International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2017 IJSRCSEIT | Volume 2 | Issue 5 | ISSN : 2456-3307

Privacy-Preserving Distributed Mining Technique of Association Rules on Horizontally Partitioned Data

P. Kavitha, M.C.A., M.Phil
Assistant Professor, Department of Computer Science, Arulmigu Palaniandavar Arts College For Women, Palani, Tamil Nadu, India

ABSTRACT

Huge volume of detailed personal data is regularly collected and sharing of these data is proved to be beneficial for data mining application. Such data include shopping habits, criminal records, medical history, credit records etc. On one hand such data is an important asset to business organization and governments for decision making by analyzing it. On the other hand privacy regulations and other privacy concerns may prevent data owners from sharing information for data analysis. In order to share data while preserving privacy data owner must come up with a solution which achieves the dual goal of privacy preservation as well as accurate clustering result. 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 dissertation, 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. 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.

Keywords : Data Mining, Association Rile Mining, Data transformation.

I. INTRODUCTION

The sharing of association rules is often beneficial in industry, but requires privacy safe-guards. 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 re-main 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. We describe some scenarios that demonstrate the need for techniques to protect collective privacy (e.g., sensitive knowledge) in association rule mining. 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. We introduce 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 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. The sanitization removes not only all sensitive

patterns but also blocks other patterns that could be used to infer the sensitive hidden ones.

II. MOTIVATION FOR PRIVACY-PRESERVING ASSOCIATION RULE MINING

Today, collaboration has become prevalent in the competitive commercial world since it brings mutual benefits. 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. Two companies have a very large dataset of records of their customer's 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 association's 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.

III. 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-1, the framework encompasses an inverted le 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.

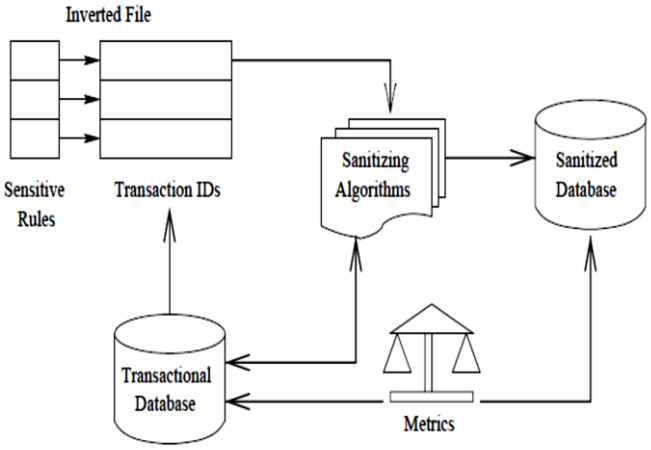

Figure-1: The sketch of the framework for privacy-preserving association rule mining.

3.3.1 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). 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 -2(b) shows an example of an inverted le corresponding to the sample transactional database shown in Figure-2(a). For this example, we assume that the sensitive rules are A,B → D and A,C →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

Figure-2(a): A sample transactional database

Inverted file

A,B → D	→	T1, T2
A,C → D	→	T1, T4
Sensitive Rules		Transaction IDs

Figure-2(b): The corresponding inverted file

Note that once the inverted file is built, a data owner will sanitize only the sensitive transactions who's 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.

3.3.2 Sanitizing Algorithms

In our framework, the sanitizing algorithms modify some transactions to hide sensitive rules based on a disclosure threshold 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 ψ = 50% then half of the sensitive transactions will be sanitized, when ψ = 0% all the sensitive transaction will be sanitized, and when ψ = 100% no sensitive transaction will be sanitized. In other words, 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. We classify our algorithms into two major groups: data sharing-based algorithms and pattern sharing-based algorithms.

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

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.

3.3.3 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: (a) Data sharing-based

metrics and (b) Pattern sharing-based metrics.

a) Data sharing-based metrics are related to the problems illustrated in Figure 4.3. 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 $\square S_R$, as well as the set R' of 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.

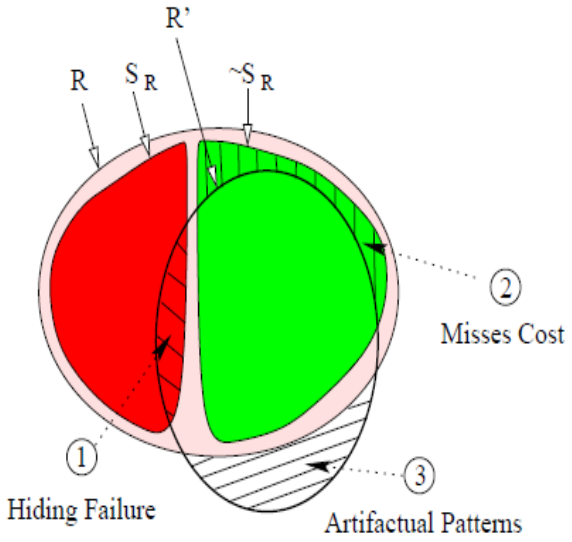

Figure-3: Data sharing-based sanitization problems.

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:

$$HF = \frac{\#S_R(D')}{\#S_R(D)} \qquad (4.1)$$

where $\#S_R(X)$ denotes the number of non-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:

$$MC = \frac{\#\square S_R(D) - \#\square S_R(D')}{\#\square S_R(D)}$$

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→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:

$$AP = \frac{|R'| - |R \cap R'|}{|R'|} \qquad (4.3)$$

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:

$$Dif(D, D') = \frac{1}{\sum_{i=1}^{n} f_D(i)} \times \sum_{i=1}^{n} [f_D(i) - f_{D'}(i)] \quad (4.4)$$

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-4 Problem 1 conveys the non-sensitive rules (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.

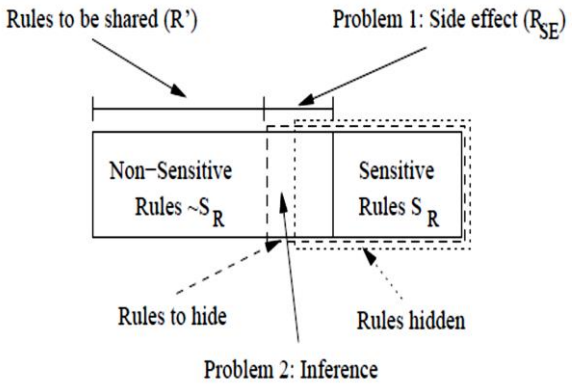

Figure-4: Pattern sharing-based sanitization problems.

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:

$$SEF = \frac{(|R| - (|R'| + |S_R|))}{(|R| - |S_R|)} \quad (4.5)$$

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.

IV. DATA SHARING-BASED SANITIZING ALGORITHMS

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

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

The optimal sanitization is an NP-hard problem. To alleviate the complexity of the optimal sanitization, we could use some heuristics. A heuristic does not guarantee the optimal solution, but usually finds a solution close to the best one in a faster response time. Our first heuristic for data sanitization is based on the fact that, in many cases, a sensitive transaction 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:

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 $t \varepsilon S_T$, is defined as the number of sensitive association rules that can be found in t.

Round Robin, Random, and Item Grouping sanitizing algorithms is presented which 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 controlled by the database owner, while the output is the sanitized database D'.

4.4.2 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, the IGA builds the inverted index, and the IGA computes the frequencies of all items in the database D.

The goal of step 4 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, 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. 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. However, "ABC" could also be in another group if there was one where sensitive rules shared "C", 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:

Item Grouping Algorithm
Input D, S_R, ψ
Output: D'.
Step-1: begin
Step-2: Identifying sensitive transactions and building

index T

Step-3: foreach transaction t ε D do

For k = 1 to size(t) do

$Sup(item_k, D) \leftarrow Sup(item_k, D) + 1$;

Sort the items in t is alphabetic order;

Step-4: for each sensitive association rule $sr_i \varepsilon S_R$ 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-5: Selecting the number of sensitive transactions

For each sensitive association rule $sr_i \varepsilon S_R$ do

Sort the vector $T[sr_i].tid\_list$ in descending order of degree;

$NumbTrans_{sri} \leftarrow [T[sr_i]] \times (1 - \psi)$;

$|T[sr_i]|$ is the number of sensitive transactions for sr_i

end

Step-6: Identifying victim items for each sensitive transaction

Group sensitive rules in a set of groups GP such that $\forall$ G ε GP,

$\forall sr_i, sr_j \varepsilon G$, sr_i and sr_j share the same itemset I. Give the class label

α to G such that $\alpha \varepsilon I$ and $\forall \beta \varepsilon I$, $sup(\alpha, D) \leq sup(\beta, D)$;

Step-7: Order the groups in GP by size in terms of number of sensitive rules in the group;

Step-8: for all $sr_k \varepsilon G_i \cap G_j$ 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 α such that $sup(\alpha, D) \geq sup(\beta, D)$

and α, β are class labels of either G_i, G_j;

end

end

Step-9: for each sensitive association rule $sr_i \varepsilon S_R$ do

for j = 1 to $NumbTrans_{sri}$ do

ChosenItem← α such that α is the class label of G and $sr_i \varepsilon G$;

$Victims[T[sr_i,j]].item\_list \leftarrow Victims[T[sr_i,j]].item\_list \cup ChosenItem$;

end

end

Step-10: $D' \leftarrow D$

Sort the vector Victims in ascending order of t_{ID};

$j \leftarrow 1$

foreach transaction t ε D do

if $t_{ID} == Victims[j].t_{ID}$ then

$t \leftarrow (t - Victims[j].item\_list)$;

$j \leftarrow j + 1$;

end

end

end

Let us consider the sample transactional database. 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 $\{T_1, T_3, T_4\}$. The degrees of the transactions T_1, T_3 and T_4 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 T_1 and T_4.

Step 2: Suppose that we set the disclosure threshold 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 T_1 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 T_1 the sensitive rules will be hidden from T_1 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.

TID	Items
T_1	B C D
T_2	A B C

T_3	A B D
T_4	A C D
T_5	A B C
T_6	B D

Figure-5(a): The sanitized databases using the Round Robin algorithm

TID	Items
T_1	B D
T_2	A B C
T_3	A B D
T_4	A C D
T_5	A B C
T_6	B D

Figure-5(b): The sanitized databases using the Random algorithm

TID	Items
T_1	A B C
T_2	A B C
T_3	A B D
T_4	A C D
T_5	A B C
T_6	B D

Figure-5(c): The sanitized databases using the Item Grouping algorithm

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.

V. CONCLUSION

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. 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. 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.

VI. REFERENCES

[1]. C. Clifton, M. Kantarciofiglu, J. Vaidya, X. Lin, and M. Y. Zhu. Tools For Privacy Preserving Distributed Data Mining. SIGKDD Explorations, 4(2):28-34, 2002.

[2]. 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, 1987.

[3]. 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.

[4]. B. Pinkas. Cryptographic Techniques For Privacy-Preserving Data Mining. SIGKDD Explorations, 4(2):12-19, December 2002.

[5]. S. Goldwasser. Multi-party Computations: Past and Present. In Proc. of the 16th Annual ACM Symposium on Principles of Distributed Computing, pages 1-6, SantaBarbara, CA, August 1997.

[6]. 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.

[7]. 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.

[8]. D. Chaum, C. Crepeau, 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.

[9]. Y. Lindell and B. Pinkas. Privacy Preserving Data Mining. In Crypto 2000, Springer-Verlag (LNCS 1880), pages 36-54, Santa Barbara, CA, August 2000.

[10]. 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.

[11]. M. Kantarcioglu 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.

[12]. 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.

[13]. 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.

[14]. M. Kantarcioglu and J. Vaidya. Privacy Preserving Naive 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.