Design a Weight Based Sorting Distortion Algorithm for Privacy Preserving Data Mining

R. Sugumar, A. Rengarajan and C. Jayakumar

1Department of Computer Science and Engineering, Velammal Institute of Technology, Panchetti, Thiruvallur District, Tamilnadu, India
2Department of Computer Science and Engineering, Veltech Multitech Dr. Rangarajan Dr. Sakunthala Engineering College, Chennai, India
3Department of Computer Science and Engineering, RMK Engineering College, Kavarapettai, Thiruvallur District, Tamilnadu, India

Abstract: The security of the large database that contains certain crucial information, it will become a serious issue when sharing data to the network against unauthorized access. Privacy preserving data mining is a new research trend in privacy data for data mining and statistical database. Association analysis is a powerful tool for discovering relationships which are hidden in large database. Association rules hiding algorithms get strong an efficient performance for protecting confidential and crucial data. Data modification and rule hiding is one of the most important approaches for secure data. The objective of the proposed Weight Based Sorting Distortion (WBSD) algorithm is to distort certain data which satisfies a particular sensitive rule. Then hide those transactions which support a sensitive rule and assigns them a priority and sorts them in ascending order according to the priority value of each rule. Then it uses these weights to compute the priority value for each transaction according to how weak the rule is that a transaction supports. Data distortion is one of the important methods to avoid this kind of scalability issues.

Key words: Association rule hiding · Apriori algorithm · Privacy Preserving Data mining · K-anonymity

INTRODUCTION

In recent years, large amount of data about individuals have become available with corporations as well as public entities. This has led to serious concerns about the misuse and privacy of such data. Privacy preserving data mining has become an important problem in recent years, because of the large amount of consumer data tracked by automated systems on the internet. In addition, advances in hardware technology have also made it feasible to track information about individuals from transactions in everyday life. For example, a simple transaction such as using the credit card results in automated storage of information about user buying behavior. In many cases, users are not willing to supply such personal data unless its privacy is guaranteed. Therefore, in order to ensure effective data collection, it is important to design methods which can mine the data with a guarantee of privacy. Another interesting method for privacy preserving data mining is the k-anonymity model. In the k-anonymity model, domain generalization hierarchies are used in order to transform and replace each record value with a corresponding generalized value. There are two ways to implement k-anonymity namely k-anonymity using generalization and k-anonymity using suppression. The Weight Based Sorting Distortion (WBSD) algorithm replaces the k-anonymity method in order to overcome high information loss when the data are hidden and retrieved back.

The rest of the paper is organized as follows. In Section 2 association rule hiding and the related works are discussed. Section 3 gives the general problem formulation and the basic definitions of association rule mining. In Section 4, Weight Based Sorting Distortion (WBSD) algorithm for sensitive item modification is given.
The effectiveness of the algorithm is evaluated and the experimental results of the proposed technique are discussed in Section 5. Conclusions are given in Section 6.

Related Work

K-anonymity: K-anonymization techniques have been the focus of intense research in the last few years. In order to ensure anonymization of data while at the same time minimizing the information loss resulting from data modifications, several extending models are proposed, which are discussed as follows. To protect individual s’identity when releasing sensitive information, data holders often encrypt or remove explicit identifiers, such as names and unique security numbers. However, unencrypted data provides no guarantee for anonymity. In order to preserve privacy, k-anonymity model has been proposed by Sweeney [1-8] which achieves k-anonymity using generalization and suppression [6]. In K -anonymity, it is difficult for an imposter to determine the identity of the individuals in collection of data set containing personal information. Each release of data contains every combination of values of quasi-identifiers and that is indistinctly matched to at least k -1 respondents. Generalization involves replacing a value with a less specific (generalized) but semantically reliable value. For example, the age of the person could be generalized to a range such as youth, middle age and adult without specifying appropriately, so as to reduce the risk of identification [6]. Suppression involves reduce the exactness of applications and it does not liberate any information. By using this method it reduces the risk of detecting exact information.

K-anonymity is one of the most classic models, which technique that prevents joining attacks by generalizing and/or suppressing portions of the released microdata so that no individual can be uniquely distinguished from a group of size k. In the k-anonymous tables, a data set is k-anonymous (k > 1) if each record in the data set is indistinguishable from at least (k -1) other records within the same data set [9-14].

Cryptographic Techniques: Another branch of privacy preserving data mining which using cryptographic techniques was developed. This branch became hugely popular for two main reasons: Firstly, cryptography offers a well-defined model for privacy, which includes methodologies for proving and quantifying it. Secondly, there exists a vast toolset of cryptographic algorithms and constructs to implement privacy preserving data mining algorithms. However, recent work [14] has pointed that cryptography does not protect the output of a computation. Instead, it prevents privacy leaks in the process of computation. Thus, it falls short of providing a complete answer to the problem of privacy preserving data mining.

In the context of PPDM over distributed data, cryptography based techniques have been developed to solve problem 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 (SMC) problem (Goldreich, Micali, & Wigderson, 1987). The technique proposed in (Lindell & Pinkas, 2000) address privacy-preserving classification, while the techniques proposed in (Kantarcioulu & Clifton, 2002; Vaidya & Clifton, 2002) address privacy-preserving association rule mining and the technique in (Vaidya & Clifton, 2003) addresses privacy preserving clustering.

Randomization Method: Randomization method is a popular method in current privacy preserving data mining studies. It masks the values of the records by adding noise to the original data. The noise added is sufficiently large so that the individual values of the records can no longer be recovered. However, the probability distribution of the aggregate data can be recovered and subsequently used for privacy-preservation purposes. In general, randomization method aims at finding an appropriate balance between privacy preservation and knowledge discovery. Representative randomization methods include random-noise-based perturbation and Randomized Response scheme. When the randomization method is carried out, the data collection process consists of two steps. The first step is for the data providers to randomize their data and transmit the randomized data to the data receiver. In the second step, the data receiver estimates the original distribution of the data by employing a distribution reconstruction algorithm. The randomization method is more efficient. However, it results in high information loss [9-16].

Randomization technique is an inexpensive and efficient approach for privacy preserving data mining (PPDM). In order to assure the performance of data mining and to preserve individual privacy, this randomization schemes need to be implemented.
Table 3: Transactional Database

<table>
<thead>
<tr>
<th>T id</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{crème, sugar, coffee, beer}</td>
</tr>
<tr>
<td>2</td>
<td>{bread, chips, cheese, milk}</td>
</tr>
<tr>
<td>3</td>
<td>{oranges, sugar, crème, beer}</td>
</tr>
<tr>
<td>4</td>
<td>{apples, beer, crème, sugar}</td>
</tr>
<tr>
<td>5</td>
<td>{eggs, milk, coffee, sugar}</td>
</tr>
</tbody>
</table>

The randomization approach protects the customers’ data by letting them arbitrarily alter their records before sharing them, taking away some true information and introducing some noise. Some methods in randomization are numerical randomization and item set randomization. Noise can be introduced either by adding or multiplying random values to numerical records or by deleting real items and adding “fake” values to the set of attributes [17-19].

Problem Formulation

Formulation of Association Rule: Association rule hiding refers to the process of modifying the original database in such a way that certain sensitive association rules disappear without seriously affecting the data and the nonsensitive rules. Association rule mining is defined as:

Let be a set of n binary attributes called items. Let be a set of transactions called the database. Each transaction in D has a unique transaction ID and contains a subset of the items in I. A rule is defined as an implication of the form X → Y where X and Y are called antecedent (left-hand-side or LHS) and consequent (right-handside or RHS) of the rule respectively.

For example T = {T1, T2, T3, T4, T5}. I = {crème, sugar, coffee, beer, bread, chips, cheese, milk, oranges, apples, eggs}.

Support measure of X is denoted as Support(X).

\[ \text{Support}(X) = \frac{\text{Support count}(X)}{n} * 100 \]

The confidence of a rule is defined

\[ \text{conf}(X \rightarrow Y) = \frac{\text{supp}(X \cup Y)}{\text{supp}(X)} \]

From the Table 1 the item set {milk, sugar} has a support of 1 / 5 = 0.2 since it occurs in 20% of all transactions (3 out of 5 transactions). The rule {milk, sugar} → coffee has a confidence of 1 / 2 = 0.5 in the database, which means that for 50% of the transactions containing milk and sugar the rule is correct.

Apriori Algorithm: Apriori is a classic algorithm for learning association rules. Apriori is designed to operate on databases containing transactions (for example, collections of items bought by customers, or details of a website frequentation). Apriori algorithm is the most popular algorithm to find all the frequent sets. It makes use of the downward closure property. Apriori algorithm is a bottom-up search, moving upward level-wise in the lattice. Before reading the database at every level it graciously prunes many of the sets which are unlikely to be frequent sets. The Apriori frequent item set discovery algorithm uses the two functions namely candidate generation and pruning at every iteration. It moves upward in the lattice starting from level 1 till level k, where no candidate set remains after pruning. It has two processes such as Candidate Generation, Pruning.

The first pass of the algorithm calculates single item frequencies to determine the frequent 1-itemsets. Each subsequent pass k discovers frequent itemsets of size k. To do this, the frequent itemsets L_k-1 found in the previous iteration are joined to generate the candidate itemsets C_k. Next, the support for candidates in C_k is calculated through one sweep of the transaction list. From L_{k-1}, the set of all frequent (k-1) itemsets, the set of candidate k-itemsets is created.

Consider a given transactional database D, minimum support threshold value SUP_{min}, minimum confidence threshold value CONF_{min}, a set of association rules AR can be mined from D and a set of sensitive association rules AR_{sen} mined from D and set of sensitive rules AR_{sen} AR to be hidden, generate a new database D\^\{\}, such that the rules in AR_{sen}\^\{\} = AR\^\{\} - AR_{sen} can mined from D\^\{\} under the same SUP_{min} and CONF_{min}. C. No normal rules in AR_{sen}\^\{\} are falsely hidden (lost rules) and no extra fake rules (ghost rules) are mistakenly will mined after the rule hiding process [17].

Proposed Solution

Objectives:

- To minimize the changes in the values of the database.
To minimize the side effects (rules and itemsets lost or created) by selecting the items in the appropriate transactions to change.

To modify the database in a way that an adversary cannot recover the original values of the database.

**WBSD Algorithm:** The existing Priority Based Distortion (PBD) algorithm reduces the confidence of the rule it hides by reversing 1’s to 0’s in items in the right itemset of each sensitive rule. First, it finds all the transactions that support a rule Ri and all the possible items in them that can be hidden its each step. Then it finds how many rules will be lost if a specific item will be reversed. After that it chooses to hide the item that will cause the least or no side effects to other rules. In order to compute these rules correctly, the algorithm holds in memory a hash table with all the rules with common items with the rule we hide. But this method consumes lot of memory space since it is making use of hash tables to store more number of records.

In the Distortion Based Algorithms, the database sanitization (i.e. the operation by which a sensitive rule is hidden), it is obtained by distorting of some values contained in the transactions supporting the rule to be hidden. Once a good item is identified, a certain number of transactions supporting the rule are modified changing the item value from 1 to 0 until the confidence of the rule is under a certain threshold [18].

The WBSD algorithm concentrates on the optimization of the hiding techniques of the hiding process so as to achieve the least side effects and the minimum complexity. The WBSD algorithm finds the transactions which support the rule, hides those transactions, assigns them a priority and sorts them in ascending order according to the priority value that each rule. Then uses these weights to compute the priority value for each transaction according to how weak are the rules that a transaction supports.

In the Figure 1, the sample database has original data and whichever data satisfies the sensitive rule will be distorted, replaced by 0. For sample database, the Rule $A \rightarrow C$ has the Support of 80% and Confidence of 100%. Whereas, for distorted database, the same rule $A \rightarrow C$ has Support of 40% and Confidence of 50%.

In this paper, WBSD algorithm depends on the assumption that in order to hide a rule $A \rightarrow B$ either the support of the item set $A \cup B$ should be decreased below the MST or the confidence of the rule should be decreased below the MCT. In order to hide an association rule, $X \cup Y$, either decrease its support or its confidence to be smaller than user-specified MST and MCT. To decrease the confidence of a rule, either increase the support of $X$, the left hand side of the rule, but not the support of $X \cup Y$, or decrease the support of the item set $X \cup Y$. For the second case, to decrease the support of $Y$, the right hand side of the rule, it would reduce the confidence faster than simply reducing the support of $X \cup Y$. To decrease support of an item, it will modify one item at a time by changing from 1 to 0 or from 0 to 1 in a selected transaction.

The following steps are required for the proposed solution [19].

**WBSD Algorithm**

**Input:** Transactions $T \in D$, Rules Set $RS$, Negative Border Rules Set $NBRS$, Rules to hide $RH$, Threshold $SM$, Thresholds $MCT$ and $MST$, Proportion $A01$

**Output:** Modified Database $DM$

**For each** $Ri \in RH$

**Do**

**{**

**Step 1:** Compute $N1$.

**Step 2:**

**2.1** Find $TR$ for $Ri$.

**2.2** For each transaction in $TR$ find the rules $Rcommon \in RS$ with at least one common item with $IR$.

**Step 3:** Sort $T \in TR$ starting from them with lowest $PTi$.

**Step 4:** For the first $N1$ sorted $T \in TR$ reverse an item $I \in IR$.

**Step 5:** Update $conf(Ri)$, $AMCT(R)$, for all the other rules that have been affected.

**}**
WBSD Algorithm Description

1st Step:
- Retrieve the set of transactions which support sensitive rule RS.
- For each sensitive rule RS find the number N1 of transaction in which, one item that supports the rule will be deleted.

2nd Step:
- For each rule Ri in the Database with common items with RS compute a weight w that denotes how strong is Ri.
- For each transaction that supports RS compute a priority Pi, that denotes how many strong rules this transaction supports.

3rd Step:
- Sort the N1 transactions in ascending order according to their priority value Pi.

4th Step:
- For the first N1 transactions hide an item that is contained in RS.

5th Step:
- Update confidence and support values for other rules in the database.

Explanation of the Hiding Process in the WBSD: At step 1, the N1 is calculated. The algorithm must make N1, transactions no longer support the Ri. So if

\[ N_1 = \left( |TR_i| - (MCT - SM) \times |TIL| \right) \]

Solving equation (1) for N1, we have:

\[ \text{CONF}(R)_{it} = \frac{IT_{Ri}}{T} = \frac{IT_{Ri} \times |T_{il}|}{T} \leq (MCT - SM) \]

At step 2.1, the Weight-based Sorting Distortion Algorithm recovers the transactions that support the rule Ri which are denoted as TR. More specifically, at step 2.2 the algorithm scans this set of transactions to collect the rules which may be affected by the hiding process. Each one of the rules that is collected has A_{MCT}(R) transactions above MCT. Then, at step 2.3 the algorithm assigns a w for each rule in Rcommon.

Each w, is assigned as follows:

\[ W_i = \frac{A_{MCT}(R)}{A_{MCT}(R)_{max}} \times \frac{A_{MCT}(R)}{A_{MCT}(R)_{min}} \]

where A_{MCT}(R)Max the maximum A_{MCT}(R) value between Rcommon rules and A_{MCT}(R)Min the minimum.

The closer the confidence of a rule is above the MCT (or the number of transactions A_{MCT}(R) above MCT) the more possible is for that rule to be eliminated if some items are turned to 0 in the victim transactions, so this rule gets a higher weight.

At step 2.4, the algorithm computes the priority value for each transaction. The reason for this assignment is to give lower PT to transactions that support fewer and stronger rules than others, in order to implicitly minimize the rules that will be eliminated.

At step 3, the algorithm sorts the transactions in ascending order according the Pi value of each transaction. In that way the transactions with the lowest PT will come on top. For the sorting process it can be used an already known algorithm such as quick sort, or bubble sort.

At step 4, N1 items of IL in TєTR are reversed in order to reduce the conf (Ri).

Finally at step 5 needs O (|R|), where |R| the number of rules in the database.

Experimental Results: The proposed system has performed extensive experiments in order to compare the effectiveness of the algorithms presented in above. The system run these algorithms in Linux operating system at a Pentium 4 at 2.40 Ghz with 2 GB RAM using ARMADA tool. The database had 50 items and the average number of items per transaction was 13.

The system has converted the results into a binary database appropriate for this implementation. In order to decrease the execution time, the system has created an inverted index with the items and the transactions that support each item. In addition, during the discovery of the rules in the database we created an index with the association rules and the items that each rule contains. In the 5k database, chose to hide 5 rules out 299. The MST was 9% and the MCT was 65%. The following rules are chosen randomly for hiding:

Rules Lost or Created in the Database and Large Itemsets That Remained Unaffected: The Figure 2 shows the side effects for different safety margin values 10%, 20%, 30%, 40% and 50%. From the Table 6, when the number of rules changed has reduced for WBSD algorithm compared to PDA. WBSD algorithm also reduces the side effects in other rules in the database. The Figure 2 shows the trade-off between privacy and
Table 5: Confidence rule table for 5k database.

<table>
<thead>
<tr>
<th>Rules</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 =&gt; 33</td>
<td>80.6%</td>
</tr>
<tr>
<td>34 =&gt; 3</td>
<td>68.9%</td>
</tr>
<tr>
<td>133 =&gt; 48</td>
<td>100%</td>
</tr>
<tr>
<td>622 =&gt; 11</td>
<td>78.1%</td>
</tr>
<tr>
<td>741 =&gt; 3</td>
<td>98%</td>
</tr>
</tbody>
</table>

Table 6: Rules lost or created after the hiding process for 5k database

<table>
<thead>
<tr>
<th>Safety margin</th>
<th>Rules lost for PDA (%)</th>
<th>Rules created for WBSD (%)</th>
<th>% of Rules changed has reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>49</td>
<td>31</td>
<td>0.37</td>
</tr>
<tr>
<td>20%</td>
<td>59</td>
<td>36</td>
<td>0.39</td>
</tr>
<tr>
<td>30%</td>
<td>68</td>
<td>40</td>
<td>0.41</td>
</tr>
<tr>
<td>40%</td>
<td>79</td>
<td>45</td>
<td>0.43</td>
</tr>
<tr>
<td>50%</td>
<td>91</td>
<td>50</td>
<td>0.45</td>
</tr>
</tbody>
</table>

Table 7: Large Itemsets remained after the hiding process

<table>
<thead>
<tr>
<th>Safety margin</th>
<th>No. of itemsets in PDA</th>
<th>No. of itemsets in WBSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>590</td>
<td>600</td>
</tr>
<tr>
<td>20%</td>
<td>510</td>
<td>520</td>
</tr>
<tr>
<td>30%</td>
<td>480</td>
<td>490</td>
</tr>
<tr>
<td>40%</td>
<td>440</td>
<td>450</td>
</tr>
<tr>
<td>50%</td>
<td>390</td>
<td>400</td>
</tr>
</tbody>
</table>

Table 8: Execution time of WBSD compared with PDA for average number of items per transaction is 13/50

<table>
<thead>
<tr>
<th>Number of transactions</th>
<th>Execution time of PDA (Seconds)</th>
<th>Execution time of WBSD (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2500</td>
<td>180</td>
<td>100</td>
</tr>
<tr>
<td>5000</td>
<td>390</td>
<td>270</td>
</tr>
<tr>
<td>7500</td>
<td>630</td>
<td>440</td>
</tr>
<tr>
<td>10000</td>
<td>830</td>
<td>700</td>
</tr>
</tbody>
</table>

Table 9: Execution time of WBSD compared with PDA for average number of items per transaction is 20/50

<table>
<thead>
<tr>
<th>Number of transactions</th>
<th>Execution time of PDA (Seconds)</th>
<th>Execution time of WBSD (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2500</td>
<td>250</td>
<td>200</td>
</tr>
<tr>
<td>5000</td>
<td>600</td>
<td>450</td>
</tr>
<tr>
<td>7500</td>
<td>890</td>
<td>630</td>
</tr>
<tr>
<td>10000</td>
<td>1125</td>
<td>900</td>
</tr>
</tbody>
</table>

data loss since, if the safety margin has increased then more side effects will be created. The Figure 3 shows how many large itemsets have been remained after the hiding process.

**Time Experiments:** The system compared the execution time of PDA and WBSD with eight different databases. The average number of items in each transaction in the first four databases is 13 and in the other four database are 20. The smaller database had 2.5k transactions and the largest had 10k transactions. The WBSD algorithm is faster than PDA because WBSD sorts the transactions at step 3 and then consecutively hides the items in those
transactions and on the other hand PDA scans all the set of victim transactions each time it hides an item. The WBSD algorithm needs less time to execute than PDA. WBSD run in linear time according to the input. In contrast, PDA seems to consume much more time as the input transactions or safety margin increase.

The Table 8 shows Execution time of WBSD compared with PDA for average number of items per transaction is 13/50. The proposed algorithm’s performance is better than PDA in terms of execution time. And the Figure 4 shows the execution time of WBSD compared with PDA. Similarly, Table 9 shows Execution time of WBSD compared with PDA for average number of items per transaction is 20/50. And the Figure 5 shows the execution time of WBSD compared with PDA.

**CONCLUSION**

We have proposed Weight Based Sorting Distortion (WBSD) algorithm in this paper for generating association rule. This work describes a method that reduces loss of data and minimizes the undesirable side effects by selecting the items in the appropriate transactions to change and maximize the desirable side effects. The purpose of the WBSD algorithm for privacy preserving data mining is to hide certain crucial information so they cannot discovered through association rule.

**REFERENCES**