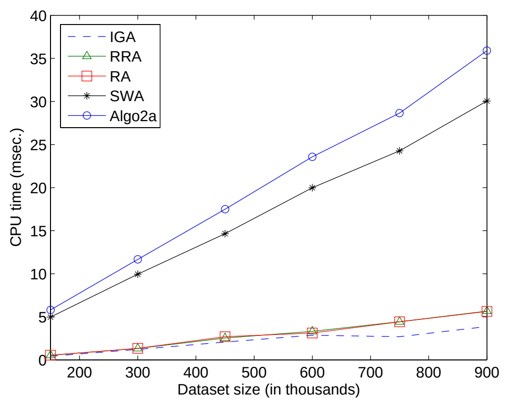
An important observation drawn from our special cases of data sanitization is that the values of misses cost can be improved. In the case of the algorithm SWA, having different disclosure thresholds reduces the values of misses cost. Similarly, sliding the disclosure threshold ψ improves the values of misses cost. On the other hand, the values of hiding failure increase since misses cost and hiding failure are typically contradictory measures, i.e., improving one usually incurs a cost in the other.

7.2.6 CPU Time for the Sanitization Process

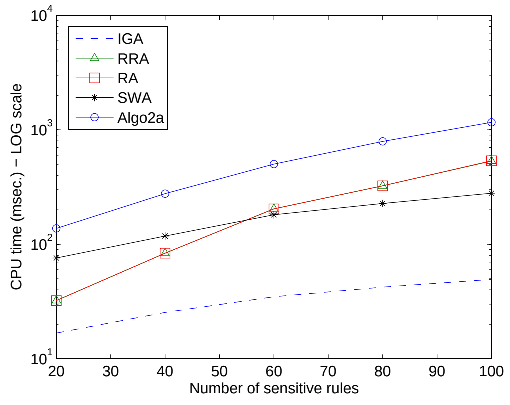
We tested the scalability of the sanitization algorithms vis-à-vis the size of the database as well as the number of rules to hide. To do so, we selected the Kosarak dataset since it is the largest one used in our experiments. Our comparison study also includes the algorithm Algo2a.

We varied the size of the original database D from 150K transactions to 900K transactions, while fixing the disclosure threshold $\psi = 0\%$ and keeping the set of sensitive rules constant (6 original sensitive rules that are mutually exclusive). Figure 7.1(a) shows that our algorithms scale well with the database size. The algorithms IGA, RRA and RA yielded lower CPU time than that for SWA and Algo2a. In particular, Algo2a requires six scans over the original database (one to hide each sensitive rule), while the algorithms IGA, RRA and RA require only two.

Although the algorithm SWA requires only one scan, it performs many operations in memory (e.g., sorting transactions in ascending order of size for each window), which demands more CPU time as the dataset increases. Even though IGA, RRA, and RA require two scans, they are faster than SWA. The main reason is that these algorithms perform a sort in memory only once.



(a)



(b)

Figure 7.1: Results of CPU time for the sanitization process.

As can be observed, the algorithms IGA, RRA, and RA increase CPU linearly, even though their complexity in main memory is not linear. If we increase the number of sensitive rules or even if we select a group of sensitive rules with very high support, these algorithms may not scale linearly. However, there is no compelling need for sanitization to be a fast operation since it can be done offline.

The I/O time (scans over the dataset) is also considered in these figures. This demonstrates good scalability with the cardinality of the transactional database.

We also varied the number of sensitive rules to hide from approximately 20 to 100 selected randomly, while fixing the size of the dataset Kosarak and fixing the support and disclosure thresholds to $\psi = 0\%$. Figure 7.1(b) shows that our algorithms scale well with the number of rules to hide. The values are plotted in logarithmic scale because the algorithm Algo2a requires one scan for each rule to hide.

Although IGA requires 2 scans, it was faster than SWA in all the cases. The main reason is that the SWA performs a number of operations in main memory to fully sanitize a database. The IGA requires one scan to build an inverted index where the vocabulary contains the sensitive rules and the occurrences contain the transaction IDs. In the second scan, IGA sanitizes only the transactions marked in the occurrences. Another interesting result observed was that over 40 rules, the SWA performed better than the algorithms RRA and RA. The reason is that the heuristic behind the SWA is optimized especially when there are rules with the intersection of items. Note that when the number of sensitive rules increases, the intersection of items among the rules tends to increase as well. In this case, the SWA touches fewer transactions than RRA and RA. As a result, SWA improves the performance as the number of rules to hide increases since the number of sorts in memory is the same (one by window size) for the dataset.

We should point out that the scalability of our sanitizing algorithms is mainly due to the inverted files we use in our approaches for indexing the sensitive transaction IDs per sensitive rule. There is no need to scan the database again whenever we want to access a transaction for sanitization purposes. The inverted file gives direct access with pointers to the relevant transactions. On the other hand, the CPU time for Algo2a is more expensive due to the number of scans over the database.

7.2.7 Discussion on the Data Sharing-Based Algorithms

We have evaluated our data sharing-based algorithms by performing a broad set of experiments using real datasets. This evaluation was carried out to suggest guidance on which algorithms perform best under different conditions.

Our experiments demonstrated that sanitization is not a trivial task. It can render the released database almost useless when not done properly. For this reason, we investigated different conditions under which a data owner can tune the parameters of the sanitizing algorithms to get the most out of the sanitization process.

We have learned several lessons from the experiments with our data sharing-based algorithms. In particular,

- Large datasets are our friends: our results typically show that the best results of misses cost and hiding failure can be obtained as the dataset increases. The Kosarak dataset is a typical example.

- Our algorithms scale well: in the worst case, we scan a transactional dataset twice, one to build the indexes and the other to sanitize the dataset. The SWA algorithm requires only one scan.

- The algorithm IGA performs very well: our experiments have demonstrated its outstanding performance. In almost all the cases, IGA yielded the best results in terms of misses cost and hiding failure. Exceptions occur in scenarios in which sensitive rules contain items with very high support. In this particular case, the algorithms SWA, RA, and RRA may present better results for misses cost.

- The data sanitization paradox is real: minimizing the impact on the sanitized datasets does not guarantee the best results in terms of misses cost. We showed that even though the SWA has yielded the best results for the differential between the original and the sanitized datasets, it has not achieved the best results for misses cost.

7.3 Evaluation of the Pattern Sharing-Based Algorithms

After evaluating our data sharing-based algorithms, we now move on to evaluate our pattern sharing-based algorithm, called the Downright Sanitizing Algorithm (DSA), which is based on our Heuristic 3 (Section 5.4.1).

7.3.1 Sanitizing Algorithms

Recall that our sanitizing algorithms are classified into two major groups: *Data-Sharing approach* and *Pattern-Sharing approach*. In the former, the sanitization acts on the data to hide the group of sensitive association rules that contain sensitive knowledge. In the latter, the sanitizing algorithm acts on the rules mined from a database instead of the data itself. In this section, we focus on our algorithm DSA, which relies on the latter category. The DSA removes sensitive rules before the pattern-sharing process. This sanitization process blocks possibilities of inferring sensitive rules that we call inference channels.

To our best knowledge, there are no known pattern sharing-based algorithms for rule sanitization in the literature. However, data sharing-based algorithms can be used for this purpose. Indeed, in order to sanitize a set of sensitive rules S_R (before sharing the patterns), one could use data sanitization to transform a database D into D' and then mine D' to get the patterns to share. We used this idea to compare our algorithm to existing data sanitization approaches. In particular, we compare our algorithm DSA with IGA since the latter has yielded the best results for data sanitization, as we reported in the previous section.

7.3.2 Methodology

We performed two series of experiments: the first to evaluate the effectiveness of DSA, and the second to measure its efficiency and scalability.

We considered the same datasets used in the performance evaluation for our data sharing-based algorithms. In addition, we used the same sensitive rules selected for the validations of our data sharing-based algorithms. Recall that such sensitive rules were selected based on four different scenarios (S1-S4).

Our comparison study was carried out through the following steps:

Steps for IGA :

- *Step 1*: We used the algorithm IGA to sanitize the sets of sensitive rules in the four initial datasets.
- *Step 2*: We applied an association rule mining algorithm on the sanitized datasets to extract the rules/patterns to share.

Steps for DSA :

- *Step 1*: We applied an association rule mining algorithm to extract rules from the four initial datasets.

- *Step 2*: We used DSA to sanitize these rules before sharing the rules/patterns.

The goal of our experiments here is to answer the same question raised in the previous section: under which conditions can one use IGA or DSA to protect sensitive knowledge mined from transactional databases?

7.3.3 Measuring Effectiveness

The effectiveness is measured in terms of sensitive associations rules that can be recovered by an adversary, as well as the proportion of non-sensitive rules hidden inadvertently due to the sanitization.

In order to compare the sanitizing algorithms IGA and DSA under the same conditions, we set the disclosure thresholds ψ of the algorithm IGA to 0%. In this case, all sensitive rules are completely sanitized. We purposely set these thresholds to zero because DSA always sanitizes all the sensitive rules.

Table 7.9 provides a summary of the best sanitizing algorithms in terms of misses cost when fixing the number of sensitive rules to be sanitized (6 rules). Further information on the values of misses cost can be found in Appendix B.3.1. The algorithm DSA yielded the best results in almost all the cases. The exceptions are the scenarios S1 (the dataset Kosarak) and S4 (the datasets Retail and Reuters) in which the values of misses cost for IGA are slightly better than those in DSA. In particular, we observed that IGA yielded the best results only when the sensitive rules had items with low support. This is the typical case in Scenario S4. The same case occurred in Scenario S1 for the dataset Kosarak in which the rules selected were composed of items with low support. On the other hand, we can note that in Scenarios S2 (rules selected randomly) and S3 (rules with high support items), the algorithm DSA yielded the best results in all the cases, as expected.

Dataset	$\psi = 0\%$, 6 sensitive rules			
	S1	S2	S3	S4
Kosarak	IGA	DSA	DSA	DSA
Retail	DSA	DSA	DSA	IGA
Reuters	DSA	DSA	DSA	IGA
BMS-1	DSA	DSA	DSA	DSA

Table 7.9: Summary of the best algorithms in terms of misses.

We also observed the same behaviour of the sanitizing algorithms IGA and DSA in terms of misses cost when varying the number of rules to be hidden. Table 7.10 shows a summary of the effect of misses cost on the four datasets. The results confirm the same finding we reported previously for the algorithms IGA and DSA. The only exceptions are the scenarios S1, S3, and S4 for the datasets Reuters, Retail and Kosarak, respectively. In these cases, IGA yielded the best results in terms of misses cost up to 3 sensitive rules being sanitized.

As the number of sensitive rules increases, DSA yielded the best values in terms of misses cost.

Dataset	$\psi = 0\%$, varying the no. of rules			
	S1	S2	S3	S4
Kosarak	IGA	DSA	DSA	IGA/DSA
Retail	DSA	DSA	IGA/DSA	IGA
Reuters	IGA/DSA	DSA	DSA	IGA
BMS-1	DSA	DSA	DSA	DSA

Table 7.10: Summary of IGA and DSA in terms of misses cost varying the number of rules.

We did not compare the algorithms IGA and DSA when varying the minimum support threshold σ because DSA does not work on this condition. Once a dataset is sanitized by DSA and the remain rules/patterns are shared for mining, a data analyst can no longer change the parameter σ to mine new rules/patterns. In addition, we did not compare the algorithms IGA and DSA in terms of dif(D, D') because this metric does not apply to DSA.

After comparing the algorithms IGA and DSA in terms of misses cost, we compared them in terms of the side effect factor. Table 7.11 summarizes the results we observed in terms of the side effect factor. The details concerning the values of side effect factor for IGA and DSA can be found in Appendix B.6.

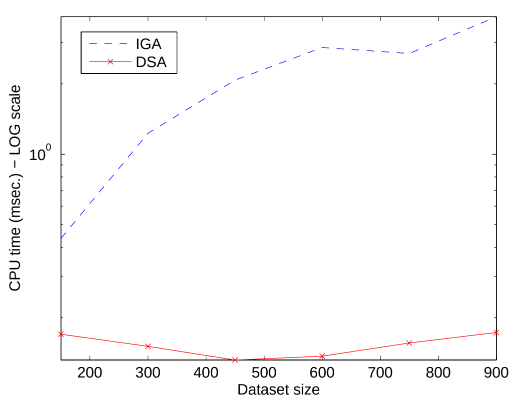
Note that the values in Table 7.11 are exactly the same as those in Table 7.9. These results were expected since misses cost and side effect factor are very similar measures.

Dataset	$\psi = 0\%$, 6 sensitive rules			
	S1	S2	S3	S4
Kosarak	IGA	DSA	DSA	DSA
Retail	DSA	DSA	DSA	IGA
Reuters	DSA	DSA	DSA	IGA
BMS-1	DSA	DSA	DSA	DSA

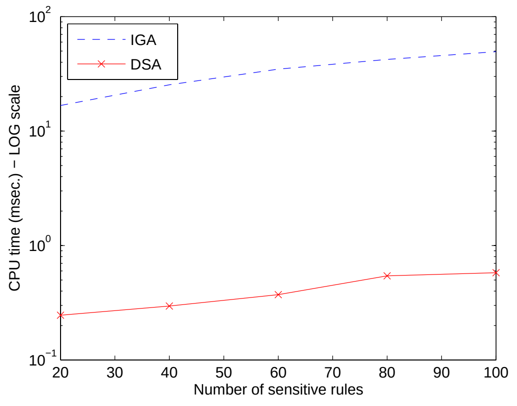
Table 7.11: Summary of the best algorithms in terms of side effect factor.

After identifying the side effect factor, we evaluated the recovery factor for DSA. This measure is not applied to IGA since this algorithm relies on data sanitization instead of pattern sanitization. Thus, once the data are shared for mining, there is no restriction about the rules discovered from a sanitized database.

In the case of pattern sanitization, some inference channels can occur, as discussed in Section 5.4.2. We ran a checklist procedure to evaluate the effectiveness of the sanitization performed by DSA. We then checked for the existence of any subset of the sensitive rules removed in order to identify the recovery factor. If all subsets of a rule were found, we assumed the rule could be recovered. As expected, DSA blocked both forward-inference and the backward-inference attacks. The results suggested that an adversary is highly unlikely



(a)



(b)

Figure 7.2: Results of CPU time for the sanitization process.

to be able to reconstruct the sensitive rules after the sanitization performed by DSA.

7.3.4 CPU Time for the Sanitization Process

We tested the scalability of the sanitization algorithms vis-à-vis the size of the database as well as the number of rules to hide. Again, we used the Kosarak dataset since it is the largest one used in our experiments.

We varied the size of the original database D from 150K transactions to 900K transactions (150K, 300K, 450K, 600K, 750K, and 900K), while fixing the disclosure threshold $\psi = 0\%$ for IGA and keeping the set of sensitive rules constant (6 original sensitive rules that were mutually exclusive). The transactions in the six sub-datasets were selected randomly from the Kosarak dataset. Figure 7.2(a) shows that our algorithms scale well with the database size. In particular, the CPU time for the DSA decreases significantly as the size of the datasets increased. Note that the CPU time for the DSA strongly changed. The main reason is that the number of rules in these datasets did not increase linearly for the same value of ψ. For instance, the dataset with 150k had many more rules than the dataset with 600K, resulting in this unexpected behaviour of the DSA. In contrast, the CPU time for the IGA increased linearly, as can be seen in Figure 7.2(a). Note that the IGA sanitizes transactions which increase linearly in our example, while the DSA sanitizes rules generated from the sub-datasets.

We also varied the number of sensitive rules to hide from approximately 20 to 100 selected randomly, while fixing the size of the dataset Kosarak and fixing the support and disclosure thresholds to $\psi = 0\%$. Figure 7.2(b) shows that our algorithms scale well with the number of rules to hide. The values are plotted in logarithmic scale because of the significant difference between the CPU time for both algorithms. The absolute values of CPU time for the algorithms IGA and DSA can be found in Appendix B.5.

The I/O time (scans over the dataset) is also considered in these figures. This demonstrates the good scalability of both algorithms with the cardinality of the transactional database and of the number of sensitive rules to be sanitized.

7.3.5 Discussion on the Pattern Sharing-Based Algorithms

We have evaluated our data pattern-based algorithms by performing a broad set of experiments using real datasets. Our experiments demonstrated the evidence of attacks (inference channels) in sanitized databases. The figures revealed that DSA is a promising solution to protect sensitive knowledge before sharing association rules.

DSA has a low value for side effect factor (and misses cost) and a very low recovery factor. We have identified some advantages of DSA over the previous data sharing-based sanitizing algorithms in the literature as follows:

- Using DSA, a database owner would share patterns (results) instead of the data itself.
- By sanitizing rules, one drastically reduces the possibility of inference channels since the support threshold and the mining algorithm are previously selected by the database owner.
- Sanitizing rules instead of data results in no alteration in the support and confidence of the non-sensitive rules, i.e., the released rules have the original support and confidence. As a result, the released rules seem more interesting for practical applications. Note that the other approaches reduce the support and the confidence of the rules as a side effect of the sanitization process.

On the other hand, DSA reduces the flexibility of information sharing since each time a third party wants to try a different set of support and confidence levels, it has to request the rules/patterns from the data owner.

7.4 Evaluation of the Dimensionality Reduction-Based Transformation

We evaluate the dimensionality reduction-based transformation (DRBT) in this section. This data transformation method was proposed to address PPC. Our goal here is to show that by using DRBT, a data owner can achieve the dual-goal: protecting the underlying data values subjected to clustering and obtaining acceptable clustering results.

We do not evaluate the OSBR because this solution requires a simple computation of a dissimilarity matrix and the suppression of identifiers before the release of data for clustering. Most importantly, in Section 6.4 we showed analytically the security and accuracy of the OSBR. In addition, we analyzed the space requirements and communication costs for the

OSBR. We concluded that section describing the advantages and disadvantages of using the OSBR for PPC. In particular, we showed that the OSBR is feasible to address PPC over centralized data only. In the context of PPC over vertically centralized data, the OSBR presents some limitations in terms of communication costs and security.

7.4.1 Methodology

We performed two series of experiments to evaluate the effectiveness of DRBT when addressing PPC over centralized data and PPC over vertically partitioned data. Our evaluation approach focused on the overall quality of generated clusters after dimensionality reduction. One question that we wanted to answer was:

> *What is the quality of the clustering results mined from the transformed data when the data are both sparse and dense?*

Our performance evaluation was carried out through the following steps:

- *Step 1*: we selected the datasets Chess, Mushroom, Pumsb, Connect, and Accidents for our performance evaluation. These datasets are described in Section 7.1.
- *Step 2*: we normalized the attribute values of the five real datasets mentioned in the previous step. To do so, we used the z-score normalization given in Equation (2.3). The results presented in the next sections were obtained after normalization.
- *Step 3*: we considered random projection based on two different approaches. First, the traditional way to compute random projection, by setting each entry of the random matrix R_1 to a value drawn from an i.i.d. $N(0,1)$ distribution and then normalizing the columns to unit length. Second, we used the random matrix R_2 where each element r_{ij} is computed using Equation (2.12). We refer to the former random projection as RP_1 and the latter as RP_2. We repeated each experiment (for random projection) 5 times. In the next section, we present results by showing only the average value.
- *Step 4*: we computed the relative error that the distances in *d-k* space suffer from, on the average, by using the stress function given in Equation (2.9). The stress function was computed for each dataset.
- *Step 5*: we selected K-means to find the clusters in our performance evaluation. Our selection was influenced by the following aspects: (a) K-means is one of the most known clustering algorithm and is scalable; (b) When using random projection, a perfect reproduction of the Euclidean distances may not be the best possible result. However, the rank order of the distances between the vectors is meaningful. Thus, when running K-means over the transformed data, one can find the clusters that would be mined from the original datasets with a reasonable accuracy.

- *Step 6*: we compared how closely each cluster in the transformed dataset matches its corresponding cluster in the original dataset. We expressed the quality of the generated clusters by computing the F-measure given in Equation (6.6). Considering that K-means is not deterministic (due to its use of random seed selection), we repeated each experiment 10 times. We then computed the minimum, average, maximum, and standard deviation for each measured value of F-measure.

We should point out that the steps described above were performed to evaluate the effectiveness of the DRBT when addressing PPC over centralized and vertically partitioned data.

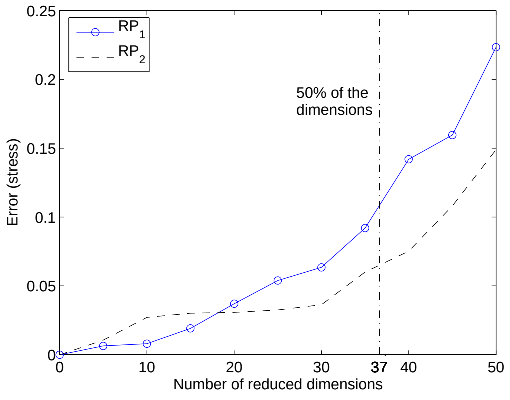
7.4.2 Measuring the Effectiveness of the DRBT over Centralized Data

To measure the effectiveness of DRBT in PPC over centralized data, we started by computing the relative error that the distances in d-k space suffer from, on the average. To do so, we used the two random projection approaches (RP_1 and RP_2) mentioned in Step 3 of Section 7.4.1.

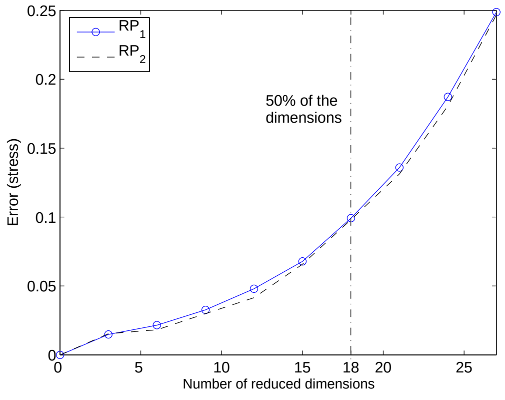
A word of notation: hereafter we denote the original dimension of a dataset as d_o and reduced dimension of the transformed dataset as d_r. This notation is to avoid confusion between the reduced dimension of a dataset (k) and the number of clusters used as input of the algorithm K-means.

An important feature of the DRBT is its versatility to trade privacy, accuracy, and communication cost. The privacy preservation is assured because random projection is a non-invertible transformation, as discussed in Section 6.5.4. We here study the trade-off between accuracy and communication cost. The accuracy is represented by the error that the distances in d_o-d_r space suffer from, while the communication cost is represented by the number of dimensions that we reduce in the datasets. We selected two datasets: Pumsb and Chess with 74 and 37 dimensions, respectively. We reduced the dimensions of these datasets and computed the error. Figure 7.3(a) shows the error produced by RP_1 and RP_2 on the dataset Pumsb and Figure 7.3(b) shows the error produced by RP_1 and RP_2 on the dataset Chess. These results represent the average value of five trials. The error produced by RP_1 and RP_2 on the other datasets are available at Appendix C.1.

We observed that, in general, RP_2 yielded the best results in terms of the error produced on the datasets (the lower the better). In the dataset Chess the difference between RP_2 and RP_1 was not significant. These results confirm the same findings in [16] and backup the theory of random projection (the choice of the random matrix) proposed in [1]. We noticed from the figures that the DRBT trades well accuracy (error) and communication cost (number of reduced dimensions) when the data are reduced up to 50% of the dimensions.



(a)



(b)

Figure 7.3: (a) The error produced on the dataset *Pumsb* ($d_o = 74$); (b) The error produced on the dataset *Chess* ($d_o = 37$).

In this case, the trade-off between the error and the communication cost is linear. However, reducing more than 50% of the dimensions, the communication cost is improved but the accuracy is compromised since the error produced on the datasets grows faster. Therefore, a data owner should consider carefully this trade-off before releasing some data for clustering.

After evaluating the error produced on the datasets, we used the algorithm K-means to find the clusters in the original and transformed datasets. We varied the number of clusters from 2 to 5 in the five datasets. Subsequently, we compared how closely each cluster in the transformed dataset matches its corresponding cluster in the original dataset by computing F-measure given in Equation (6.6).

Table 7.12 shows the results of F-measure for the Accidents dataset. We reduced the original 18 dimensions to 12. We repeated each experiment 10 times and computed the minimum, average, maximum, and standard deviation for each measured value of F-measure. We simplify the results by showing only one dataset (Accidents). The values of F-measure for the other datasets can be found in Appendix C.2. Note that we computed the values of F-measure only for the random projection RP_2 since its results were slightly better than those yielded by RP_1.

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.931	0.952	0.941	0.014	0.903	0.921	0.912	0.009

Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.870	0.891	0.881	0.010	0.878	0.898	0.885	0.006

Table 7.12: Average of F-measure (10 trials) for the Accidents dataset ($d_o = 18, d_r = 12$).

We noticed that the values of F-measure for the Chess and Connect datasets (see Appendix C.2) were relatively low when compared with the results of F-measure for the other datasets. The main reason is that the data points in these datasets are densely distributed. Thus, applying a partitioning clustering algorithm (e.g., K-means) to datasets of this nature increases the number of misclassified data points. On the other hand, when the attribute values of the objects are sparsely distributed, the clustering results are much better. Consider, for example, the Iris dataset available at the UCI Repository of Machine Learning Databases [17]. Iris is perhaps the best known database to be found in the pattern recognition literature. This dataset has two clusters well defined and the data are sparsely distributed. We reduced the original 5 dimensions to 3. Then we applied random projection RP_2 to the Iris dataset and computed the minimum, average, maximum, and standard deviation for each measured value of F-measure. We repeated each experiment 10 times. Table 7.13 shows that the standard deviation for two clusters ($k = 2$) was zero and the average of F-measure was one.

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	1.000	1.000	1.000	0.000	0.094	0.096	0.948	0.010

Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.773	0.973	0.858	0.089	0.711	0.960	0.833	0.072

Table 7.13: Average of F-measure (10 trials) for the Iris dataset ($d_o = 5, d_r = 3$).

7.4.3 Measuring the Effectiveness of the DRBT over Vertically Partitioned Data

Now we move on to measure the effectiveness of DRBT to address PPC over vertically partitioned data. To do so, we split the Pumsb dataset (74 dimensions) from 1 up to 4 parties (partitions) and fixed the number of dimensions to be reduced (38 dimensions). Table 7.14 shows the number of parties, the number of attributes per party, and the number of attributes in the merged dataset which is subjected to clustering. Recall that in a vertically partitioned data approach, one of the parties will centralize the data before mining.

No. of parties	No. of attributes per party	No. of attributes in the merged dataset
1	1 partition with 74 attributes	38
2	2 partitions with 37 attributes	38
3	2 partitions with 25 and 1 with 24 attributes	38
4	2 partitions with 18 and 2 with 19 attributes	38

Table 7.14: An example of partitioning for the Pumsb dataset.

In this example, each partition with 37, 25, 24, 19, and 18 attributes was reduced to 19,

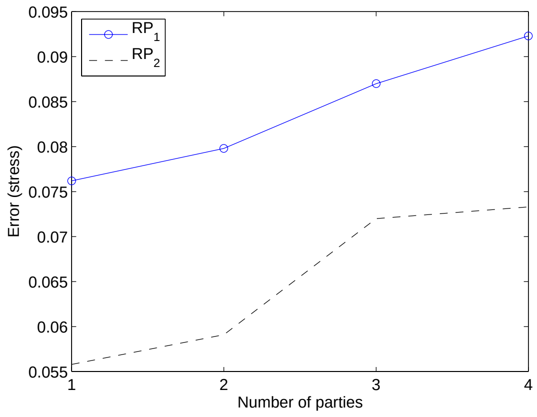


Figure 7.4: The error produced on the dataset *Pumsb* over vertically partitioned data.

13, 12, 10, and 9 attributes, respectively. We applied the random projections RP_1 and RP_2 to each partition and then merged the partitions in one central repository. Subsequently, we computed the stress error on the merged dataset and compared the error with that one produced on the original dataset (without partitioning). Figure 7.4 shows the error produced on the Pumsb dataset in the vertically partitioned data approach. As we can see, the results yielded by RP_2 were again slightly better than those yielded by RP_1.

Note that we reduced approximately 50% of the dimensions of the dataset Pumsb and the trade-off between accuracy and communication cost is still efficient for PPC over vertically partitioned data.

We also evaluated the quality of clusters generated by mining the merged dataset and comparing the clustering results with those mined from the original dataset. To do so, we computed F-measure for the merged dataset in each scenario, i.e., from 1 up to 4 parties. We varied the number of clusters from 2 to 5. Table 7.15 shows values of F-measure (average and standard deviation) for the Pumsb dataset over vertically partitioned data. These values represent the average of 10 trials considering the random projection RP_2.

No. of parties	k = 2		k = 3		k = 4		k = 5	
	Avg	Std	Avg	Std	Avg	Std	Avg	Std
1	0.909	0.140	0.965	0.081	0.891	0.028	0.838	0.041
2	0.904	0.117	0.931	0.101	0.894	0.059	0.840	0.047
3	0.874	0.168	0.887	0.095	0.873	0.081	0.801	0.073
4	0.802	0.155	0.812	0.117	0.866	0.088	0.831	0.078

Table 7.15: Average of F-measure (10 trials) for the Pumsb dataset over vertically partitioned data.

We notice from Table 7.15 that the results of F-measure slightly decrease when we increase the number of parties in the scenario of PPC over vertically partitioned data. Despite this fact, the DRBT is still effective to address PPC over vertically partitioned data in preserving the quality of the clustering results as measured by F-measure.

7.4.4 Discussion on the DRBT When Addressing PPC

The evaluation of the DRBT involves three important issues: security, communication cost, and quality of the clustering results. We discussed the issues of security in Section 6.5.4, based on Lemma 4, and the issues of communication cost and space requirements in Section 6.5.6. In this Section, we have focused on the quality of the clustering results.

We have evaluated our proposed data transformation method (DRBT) to address PPC. We have learned some lessons from this evaluation, as follows:

- *The application domain of the DRBT*: we observed that the DRBT does not present acceptable clustering results in terms of accuracy when the data subjected to clustering are dense. Slightly changing the distances between data points by random projection results in misclassification, i.e., points will migrate from one cluster to another in the transformed dataset. This problem is somehow understandable since partitioning clustering methods are not effective to find clusters in dense data. The Connect dataset is one example which confirms this finding. On the other hand, our experiments demonstrated that the quality of the clustering results obtained from sparse data is promising.

- *The versatility of the DRBT*: using the DRBT, a data owner can tune the number of dimensions to be reduced in a dataset trading privacy, accuracy, and communication costs before sharing the dataset for clustering. Most importantly, the DRBT can be used to address PPC over centralized and vertically partitioned data.

- *The choice of the random matrix*: from the performance evaluation of the DRBT we noticed that the random projection RP_2 yielded the best results for the error produced on the datasets and the values of F-measure, in general. The random projection RP_2 is based on the random matrix proposed in Equation (2.12).

7.5 Summary

In this chapter, we have empirically evaluated our proposed privacy-preserving data transformation (PPDT) methods for privacy-preserving association rule mining (PPARM) and privacy-preserving clustering (PPC).

In the context of PPARM, we noticed that sanitization is a challenging task. It can render a released database almost useless when not done properly. To alleviate the difficulties of sanitizing a database, we investigated different conditions under which a data owner can tune the parameters of our sanitizing algorithms to get the most of the sanitization process.

We evaluated our data sharing-based and pattern sharing-based algorithms by performing a broad set of experiments against real datasets. This evaluation was carried out to suggest guidance on which algorithms perform best under different conditions. In particular, we observed that the algorithm IGA (Item Grouping Algorithm) presented an outstanding performance in our experiments. In almost all the cases, IGA yielded the best results in terms of misses cost and hiding failure. Another interesting point observed from our experiments was the fact that the best results of misses cost and hiding failure were obtained as we increased the size of the datasets.

Regarding our proposed pattern sharing-based algorithm (DSA), we noticed that this algorithm has a low value for side effect factor (and misses cost) and a very low recovery factor. Our experiments demonstrated the evidence of attacks (inference channels) in database sanitized by DSA. We discussed the advantages and disadvantages of DSA and showed in which scenarios a data owner could use this algorithm.

In the context of PPC, we did not evaluate the Object Similarity-Based Representation (OSBR) because this solution requires a simple computation of a dissimilarity matrix and the suppression of identifiers before the release of data for clustering. Rather, we analytically showed in Section 6.4 the aspects of security and accuracy of the OSBR and analyzed the space requirements and communication costs required by the OSBR. The key finding of our investigation was that the OSBR is feasible to address PPC over centralized data only since the OSBR presents some limitations in terms of communication costs and security when addressing PPC over vertically partitioned data.

On the other hand, we dedicated special attention to evaluate the dimensionality reduction-based transformation (DRBT). We empirically evaluated this data transformation method to address PPC. We showed that using the DRBT a data owner can not only protect the underlying data values subjected to clustering but also obtain valid clustering results.

The evaluation of the DRBT was based on three important issues: security, communication cost, and quality of the clustering results. We discussed the issues of security in Section 6.5.4 and the issues of communication cost and space requirements were discussed in Section 6.5.6. We then dedicated this section to study the effectiveness of the DRBT in terms of the quality of the clustering results. From the performance evaluation, we suggested guidance on which scenario a data owner can achieve the best quality of the clustering when using the DRBT. We also suggested guidance on the choice of the random matrix to obtain the best results in terms of the error produced on the datasets and the values of F-measure.

The highlights of the DRBT are as follows: a) it is independent of distance-based clustering algorithms; b) it has a sound mathematical foundation; c) it does not require CPU-intensive operations; and d) it can be applied to address PPC over centralized data and PPC over vertically partitioned data.

Chapter 8

Conclusions and Future Work

There are two kinds of people,
those who finish what they start and so on.
– Robert Byrne

The end is the crown of any work.
– Russian Proverb

8.1 Summary

Privacy-preserving data mining (PPDM) is one of the newest trends in privacy and security research. It is driven by one of the major policy issues of the information era - *the right to privacy.* Although this research field is very new, we have already seen great interests in it: a) the recent proliferation in PPDM techniques is evident; b) the interest from academia and industry has grown quickly; and c) separate workshops and conferences devoted to this topic have emerged in the last few years.

Privacy issues have posed new challenges for novel uses of data mining technology. These technical challenges cannot simply be addressed by restricting data collection or even by restricting the secondary use of information technology. An approximate solution could be sufficient, depending on the application since the appropriate level of privacy can be interpreted in different contexts. In some applications (e.g., association rules, classification, or clustering), an appropriate balance between a need for privacy and knowledge discovery should be found.

In this thesis, we addressed the problem of transforming a database into a new one that conceals sensitive information while preserving the general patterns and trends from the original database. The sensitive information is not limited to personal data, but may reflect customers' purchasing behaviour, financial, medical, and insurance liability information and sensitive patterns, considered sensitive patterns for strategic or competitive reasons by the caretaker or owner of the data.

The transformation applied to the database occurs before the sharing of data for mining. We focused primarily on PPDM, notably in the context of the mining tasks: a) *association rules* which describe interesting relationships among items grouped together in a sufficient number of examples; and b) *clustering* which is concerned with grouping objects into classes of similar objects.

We investigated the feasibility of achieving PPDM by data transformation. Our thesis was that *privacy preservation in data mining, by data transformation, is to some extent possible*. We demonstrated empirically and theoretically the practicality and feasibility of achieving PPDM. In particular, we showed that a trade-off between privacy preservation and knowledge discovery can be accomplished when addressing privacy preservation in association rule mining and clustering.

Four major issues were addressed to support the central thesis statement of this research, as follows:

1. It is possible to transform a database by protecting the attribute values of objects subjected to clustering and get valid clustering results, i.e., the clusters generated in the transformed database are very similar to those mined from the original database.

2. It is possible to protect sensitive knowledge discovered from databases without losing the benefit of mining the transformed database.

3. It is possible to quantify the disclosure of sensitive knowledge discovered from a transformed database.

4. It is possible to measure the information loss in a transformed database available for association rule mining.

8.2 Contributions

We now summarize the main contributions accomplished in this research. These contributions are parts of a novel framework for privacy preservation in data mining:

1. **Toward foundations of PPDM**: We have laid down the foundations for further research in the area of PPDM in Chapter 3. In particular, we described the problems we face in defining what information is private in data mining, and discuss how privacy can be violated in data mining. We described the basis of PPDM including the historical roots, the definition of privacy preservation in data mining, and models of data miners in PPDM. We then analyzed the implications of standard privacy principles in knowledge discovery and suggested some policies for PPDM based on these privacy principles. Subsequently, we suggested some desirable privacy requirements that are

related to industrial initiatives. These requirements are essential for the development and deployment of technical solutions and will allow vendors and developers to make solid advances in the future of PPDM.

2. **A taxonomy of PPDM techniques**: We surveyed the existing PPDM techniques in the literature and proposed a taxonomy including such techniques, which is described in Chapter 4.

3. **A family of privacy-preserving methods**: Addressing privacy preservation in data mining requires different kinds of data transformation since the mining tasks are versatile. We proposed a family of privacy-preserving data transformation (PPDT) methods for protecting privacy before data are shared for association rule mining and clustering. These methods are described in Chapter 5 and Chapter 6, respectively. Although our framework has been designed to address privacy issues in association rules and clustering, it can be extended to encompass data transformation for privacy-preserving classification.

4. **A library of algorithms**: To attain privacy preservation in association rule mining and clustering, we proposed a library of algorithms. Such algorithms were designed taking into account heuristics for our PPDT methods presented in Chapter 5.

5. **Retrieval facilities**: These retrieval facilities were specifically designed for our sanitizing algorithms described in Chapter 5. These algorithms are applied to protect sensitive knowledge in association rule mining. As mentioned in Section 2.1.2, pattern discovery may require various scans over a transactional database. To speed the process of hiding sensitive knowledge in transactional databases, our framework is built on an index. As a consequence, our algorithms require only two scans to protect sensitive knowledge regardless the number of association rules to be hidden: one scan to build an inverted index, and the other scan to hide the sensitive rules. Other techniques in the literature require multiple scans [12, 37, 126].

6. **A set of metrics**: Since there is no exact solution to address privacy preservation in data mining, we need to be able to measure how much sensitive information is disclosed and verify the usefulness of the data after the transformation process. To evaluate our method for association rule mining, we proposed a set of metrics to measure not only how much sensitive knowledge has been disclosed, but also to measure the effectiveness of the proposed algorithms in terms of information loss and in terms of non-sensitive rules removed as a side effect of the transformation process. In the context of privacy-preserving clustering, we used standard measures to evaluate information loss (e.g.,

stress function) and the similarity between the clusters generated before and after dimensionality reduction (e.g., F-measure).

8.3 Future Research

Several directions can be exploited as a continuation of this research. We discuss a few technical challenges in this section and categorize them into two major groups: *Challenges left to explore* and *Future research trends*. In the former, we describe some issues and open questions left to explore in the context of our framework. In the latter, we point to some technical challenges as future research trends in PPDM.

8.3.1 Challenges Left to Explore

We describe some challenges and open questions left to explore in the domain of our framework, as follows:

- **Privacy definition**: The concept of privacy is often more complex than realized. In data mining, the definition of privacy preservation is still unclear and there is no consensus about this topic. There is very little literature related to this topic. A notable exception are the contributions presented in [29] and in [108]. Although our work described in [108] is preliminary and conceptual in nature, it is a vital prerequisite for the development and deployment of new techniques. Issues of privacy in data mining will certainly play a significant role in the future of this new area.

- **Combining sanitization and randomization**: The optimal sanitization is an NP-hard problem [12]. We have showed that sanitization is a challenging problem and it is sometimes restrictive. To reduce the side effect of the sanitization process one could combine sanitization and randomization under the same framework. On the one hand, randomization does not remove items from a dataset. On the other hand, randomization in general introduce false drops to the data, i.e., some patterns that are not supposed to exist in the original database. A hybrid approach balancing sanitization and randomization would be interesting to expand our framework with probabilistic analysis to supplement the empirical and theoretical results. The technical question raised by this approach is: how much can this hybrid approach improve the trade-off between privacy preservation and knowledge discovery? This problem deserves further exploration.

- **New method for PPC**: Is it possible to combine data perturbation methods or even the k-anonymity model with isometries to address PPC? The essential characteristic of an isometry is that distances between objects are preserved in the process of moving them in a d-dimensional Euclidean space (e.g., rotation, translation). The challenge here is due to the fact that, in general, isometries are invertible [35, 96]. Although one approach based on isometries maintains the distances between data points, such an approach cannot offer formal proofs of security. Regarding k-anonymity, this model ensures that a release provides k-anonymity protection if the information for each person contained in the release cannot be distinguished from at least $k-1$ individuals whose information also appears in the release [134]. One drawback of this approach is that a data owner would have to greatly reduce a released database to achieve k-anonymity. Regarding data perturbation methods, such methods modify the similarity between objects to preserve privacy. We believe that combining the strength of data perturbation, isometries, and k-anonymity is to some extent possible to address PPC trading privacy preservation and knowledge discovery. This investigation requires further examination.

8.3.2 Future Research Trends

Here we discuss some future research trends in PPDM, as follows:

- **Privacy preservation in spoken language databases**: Goal-oriented spoken dialog systems aim at identifying intents of humans expressed in natural language and taking actions accordingly [136]. In general, spoken dialog systems are trained using large amounts of task data which are usually transcribed and then labeled by humans. In the customer care domain, "labeling" means assigning one or more of the defined intents to each utterance. As an example, consider the utterance *I would like to pay my bill*, in a customer care application. Assuming that the utterance is recognized correctly, the corresponding intent would be *Pay(Bill)* and the action would be learning the caller's account number and credit card number. The transcribed and labeled data are then used to train automatic speech recognition. Clearly, the sharing and reusing of such data is extremely important for research and development in spoken language processing. However, the sharing of such data requires privacy safeguards since the data may include personal information such as social security numbers and credit card numbers, that must be sanitized and/or anonymized. In this scenario, the sanitization process should act on the text databases to hide personal information meeting some privacy requirements. On the other hand, the quality of the data must be preserved after the sanitization. How can one hide task-dependent named entities to preserve the privacy of speakers and maintain the text databases useful for training spoken dialog systems?

- **Sanitization of document repositories**: A more challenging scenario is to sanitize document repositories on the Web to protect private information against sophisticated tools to browse and analyze data. Preserving privacy on the Web has an important impact on many Web activities and Web applications. In particular, mining Web activities while preserving privacy is a complex problem. The ease access to information on the Web, coupled with the ready availability of personal data, also made it easier and more tempting for interested parties (e.g., businesses and governments) to willingly or inadvertently intrude on individuals' privacy in unprecedented ways. Undoubtedly, public data are very important in the context of machine learning, data mining, information retrieval, and natural language processing research. However, before the sharing of such data for research purposes, sanitization could be used to protect the privacy of individuals. One important example is the sharing of medical reports of patients which facilitates the medical research significantly [122]. The primary goal in this context is how to sanitize document repositories to protect private information that exists as a result of availability of document repositories.

- **PPDM on the Web**: Privacy issues on the Web have attracted a lot of attention due to the growth of e-commerce and e-business. These issues are further complicated by the global and self-regulatory nature of the Web.

  Clearly, privacy issues on Web data is an umbrella that encompasses many Web applications such as e-commerce, stream data mining, multimedia mining, among others. In particular, a common framework for PPDM should be conceived, notably in terms of definitions, principles, policies, and requirements. The advantages of a framework of that nature are as follows: (a) a common framework will avoid confusing developers, practitioners, and many others interested in PPDM on the Web; (b) adoption of a common framework will inhibit inconsistent efforts in different ways, and will enable vendors and developers to make solid advances in the future of research in PPDM on the Web.

  The success of a framework of this nature can only be guaranteed if it is backed up by a legal framework, such as the Platform for Privacy Preferences (P3P) Project [79, 115]. This project is emerging as an industry standard providing a simple, automated way for users to gain more control over the use of personal information on Web sites they visit.

  The European Union has taken a lead in setting up a regulatory framework for Internet Privacy and has issued a directive which sets guidelines for processing and transfer of personal data [47].

Bibliography

[1] D. Achlioptas. Database-Friendly Random Projections. In *Proc. of the 20th ACM Symposium on Principles of Database Systems*, pages 274–281, Santa Barbara, CA, USA, May 2001.

[2] M. Ackerman, L. Cranor, and J. Reagle. Privacy in E-Commerce: Examining User Scenarios and Privacy Preferences. In *Proc. of the ACM Conference on Electronic Commerce*, pages 1–8, Denver, Colorado, USA, November 1999.

[3] N. R. Adam and J. C. Worthmann. Security-Control Methods for Statistical Databases: A Comparative Study. *ACM Computing Surveys*, 21(4):515–556, December 1989.

[4] D. Agrawal and C. C. Aggarwal. On the Design and Quantification of Privacy Preserving Data Mining Algorithms. In *Proc. of ACM SIGMOD/PODS*, pages 247–255, Santa Barbara, CA, May 2001.

[5] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 94–105, Seattle, WA, 1998.

[6] R. Agrawal, T. Imielinski, and A. N. Swami. Mining Association Rules Between Sets of Items in Large Databases. In *Proc. of ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington, D.C., May 1993.

[7] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocratic Databases. In *Proc. of the 28th Conference on Very Large Data Bases*, Hong Kong, China, August 2002.

[8] R. Agrawal and R. Srikant. Fast Algorithms for Mining Association Rules. In *Proc. of the 20th International Conference on Very Large Data Bases*, pages 487–499, Santiago, Chile, Sep. 1994.

[9] R. Agrawal and R. Srikant. Privacy-Preserving Data Mining. In *Proc. of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 439–450, Dallas, Texas, May 2000.

[10] S. Agrawal, V. Krishnan, and J. R. Haritsa. On Addressing Efficiency Concerns in Privacy-Preserving Mining. In *Proc. of the 9th International Conference on Database Systems for Advanced Applications (DASFAA-2004)*, Jeju Island, Korea, March 2004.

[11] M. P. Armstrong, G. Rushton, and D. L. Zimmerman. Geographically Masking Health Data to Preserve Confidentiality. *Statistics in Medicine*, 18:497–525, 1999.

[12] M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim, and V. Verykios. Disclosure Limitation of Sensitive Rules. In *Proc. of IEEE Knowledge and Data Engineering Workshop*, pages 45–52, Chicago, Illinois, November 1999.

[13] J. W. Auer. *Linear Algebra With Applications*. Prentice-Hall Canada Inc., Scarborough, Ontario, Canada, 1991.

[14] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Limited, England, 1999.

[15] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation. In *Proc. of the 20th ACM Symposium on Theory of Computing*, pages 1–10, Chicago, Illinois, USA, 1988.

[16] E. Bingham and H. Mannila. Random Projection in Dimensionality Reduction: Applications to Image and Text Data. In *Proc. of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 245–250, San Francisco, CA, USA, 2001.

[17] C.L. Blake and C.J. Merz. UCI Repository of Machine Learning Databases, University of California, Irvine, Dept. of Information and Computer Sciences, 1998.

[18] L. Brankovic and V. Estivill-Castro. Privacy Issues in Knowledge Discovery and Data Mining. In *Proc. of Australian Institute of Computer Ethics Conference (AICEC99)*, Melbourne, Victoria, Australia, July 1999.

[19] T. Brijs, G. Swinnen, K. Vanhoof, and G. Wets. Using Association Rules for Product Assortment Decisions: A Case Study. In *Knowledge Discovery and Data Mining*, pages 254–260, 1999.

[20] S. Brin, R. Motwani, J. D. Ullman, and S. Tsur. Dynamic Itemset Counting and Implication Rules for Market Basket Data. In *Proc. of ACM SIGMOD International Conference on Management of Data*, pages 255–264, Tucson, Arizona, USA, May 1997.

[21] B. Brumen, I. Golob, T. Welzer, I. Rozman, M. Družovec, and H. Jaakkola. An Algorithm for Protecting Knowledge Discovery Data. *INFORMATICA*, 14(3):277–288, December 2003.

[22] T. S. Caetano. *Graphical Models and Point Set Matching.* PhD thesis, Federal University of Rio Grande do Sul, Porto Alegre, RS, Brazil, July 2004.

[23] S. Castano, M. Fugini, G. Martella, and P. Samarati. *Database Security.* Addison-Wesley Longman Limited, England, 1995.

[24] D. Chaum, C. Crépeau, and I. Damgard. Multiparty Unconditionally Secure Protocols. In *Proc. of the 20th ACM Symposium on Theory of Computing*, pages 11–19, Chicago, Illinois, USA, 1988.

[25] M.-S. Chen, J. Han, and P. S. Yu. Data Mining: An Overview from a Database Perspective. *IEEE Transactions on Knowledge and Data Engineering*, 8(6):866–883, 1996.

[26] Z. Chen. *Data Mining and Uncertain Reasoning.* John Wiley and Sons, Inc., New York, NY, 2001.

[27] C. Clifton. Using Sample Size to Limit Exposure to Data Mining. *Journal of Computer Security*, 8(4):281–307, November 2000.

[28] C. Clifton, W. Du, M. Atallah, M. Kantarcioǧlu, X. Lin, and J. Vaidya. Distributed Data Mining to Protect Information Privacy. Proposal to the National Science Foundation, December 2001.

[29] C. Clifton, M. Kantarcioǧlu, and J. Vaidya. Defining Privacy For Data Mining. In *Proc. of the National Science Foundation Workshop on Next Generation Data Mining*, pages 126–133, Baltimore, MD, USA, November 2002.

[30] C. Clifton, M. Kantarcioǧlu, J. Vaidya, X. Lin, and M. Y. Zhu. Tools For Privacy Preserving Distributed Data Mining. *SIGKDD Explorations*, 4(2):28–34, December 2002.

[31] C. Clifton and D. Marks. Security and Privacy Implications of Data Mining. In *Workshop on Data Mining and Knowledge Discovery*, pages 15–19, Montreal, Canada, February 1996.

[32] S. Cockcroft and P. Clutterbuck. Attitudes Towards Information Privacy. In *Proc. of the 12th Australasian Conference on Information Systems*, Coffs Harbour, NSW, Australia, December 2001.

[33] C. Collberg, C. Thomborson, and D. Low. A Taxonomy of Obfuscating Transformations. Technical report, TR–148, Department of Computer Science, University of Auckland, New Zealand, July 1997.

[34] T. H. Cormen, C. E. Leiserson, and R. L. Rivest. *Introduction to Algorithms.* The MIT Press: Cambridge, Massachusetts, USA, 1992.

[35] H. T. Croft, K. J. Falconer, and R. K. Guy. *Unsolved Problems in Geometry: v.2.* New York: Springer-Verlag, 1991.

[36] M. J. Culnan. How Did They Get My Name?: An Exploratory Investigation of Consumer Attitudes Toward Secondary Information. *MIS Quartely*, 17(3):341–363, September 1993.

[37] E. Dasseni, V. S. Verykios, A. K. Elmagarmid, and E. Bertino. Hiding Association Rules by Using Confidence and Support. In *Proc. of the 4th Information Hiding Workshop*, pages 369–383, Pittsburg, PA, April 2001.

[38] D. E. Denning and J. Schlörer. Inference Controls for Statistical Databases. *IEEE Computer*, 16(7):69–82, July 1983.

[39] V. Dhar and A. Tuzhilin. Abstract-Driven Pattern Discovery in Databases. *IEEE Transactions on Knowledge and Data Engineering*, 5(6):926–938, December 1993.

[40] W. Du and M. J. Atallah. Secure Multi-Party Computation Problems and their Applications: A Review and Open Problems. In *Proc. of 10th ACM/SIGSAC 2001 New Security Paradigms Workshop*, pages 13–22, Cloudcroft, New Mexico, September 2001.

[41] S. Džeroski. Data Mining in a Nutshell, Chapter 1 of "Relational Data Mining", S. Džeroski and N. Lavrač (eds.), Springer-Verlag, Germany, 2001, pages 3-27.

[42] S. Džeroski and N. Lavrač (editors). *Relational Data Mining.* Springer-Verlag, Germany, 2001.

[43] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of the 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, Portland, Oregon, 1996.

[44] V. Estivill-Castro. Why So Many Clustering Algorithms - A Position Paper. *SIGKDD Explorations*, 4(1):65–75, June 2002.

[45] V. Estivill-Castro and L. Brankovic. Data Swapping: Balancing Privacy Against Precision in Mining for Logic Rules. In *Proc. of Data Warehousing and Knowledge Discovery DaWaK-99*, pages 389–398, Florence, Italy, August 1999.

[46] V. Estivill-Castro, L. Brankovic, and D. L. Dowe. Privacy in Data Mining. *Privacy Law and Policy Reporter*, 6(3):33–35, September 1999.

[47] European Comission. The directive on the protection of individuals with regard of the processing of personal data and on the free movement of such data, 1998. Available at http://www2.echo.lu.

[48] A. Evfimievski. Randomization in Privacy Preserving Data Mining. *SIGKDD Explorations*, 4(2):43–48, December 2002.

[49] A. Evfimievski, J. Gehrke, and R. Srikant. Limiting Privacy Breaches in Privacy Preserving Data Mining. In *Proc. of the 22nd ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2003)*, San Diego, CA, USA, June 2003.

[50] A. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy Preserving Mining of Association Rules. In *Proc. of the 8th ACM SIGKDD Intl. Conf. on Knowlegde Discovery and Data Mining*, pages 217–228, Edmonton, AB, Canada, July 2002.

[51] C. Faloutsos and K.-I. Lin. FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets. In *Proc. of the 1995 ACM SIGMOD International Conference on Management of Data*, pages 163–174, San Jose, CA, USA, June 1995.

[52] C. Farkas and S. Jajodia. The Inference Problem: A Survey. *SIGKDD Explorations*, 4(2):6–11, December 2002.

[53] U. Fayyad. Knowledge Discovery in Databases: An Overview, Chapter 2 of "Relational Data Mining", S. Džeroski and N. Lavrač (eds.), Springer-Verlag, Germany, 2001, pages 28-47.

[54] U. M. Fayyad, S. G. Djorgovski, and N. Weir. Automating the Analysis and Cataloging of Sky Surveys. In *Advances in Knowledge Discovery and Data Mining. U. M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy (eds.)*, pages 471–493, MIT Press, Cambridge, MA, 1996.

[55] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From Data Mining to Knowledge Discovery: An Overview. In *Advances in Knowledge Discovery and Data Mining. U. M. Fayyad, G. Piatetsky-Shapiro, P. Smith, and R. Uthurusamy (eds.)*, pages 1–34, MIT Press, Cambridge, MA, 1996.

[56] A. P. Felty and S. Matwin:. Privacy-Oriented Data Mining by Proof Checking. In *Proc. of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD)*, pages 138–149, Helsinki, Finland, August 2002.

[57] X. Z. Fern and C. E. Brodley. Random Projection for High Dimensional Data Clustering: A Cluster Ensemble Approach. In *Proc. of the 20th International Conference on Machine Learning (ICML 2003)*, Washington DC, USA, August 2003.

[58] D. F. Ferraiolo and R. Kuhn. Role-Based Access Control: Features and Motivations. In *Proc. of the 11th Annual Computer Security Applications Conference*, pages 241–248, New Orleans, LA, USA, Dec. 1995.

[59] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. 2nd. Edition. Academic Press, 1990.

[60] S. Garfinkel. *Database Nation: The Death of the Privacy in the* 21^{st} *Century*. O'Reilly & Associates, Sebastopol, CA, USA, 2001.

[61] O. Goldreich. Secure Multi-Party Computation. Working Draft, 2000.

[62] O. Goldreich, S. Micali, and A. Wigderson. How to Play Any Mental Game - A Completeness Theorem for Protocols with Honest Majority. In *Proc. of the 19th Annual ACM Symposium on Theory of Computing*, pages 218–229, New York City, USA, May 1987.

[63] S. Goldwasser. Multi-party Computations: Past and Present. In *Proc. of the 16th Annual ACM Symposium on Principles of Distributed Computing*, pages 1–6, Santa Barbara, CA, August 1997.

[64] S. Guha, R. Rastogi, and K. Shim. CURE: An Efficient Clustering Algorithm For Large Databases. In *Proc. of the ACM SIGMOD International Conference on Management of Data*, pages 73–84, Seattle, WA, June 1998.

[65] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, San Francisco, CA, 2001.

[66] D. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. The MIT Press, Cambridge, Massachusetts, 2001.

[67] P. Indyk and R. Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *Proc. of the 30th Annual ACM Symposium on Theory of Computing*, pages 604–613, 1998.

[68] Md. Z. Islam, P. M. Barnaghi, and L. Brankovic. Measuring Data Quality: Predictive Accuracy vs. Similarity of Decision Trees. In *Proc. of the 6th International Conference on Computer And Information Technology (ICCIT 2003)*, Dhaka, Bangladesh, December 2003.

[69] Md. Z. Islam and L. Brankovic. Noise Addition for Protecting Privacy in Data Mining. In *Proc. of the 6th Engineering Mathematics and Applications Conference (EMAC 2003)*, Sydney, Australia, 2003.

[70] V. S. Iyengar. Transforming Data to Satisfy Privacy Constraints. In *Proc. of the 8th ACM SIGKDD Intl. Conf. on Knowlegde Discovery and Data Mining*, pages 279–288, Edmonton, AB, Canada, July 2002.

[71] J. Hipp and U. Güntzer and G. Nakhaeizadeh. Algorithms for Association Rule Mining - A General Survey and Comparison. *SIGKDD Explorations*, 2(1):58–64, July 2000.

[72] H. V. Jagadish. A Retrieval Technique For Similar Shapes. In *Proc. of the 1991 ACM SIGMOD International Conference on Management of Data*, pages 208–217, Denver, Colorado, USA, May 1991.

[73] P. Jefferies. Multimedia, Cyberspace & Ethics. In *Proc. of International Conference on Information Visualisation (IV2000)*, pages 99–104, London, England, July 2000.

[74] G. H. John. Behind-the-Scenes Data Mining. *Newletter of ACM SIG on KDDM*, 1(1):9–11, June 1999.

[75] W. B. Johnson and J. Lindenstrauss. Extensions of Lipshitz Mapping Into Hilbert Space. In *Proc. of the Conference in Modern Analysis and Probability*, pages 189–206, volume 26 of Contemporary Mathematics, 1984.

[76] T. Johnsten and V. V. Raghavan. Impact of Decision-Region Based Classification Mining Algorithms on Database Security. In *Proc. of 13th Annual IFIP WG 11.3 Working Conference on Database Security*, pages 177–191, Seattle, USA, July 1999.

[77] T. Johnsten and V. V. Raghavan. Security Procedures for Classification Mining Algorithms. In *Proc. of 15th Annual IFIP WG 11.3 Working Conference on Database and Applications Security*, pages 293–309, Niagara on the Lake, Ontario, Canada, July 2001.

[78] T. Johnsten and V. V. Raghavan. A Methodology for Hiding Knowledge in Databases. In *Proc. of the IEEE ICDM Workshop on Privacy, Security, and Data Mining*, pages 9–17, Maebashi City, Japan, December 2002.

[79] R. Joseph and C. L. Faith. The Platform for Privacy Preferences. 42(2):48-55, 1999.

[80] M. Kantarcioğlu and C. Clifton. Privacy-Preserving Distributed Mining of Association Rules on Horizontally Partitioned Data. In *Proc. of The ACM SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, Madison, Wisconsin, June 2002.

[81] M. Kantarcioğlu, J. Jin, and C. Clifton. When Do Data Mining Results Violate Privacy? In *Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 599–604, Seattle, WA, USA, August 2004.

[82] M. Kantarcioğlu and J. Vaidya. Privacy Preserving Naïve Bayes Classifier for Horizontally Partitioned Data. In *Proc. of the IEEE ICDM Workshop on Privacy Preserving Data Mining*, pages 3–9, Melbourne, FL, USA, November 2003.

[83] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the Privacy Preserving Properties of Random Data Perturbation Techniques. In *Proc. of the 3rd IEEE International Conference on Data Mining (ICDM'03)*, pages 99–106, Melbourne, Florida, USA, November 2003.

[84] G. Karypis, E.-H. Han, and V. Kumar. Chameleon: A Hierarchical Clustering Algorithm Using Dynamic Modeling. *IEEE Computer*, 32(8):68–75, 1999.

[85] S. Kaski. Dimensionality Reduction by Random Mapping. In *Proc. of the International Joint Conference on Neural Networks*, pages 413–418, Anchorage, Alaska, May 1999.

[86] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data- An Introduction to Cluster Analysis.* John Wiley & Sons, Inc., 1990.

[87] W. Klösgen. Anonymization Techniques for Knowledge Discovery in Databases. In *Proc. of the First International Conference on Knowledge Discovery and Data Mining (KDD-95)*, pages 186–191, Montreal, Canada, August 1995.

[88] W. Klösgen. KDD: Public and Private Concerns. *IEEE EXPERT*, 10(2):55–57, April 1995.

[89] J. B. Kruskal and M. Wish. *Multidimensional Scaling.* Sage Publications, Beverly Hills, CA, USA, 1978.

[90] B. Larsen and C. Aone. Fast and Effective Text Mining Using Linear-Time Document Clustering. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 16–22, San Diego, CA, USA, August 1999.

[91] K. C. Laudon. Markets and Privacy. *Communication of the ACM*, 39(9):92–104, September 1996.

[92] Y. Lindell and B. Pinkas. Privacy Preserving Data Mining. In *Crypto 2000, Springer-Verlag (LNCS 1880)*, pages 36–54, Santa Barbara, CA, August 2000.

[93] J. Macqueen. Some Methods for Classification and Analysis of Multivariate Observations. In *Proc. of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, Berkeley: University of California Press, Vol. 1, 1967.

[94] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, LLC, Fifth Printing, August 2001.

[95] S. Meregu and J. Ghosh. Privacy-Preserving Distributed Clustering Using Generative Models. In *Proc. of the 3rd IEEE International Conference on Data Mining (ICDM'03)*, pages 211–218, Melbourne, Florida, USA, November 2003.

[96] M. E. Mortenson. *Geometric Transformations.* New York: Industrial Press Inc., 1995.

[97] A. Mucsi-Nagy and S. Matwin. Digital Fingerprinting for Sharing of Confidential Data. In *Proc. of the Workshop on Privacy and Security Issues in Data Mining*, pages 11–26, Pisa, Italy, September 2004.

[98] Office of the Information and Privacy Commissioner. Data Mining: Staking a Claim on Your Privacy, Toronto, Ontario, January 1998.

[99] D. E. O'Leary. Knowledge Discovery as a Threat to Database Security. In G. Piatetsky-Shapiro and W. J. Frawley (editors): Knowledge Discovery in Databases. AAAI/MIT Press, pages 507-516, Menlo Park, CA, 1991.

[100] D. E. O'Leary. Some Privacy Issues in Knowledge Discovery: The OECD Personal Privacy Guidelines. *IEEE EXPERT*, 10(2):48–52, April 1995.

[101] S. R. M. Oliveira and O. R. Zaïane. Foundations for an Access Control Model for Privacy Preservation in Multi-Relational Association Rule Mining. In *Proc. of the IEEE ICDM Workshop on Privacy, Security, and Data Mining*, pages 19–26, Maebashi City, Japan, December 2002.

[102] S. R. M. Oliveira and O. R. Zaïane. Privacy Preserving Frequent Itemset Mining. In *Proc. of the IEEE ICDM Workshop on Privacy, Security, and Data Mining*, pages 43–54, Maebashi City, Japan, December 2002.

[103] S. R. M. Oliveira and O. R. Zaïane. Algorithms for Balancing Privacy and Knowledge Discovery in Association Rule Mining. In *Proc. of the 7th International Database Engineering and Applications Symposium (IDEAS'03)*, pages 54–63, Hong Kong, China, July 2003.

[104] S. R. M. Oliveira and O. R. Zaïane. Privacy Preserving Clustering By Data Transformation. In *Proc. of the 18th Brazilian Symposium on Databases*, pages 304–318, Manaus, Brazil, October 2003.

[105] S. R. M. Oliveira and O. R. Zaïane. Protecting Sensitive Knowledge By Data Sanitization. In *Proc. of the 3rd IEEE International Conference on Data Mining (ICDM'03)*, pages 613–616, Melbourne, Florida, USA, November 2003.

[106] S. R. M. Oliveira and O. R. Zaïane. Achieving Privacy Preservation When Sharing Data For Clustering. In *Proc. of the Workshop on Secure Data Management in a Connected World (SDM'04) in conjunction with VLDB'2004*, pages 67–82, Toronto, Ontario, Canada, August 2004.

[107] S. R. M. Oliveira and O. R. Zaïane. Privacy-Preserving Clustering by Object Similarity-Based Representation and Dimensionality Reduction Transformation. In *Proc. of the Workshop on Privacy and Security Aspects of Data Mining (PSADM'04) in conjunction with the Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 21–30, Brighton, UK, November 2004.

[108] S. R. M. Oliveira and O. R. Zaïane. Toward Standardization in Privacy-Preserving Data Mining. In *Proc. of the 3nd Workshop on Data Mining Standards (DM-SSP 2004), in conjuction with KDD 2004*, pages 7–17, Seattle, WA, USA, August 2004.

[109] S. R. M. Oliveira, O. R. Zaïane, and Y. Saygin. Secure Association Rule Sharing. In *Proc. of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'04)*, pages 74–85, Sydney, Australia, May 2004.

[110] C. H. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala. Latent Semantic Indexing: A Probabilistic Analysis. In *Proc. of the 17th ACM Symposium on Principles of Database Systems*, pages 159–168, Seattle, WA, USA, June 1998.

[111] G. Piatetsky-Shapiro. Discovery, Analysis and Presentation of Strong Rules. In G. Piatetsky-Shapiro and W. J. Frawley (editors): Knowledge Discovery in Databases. AAAI/MIT Press, pages 229-248, Menlo Park, CA, 1991.

[112] G. Piatetsky-Shapiro. Knowledge Discovery in Personal Data vs. Privacy: A Mini-Symposium. *IEEE Expert*, 10(2):46–47, 1995.

[113] G. Piatetsky-Shapiro. The Data-Mining Industry Coming of Age. *IEEE Expert*, 14(6):32–34, 1999.

[114] B. Pinkas. Cryptographic Techniques For Privacy-Preserving Data Mining. *SIGKDD Explorations*, 4(2):12–19, December 2002.

[115] Platform for Privacy Preferences (P3P) Project. Available at http://www.w3.org/P3P/.

[116] J. R. Quinlan. Learning Efficient Classification Procedures and Their Application to Chess end Games. In *R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, eds., Machine Learning – An Artificial Intelligence Approach*, pages 463–482, Tioga, Palo Alto, CA, 1983.

[117] R. Ramakrishnan and J. Gehrke. *Database Management Systems (second edition)*. McGraw-Hill, 2000.

[118] Amy C. Rea. The Cooperative Side of Competition: Coopetition. In *Management World: The Online Publication for Certified Managers*. Institute for Certified Professional Managers, Harrisonburg, Virginia, USA, May 2002.

[119] A. Rezgui, A. Bouguettaya, and M. Y. Eltoweissy. Privacy on the Web: Facts, Challenges, and Solutions. *IEEE Security & Privacy*, 1(6):40–49, Nov-Dec 2003.

[120] S. J. Rizvi and J. R. Haritsa. Maintaining Data Privacy in Association Rule Mining. In *Proc. of the 28th International Conference on Very Large Data Bases*, Hong Kong, China, August 2002.

[121] A. Rosenberg. Privacy as a Matter of Taste and Right. In E. F. Paul, F. D. Miller, and J. Paul, editors, The Right to Privacy, pages 68-90, Cambridge University Press, 2000.

[122] P. Ruch, R. H. Baud, and A.-M. Rassinoux. Medical Document Anonymization With a Semantic Lexicon. *Journal of American Medical Informatics Association (Symposium Supplement)*, pages 729–733, 2000.

[123] P. Samarati. Protecting Respondents' Identities in Microdata Release. *IEEE Transactions on Knowledge and Data Engineering*, 13(6):1010–1027, 2001.

[124] P. Samarati and L. Sweeney. Protecting Privacy When Disclosing Information: K-anonymity and its Enforcement Through Generalization and Suppression. In *Proc. of the IEEE Symposium on Research in Security and Privacy*, May 1998.

[125] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-Based Access Control Models. *IEEE Computer*, 20(2):38–47, 1996.

[126] Y. Saygin, V. S. Verykios, and C. Clifton. Using Unknowns to Prevent Discovery of Association Rules. *SIGMOD Record*, 30(4):45–54, December 2001.

[127] Y. Saygin, V. S. Verykios, and A. K. Elmagarmid. Privacy Preserving Association Rule Mining. In *Proc. of the 12th International Workshop on Research Issues in Data Engineering: Engineering E-Commerce/E-Business Systems (RIDE'02)*, pages 151–158, San Jose, CA, USA, February 2002.

[128] B. Schneier. *Applied Cryptography: Protocols, Algorithms, and Source Code in C.* John Wiley & Sons, Inc., Second Edition, 1996.

[129] F. D. Schoeman. Philosophical Dimensions of Privacy, Cambridge Univ. Press, 1984.

[130] L.J. Schulman. Clustering for Edge-Cost Minimization. In *Proc. of the 32nd Annual ACM Symposium on Theory of Computing* , pages 547–555, Portland, OR, USA, 2000.

[131] G. Sheikholeslami, S. Chatterjee, and A. Zhang. WaveCluster: A Multi-Resolution Clustering Approach for Very Large Spatial Databases. In *Proc. 24th International Conference on Very Large Data Bases*, pages 428–439, New York City, August 1998.

[132] P. Soldacki and G. Protaziuk. Discovering Interesting Rules from Financial Data. In *Proc. of the Intelligent Information Systems (IIS'2002)*, pages 109–119, Sopot, Poland, June 2002.

[133] L. Sweeney. Achieving k-Anonymity Privacy Protection using Generalization and Suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):571–588, 2002.

[134] L. Sweeney. k-Anonymity: A Model for Protecting Privacy. *International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):557–570, 2002.

[135] P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the Right Interestingness Measure for Association Patterns. In *Proc. of the 8th ACM SIGKDD Intl. Conf. on Knowlegde Discovery and Data Mining*, pages 32–41, Edmonton, AB, Canada, July 2002.

[136] M. Tang, D. Hakkani-Tür, and G. Tur. Preserving Privacy in Spoken Language Databases. In *Proc. of the Workshop on Privacy and Security Issues in Data Mining*, pages 27–36, Pisa, Italy, September 2004.

[137] P. Tendick and N. S. Matloff. Recent Results on the Noise Addition Method for Database Security. In *Proc. of the 1987 Joint Meetings, American Statistical Association / Institute of Mathematical Statistics (ASA/IMA)*, pages 406–409, Washington, DC, USA, 1987.

[138] E. Turban and J. E. Aronson. *Decision Support Systems and Intelligent Systems.* Prentice-Hall, New Jersey, USA, 2001.

[139] J. Vaidya and C. Clifton. Privacy Preserving Association Rule Mining in Vertically Partitioned Data. In *Proc. of the 8th ACM SIGKDD Intl. Conf. on Knowlegde Discovery and Data Mining*, pages 639–644, Edmonton, AB, Canada, July 2002.

[140] J. Vaidya and C. Clifton. Privacy-Preserving K-Means Clustering Over Vertically Partitioned Data. In *Proc. of the 9th ACM SIGKDD Intl. Conf. on Knowlegde Discovery and Data Mining*, pages 206–215, Washington, DC, USA, August 2003.

[141] A. A. Veloso, W. Meira Jr., S. Parthasarathy, and M. B. Carvalho. Efficient, Accurate and Privacy-Preserving Data Mining for Frequent Itemsets in Distributed Databases. In *Proc. of the 18th Brazilian Symposium on Databases*, pages 281–292, Manaus, Brazil, October 2003.

[142] V. S. Verykios, A. K. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni. Association Rule Hiding. *IEEE Transactions on Kowledge Discovery and Data Engineering*, 16(4):434–447, April 2004.

[143] W. Wang, J. Yang, and R. R. Muntz. STING: A Statistical Information Grid Approach to Spatial Data Mining. In *Proc. of the 23rd International Conference on Very Large Data Bases*, pages 186–195, Athens, Greece, 1997.

[144] S. D. Warren and L. D. Brandeis. The Right to Privacy. *Harvard Law Review*, 4(5):193–220, 1890.

[145] G. A. Watson. An Algorithm for the Single Facility Location Problem Using the Jaccard Metric. *SIAM J. Sci. Stat. Comput.*, 4(4), December 1983.

[146] A. F. Westin. The Right to Privacy, Atheneum, 1967.

[147] L. Willenborg and T. D. Waal. *Statistical Disclosure Control in Practice.* Springer-Verlag, 1996.

[148] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, 1999.

[149] C. W. Wu. Privacy Preserving Data Mining: A Signal Processing Perspective and a Simple Data Perturbation Protocol. In *Proc. of the IEEE ICDM Workshop on Privacy Preserving Data Mining*, pages 10–17, Melbourne, FL, USA, November 2003.

[150] A.C.-C. Yao. How to Generate and Exchange Secrets. In *Proc. of the 27th IEEE Symposium of Foundations of Computer Science*, pages 162–167, Toronto, Ontario, Canada, October 1986.

[151] F. W. Young. *Multidimensional Scaling.* Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1987.

[152] O. R. Zaïane and A. Foss. Data Clustering Analysis, from Simple Groupings to Scalable Clustering with Constraints. In *Conference Tutorial Notes: 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD'02)*, pages 137–201, Taipei, Taiwan, 2002.

[153] N. Zang, S. Wang, and W. Zhao. A New Scheme on Privacy Preserving Association Rule Mining. In *Proc. of the 15th European Conference on Machine Learning (ECML) and the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)*, Pisa, Italy, September 2004.

[154] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: An Efficient Data Clustering Method for Very Large Databases. In *Proc. of the 1996 ACM SIGMOD International Conference on Management of Data*, pages 103–114, Montreal, Canada, 1996.

[155] Z. Zheng, R. Kohavi, and L. Mason. Real World Performance of Association Rules Algorithms. In *Proc. of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA, USA, August 2001.

[156] Y. Zhu and L. Liu. Optimal Randomization for Privacy Preserving Data Mining. In *Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 761–766, Seattle, WA, USA, August 2004.

Appendix A

Complexity of the Sanitizing Algorithms

A.1 Analysis of the Data Sharing-Based Algorithms

A.1.1 The Running Time of the Round Robin Algorithm

Theorem 1 *The running time of the Round Robin algorithm is $O(n_1 \times N \times log\ N)$, in the worst case, where n_1 is the number of sensitive rules and N is the number of transactions in the database.*

Proof. Let D be a transactional database, N the number of transactions in D, n_1 the number of sensitive rules in D, n_2 the maximum number of items in a sensitive rule, and let n_3 the maximum number of items in one transaction.

In Step 1, first the items in each transaction are sorted in alphabetic order. So line 4 takes $n_3\ log\ n_3$. Then, for each association rule $sr_i \in S_R$, the algorithm verifies if such rule is present in the current transaction t. To do so, in line 6 the algorithm binary searches, in the worst case, all items of the rule sr_i in the transaction t to make sure that this rule is present in t. This entire process encompasses n_2 binary searches over a transaction with at most n_3 items, so that line 6 takes $n_2 \times log\ n_3$. Line 7 contains a straightforward computation $O(1)$. Because the loop from line 5 to 9 is executed n_1 times, this whole loop takes $n_1 \times n_2 \times log\ n_3 + n_1$. Now the outer loop from line 3 to 10 is performed N times. So the running time for Step 1 takes $O(N \times n_3 \log\ n_3 + N \times n_1 \times n_2 \times log\ n_3 + N \times n_1)$. When N and n_1 are large, $N \times n_1 \times n_2 \times log\ n_3$ grows faster than $N \times n_3 \log\ n_3$ and $N \times n_1$. Therefore, Step 1 takes $O(N \times n_1 \times n_2 \times log\ n_3)$.

In Step 2, line 13, in the worst case all transactions in D are sensitive. Thus, the sort of sensitive transactions takes $NlogN$. Line 14 is a straightforward computation and takes $O(1)$. The entire loop from line 12 to 16 takes $O(n_1 \times NlogN + n_1)$ that can be simplified to $O(n_1 \times NlogN)$.

In step 3, the inner loop from line 19 to 23 is performed N times, i.e., in the worst case all transactions are marked to be sanitized. Lines 20 and 22 encompass straightforward computations. So the inner loop in step 3 takes $O(N)$. The loop from line 18 to 24 is performed n_1 times. Thus, the running time for step 3 takes $O(n_1 \times N)$.

In step 4, line 26, the algorithm sorts the vector $Victims$ in alphabetic order of t_{ID}. This sort takes $NlogN$ considering that all transactions are marked to be sanitized. The loop from line 28 to 33 is performed N times (second scan). In line 30, considering that the items in all transactions in D were sorted in step 1 (line 4), the process of sanitizing the items in line 30 encompasses at most n_3 binary searches in case of all items are victim items. So line 30 takes $n_3\ log\ n_3$ and the loop from line 28 to 33 takes $N \times n_3\ log\ n_3$. Thus, the running time for step 4 takes $O(NlogN + N \times n_3\ log\ n_3)$. When N is large, $NlogN$ grows faster than $N \times n_3\ log\ n_3$. Thus, step 4 takes $O(NlogN)$.

The running time of the Round Robin algorithm is the sum of running times for each step executed, i.e., $O(N \times n_1 \times n_2 \times log\ n_3 + n_1 \times NlogN + n_1 \times N + NlogN)$. When N and n_1 are large, $n_1 \times NlogN$ grows faster than $N \times n_1 \times n_2 \times log\ n_3$, $n_1 \times N$, and $NlogN$. Hence, the running time of the Round Robin algorithm takes $O(n_1 \times N \times log\ N)$. □

A.1.2 The Running Time of the Item Grouping Algorithm

Theorem 2 *The running time of the Item Grouping algorithm is $O(n_1 \times N \times log\ N)$, in the worst case, where n_1 is the number of sensitive rules and N is the number of transactions in the database.*

Proof. Let D be the source database, N the number of transactions in D, n_1 the number of sensitive rules in D, n_2 the maximum number of items in a sensitive rule, and n_3 the maximum number of items in one transaction.

The only difference in step 1, between the Item Grouping algorithm (IGA) and the Round Robin algorithm (RRA), is in lines 4 and 5 of IGA. In these lines, IGA keeps the frequencies of all the items in the database D. This computation is simple and takes $O(n_3 \times N)$ since it is performed N times. However, the loop from line 7 to 11 is still the most expensive computation in step 1. From Theorem 1, we know that this computation takes $O(N \times n_1 \times n_2 \times log\ n_3)$.

From Theorem 1, we also know that step 2 takes $O(n_1 \times NlogN)$ and step 4 takes $O(NlogN)$.

Now we move on to analyzing step 3. From step 3.1 to 3.3, the following computations are performed: (a) group the sensitive rules in groups sharing the same items. To optimize this step, IGA verifies which rules share the item with highest frequency in all sensitive rules S_R. IGA then clusters the next sensitive rules sharing the second highest frequency in all sensitive rules S_R, and this process continues until the clusters with one sensitive rule are formed; (b) If two or more clusters share more than one item and these clusters have the same size, IGA favors the cluster whose rules share the lowest frequency item. The rationale behind this selection is that this item will be the victim item of these rules (for all sensitive transactions), and removing this item results in less combinations of rules affected in the sanitized database. In each cluster formed, IGA eliminates directly the rules already clustered which greatly optimizes the step 3.3. For instance, if $rule1$, $rule2$, and $rule3$ share the highest frequent item in all sensitive rules S_R, these rules are automatically eliminated from the other clusters with rules sharing item(s) with lower frequencies. Before clustering the rules in step 3.1, IGA scans all items in the sensitive rules to compute their frequencies. This procedure takes $O(n_1 \times n_2)$. IGA then sorts the items in descending order of frequencies which takes $O(n_1 \times n_2 \times log\ n_1 \times n_2)$. Steps 3.2 and 3.3 are computed as follows: IGA scans the sensitive rules again and checks whether the highest frequent item is one of the items in the current rule. If so, that rule is marked, i.e., it is grouped in the cluster sharing the item with highest frequency and this rule will no longer be scanned again. This process continues until the clusters with one sensitive rule are formed. Thus, steps 3.2 and 3.3 takes $O(n_1 \times n_2)$ since IGA scans all items (n_2) in all rules (n_1), in the worst case. In doing so, the rules will be grouped (in case some of them share one or more items) in one scan. Therefore, steps 3.1 to 3.3 takes $O(n_1 \times n_2 \times log\ n_1 \times n_2)$. Now, we know from Theorem 1 that step 3.4 (lines 33-38) takes $O(n_1 \times N)$. When N and n_1 are large, $n_1 \times N$ grows faster than $n_1 \times n_2 \times log\ n_1 \times n_2$. Thus, the running time for step 3 is $O(n_1 \times N)$.

The running time of the Item Grouping Algorithm is the sum of the running times for each step executed, i.e., $O(N \times n_1 \times n_2 \times log\ n_3 + n_1 \times NlogN + n_1 \times N + NlogN)$. When N and n_1 are large, $n_1 \times NlogN$ grows faster than $N \times n_1 \times n_2 \times log\ n_3$, $n_1 \times N$, and $NlogN$. Hence, the running time of the Item Grouping algorithm takes $O(n_1 \times N \times log\ N)$. $\square$

A.1.3 The Running Time of the Sliding Window Algorithm

Theorem 3 *The running time of the SWA is $O(n_1 \times N \times log\ K)$ when $\psi \neq 0$ and $O(n_1^2 \times N \times K)$ when $\psi = 0$, where n_1 is the initial number of sensitive rules in the database D, K is the window size chosen, and N is the number of transactions in D.*

Proof. Let D be the source database, N the number of transactions in D, n_1 the initial number of sensitive association rules selected in D, n_2 the maximum number of items in a sensitive association rule, n_3 the maximum number of items in a transaction $t \in D$, K the number of transactions in a sliding window, and M_P a set of mining permissions in which each permission is defined as $mp = < sr_i, \psi_i >$, where $\forall i\ sr_i$ is a sensitive rule and ψ_i is the corresponding disclosure threshold.

In line 5 of Step 1, first the items in each transaction are sorted in alphabetic order. So line 5 takes $n_3\ log\ n_3$. Then, for each sensitive association rule $sr_i \in S_R$, the algorithm verifies if such rule is present in the current transaction t. To do so, the algorithm performs, in the worst case, n_2 (items in sr_i) binary searches in the transaction t to make sure that the items in sr_i are present in t. If the transaction t is sensitive (generates at least one sensitive rule), its ID is added to the list of transaction IDs of the corresponding sr_i. In addition, the size of the transaction is added to the list of transaction size, and the frequency of $item_j \in sr_i$ and $item_j \in t$ is updated. Line 7 encompasses n_2 binary searches over a transaction with at most n_3 items. Lines 8, 9, 10, and 11 are straightforward computations, so that the loop from line 6 to 13 takes $O(n1 \times (n_2 \times log\ n_3 + c))$, where c is a constant. Because step 1 is performed K times, the running time of step 1 takes $O(K \times (n_3\ log\ n_3 + n_1 \times n_2 \times log\ n_3 + c))$. When n_1 is large, step 1 takes $O(K \times n_1 \times n_2 \times log\ n_3)$.

In line 16 of Step 2, the vector of frequencies computed in the previous step contains at most $n_1 \times n_2$ items, when all items in the sensitive rules are present in the transaction t. The running time to sort this vector takes $O(n_1 \times n_2 \times log\ (n_1 \times n_2))$. The selection of the victim item in line 21 and line 23 are straightforward computations that takes $O(1)$ per sensitive rule since the vector of frequencies is already sorted. Thus, the loop from line 17 to 25 takes $O(n_1)$ since it is executed n_1 times. Considering that step 2 is executed K times, the running time in step 2 takes $O(K \times (n_1 \times n_2 \times log\ (n_1 \times n_2)) + n_1)$, which can be simplified to $O(K \times n_1 \times n_2 \times log\ (n_1 \times n_2))$.

Line 31 of Step 3 contains a straightforward computation that takes $O(1)$ per sensitive rule. In line 32, SWA sorts the vector T in ascending order of size, which takes $K\ logK$ if all the K transactions are sensitive. Because step 3 is performed n_1 times, the running time for this step takes $O(n_1 \times K\ logK)$.

In line 37 of Step 4, SWA needs one access to find the victim item in the sensitive transaction, and one more computation to remove it. Recall that the sensitive transactions are sorted in step 1 and are implemented in an array fashion. If all the K transactions are sensitive, line 37 takes $O(K)$.

First, let us consider the case where $\psi \neq 0$. In this case, Step 4 takes $O(n_1 \times K)$. The running time of the Sliding Window algorithm is the sum of running times for each step, i.e., $O(K \times n_1 \times n_2 \times log\ n_3 + K \times n_1 \times n_2 \times log\ (n_1 \times n_2) + n_1 \times K\ logK + n_1 \times K)$. When n_1 are K large, the running time can be simplified to $O(n_1 \times K\ logK)$. Considering that the whole database contains N transactions, and there exists N/K windows, the running time of the Sliding Window Algorithm takes $O(n_1 \times N \times log\ K)$, when $\psi \neq 0$.

When $\psi = 0$, SWA performs the look ahead procedure. In the worst case, all K transactions are marked to be sanitized in all sensitive rules. We know that line 37 takes $n_1 \times K$. Now we need to analyze the loop from line 38 to 40. In this loop, SWA checks if the current victim item sanitized is present in the next sensitive rules. This procedure takes $(n_1 - 1) \times K$ for the first sensitive rule, $(n_1 - 2) \times K$ for the second rule, and it repeats until the last sensitive rule. So the number of times that the look ahead procedure is performed is the sum of the terms of an arithmetic progression, i.e., $((n_1 - 1) \times ((n_1 - 1) \times K + K))/2$. However, this loop is performed K times (the size of the window). In this case, Step 4 is the sum of running times for each step and takes $O(n_1 \times K + n_1^2 \times K^2)$, which can be simplified to $O(n_1^2 \times K^2)$. Considering that the whole database contains N transactions, and there exists N/K windows, the running time of the Sliding Window Algorithm takes $O(n_1^2 \times N \times K)$, when $\psi = 0$. □

A.2 Analysis of the Pattern Sharing-Based Algorithm

A.2.1 The Running Time of the Downright Sanitizing Algorithm

Theorem 4 *The running time of the Downright Sanitizing Algorithm is $O(n \times (k^2 + m \times log\ k))$, where n is the number of sensitive rules to be sanitized, m is the number of itemsets in a frequent itemsets graph G, and k the maximum number of items in a frequent itemset in G.*

Proof. Let G be a frequent itemset graph corresponding to a transactional database D mined with a minimum support threshold σ, n the initial number of sensitive association rules to be sanitized, k the maximum number of items in a frequent itemset $c_j \in G$, m the number of frequent itemsets in G, and S_R a set of sensitive rule.

In line 4 of step 1, the algorithm converts each sensitive rule $sr_i \in S_R$ into a frequent itemset c_i. For each sensitive rule, the DSA scans at most k items, since k is the maximum number of items in an itemset/rule $c_j \in G$. The entire loop from line 3 to 5 is performed n times. Thus, step 1 takes $O(n \times k)$.

In line 8 of step 2, the algorithm computes the item pairs of each sensitive itemset c_i. This computation takes $C_{k,2} = (k \times (k-1))/2$ for each sensitive itemset c_i. In line 9, to make sure that at least one item pair is marked in the list $MarkedPair$, the DSA scans in the worst case all the pairs of a sensitive itemset c_i. This operation is performed $(k \times (k-1))/2$ times. If none of the item pairs is marked in the list $MarkedPair$, the algorithm selects one pair randomly. This selection in line 10 is straightforward and takes $O(1)$. The update of the list $MarkedPair$ in line 11 also takes $O(1)$. Thus, in the worse case, the loop from line 7 to 13 takes $O(n \times (k \times (k-1))/2 + n \times (k \times (k-1))/2)$. When n and k are large, the running time of the DSA takes $O(n \times k^2)$.

In Step 3, in our implementation first the items in each itemset $c_j \in G$ are sorted in ascending order. Thus line 17 takes $k\ log\ k$. Considering that there are m frequent itemsets in G, the loop from line 16 to 18 takes $O(m \times k\ log\ k)$. The loop from line 19 to 23 is performed m times. In line 20 of step 3, for all marked pairs in the list $MarkedPair$, the algorithm verifies if at least one pair p is a subset of the current frequent itemset c_j. To do so, the algorithm binary searches, in the worst case, all the items in each pair p of the list $MarkedPair$ to make sure that p is a subset of c_j. In the worst case, there are n marked pairs in the list $MarkedPair$, one corresponding to each sensitive itemset c_i. Thus, this entire process encompasses n binary searches over a frequent itemset $c_j \in G$, so that line 20 is performed $n \times log\ k$. Thus step 3 takes $O(m \times k\ log\ k + m \times n \times log\ k)$. When m and n are large, $m \times n \times log\ k$ grows faster than $m \times k\ log\ k$. Thus step 3 takes $O(m \times n \times log\ k)$.

The running time of the Downright Sanitizing Algorithm is the sum of running times for each step. Thus, the running time of DSA takes $O(n \times k + n \times k^2 + n \times m \times log\ k)$. Considering that $n \times k^2$ grows faster that $n \times k$, the running time of the Downright Sanitizing Algorithm takes $O(n \times k^2 + n \times m \times log\ k)$, which can be simplified to $O(n \times (k^2 + m \times log\ k))$. □

Appendix B

Results of the Sanitization in Real Datasets

B.1 Sensitive Rules

B.1.1 The BMS-Web-View-1 Dataset

No.	Sensitive Rules	Support (%)	Confidence (%)
1	310,306,311,314,308 => 307	0.1	96.8
2	345,348 => 347	0.1	90.3
3	25 => 27	0.2	63.3
4	113,108,107 => 111	0.1	83.3
5	7,5,6,2,120 => 122	0.1	79.7
6	41,64 => 63	0.1	70.5

Table B.1: A set of 6 sensitive rules mutually exclusive

No.	Sensitive Rules	Support (%)	Confidence (%)
1	311,307,308 => 315	0.1	72.6
2	319,305,304 => 307	0.1	62.1
3	315,6,122 => 301	0.1	60.3
4	306,311,308,312 => 345	0.1	62.9
5	310,311,307,314 => 319	0.1	64.3
6	168,6,2,3 => 7	0.1	64.2

Table B.2: A set of 6 sensitive rules selected randomly

No.	Sensitive Rules	Support (%)	Confidence (%)
1	5,3 => 2	0.8	64.6
2	73,301 => 122	0.8	62.3
3	6,72 => 73	0.8	66.1
4	310,306 => 315	0.7	67.0
5	168,2 => 5	0.7	61.8
6	73,318 => 122	0.6	64.1

Table B.3: A set of 6 sensitive rules with high support

No.	Sensitive Rules	Support (%)	Confidence (%)
1	342,343 => 328	0.1	60.0
2	317,23 => 21	0.1	60.0
3	347,348 => 306	0.1	60.0
4	207 => 201	0.1	60.0
5	57,63 => 41	0.1	61.1
6	251 => 176	0.1	62.7

Table B.4: A set of 6 sensitive rules with low support

B.1.2 The Retail Dataset

No.	Sensitive Rules	Support (%)	Confidence (%)
1	41,43,172 => 40	0.7	98.6
2	4032 => 50	0.1	82.6
3	1086 => 1078	0.1	69.9
4	4741 => 6175	0.1	70.7
5	7990 => 1386	0.1	69.7
6	1820 => 797	0.3	66.4

Table B.5: A set of 6 sensitive rules mutually exclusive

No.	Sensitive Rules	Support (%)	Confidence (%)
1	50,792 => 40	0.4	97.5
2	43,2201 => 50	0.2	71.4
3	311,1381,1382 => 1380	0.1	77.2
4	3898 => 41	0.2	61.1
5	589 => 50	0.3	62.5
6	833 => 41	0.5	69.2

Table B.6: A set of 6 sensitive rules selected randomly

No.	Sensitive Rules	Support (%)	Confidence (%)
1	50,34 => 41	9.1	67.2
2	67 => 41	5.1	62.3
3	40,43 => 50	4.4	60.9
4	91 => 41	4.4	71.6
5	38 => 40	3.3	95.0
6	16013 => 16012	0.8	97.3

Table B.7: A set of 6 sensitive rules with high support

No.	Sensitive Rules	Support (%)	Confidence (%)
1	41,3163 => 50	0.1	66.7
2	41,40,34,43,112 => 50	0.1	78.7
3	43,178 => 41	0.1	63.4
4	3879 => 6084	0.1	68.1
5	1386 => 1387	0.1	67.9
6	1381,1271 => 311	0.1	75.8

Table B.8: A set of 6 sensitive rules with low support

B.1.3 The Reuters Dataset

No.	Sensitive Rules	Support (%)	Confidence (%)
1	8916 => 17586	5.5	65.3
2	593 => 2	5.8	85.7
3	11820,20447 => 19096	6.3	76.8
4	7563 => 4977	5.9	63.0
5	14061,15273,30,5927,25727,21822,13981 => 7136	6.2	67.6
6	7086 => 19094	6.5	70.9

Table B.9: A set of 6 sensitive rules mutually exclusive

No.	Sensitive Rules	Support (%)	Confidence (%)
1	14061,7136,5927,25727 => 20447	7.9	69.4
2	11838 => 25727	5.9	78.5
3	14061,18657 => 7086	7.0	61.5
4	5927,7086 => 17535	5.6	82.6
5	7563 => 4977	5.9	63.0
6	14061,15273,30,21822,19096 => 5927	6.0	94.0

Table B.10: A set of 6 sensitive rules selected randomly

No.	Sensitive Rules	Support (%)	Confidence (%)
1	21636 => 7136	15.9	68.1
2	30,11820 => 5927	15.1	80.9
3	14061,15273,25727 => 19096	14.9	71.0
4	15273,30,21822,16400 => 25727	9.2	98.7
5	5927,21288,13981 => 15273	8.7	73.4
6	15273,18735 => 30	8.3	94.0

Table B.11: A set of 6 sensitive rules with high support

No.	Sensitive Rules	Support (%)	Confidence (%)
1	41,3163 => 50	0.1	66.7
2	41,40,34,43,112 => 50	0.1	78.7
3	43,178 => 41	0.1	63.4
4	3879 => 6084	0.1	68.1
5	1386 => 1387	0.1	67.9
6	1381,1271 => 311	0.1	75.8

Table B.12: A set of 6 sensitive rules with low support

B.1.4 The Kosarak Dataset

No.	Sensitive Rules	Support (%)	Confidence (%)
1	6,11,324,338,339,331,332 => 327	0.3	68.6
2	1519 => 530	0.3	62.3
3	529 => 528	0.4	62.0
4	218 => 371	0.3	69.3
5	1529 => 27	0.3	80.2
6	1518 => 2284	0.3	64.0

Table B.13: A set of 6 sensitive rules mutually exclusive

No.	Sensitive Rules	Support (%)	Confidence (%)
1	148,491 => 1	0.4	62.4
2	77,737 => 11	0.4	86.5
3	338,339,330 => 332	0.3	89.2
4	6,87,32 => 27	0.4	85.8
5	6,11,205,1956 => 7	0.4	78.0
6	6,11,1,514 => 218	0.3	70.9

Table B.14: A set of 6 sensitive rules selected randomly

No.	Sensitive Rules	Support (%)	Confidence (%)
1	1,3 => 6	8.6	78.6
2	6,218 => 11	7.8	78.1
3	6,148 => 11	6.5	85.3
4	6,7,27 => 11	3.8	81.7
5	1,218 => 148	3.2	78.6
6	205 => 27	2.2	67.9

Table B.15: A set of 6 sensitive rules with high support

No.	Sensitive Rules	Support (%)	Confidence (%)
1	6,11,324,338,339,331,332 => 335	0.3	73.5
2	1519 => 530	0.3	62.3
3	11,25,747 => 1	0.3	66.2
4	6,11,27,148,83 => 218	0.3	92.1
5	6,11,27,148,205 => 7	0.3	81.3
6	334 => 325	0.3	68.8

Table B.16: A set of 6 sensitive rules with low support

B.2 Evaluation of Window Size for SWA

Kosarak	**Window Size**								
	500	1000	5000	10000	20000	40000	60000	80000	100000
Dif.	1.69	1.67	1.66	1.65	1.65	1.65	1.65	1.65	1.65
MC	44.90	42.81	40.47	40.09	39.92	39.87	39.87	39.87	39.87
HF	12.01	14.96	17.24	17.71	17.71	17.83	17.83	17.83	17.83

Reuters	**Window Size**								
	500	1000	5000	10000	20000	40000	60000	80000	100000
Dif.	0.65	0.64	0.64	0.64	0.64	0.64	0.64	0.64	0.64
MC	66.91	66.87	66.62	66.62	66.62	66.62	66.62	66.62	66.62
HF	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Retail	**Window Size**								
	500	1000	5000	10000	20000	40000	60000	80000	100000
Dif.	1.09	1.08	1.07	1.07	1.07	1.07	1.07	1.07	1.07
MC	30.52	30.17	28.75	28.13	28.07	27.74	27.74	27.74	27.74
HF	14.56	15.06	15.56	16.05	15.89	15.89	15.89	15.89	15.89

BMS-1	**Window Size**								
	500	1000	5000	10000	20000	40000	60000	80000	100000
Dif.	0.79	0.74	0.70	0.70	0.70	0.69	0.69	–	–
MC	44.94	39.20	27.24	25.27	22.96	10.30	10.30	–	–
HF	0.00	0.00	2.18	2.49	5.92	11.54	11.54	–	–

Table B.17: Effect of window size on the difference between D and D', misses cost and hiding failure.

B.3 Results of Misses Cost on the Datasets

B.3.1 Condition C1 (A set of 6 Sensitive Rules)

Kosarak	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	2.06	28.56	62.11	29.88
RRA	28.98	42.22	74.42	37.92
RA	28.80	42.62	74.37	38.02
SWA	29.15	42.49	72.70	37.97
Algo2a	26.03	45.14	62.58	36.53
DSA	5.41	19.34	24.02	8.05

Table B.18: Results of misses cost on the dataset Kosarak for condition C1.

Retail	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	1.03	9.16	66.31	3.11
RRA	2.66	5.87	64.02	3.25
RA	2.48	5.77	63.86	3.10
SWA	2.77	5.64	65.29	3.15
Algo2a	5.05	10.04	82.43	3.97
DSA	0.19	0.47	46.03	9.31

Table B.19: Results of misses cost on the dataset Retail for condition C1.

Reuters	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	45.67	46.96	67.10	45.00
RRA	64.36	67.47	89.00	49.06
RA	64.47	66.45	89.03	50.15
SWA	64.50	64.46	75.22	47.32
Algo2a	47.35	66.81	77.32	45.60
DSA	32.85	37.34	35.85	51.81

Table B.20: Results of misses cost on the dataset Reuters for condition C1.

BMS-1	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	21.73	15.36	28.01	15.36
RRA	39.79	41.30	53.13	41.30
RA	37.77	43.05	50.35	43.05
SWA	40.77	32.84	49.80	32.84
Algo2a	24.67	42.25	46.57	42.25
DSA	7.06	5.68	8.17	0.15

Table B.21: Results of misses cost on the dataset BMS-1 for condition C1.

B.3.2 Condition C2 (Varying the Number of Sensitive Rules)

Algorithm	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	1.73	1.73	1.80	1.87	1.89	2.06
R. Robin	22.95	23.18	24.09	28.17	27.88	28.07
Random	22.74	22.96	23.86	28.25	28.01	28.25
SWA	22.85	22.92	23.94	28.16	29.05	29.14
Algo2a	15.38	15.65	16.65	24.51	25.87	26.02
DSA	5.40	5.40	5.41	5.41	5.41	5.42

Table B.22: Results of misses cost on the Kosarark dataset for condition C2 (S1)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.57	0.58	0.59	0.64	0.65	1.03
R. Robin	1.65	1.98	1.99	2.10	1.91	2.38
Random	1.71	2.08	2.09	2.20	2.04	2.53
SWA	1.65	1.76	2.16	1.97	2.20	2.76
Algo2a	3.85	4.33	4.35	4.46	4.50	5.04
DSA	0.19	0.19	0.19	0.19	0.20	0.20

Table B.23: Results of misses cost on the Retail dataset for condition C2 (S1)

Algorithm	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.02	23.73	23.75	45.40	45.44	45.66
R. Robin	0.07	44.35	44.50	63.48	60.46	61.49
Random	0.04	44.16	44.54	63.49	59.68	60.87
SWA	0.06	44.72	44.93	62.97	63.86	64.50
Algo2a	0.10	39.54	40.08	45.99	46.44	47.34
DSA	0.02	25.67	25.68	32.82	32.83	32.85

Table B.24: Results of misses cost on the Reuters dataset for condition C2 (S1)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	6.86	21.27	21.31	21.45	21.46	21.72
R. Robin	32.39	40.86	40.41	40.71	38.99	40.24
Random	31.21	40.78	39.75	40.46	37.51	39.34
SWA	32.05	39.54	39.48	40.33	41.42	40.76
Algo2a	21.08	23.21	23.28	24.01	24.02	24.66
DSA	5.13	6.93	6.99	7.03	7.04	7.05

Table B.25: Results of misses cost on the BMS-1 dataset for condition C2 (S1)

Algorithm	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.02	0.03	40.71	40.94	53.61	57.29
R. Robin	1.28	8.36	79.74	80.62	80.78	83.13
Random	1.31	8.90	79.65	80.16	81.03	83.56
SWA	1.26	9.02	79.58	80.42	81.68	84.03
Algo2a	1.70	19.85	72.27	75.42	77.69	81.73
DSA	0.01	0.02	17.66	17.85	17.87	19.34

Table B.26: Results of misses cost on the Kosarark dataset for condition C2 (S2)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.79	1.00	5.16	5.16	6.29	9.16
R. Robin	1.78	2.98	3.72	4.05	4.34	5.27
Random	1.85	3.18	3.92	4.27	4.40	5.43
SWA	1.80	3.02	3.70	4.11	4.75	5.64
Algo2a	1.71	4.69	6.08	6.64	7.81	10.04
DSA	0.29	0.40	0.47	0.47	0.47	0.47

Table B.27: Results of misses cost on the Retail dataset for condition C2 (S2)

Algorithm	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	29.18	48.05	65.92	48.18	48.21	46.96
R. Robin	48.32	56.24	65.34	67.11	64.24	64.10
Random	48.19	55.37	63.65	64.75	62.62	62.95
SWA	48.58	51.38	63.10	64.52	65.13	64.46
Algo2a	43.97	62.20	64.50	65.72	66.27	66.81
DSA	27.02	27.02	36.99	37.11	37.12	37.34

Table B.28: Results of misses cost on the Reuters dataset for condition C2 (S2)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	6.95	11.12	12.02	15.61	12.50	15.36
R. Robin	20.33	28.43	31.92	34.50	37.62	40.27
Random	20.82	29.31	31.30	32.25	37.81	40.57
SWA	19.86	20.52	20.93	25.09	26.38	32.84
Algo2a	4.20	8.13	8.26	20.47	32.02	42.25
DSA	1.09	1.39	1.60	3.41	3.67	5.68

Table B.29: Results of misses cost on the BMS-1 dataset for condition C2 (S2)

Algorithm	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	15.94	48.69	47.51	51.29	61.36	62.11
R. Robin	25.65	56.13	60.81	71.69	70.56	72.34
Random	25.67	56.21	60.79	71.63	70.59	72.36
SWA	25.49	53.16	46.51	65.39	71.77	72.70
Algo2a	33.36	48.69	47.53	51.29	58.44	62.58
DSA	4.87	11.56	18.58	24.06	24.55	24.02

Table B.30: Results of misses cost on the Kosarak dataset for condition C2 (S3)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	6.66	56.16	51.68	66.87	66.18	66.31
R. Robin	34.03	43.84	55.04	63.54	58.37	58.81
Random	33.66	43.15	55.04	63.68	58.23	58.27
SWA	33.22	44.30	55.57	64.82	65.42	65.29
Algo2a	44.66	56.16	77.06	81.99	82.30	82.43
DSA	41.37	41.70	45.81	45.79	45.97	46.03

Table B.31: Results of misses cost on the Retail dataset for condition C2 (S3)

Algorithm	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.41	45.61	65.28	71.36	67.15	67.10
R. Robin	14.56	48.21	78.92	85.07	85.91	87.07
Random	12.38	47.98	78.69	85.34	86.19	87.30
SWA	12.38	46.09	78.48	75.11	75.55	75.22
Algo2a	23.43	53.77	75.71	77.39	77.29	77.32
DSA	0.02	2.83	21.69	20.13	30.72	35.85

Table B.32: Results of misses cost on the Reuters dataset for condition C2 (S3)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	1.44	2.90	11.19	25.15	27.93	28.01
R. Robin	9.52	14.54	17.89	45.84	42.39	53.81
Random	10.18	14.04	18.62	43.24	42.18	53.39
SWA	10.04	13.40	17.53	41.91	38.71	49.80
Algo2a	9.27	12.82	13.61	34.72	40.03	46.57
DSA	0.41	2.84	5.32	6.88	6.39	8.17

Table B.33: Results of misses cost on the BMS-1 dataset for condition C2 (S3)

Algorithm	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	1.77	1.77	20.91	26.65	28.91	29.88
R. Robin	23.32	23.62	29.82	35.15	36.05	36.54
Random	23.14	23.43	29.55	35.08	35.91	36.52
SWA	22.98	23.48	29.33	34.74	37.35	37.97
Algo2a	15.53	15.80	26.47	32.53	34.98	36.53
DSA	3.88	3.88	4.49	6.30	8.04	8.05

Table B.34: Results of misses cost on the Kosarak dataset for condition C2 (S4)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.01	2.18	2.20	2.21	2.24	3.11
R. Robin	0.39	1.05	1.46	1.51	1.27	2.96
Random	0.30	0.82	1.25	1.32	1.32	2.91
SWA	0.46	1.03	1.21	1.24	1.29	3.15
Algo2a	0.47	1.40	2.20	2.24	2.28	3.97
DSA	0.01	8.93	8.95	8.95	8.95	9.30

Table B.35: Results of misses cost on the Retail dataset for condition C2 (S4)

Algorithm	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.01	0.99	1.00	41.92	39.47	45.00
R. Robin	0.44	3.19	3.49	46.09	42.13	44.72
Random	0.49	3.14	3.32	46.83	42.20	44.91
SWA	0.35	3.10	3.35	44.53	44.32	47.32
Algo2a	0.61	4.12	4.50	42.64	43.08	45.59
DSA	0.01	27.68	27.69	39.30	39.69	51.81

Table B.36: Results of misses cost on the Reuters dataset for condition C2 (S4)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.01	0.05	0.20	0.20	0.47	0.47
R. Robin	0.27	8.93	18.33	18.38	19.39	19.37
Random	0.19	8.81	20.01	19.90	21.12	21.04
SWA	0.23	9.33	18.01	19.48	18.58	20.63
Algo2a	0.35	9.82	21.97	21.98	22.57	22.58
DSA	0.01	0.01	0.07	0.07	0.15	0.15

Table B.37: Results of misses cost on the BMS-1 dataset for condition C2 (S4)

B.3.3 Condition C3 (Varying the Minimum Support Threshold)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.1\%$	$\sigma = 0.15\%$	$\sigma = 0.2\%$	$\sigma = 0.25\%$	$\sigma = 0.3\%$
IGA	31.27	50.12	18.95	5.85	2.06
R. Robin	61.34	88.36	87.74	54.44	29.00
Random	61.64	88.43	87.79	54.47	29.19
SWA	61.30	88.30	87.72	54.14	29.15
Algo2a	53.25	51.54	48.91	38.07	25.86

Table B.38: Effect of σ on misses cost in the Kosarak dataset for condition C3 (S1)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.08\%$	$\sigma = 0.09\%$	$\sigma = 0.1\%$	$\sigma = 0.11\%$	$\sigma = 0.12\%$
IGA	0.84	0.91	1.03	1.12	1.09
R. Robin	2.73	2.43	2.66	2.93	2.93
Random	3.04	2.71	2.72	2.99	3.28
SWA	2.81	2.58	2.77	2.80	3.00
Algo2a	4.87	4.96	4.94	4.83	5.24

Table B.39: Effect of σ on misses cost in the Retail dataset for condition C3 (S1)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 4.5\%$	$\sigma = 5.0\%$	$\sigma = 5.5\%$	$\sigma = 6.0\%$	$\sigma = 6.5\%$
IGA	51.77	48.33	45.67	43.51	41.60
R. Robin	67.46	65.38	64.54	67.62	67.75
Random	66.95	64.91	64.59	67.28	67.64
SWA	66.54	64.72	64.50	67.26	67.75
Algo2a	48.29	47.19	47.33	46.91	44.48

Table B.40: Effect of σ on misses cost in the Reuters dataset for condition C3 (S1)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.08\%$	$\sigma = 0.09\%$	$\sigma = 0.1\%$	$\sigma = 0.11\%$	$\sigma = 0.12\%$
IGA	30.49	22.30	21.73	18.94	18.50
R. Robin	57.73	47.26	41.21	39.47	37.79
Random	59.36	48.78	43.24	40.31	37.49
SWA	56.90	88.05	40.77	38.94	37.85
Algo2a	30.69	26.75	24.67	25.25	23.83

Table B.41: Effect of σ on misses cost in the BMS-1 dataset for condition C3 (S1)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.1\%$	$\sigma = 0.15\%$	$\sigma = 2.0\%$	$\sigma = 2.5\%$	$\sigma = 3.0\%$
IGA	45.50	53.11	57.29	42.37	28.56
R. Robin	61.48	84.07	83.62	61.91	42.21
Random	61.84	84.61	84.16	62.04	42.78
SWA	62.08	84.49	84.03	61.90	42.49
Algo2a	66.86	78.99	81.59	60.27	44.83

Table B.42: Effect of σ on misses cost in the Kosarak dataset for condition C3 (S2)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.08\%$	$\sigma = 0.09\%$	$\sigma = 0.1\%$	$\sigma = 0.11\%$	$\sigma = 0.12\%$
IGA	9.15	9.19	9.16	8.78	8.89
R. Robin	6.14	6.22	5.83	5.26	5.23
Random	6.33	6.09	5.98	5.23	5.32
SWA	6.03	6.06	5.64	5.10	4.87
Algo2a	10.07	10.20	10.04	9.45	9.31

Table B.43: Effect of σ on misses cost in the Retail dataset for condition C3 (S2)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
Algorithm	$\sigma = 4.5\%$	$\sigma = 5.0\%$	$\sigma = 5.5\%$	$\sigma = 6.0\%$	$\sigma = 6.5\%$
IGA	46.04	45.17	46.16	46.09	45.08
R. Robin	70.89	68.38	67.48	70.20	70.76
Random	69.15	67.16	66.05	68.63	69.39
SWA	65.69	64.06	64.46	66.28	66.21
Algo2a	66.49	65.73	66.69	67.65	67.44

Table B.44: Effect of σ on misses cost in the Reuters dataset for condition C3 (S2)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
Algorithm	$\sigma = 0.08\%$	$\sigma = 0.09\%$	$\sigma = 0.1\%$	$\sigma = 0.11\%$	$\sigma = 0.12\%$
IGA	22.53	17.44	15.36	14.70	15.73
R. Robin	58.64	48.15	41.64	40.01	39.61
Random	58.89	48.22	42.79	40.93	38.50
SWA	45.94	85.76	32.84	31.73	31.86
Algo2a	52.11	45.84	42.25	41.80	39.19

Table B.45: Effect of σ on misses cost in the BMS-1 dataset for condition C3 (S2)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.1\%$	$\sigma = 0.15\%$	$\sigma = 2.0\%$	$\sigma = 2.5\%$	$\sigma = 3.0\%$
IGA	55.71	52.39	52.04	57.37	62.11
R. Robin	65.04	63.81	63.00	73.31	74.41
Random	65.07	63.89	63.02	73.32	74.37
SWA	64.38	62.06	65.55	71.85	72.69
Algo2a	51.28	52.29	51.97	58.06	62.49

Table B.46: Effect of σ on misses cost in the Kosarak dataset for condition C3 (S3)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.08\%$	$\sigma = 0.09\%$	$\sigma = 0.1\%$	$\sigma = 0.11\%$	$\sigma = 0.12\%$
IGA	65.65	66.20	66.31	67.10	67.51
R. Robin	62.70	63.43	64.03	64.60	65.04
Random	63.09	63.86	64.27	64.67	65.04
SWA	63.43	64.39	65.28	65.41	65.76
Algo2a	80.60	81.48	81.96	82.73	83.13

Table B.47: Effect of σ on misses cost in the Retail dataset for condition C3 (S3)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
Algorithm	$\sigma = 4.5\%$	$\sigma = 5.0\%$	$\sigma = 5.5\%$	$\sigma = 6.0\%$	$\sigma = 6.5\%$
IGA	62.63	63.14	67.10	66.33	66.05
R. Robin	88.68	88.50	88.94	90.41	90.35
Random	88.74	88.45	89.03	90.69	90.41
SWA	75.04	74.06	75.22	77.43	78.73
Algo2a	75.14	76.52	77.11	78.78	77.48

Table B.48: Effect of σ on misses cost in the Reuters dataset for condition C3 (S3)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
Algorithm	$\sigma = 0.08\%$	$\sigma = 0.09\%$	$\sigma = 0.1\%$	$\sigma = 0.11\%$	$\sigma = 0.12\%$
IGA	31.52	28.67	28.00	30.05	32.34
R. Robin	57.64	53.67	54.10	57.77	58.46
Random	56.58	52.51	51.81	56.28	57.20
SWA	54.67	88.85	49.80	53.02	54.01
Algo2a	45.97	45.02	47.21	50.60	51.11

Table B.49: Effect of σ on misses cost in the BMS-1 dataset for condition C3 (S3)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.1\%$	$\sigma = 0.15\%$	$\sigma = 2.0\%$	$\sigma = 2.5\%$	$\sigma = 3.0\%$
IGA	46.64	62.38	53.32	39.24	29.88
R. Robin	67.42	92.81	89.27	61.23	37.93
Random	67.68	92.79	89.24	61.11	37.71
SWA	68.07	92.59	89.28	60.62	37.97
Algo2a	65.27	82.32	79.31	55.27	36.30

Table B.50: Effect of σ on misses cost in the Kosarak dataset for condition C3 (S4)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
	$\sigma = 0.08\%$	$\sigma = 0.09\%$	$\sigma = 0.1\%$	$\sigma = 0.11\%$	$\sigma = 0.12\%$
IGA	3.34	3.29	3.11	2.90	3.16
R. Robin	3.76	3.69	3.24	2.88	2.77
Random	3.59	3.40	3.07	2.68	2.63
SWA	3.77	3.59	3.15	2.78	2.72
Algo2a	4.34	4.25	3.97	3.63	3.82

Table B.51: Effect of σ on misses cost in the Retail dataset for condition C3 (S4)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
Algorithm	$\sigma = 4.5\%$	$\sigma = 5.0\%$	$\sigma = 5.5\%$	$\sigma = 6.0\%$	$\sigma = 6.5\%$
IGA	45.59	43.98	45.00	45.19	43.46
R. Robin	55.14	50.63	49.13	53.79	54.82
Random	55.62	51.36	49.62	53.92	55.03
SWA	53.37	48.60	47.32	52.43	53.08
Algo2a	46.85	45.11	45.45	45.96	44.86

Table B.52: Effect of σ on misses cost in the Reuters dataset for condition C3 (S4)

Algorithm	$\psi = 0\%$, $\varphi = 60\%$, 6 sensitive rules				
Algorithm	$\sigma = 0.08\%$	$\sigma = 0.09\%$	$\sigma = 0.1\%$	$\sigma = 0.11\%$	$\sigma = 0.12\%$
IGA	2.27	0.94	0.47	0.63	0.30
R. Robin	39.00	25.85	19.58	16.88	13.91
Random	40.05	27.11	21.21	18.14	14.65
SWA	39.67	83.50	20.63	17.56	14.43
Algo2a	37.80	26.96	22.58	19.39	17.28

Table B.53: Effect of σ on misses cost in the BMS-1 dataset for condition C3 (S4)

B.4 Results of the Difference Between the Original and the Sanitized Datasets

B.4.1 Condition C1 (A set of 6 Sensitive Rules)

Kosarak	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	0.16	0.21	2.13	0.15
RRA	0.16	0.21	2.33	0.16
RA	0.16	0.21	2.33	0.16
SWA	0.16	0.20	2.05	0.15
Algo2a	0.16	0.21	2.13	0.16

Table B.54: Difference(D, D') for condition C1.

Retail	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	0.12	0.12	1.69	0.05
RRA	0.12	0.12	1.77	0.05
RA	0.12	0.12	1.78	0.05
SWA	0.12	0.12	1.66	0.05
Algo2a	0.12	0.12	1.74	0.05

Table B.55: Difference(D, D') for condition C1.

Reuters	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	0.56	0.52	0.85	0.54
RRA	0.55	0.52	1.00	0.53
RA	0.55	0.52	1.01	0.53
SWA	0.55	0.44	0.84	0.46
Algo2a	0.56	0.52	0.90	0.54

Table B.56: Difference(D, D') for condition C1.

BMS-1	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	0.22	0.13	0.88	0.13
RRA	0.22	0.14	0.99	0.14
RA	0.22	0.14	0.98	0.14
SWA	0.22	0.12	0.88	0.12
Algo2a	0.22	0.17	0.89	0.17

Table B.57: Difference(D, D') for condition C1.

B.4.2 Condition C2 (Varying the Number of Sensitive Rules)

Algorithm	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.03	0.05	0.08	0.11	0.14	0.16
R. Robin	0.03	0.05	0.08	0.11	0.12	0.14
Random	0.03	0.05	0.08	0.11	0.12	0.14
SWA	0.03	0.05	0.08	0.11	0.14	0.16
Algo2a	0.03	0.05	0.08	0.11	0.14	0.16

Table B.58: Difference(D, D') on the Kosarark dataset for condition C2 (S1)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.07	0.08	0.08	0.09	0.10	0.12
R. Robin	0.07	0.08	0.08	0.09	0.09	0.10
Random	0.07	0.08	0.08	0.09	0.09	0.10
SWA	0.07	0.08	0.08	0.09	0.10	0.12
Algo2a	0.07	0.08	0.08	0.09	0.10	0.12

Table B.59: Difference(D, D') on the Retail dataset for condition C2 (S1)

Algorithm	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.08	0.17	0.27	0.38	0.46	0.56
R. Robin	0.08	0.17	0.27	0.38	0.41	0.50
Random	0.08	0.17	0.27	0.38	0.41	0.50
SWA	0.08	0.17	0.27	0.38	0.46	0.56
Algo2a	0.08	0.17	0.27	0.38	0.46	0.56

Table B.60: Difference(D, D') on the Reuters dataset for condition C2 (S1)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.04	0.07	0.11	0.15	0.19	0.22
R. Robin	0.04	0.08	0.11	0.15	0.17	0.20
Random	0.04	0.08	0.11	0.15	0.17	0.20
SWA	0.04	0.07	0.11	0.15	0.19	0.22
Algo2a	0.04	0.07	0.11	0.15	0.19	0.22

Table B.61: Difference(D, D') on the BMS-1 dataset for condition C2 (S1)

Algorithm	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.03	0.07	0.11	0.15	0.19	0.21
R. Robin	0.03	0.07	0.11	0.15	0.17	0.19
Random	0.03	0.07	0.11	0.15	0.17	0.19
SWA	0.03	0.07	0.11	0.15	0.18	0.20
Algo2a	0.03	0.07	0.11	0.15	0.18	0.21

Table B.62: Difference(D, D') on the Kosarark dataset for condition C2 (S2)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.01	0.04	0.05	0.07	0.08	0.12
R. Robin	0.01	0.04	0.05	0.07	0.08	0.11
Random	0.01	0.04	0.05	0.07	0.08	0.11
SWA	0.01	0.04	0.05	0.07	0.08	0.12
Algo2a	0.01	0.04	0.05	0.07	0.08	0.12

Table B.63: Difference(D, D') on the Retail dataset for condition C2 (S2)

Algorithm	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.12	0.17	0.29	0.34	0.42	0.52
R. Robin	0.12	0.21	0.28	0.37	0.40	0.46
Random	0.12	0.21	0.28	0.37	0.40	0.46
SWA	0.12	0.17	0.24	0.33	0.41	0.44
Algo2a	0.12	0.22	0.26	0.34	0.42	0.52

Table B.64: Difference(D, D') on the Reuters dataset for condition C2 (S2)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.04	0.05	0.07	0.09	0.10	0.13
R. Robin	0.04	0.06	0.09	0.10	0.10	0.13
Random	0.04	0.06	0.09	0.10	0.10	0.13
SWA	0.04	0.05	0.07	0.09	0.09	0.12
Algo2a	0.04	0.06	0.09	0.11	0.14	0.17

Table B.65: Difference(D, D') on the BMS-1 dataset for condition C2 (S2)

Algorithm	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.72	1.34	1.40	1.63	1.95	2.13
R. Robin	0.72	1.43	1.69	1.96	1.88	2.03
Random	0.72	1.43	1.69	1.96	1.88	2.03
SWA	0.72	1.34	1.40	1.63	1.94	2.05
Algo2a	0.72	1.34	1.40	1.63	1.95	2.13

Table B.66: Difference(D, D') on the Kosarak dataset for condition C2 (S3)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.59	0.86	1.11	1.36	1.62	1.69
R. Robin	0.59	0.89	1.13	1.41	1.53	1.59
Random	0.59	0.89	1.13	1.41	1.53	1.59
SWA	0.59	0.86	1.08	1.31	1.59	1.66
Algo2a	0.59	0.86	1.12	1.36	1.66	1.74

Table B.67: Difference(D, D') on the Retail dataset for condition C2 (S3)

Algorithm	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.23	0.49	0.72	0.77	0.78	0.85
R. Robin	0.23	0.49	0.72	0.82	0.79	0.89
Random	0.23	0.49	0.72	0.82	0.79	0.89
SWA	0.23	0.49	0.72	0.75	0.77	0.84
Algo2a	0.23	0.49	0.72	0.78	0.83	0.90

Table B.68: Difference(D, D') on the Reuters dataset for condition C2 (S3)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.21	0.41	0.55	0.74	0.84	0.88
R. Robin	0.21	0.41	0.60	0.79	0.81	0.88
Random	0.21	0.41	0.59	0.78	0.81	0.87
SWA	0.21	0.41	0.55	0.74	0.84	0.88
Algo2a	0.21	0.41	0.55	0.74	0.84	0.89

Table B.69: Difference(D, D') on the BMS-1 dataset for condition C2 (S3)

Algorithm	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.03	0.05	0.07	0.11	0.13	0.15
R. Robin	0.03	0.05	0.08	0.11	0.12	0.15
Random	0.03	0.05	0.08	0.11	0.12	0.15
SWA	0.03	0.05	0.07	0.11	0.13	0.15
Algo2a	0.03	0.05	0.08	0.11	0.13	0.16

Table B.70: Difference(D, D') on the Kosarak dataset for condition C2 (S4)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.01	0.02	0.02	0.03	0.04	0.05
R. Robin	0.01	0.02	0.02	0.03	0.04	0.05
Random	0.01	0.02	0.02	0.03	0.04	0.05
SWA	0.01	0.02	0.02	0.03	0.04	0.05
Algo2a	0.01	0.02	0.02	0.03	0.04	0.05

Table B.71: Difference(D, D') on the Retail dataset for condition C2 (S4)

Algorithm	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.08	0.20	0.29	0.38	0.38	0.54
R. Robin	0.08	0.20	0.29	0.38	0.40	0.48
Random	0.08	0.20	0.29	0.38	0.40	0.47
SWA	0.08	0.20	0.29	0.38	0.38	0.46
Algo2a	0.0.8	0.20	0.29	0.38	0.46	0.54

Table B.72: Difference(D, D') on the Reuters dataset for condition C2 (S4)

Algorithm	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$					
	1 Rule	2 Rules	3 Rules	4 Rules	5 Rules	6 Rules
IGA	0.02	0.05	0.07	0.10	0.13	0.16
R. Robin	0.02	0.05	0.07	0.10	0.12	0.15
Random	0.02	0.05	0.07	0.10	0.12	0.15
SWA	0.02	0.05	0.07	0.10	0.13	0.16
Algo2a	0.02	0.05	0.07	0.10	0.13	0.16

Table B.73: Difference(D, D') on the BMS-1 dataset for condition C2 (S4)

B.4.3 Condition C3 (Varying the Minimum Support Threshold)

Algorithm	$\psi = 0\%$, $\sigma =$ [0.1 - 0.3], $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	0.16	0.21	2.13	0.15
R. Robin	0.16	0.21	2.33	0.16
Random	0.16	0.21	2.33	0.16
SWA	0.16	0.20	2.05	0.15
Algo2a	0.16	0.21	2.12	0.16

Table B.74: Difference(D, D') on the Kosarak dataset for condition C3

Algorithm	$\psi = 0\%$, $\sigma =$ [0.08 - 0.12], $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	0.12	0.12	1.69	0.05
R. Robin	0.12	0.12	1.77	0.05
Random	0.12	0.12	1.77	0.05
SWA	0.12	0.12	1.66	0.05
Algo2a	0.12	0.12	1.73	0.05

Table B.75: Difference(D, D') on the Retail dataset for condition C3

Algorithm	$\psi = 0\%$, $\sigma =$ [4.5 - 6.5], $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	0.56	0.52	0.85	0.54
R. Robin	0.56	0.52	1.01	0.53
Random	0.56	0.52	1.01	0.53
SWA	0.56	0.44	0.84	0.46
Algo2a	0.56	0.52	0.89	0.54

Table B.76: Difference(D, D') on the Reuters dataset for condition C3

Algorithm	$\psi = 0\%$, $\sigma =$ [0.08 - 0.12], $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	0.22	0.13	0.88	0.16
R. Robin	0.22	0.14	0.99	0.16
Random	0.22	0.14	0.98	0.16
SWA	0.22	0.12	0.88	0.16
Algo2a	0.22	0.17	0.89	0.16

Table B.77: Difference(D, D') on the BMS-1 dataset for condition C3

B.4.4 Effect of ψ on MC and HF (Rules in Scenario S3)

Algorithm	$\psi = 0\%$		$\psi = 5\%$		$\psi = 10\%$		$\psi = 15\%$		$\psi = 25\%$	
	MC	HF	MC	HF	MC	HF	MC	HF	MC	HF
IGA	62.11	0.00	61.85	0.00	61.66	0.08	61.38	0.08	60.33	0.24
R. Robin	74.42	0.00	73.42	0.00	72.32	0.00	70.94	0.00	67,70	0.12
Random	74.37	0.00	73.32	0.00	72.36	0.00	70.87	0.00	67.73	0.00
SWA	72.70	0.00	67.03	0.00	59.56	0.75	53.06	3.81	39.87	17.83

Table B.78: Effect of ψ on misses cost and hiding failure in the Kosarak dataset

Algorithm	$\psi = 0\%$		$\psi = 5\%$		$\psi = 10\%$		$\psi = 15\%$		$\psi = 25\%$	
	MC	HF	MC	HF	MC	HF	MC	HF	MC	HF
IGA	66.31	0.00	64.77	0.66	63.23	0.83	60.94	1.32	56.26	1.99
R. Robin	64.02	0.00	61.18	7.28	58.15	6.46	55.12	7.62	46.46	15.73
Random	63.86	0.00	60.12	7.12	56.72	7.62	54.39	7.95	46.48	16.39
SWA	65.29	0.00	55.58	1.16	48.31	1.82	42.67	3.31	27.74	15.89

Table B.79: Effect of ψ on misses cost and hiding failure in the Retail dataset

Algorithm	$\psi = 0\%$		$\psi = 5\%$		$\psi = 10\%$		$\psi = 15\%$		$\psi = 25\%$	
	MC	HF	MC	HF	MC	HF	MC	HF	MC	HF
IGA	67.10	0.00	67.04	0.00	66.03	0.00	66.00	0.00	65.70	0.00
R. Robin	89.00	0.00	87.85	0.00	86.13	0.00	83.95	0.00	78.65	0.00
Random	89.03	0.00	87.30	0.00	85.89	0.00	83.12	0.00	78.86	0.00
SWA	75.22	0.00	75.05	0.00	73.67	0.00	71.34	0.00	66.62	0.00

Table B.80: Effect of ψ on misses cost and hiding failure in the Reuters dataset

Algorithm	$\psi = 0\%$		$\psi = 5\%$		$\psi = 10\%$		$\psi = 15\%$		$\psi = 25\%$	
	MC	HF	MC	HF	MC	HF	MC	HF	MC	HF
IGA	28.01	0.00	28.00	0.00	28.01	0.00	28.01	0.00	27.95	0.16
R. Robin	53.13	0.00	48.69	0.00	47.29	0.00	43.52	0.00	32.15	9.67
Random	50.35	0.00	47.30	0.00	46.63	0.00	42.63	0.00	33.21	10.45
SWA	49.80	0.00	44.60	0.00	39.34	0.00	34.05	0.00	10.30	11.54

Table B.81: Effect of ψ on misses cost and hiding failure in the BMS-1 dataset

B.5 CPU Time - The Kosarak Dataset

Algorithm	20 Rules	40 Rules	60 Rules	80 Rules	100 Rules
IGA	16.752	25.383	34.810	42.269	49.361
R. Robin	41.840	106.081	248.589	392.972	585.821
Random	41.784	105.979	248.555	391.882	593.128
SWA	75.875	118.013	181.294	227.537	279.167
Algo2a	137.652	276.974	501.834	792.876	1.164.714
DSA	0.246	0.296	0.372	0.543	0.579

Table B.82: CPU time for sanitization varying the number of sensitive rules (msec.)

Algorithm	150K	300K	450K	600K	750K	900K
IGA	0.436	1.229	2.074	2.861	2.698	3875
R. Robin	0.547	1.136	2.544	3.323	4.437	5.664
Random	0.549	1.347	2.698	3.125	4.428	5.627
SWA	4.976	9.955	14.645	19.986	24.271	30.046
Algo2a	5.801	11.671	17.481	23.569	28.649	35.894
DSA	0.170	0.151	0.132	0.137	0.156	0.173

Table B.83: CPU time for sanitization varying the dataset size (msec.)

B.6 Results of Side Effect Factor on the Datasets

Kosarak	$\psi = 0\%$, $\sigma = 0.2\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	2.17	28.94	69.16	29.93
DSA	5.52	17.32	38.15	8.11

Table B.84: Results of side effect factor on the dataset Kosarak.

Retail	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	1.64	9.40	69.05	3.33
DSA	0.82	0.79	50.44	9.51

Table B.85: Results of side effect factor on the dataset Retail.

Reuters	$\psi = 0\%$, $\sigma = 5.5\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	45.26	47.63	71.10	45.12
DSA	33.59	37.91	43.64	51.91

Table B.86: Results of side effect factor on the dataset Reuters.

BMS-1	$\psi = 0\%$, $\sigma = 0.1\%$, $\varphi = 60\%$			
	S1	S2	S3	S4
IGA	21.85	15.43	29.80	15.94
DSA	7.20	5.75	10.46	0.16

Table B.87: Results of side effect factor on the dataset BMS-1.

Appendix C

Results of the Effectiveness of the DRBT

C.1 Results of the Stress Function Applied to the Datasets

Chess	$d_r = 37$	$d_r = 34$	$d_r = 31$	$d_r = 28$	$d_r = 25$	$d_r = 22$	$d_r = 16$
RP_1	0.000	0.015	0.024	0.033	0.045	0.072	0.141
RP_2	0.000	0.014	0.019	0.032	0.041	0.067	0.131

Table C.1: The error produced on the *Chess* dataset ($d_o = 37$).

Mushroom	$d_r = 23$	$d_r = 21$	$d_r = 19$	$d_r = 17$	$d_r = 15$	$d_r = 13$	$d_r = 9$
RP_1	0.000	0.020	0.031	0.035	0.048	0.078	0.155
RP_2	0.000	0.017	0.028	0.029	0.040	0.079	0.137

Table C.2: The error produced on the *Mushroom* dataset ($d_o = 23$).

Pumsb	$d_r = 74$	$d_r = 69$	$d_r = 64$	$d_r = 59$	$d_r = 49$	$d_r = 39$	$d_r = 29$
RP_1	0.000	0.006	0.022	0.029	0.049	0.078	0.157
RP_2	0.000	0.007	0.030	0.030	0.032	0.060	0.108

Table C.3: The error produced on the *Pumsb* dataset ($d_o = 74$).

Connect	$d_r = 43$	$d_r = 37$	$d_r = 31$	$d_r = 25$	$d_r = 19$	$d_r = 16$	$d_r = 13$
RP_1	0.000	0.016	0.037	0.063	0.141	0.159	0.219
RP_2	0.000	0.016	0.028	0.062	0.122	0.149	0.212

Table C.4: The error produced on the *Connect* dataset ($d_o = 43$).

Accidents	$d_r = 18$	$d_r = 16$	$d_r = 14$	$d_r = 12$	$d_r = 10$	$d_r = 8$	$d_r = 6$
RP_1	0.000	0.033	0.034	0.044	0.094	0.144	0.273
RP_2	0.000	0.018	0.023	0.036	0.057	0.108	0.209

Table C.5: The error produced on the *Accidents* dataset ($d_o = 18$).

C.2 Results of F-measure for the Clusters Mined from the Transformed Datasets

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.529	0.873	0.805	0.143	0.592	0.752	0.735	0.050

Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.597	0.770	0.695	0.063	0.569	0.761	0.665	0.060

Table C.6: Average of F-measure (10 trials) for the Chess dataset ($d_o = 37, d_r = 25$).

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.972	0.975	0.974	0.001	0.689	0.960	0.781	0.105

Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.727	0.864	0.811	0.058	0.747	0.884	0.824	0.051

Table C.7: Average of F-measure (10 trials) for the Mushroom dataset ($d_o = 23, d_r = 15$).

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.611	0.994	0.909	0.140	0.735	0.991	0.965	0.081

Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.846	0.925	0.891	0.028	0.765	0.992	0.838	0.041

Table C.8: Average of F-measure (10 trials) for the Pumsb dataset ($d_o = 74, d_r = 38$).

Data Transformation	k = 2				k = 3			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.596	0.863	0.734	0.066	0.486	0.863	0.623	0.103

Data Transformation	k = 4				k = 5			
	Min	Max	Avg	Std	Min	Max	Avg	Std
RP_2	0.618	0.819	0.687	0.069	0.572	0.763	0.669	0.069

Table C.9: Average of F-measure (10 trials) for the Connect dataset ($d_o = 43, d_r = 28$).