Journal of AI and Data Mining
Vol 3, No 2, 2015, 131-140. 10.5829/idosi.JAIDM.2015.03.02.02

Data sanitization in association rule mining based on impact factor

A. Telikani[1*], A. Shahbahrami[2] and R. Tavoli[3]

[1]*Department of Electronic & Computer Engineering, Institute for Higher Education Pouyandegan Danesh, Chalous, Iran.*
[2]*Department of Computer Engineering, University of Guilan, Rasht, Iran.*
[3]*Department of Mathematics, Chalous Branch, Islamic Azad University, Chalous, Iran.*

Received 31 January 2015; Accepted 22 August 2015
**Corresponding author: akbar.telikani@gmail.com (A. Telikani).*

Abstract
Data sanitization process is used to promote the sharing of transactional databases among organizations and businesses, and alleviates concerns for individuals and organizations regarding the disclosure of sensitive patterns. It transforms the source database into a released database so that counterparts cannot discover the sensitive patterns and so data confidentiality is preserved against association rule mining method. This process strongly relies on the minimizing the impact of data sanitization on the data utility by minimizing the number of lost patterns in the form of non-sensitive patterns which are not mined from sanitized database. This study proposes a data sanitization algorithm to hide sensitive patterns in the form of frequent itemsets from the database while controlling the impact of sanitization on the data utility using estimation of impact factor of each modification on non-sensitive itemsets. The proposed algorithm has been compared with Sliding Window size Algorithm (SWA) and Max-Min1 in terms of execution time, data utility and data accuracy. The data accuracy is defined as the ratio of deleted items to the total support values of sensitive itemsets in the source dataset. Experimental results demonstrate that the proposed algorithm outperforms SWA and Max-Min1 in terms of maximizing the data utility and data accuracy and it provides better execution time over SWA and Max-Min1 in high scalability for sensitive itemsets and transactions.

Keywords: *Data Sanitization, Association Rule Hiding, Frequent Itemsets, Association Rule Mining, Privacy Preserving Data Mining.*

1. Introduction

The Knowledge Discovery from Databases (KDD) [1] is an interactive, iterative and interdisciplinary process and it discovers the knowledge from the data using different methodologies, and these discoveries are useful, novel, relevant, and actionable [2-3]. The KDD process comprises five major steps (1) data selection, (2) preprocessing, (3) transformation, (4) data mining and (5) evaluation; data mining is key step in this process, where intelligent methods are applied to extract unknown data patterns [4-5]. In a data sharing environment, where businesses decide to share their data to other in a collaborative project to achieve some certain gain, sometimes an adversary may be able to violate the sensitive knowledge related with individual privacy or competitive advantage in businesses through abusing KDD process [6]. Privacy Preserving in Data Mining (PPDM) is a process that prevents privacy breach against data mining algorithms, it produces a modified version from database in order to alleviate concerns for individuals and businesses regarding the revelation of sensitive patterns [7]. In some applications such as market basket analysis, Association Rule Mining (ARM) [8] has recently gained more attention in businesses [9] where the regularities in the customer purchasing behavior are found. On the other hand, these discovered patterns may pose a threat to the privacy of data owner [10]; therefore, these patterns should be hidden before data sharing in such a way that the adversaries cannot discover the regularities in customer purchasing behavior. There are two general viewpoints in the privacy preserving in ARM, (1) *input privacy*, the data values are perturbed randomly or the identifiers are anonymized before delivering to data mining

algorithms [11-13], (2) *output privacy*, some given transactions that generate the sensitive patterns are sanitized , and is known as Association Rule Hiding (ARH) [14-18].

There are two types of patterns in ARH, sensitive patterns and non-sensitive patterns. The sensitive patterns contain sensitive knowledge that show strategic patterns and trends, but the non-sensitive patterns are patterns that are general and do not disclosure the privacy of individuals. The ARH aims to achieve a solution for transforming a source database into a released database using data sanitization so that all sensitive patterns are hidden and all non-sensitive patterns are mined from released database after data sanitization; this problem is known as optimal solution. The optimal solution for data sanitization is NP-hard problem [19] and sanitization algorithms try to find this solution by minimizing undesirable effects in sanitization process. These side-effects include lost rule (misses cost), ghost rule (artifactual pattern) and hiding failure (false rule). The lost rules are non-sensitive patterns that cannot be discovered from sanitized database; this problem occurs when some corresponding transactions/ items for sensitive patterns that correspond with non-sensitive patterns are modified, and subsequently, the frequency of non-sensitive patterns are reduced and cannot be extracted from sanitized database. If number of lost patterns is reduced, the data utility is maximized. The artifactual patterns are non-sensitive patterns that are not discovered from the source database but can be mined from the released database. The hiding failure is measured in terms of the percentage of sensitive patterns that are not hidden by sanitization process and are discovered from sanitized database. For minimizing these side-effects, different approaches such as *border*, *exact* and *heuristic* have been presented, the *border approach* focuses on the maintaining the non-sensitive patterns and controls the impact of sanitization on the non-sensitive patterns and then performs the modifications with minimum impact [20-21]. The *exact approach* contains very complex algorithms, which conceive the hiding process as a constraint satisfaction problem and these algorithms solve the problem by using integer and linear programming or binary integer programming [4, 22]. Both the border and exact approaches have achieved good results when hiding a set of itemsets. These approaches hide a rule by hiding its productive itemset so that all rules generated from this itemset are hidden; therefore, they are efficient in minimizing the side-effects. The *heuristic approach* includes efficient, fast and scalable algorithms, but involves between two conflicting requirements: data privacy and data utility [10, 14-18, 23-28]. This approach does not guarantee the optimal solution; but it usually finds a solution near the best one in a faster response time. Main difference between data sanitization approaches is in the selection of victim items and transactions for modification in order to minimize the side-effects.

This study proposes an algorithm, namely Hiding based on Impact Factor (HIF), which tries to maximize data utility, data accuracy and execution time by combining optimal sub-solutions in the heuristic and border approaches. This algorithm like other data sanitization algorithms includes two main phases: *victim item selection* and *transaction selection*, first phase formulates a heuristic solution to select the appropriate victim item and second phase uses the combination of border approach and Impact Factor (IF), so that appropriate transactions are selected for modification (removing victim item from transaction). We defined and formulated "IF" as a criterion for selecting transactions with minimum impact on the non-sensitive patterns so that data sanitization causes the least impact on data utility. Also, this algorithm introduces the indexing technique that reduces sanitization ratio and execution time. An experiment performed on a real dataset to show the performance of the proposed algorithm in real application terms, as well as comparisons with the previous studies. The experimental results show that our algorithm hides all sensitive itemsets in sanitized database (no hiding failure) and there is no extra pattern in the sanitized dataset (new rule) but some non-sensitive patterns are lost in released database (misses cost) that these patterns are less than other algorithms. This study provides evidence that combining optimal sub-solutions in border and heuristic approaches with indexing and IF have a harmonic combination that introduces less side-effect on the sanitized database.

The paper is organized as follows: In section 2, problem formulations and data sanitization problem have been defined. Section 3 presents a brief review of previous works. The proposed algorithm is presented in section 4 that trade-offs between knowledge hiding and data sharing using a combinational solution. Section 5 discusses the data used in conducting experimental results, and analyzes the results of these experiments and

eventually, the main contents presented in this study are concluded in section 6.

2. Problem definitions and notations

The process of mining the association rules from transactional data was first introduced by Agrawal et al. [29]. Let in a transactional database (D), there are n-items:

$$I= \{i_1, i_2,\ldots,i_n\} \quad (1)$$

D is sequence of m-transaction, D= (t_1, t_2,…, t_m), where each transaction t_i is a set of items in I such that $t_i \subseteq I$. Each association rule is defined as an implication of the form X ⇒ Y, where X, Y are frequent itemsets in D, such that X⊆I, Y⊆I and X∩Y = ∅. Frequency of itemset X in D is defined as *support* of X, denoted by α(X), if α(X)> α_{min}, then itemset X is called *frequent itemset*, where α_{min} is the minimum support threshold given by user [30]. The *support* of an itemset, for example itemset X that contains {A, B}, is defined as fraction of transactions in the database that contain both items A and B, which are as follows:

$$\alpha (\{A, B\}) = |A \cup B| / m \quad (2)$$

where, m is the number of transactions in database. Association rule is other type of pattern in ARM that is derived from frequent itemsets using *confidence*, denoted by β. The *confidence* of the rule, denoted by β(X⇒ Y), is defined as the fraction of itemsets that support the rule among those that support the antecedent. If confidence of a rule exceeds β_{min}, (β (X⇒ Y) ≥ β_{min}), then the rule is known as a strong association rule, where β_{min} is the minimum confidence threshold given by a user. All association rules can directly be derived from the set of frequent itemsets [31]. The confidence is calculated as follows:

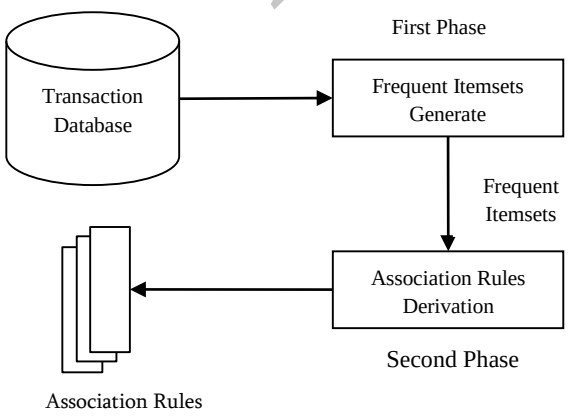


Figure 1. Association rule mining.

$$\beta (X\Rightarrow Y) = \alpha (X\Rightarrow Y) / \alpha (X) \quad (3)$$

The ARM algorithms include levelwise algorithms [32] such as apriori, and pattern-growth methods [33] such as FP-Tree and FP-Growth. The process of ARM contains two phases. In the first phase, the frequent itemsets that satisfy α_{min} are generated and then in second phase, the association rules that satisfy β_{min} are derived from the frequent itemsets, the process of ARM has been shown in figure 1.

The goal of data sanitization is to transform the source transaction database (D) to a sanitized database (D′) in order that either the support of sensitive itemset is reduced below α_{min} or the confidence of sensitive association rule is reduced below β_{min} by modifying some transactions in D, while the number of non-sensitive itemsets/ association rules with support/ confidence lower than minimum thresholds in the sanitized database is minimized. The mentioned key notations have been summarized in table 1.

Table 1. Key notations.

Notation	Description
D	Source database
m	Number of transactions
n	Number of items in database
$\mathbf{D'}$	Sanitized database
$\boldsymbol{\alpha}$	Support count of an itemset
$\boldsymbol{\alpha_{min}}$	Minimum support threshold
$\boldsymbol{\beta}$	Confidence of an association rule
$\boldsymbol{\beta_{min}}$	Minimum confidence threshold

Now, we clearly demonstrate the defined concepts using an illustrative example. Consider the sample transaction database as shown in table 2(a), which has set of items I= {a, b, c, d}. Let minimum thresholds are α_{min}=50% and β_{min} = 75%, frequent itemsets with support higher than 50% have been listed in table 2(b) and strong association rules, confidence higher than 75%, that have been derived from the frequent itemsets presented in table 2(c). Let us the itemset {b, c} be sensitive for data owner and should be hidden, and therefore the support of this itemset should be decreased in D′ to lower than α_{min} by modifying items {b} or {c} in transactions #1 or #2.

Table 2. (a) Transaction data, (b) Frequent itemsets, (c) Association rules.

(a)

ID	Items
1	{a,b,c,d}
2	{a,b,c}
3	{a,b}
4	{a,c,d}

(b)

Frequent itemset	Support
a	4(100%)
b	3(75%)
c	3(75%)
d	2(50%)
{a,b}	3(75%)
{a,c}	3(75%)
{a,d}	2(50%)
{b,c}	2(50%)
{c,d}	2(50%)
{a,b,c}	2(50%)
{a,c,d}	2(50%)

(c)

Association Rules	Confidence
a→b	75%
b→a	100%
a→c	75%
c→a	100%
d→a	100%
d→c	100%
b,c→a	100%
b→a,c	100%
d→a,c	100%
a,d→c	100%
c,d→a	100%

3. Related *w*ork

For minimizing side-effects in the sanitization process different approaches have been presented, including heuristic, border and exact. The algorithms that outperform these approaches are applied within the framework of decreasing support/confidence of itemsets/association rules and they modify the corresponding transactions using *distortion* or *blocking* techniques. In distortion, the transactions are modified via inserting or deleting items from transactions to decrease support [4, 20-23] or confidence [14, 18, 25 and 28] of patterns under given thresholds. Unlike distortion, the blocking technique modifies transactions by replacing the existing values of victim items with an unknown value [26-27].

The idea behind data sanitization has been introduced by Atallah et al. [19] to hide some frequent itemsets selectively by modifying transactions using distortion technique so that the support of a given set of sensitive itemsets is decreased below the minimum support threshold. This work has been extended to hide both sensitive itemsets and sensitive rules [23]. Most of sanitization algorithms try to find a right balance between protection of sensitive knowledge and pattern discovery. Following this purpose, *Sliding Window size Algorithm* (SWA) [17], is an efficient algorithm which outperforms sanitization process as compared to other heuristic-based algorithms in terms of speedup and it maintains data utility of the released database. Also, it hides the sensitive itemsets in only one pass through the dataset, regardless of its size or the number of sensitive itemsets that need to be protected.

The border-based approach focuses on the weight of the non-sensitive itemsets to reduce support of the sensitive itemsets while protecting the support of non-sensitive itemsets [21]. An extended approach of initial border-based approach, called Max-Min, has been proposed to decrease the side-effects, then two algorithms namely Max-Min1 and Max-Min2 have been proposed based on new extended approach. These algorithms select an itemset from non-sensitive itemsets as Max-Min itemset and modify the victim item indicated by the Max-Min itemset in such a way that the support of Max-Min itemset, if possible, is not to be changed [20].

The intersection lattice approach for hiding a specific set of association rules was first introduced by Hai and Somjit [34]. The algorithms based on intersection lattice, *Intersection Lattice-based Association Rule Hiding* (ILARH) [35], *Heuristic for Confidence and Support Reduction based on Intersection Lattice* (HCSRIL) [35] and *Algorithm of Association Rule Hiding based on Intersection Lattice* (AARHIL) [31] hide sensitive rules in three steps. The first step specifies a set of itemsets satisfying three conditions that (i) contain right-hand side of the sensitive rule, (ii) are maximal sub-itemset of a maximal itemset, and (iii) have minimal support among those sub-itemsets specified in (ii). An item in the right-hand side of the sensitive rule that is related to the specified maximal support itemset is identified as victim item. In the second step, a set of transactions supporting sensitive rule are specified. The third step removes the victim items from specified transactions until confidence of the rule is below minimum confidence threshold. In order to reduce side effects, HCSRIL sorts the set of transactions supporting the sensitive rules in ascending order of their size before sanitizing them. The AARHIL specifies the victim items based on the characteristics of the intersection lattice of frequent itemsets and identifies the transactions for data sanitization based on the weight of transactions.

4. Proposed algorithm

As discussed previously, the goal of sanitization process is to protect sensitive patterns against ARM techniques. The sanitization process which decreases the support count of sensitive itemsets

by removing items from transactions essentially includes four sub-problems:
(1) Identifying an item for each sensitive itemset to be deleted (called the victim item).
(2) Selecting the corresponding sensitive transactions to sanitize.
(3) Removing the victim item from selected transactions.
(4) Rewriting the modified database to disk.
In the next section, an efficient sanitization algorithm is proposed and outperforms these four sub-problems for hiding sensitive itemsets.

4.1. Sanitization algorithm

In order to hide an itemset, {A, B}, its support, (|A∪B|m), is decreased to below minimum support threshold. To decrease support count of an itemset, HIF modifies corresponding transactions by removing one item at a time in selected transactions. The HIF applies a heuristic to select an item as *victim item* in order to reduce the side-effects due to modification and then using border approach and an IF criterion, it controls the impact of each modification on the non-sensitive itemsets with low support. For reducing the side-effects using IF, the proposed algorithm is performed in two iterations, in first iteration, the modifications with IF=0 are done by removing victim item from transaction, if in first iteration, the support of sensitive itemsets do not decrease to below α_{min}, then HIF algorithm employs second iteration that modifications with IF=1 are done, while the support of itemset is above α_{min}, this iteration is continued. In each iteration, the HIF calculates IF for a non-sensitive itemset with minimum support that is related to victim item, instead of calculating IF for all related non-sensitive itemsets, we refer to this itemset as Minimum Support Itemset (MSI). For calculating IF, the proposed algorithm assesses the impact of each modification on the MSI, If modification affects the MSI, then IF=0, otherwise IF=1.

For selecting MSI, first the HIF selects one item from current sensitive itemset with the highest frequency as the victim item and then lists set of non-sensitive itemsets that depend on victim item, called *victim item-list* (vi-list), then from among all the listed itemsets, the itemset with minimum support is selected as MSI. Since the support of this itemset is closer to α_{min}, it takes more effect than other related non-sensitive itemsets from modification. Therefore, the impact of data sanitization on these itemsets should be controlled. To decrease execution time and also to increase data accuracy, the HIF uses indexing technique so that the sensitive itemsets with shorter length and higher support are selected for hiding. Therefore, the number of itemsets for hiding is reduced by decreasing the support of superset itemsets and so data accuracy is improved because the HIF creates less modification and so the ratio of deleted items is reduced.

The use of optimal sub-solutions and IF minimizes the lost patterns. However, the use of indexing technique minimizes sanitization ratio and execution time. Our data sanitization algorithm has essentially five steps that are applied iteratively so that all sensitive itemsets are hidden. These steps are as follows:

Step 1: *Indexing*, firstly, the sensitive itemsets are sorted in ascending order of size (number of items in itemset) and then sorted itemsets are indexed based on subset itemsets. This follows from the fact that based on apriori property, similar to all level-wise search algorithms, if an itemset is infrequent then any superset of this itemset is also infrequent, this property is the basis for indexing. For example if the itemset {a, b} is hidden, then the itemset {a, b, c} is hidden subsequently.

Step 2: *Selecting victim item*, for each itemset in root of index table, the algorithm identifies an item with maximum support as *victim item* and then creates vi-list for this item, in vi-list the algorithm lists the non-sensitive itemsets that correspond with *victim item.*

Step 3: *Selecting MSI*, from among all listed itemsets in vi-list, the algorithm selects an itemset with minimum support as MSI. If there is several MSI for a victim item, the algorithm selects first itemset as final MSI.

Step 4: *Sorting corresponding transactions*, the algorithm finds the transactions that support the sensitive itemset and then sort those in ascending order of length. This is due to the fact that if the length of transaction is the shortest, the number of itemsets that occurs in transaction is less. For example in table 2(a) transaction {a, b} has shortest length and it supports an itemset only while transaction {a, b, c, d} is longest and it supports all itemsets.

Step 5: *Calculating impact factor*, the HIF starts from first transaction and calculates the impact of removing victim item in corresponding transactions on support of the MSI. If this modification affects the support of MSI, impact factor for modification is 1, otherwise impact factor is zero. This step performs in two iterations,

in first iteration, the HIF performs modifications with impact factor equal to zero, and this operation is iterated while the support of sensitive itemset is reduced to below α_{min}. If in first iteration the support of sensitive itemset is not decreased to below α_{min} then HIF algorithm employs second iteration that modifications with IF=1 are done, while the support of itemset is above α_{min} this iteration is continued.

The pseudo-code of the HIF algorithm is shown in figure 2.

Input: Original database (*D*), minimum support threshold (α_{min}), Sensitive Itemsets (SI), non-sensitive itemsets.
Output: Sanitized database (D') that sensitive itemsets are hidden.
Step 1. Indexing sensitive itemsets
1.1. Sort *SI* based on their length
1.2. Index the sorted *SI* based on subset itemsets
While *SI* are not hidden Do{
Step 2.Create vi-list for current *SI*
Step 3.Find minimum non-sensitive itemset
3.1. Select victim item
Step 4.Find corresponding transactions for current *SI*
4.1. Sort corresponding transactions based-on length
While current *SI* is not hidden Do{
Step 5. Calculate impact factor
5.1. Find corresponding transactions for MSI
5.2. Select transactions without impact on MSI
}
}

Figure 2. The pseudo-code of the HIF.

4.2. Illustrative example

The application of the proposed data sanitization algorithm is demonstrated via an example step by step. Let us consider the sample transactional database in table 2(a) and extracted frequent itemsets with α_{min}=2 (50%) which have been shown in table 2(b). The itemsets {a, b},{a, b, c}, {a, d} are identified as sensitive itemsets, the sensitive and non-sensitive itemsets have been shown in table 3.

Table 3. Sensitive and non-sensitive itemsets.

Sensitive Itemsets	Non-Sensitive Itemsets
{a,b}	{a,c}
{a,b,c}	{b,c}
{a,d}	{c,d}
	{a,c,d}

In first step, the algorithm sorts sensitive itemsets in ascending order based on the length of itemsets and then indexes them by subset itemsets table 4(a). The HIF in second step, for itemset {a, b}, identifies the item {a} as victim item that has higher support and then non-sensitive itemsets are listed for it on table 4(b). In third step, from among listed itemsets on table 4(b) the {a, c, d} itemset has least support and it is identified as MSI. In fourth step, the algorithm finds transactions that support the sensitive itemset. The set of transactions for sensitive itemset are #1, #2 and #3. So, the support itemset {a, b} is 3(75%) and for hiding it, we need two modifications in corresponding transactions. Eventually, HIF calculates the IF for each modification, IF for transactions #1, #2 and #3 are 1, 0 and 0, respectively. Accordingly, if we modify transaction #1, this modification has undesirable effects, because transaction #1 is corresponding for MSI too. Therefore, transactions #2 and #3 are modified that introduce no-undesirable effects on the MSI.

Table 4(a). Indexed sensitive itemsets, (b) Victim items list (vi-list), (c) Sanitized data.

(a)

Root	Subset
{a,b}	{a,b,c}
{a,d}	-

(b)

Item	Non-Sensitive Itemsets
{a}	{a,c},{a,c,d}
{a}	{a,c},{a,c,d}

(c)

ID	Items
1	{b,c,d}
2	{b,c}
3	{b}
4	{a,c,d}

For itemset {a, d}, the item {a} is selected as victim item (Table 4(a)) and the itemset {a, c, d} is identified as MSI (Table 4(b)). The transactions #1 and #4 are corresponding transactions for itemset {a, d} and so we need one modification for hiding itemset, but IF for both transactions #1 and #4 is 1. therefore, HIF is performed in second iteration and transaction #1 with IF=1 is modified. After sanitizing the original data, the sanitized data have been shown on table 4(c). The algorithm hides all sensitive itemsets, but introduces itemset {a, c} as lost pattern, that cannot be extracted with α_{min}=2 (50%).

5. Experimental results

In order to evaluate the performance of the proposed algorithm, we selected two algorithms based on several parameters. Since the HIF hides sensitive patterns in the form of frequent itemsets in the framework of decreasing support in the corresponding transactions, we selected the algorithms that are performed in this framework and since HIF tries to minimize side-effects using combining optimal sub-solutions in the heuristic and border approaches, we selected SWA from

heuristic that has less computational time than other data sanitization algorithms and also Max-Min1 from border that is efficient in minimizing the side effects. The HIF, SWA and Max-Min1 were coded in Microsoft C#.net and ran on an Intel Pentium 4 with 4 GB of RAM running Windows 7 operating system at 2.53 GHz. Extensive computational tests conducted on real dataset Mushroom. In this section, we describe this dataset and analyze the results of comparisons.

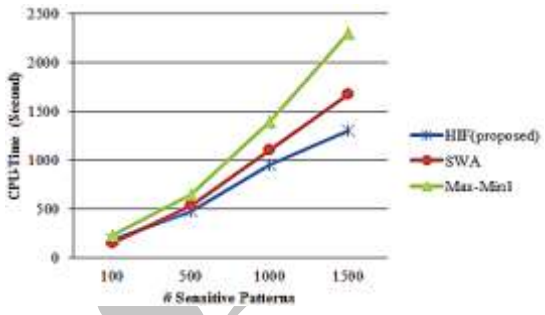
5.1. Description of the dataset

Different datasets (real or synthetic) are used to evaluate the data sanitization algorithms. One of the outcomes of the first workshop on Frequent Itemset Mining Implementations (FIMI) was the creation of the FIMI repository (http://fimi.cs.helsinki.fi) that contains real datasets made available by participants; all these datasets are described in detail in [4]. Out of these, we selected Mushroom dataset from the Irvine Machine Learning Database Repository [36] that was made available by Roberto Bayardo from the University of California. It has varying characteristics in terms of the number of transactions and items contained, and in terms of the average transaction length that this information is shown in the first four columns of table 5. By applying the apriori algorithm with α_{min} = 3161 (30%), the algorithm discovered 34154 frequent itemset. The minimum support threshold and the number of generated itemsets have been shown in the last two columns of table 5. The execution time of the apriori algorithm for this discovery was approximately 56 minutes.

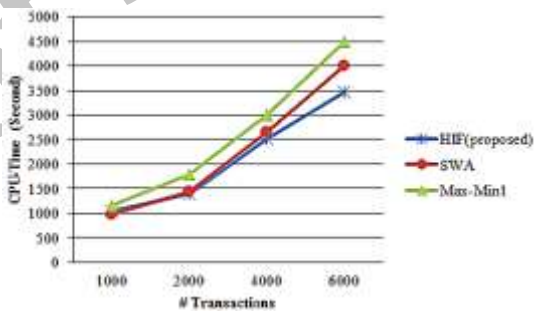
5.2. Results and analysis

We compared the HIF algorithm with the SWA and Max-Min1 to evaluate the computational complexity and side effects such as data accuracy, data utility. The computational complexity criteria were considered to evaluate the applicability of the data sanitization algorithms in the real working context. The computational complexity is the number of operations required to transform the original database into the released database. We evaluated the complexity of these algorithms by comparing the execution time required by each algorithm vis-a-vis the number of sensitive itemsets to hide as well as the size of the dataset.

In the first case, we varied the number of sensitive itemsets to hide as 100, 500, 1000 and 1500. Figure 3(a) shows that our algorithm scales well with the number of itemsets to hide, this scalability is mainly due to the index table that is used to index the sensitive itemsets per subset itemsets. Thus, there is no need to iteration for each sensitive itemset and algorithm modifies database for itemsets in the root of index table only. Generally, the execution time for SWA and Max-Min1 is more expensive due to the number of iterations for sensitive itemsets.



(a) Execution time in proportion to number of sensitive itemset.



(b) Execution time in proportion to number of transaction.

Figure 3. Execution time used by HIF, Max-Min1 and SWA.

In the second case, we varied the size of the database follow as 1000, 2000, 4000 and 6000 transactions, while fixed α_{min}=30%, and then 2% of extracted itemsets specified as sensitive itemsets. The classes of transactions are shown in table 6. Figure 3(b) shows that HIF decreases computational complexity in terms of execution time as compared to SWA and Max-Min1. As can be seen, the HIF is approximately 0.25 times rather than Max-Min1, while for low transaction sizes the execution time for HIF is near to SWA, but there is an improvement increasingly in execution time with increasing transaction sizes.

Table 5. Characteristics of the Mushroom dataset.

Dataset	# items	# Transaction	Avg. Trans. Length	Minimum Support	# Itemsets
Mushroom	119	6807	23.0	1361	34154

Therefore, the HIF is more efficient generally in proportion to number of transactions. Figure 4 shows that HIF produces the lowest lost patterns. In other words, HIF achieves the best results in minimizing the lost patterns compared with SWA and Max-Min1 algorithms. The HIF selects appropriate victim item based on maximum frequency; this selection ensures that modifying the victim item causes least impact on the set of non-sensitive patterns. Moreover, HIF applies IF to compute impact of modification on the MSI. Recall that the data utility represents the percentage of the non-sensitive itemsets that are concealed and therefore cannot be mined from the sanitized database. The data utility of released database is shown in figure 5, which directly corresponds to the lost patterns.

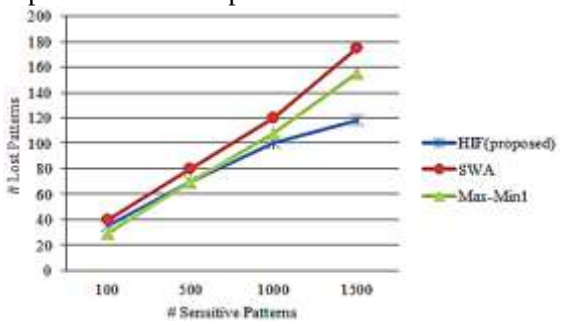


Figure 4. Number of lost patterns produced by HIF, Max-Min1 and SWA.

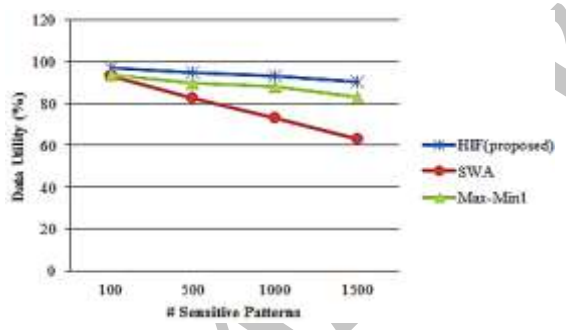


Figure 5. Data utility of dataset produced by HIF, Max-Min1 and SWA.

Figure 6 shows the comparison of HIF, SWA and Max-Min1 algorithms on the aspect of data accuracy of released dataset. With 100 sensitive itemsets for hiding, the accuracy of released dataset is high for three algorithms. This means that the hiding process causes few changes in the released dataset compared with the original dataset. Moreover, for the large-scale sensitive itemsets, the HIF algorithm achieves better accuracy compared to other algorithms and generates the most accurate sanitized database. Experimental results show that three sanitization algorithms produced no hiding failure and conceal all sensitive itemsets in the sanitized database; in the other words, these three algorithms achieved the same performance in minimizing hiding failure. Therefore, the HIF algorithm introduces 0% artifactual itemsets similar to SWA and Max-Min1. The artifactual itemsets is provable by *Theorem1*.

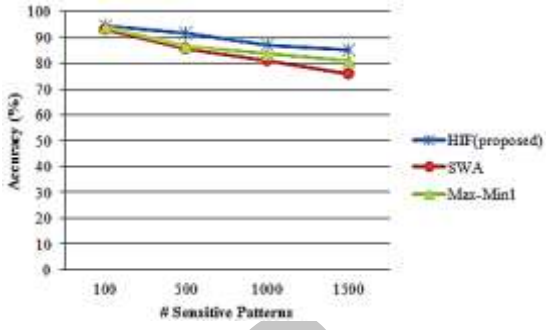


Figure 6. Accuracy of dataset produced by HIF, Max-Min1 and SWA.

Theorem1. A sanitization process, which conceals sensitive itemsets by deleting items from the source database does not generate any artifactual pattern. In summary, the results show that the HIF algorithm outperforms the SWA and Max-Min1 in terms of minimizing the side effects and execution time.

6. Conclusions

Data sanitization in association rule mining is guided by the need to minimize the impact on the data utility of the released database. In this paper we introduced a data sanitization algorithm namely HIF that incorporates optimal sub-solutions in heuristic and border approaches with indexing techniques and IF creation to balance between optimal sanitization and speedup. The experimental results show that HIF outperforms previous works, SWA and Max-Min1, in terms of maximizing data utility, data accuracy and minimizing execution time. The HIF uses heuristic approach and indexing technique for achieving to better execution time; also, it uses a border approach and IF criterion for maximizing data utility by controlling the impact of modification on the database and then selects the modifications with less IF for sanitization. Also, the indexing technique maximizes data accuracy in the released database. This contribution creates more suitable situations for businesses to share their data with other for mutual benefit and provides leverage to develop their businesses.

References

[1] Piatetsky-Shapiro, G. (1991). Report on the AAAI-91 Workshop on Knowledge Discovery in Databases. Technical Report 6.

[2] Mariscal, G., Marban, O. & Fernandez, C. (2010). A survey of data mining and knowledge discovery process models and methodologies. The Knowledge Engineering Review, vol. 5, no. 2, pp. 137-166.

[3] Vignani, B. & Satapathy, S. C. (2014). D-pattern evolving and inner pattern evolving for high performance text mining. Advances in Intelligent Systems and Computing, vol. 247, pp. 501-507.

[4] Menon, S., Sarkar, S. & Mukherjee, S. (2005). Maximizing accuracy of shared databases when concealing sensitive patterns. Information Systems Research, vol. 16, no. 3, pp. 256-270.

[5] Truta, T. M. & Campan, A. (2010). Avoiding Attribute Disclosure with the (Extended) p-Sensitive k-Anonymity Model, In: Stahlbock, R., and Crone, S. F. and Lessmann, S. (Eds.), Data Mining: Special Issue in Annals of Information Systems, pp. 353-373.

[6] Jena, L. K., Kamila, N. & Mishra, S. (2014). Privacy preserving distributed data mining with evolutionary computing. Advances in Intelligent Systems and Computing, vol. 247, pp. 259-267.

[7] Bertino, E., Fovino, I. N. & Provenza, L. A. (2005). A framework for evaluating privacy preserving data mining algorithms. Data Mining and Knowledge Discovery, vol. 11, no. 2, pp. 121-154.

[8] Agrawal, R., Imielinski, T. & Swami, A. (1993). Mining association rules between sets of items in large databases. ACM SIGMOD Conference on Management of Data, pp. 207-216.

[9] Lijffijt, J., Papapetrou, P. & Puolamäki, K. (2014). A statistical significance testing approach to mining the most informative set of patterns. Data Mining and Knowledge Discovery, vol. 28, pp. 238-263.

[10] Amiri, A. (2007). Dare to share: Protecting sensitive knowledge with data sanitization. Decision Support Systems, vol. 43, no. 1, pp. 181-191.

[11] Rizvi, S. & Haritsa, R. (2002). Maintaining data privacy in association rule mining. 28th International Conference on Very Large Databases, pp. 682-693.

[12] Evfimievski, A., Srikant, R., Agrawal, R. & Gehrke, J. (2004). Privacy preserving mining of association rules. Information Systems, vol. 29, pp. 343-364.

[13] Lin, J, & Cheng, Y. (2009). Privacy preserving itemset mining through noisy items. Expert Systems with Applications 36, pp. 5711-5717.

[14] Wang, S. L., Maskey, R., Jafari, A. & Hong, T. P. (2007a). Efficient sanitization of informative association rules. Expert Systems with Applications 35(1), doi:10.1016/j.eswa.2007.07.039.

[15] Saygin, Y., Verykios, V. S. & Elmagarmid, A. K. (2002). Privacy preserving association rule mining. 12th Workshop on Research Issues in Data Engineering.

[16] Oliveira, S. R. M. & Zaiane, O. R. (2002). Privacy preserving frequent itemset mining. IEEE ICDM Workshop on Privacy, Security and Data Mining, vol. 14, pp. 43- 54, Maebashi City, Japan.

[17] Oliveira, S. R. M. & Zaiane, O. R. (2003). Protecting sensitive knowledge by data sanitization. 3rd IEEE International Conference on Data Mining, pp. 613–616, Florida, USA.

[18] Oliveira, S. R. M. & Zaiane, O. R. (2003). Algorithms for balancing privacy and knowledge discovery in association rule mining. 7th International Database Engineering and Applications Symposium, pp. 54-63, Hong Kong, China.

[19] Atallah, M., Bertino, E., Elmagarmid, A. K., Ibrahim, M. & Verykios, V. S. (1999). Disclosure limitation of sensitive rules. IEEE Knowledge and Data Engineering Exchange Workshop, Chicago, USA.

[20] Moustakides, G. V. & Verykios, V. S. (2006). A max-min approach for hiding frequent itemsets. 6th IEEE International Conference on Data Mining Workshop, pp. 502–506.

[21] Sun, X. & Yu, P. S. (2005). A border-based approach for hiding sensitive frequent itemsets. 5th IEEE International Conference on Data Mining, pp. 426-433.

[22] Divanis, A. G. & Verykios, V. (2006). An integer programming approach for frequent itemset hiding. 15th ACM Conference on Information and Knowledge Management.

[23] Dasseni, E., Verykios, V. S., Elmagarmid, A. K. & Bertino, E. (2001). Hiding association rules by using confidence and support. 4th Information Hiding Workshop.

[24] Pontikakis, E. D., Tsitsonis, A. A. & Verykios, V. S. (2004). An experimental study of distortion-based techniques for association rule hiding. 18th Conference on Database Security, pp. 325-339.

[25] Wang, S. L., Patel, D., Jafari, A. & Hong, T. P. (2007). Hiding collaborative recommendation association rules. Applied Intelligence, vol. 26, no. 1, pp. 66-77.

[26] Saygin, Y., Verykios, V. S. & Clifton, C. (2001).Using unknowns to prevent discovery of association rules. ACM SIGMOD Record, vol. 30, no. 4, pp. 45-54.

[27] Wang, S. L. & Jafari, A. (2005). Using unknowns for hiding sensitive predictive association rules. IEEE International Conference on Information Reuse and Integration, pp. 223-228.

[28] Wang, S. L., Parikh, B. & Jafari, A. (2007). Hiding informative association rule sets. Expert Systems with Applications, vol. 33, no. 2, pp. 316-323.

[29] Agrawal, R., Imielinski, T. & Swami, A. (2013). Mining association rules between sets of items in large

databases. ACM SIGMOD Conference on Management of Data, pp. 207–216.

[30] Zeng, Y., Yin, S., Liu, J. & Zhang, M. (2015). Research of Improved FP-Growth Algorithm in Association Rules Mining. Scientific Programming, http://dx.doi.org/10.1155/2015/910281.

[31] Hai, L. Q., Somjit, A. & Ngamnij, A. (2013). Association rule hiding based on intersection lattice. Mathematical Problems in Engineering, http://dx.doi.org/10.1155/2013/210405.

[32] Park, J. S., Chen, M. S. & Yu, P. S. (1995). An effective hash-based algorithm for mining association rules. ACM-SIGMOD International Conference on Management of Data, pp. 175–186.

[33] Han, J., Pei, J., Yin, Y. & Mao, R. (2004). Mining frequent pattern without candidate generation: a frequent pattern tree approach. Data Mining and Knowledge Discovery, vol. 8, no. 1, pp. 53–87.

[34] Hai, L. Q. & Somjit, A. (2012). A conceptual framework for privacy preserving of association rule mining in e-commerce. 7th IEEE Conference on Industrial Electronics and Applications, pp. 1999–2003.

[35] Hai, L. Q., Somjit, A., Huy, X. N. & Ngamnij, A. (2013). Association rule hiding in risk management for retail supply chain collaboration. Computers in Industry, vol. 64, pp. 776–784.

[36] Bayardo, R. (1998). Efficiently mining long patterns from databases. ACM-SIGMOD International Conference on Management of Data, Seattle, WA.

نشریه هوش مصنوعی و داده کاوی

دانشگاه شاهرود

پالایش داده‌ها در کاوش قوانین انجمنی مبتنی بر ضریب تاثیر

اکبر تلیکانی[1,*] ، اسدالله شاه بهرامی[2] و رضا طاولی[3]

[1] دانشکده مهندسی کامپیوتر و الکترونیک، موسسه آموزش عالی پویندگان دانش، چالوس، ایران.

[2] دانشکده مهندسی کامپیوتر، دانشگاه گیلان، رشت، ایران.

[3] دانشکده ریاضیات، دانشگاه آزاد واحد چالوس، چالوس، ایران.

ارسال ۲۰۱۵/۰۱/۳۱؛ پذیرش ۲۰۱۵/۰۸/۲۲

چکیده:

پالایش داده‌ها برای ترویج اشتراک‌گذاری پایگاه داده‌های تراکنشی بین سازمان‌ها و کسب‌وکارها استفاده می‌شود و نگرانی‌ها برای اشخاص و سازمان‌ها پیرامون افشاء الگوهای حساس را کاهش می‌دهد. این فرآیند پایگاه داده اولیه را به یک پایگاه داده اصلاح شده تبدیل می‌کند تا دیگران نتوانند الگوهای حساس را اکتشاف کنند و در نتیجه محرمانگی داده‌ها در برابر کاوش قوانین انجمنی محافظت می‌شود. این فرآیند تلاش دارد تا تاثیر پالایش بر روی سودمندی داده‌ها را از طریق کاهش تعداد الگوهای مفقود شده در قالب الگوهای حساسی که از پایگاه داده پالایش شده استخراج نمی‌شوند کمینه کند. این پژوهش یک الگوریتم پالایش داده‌ها برای پنهان‌سازی الگوهای حساس در قالب مجموعه فقره‌های تکرارشونده ارائه می‌دهد که تاثیر پالایش بر روی سودمندی داده‌ها را با استفاده از اندازه‌گیری ضریب تاثیر هر اصلاح بر روی مجموعه فقره‌های غیرحساس کنترل می‌کند. الگوریتم پیشنهادی با الگوریتم‌های اندازه پنجره لغزان (SWA) و Max-Min1 برحسب زمان اجرا، سودمندی داده‌ای و دقت داده‌ای مقایسه شده است. دقت داده‌ای به عنوان نرخ مقادیر فقره‌های حذف شده به کل مقادیر فقره‌هایی که از مجموعه فقره‌های حساس در پایگاه داده اولیه پشتیبانی می‌کنند محاسبه می‌شود. نتایج تجربی نشان می‌دهد که الگوریتم پیشنهادی نسبت به الگوریتم های SWA و Max-Min1 برحسب سودمندی داده ها و دقت داده‌ای اجرای بهتری را به همراه دارد و برای مقیاس‌های بالای مجموعه فقره‌های حساس و تعداد تراکنش‌ها زمان اجرای بهتری نسبت به الگوریتم های SWA و Max-Min1 دارد.

کلمات کلیدی: پالایش داده‌ها، پنهان‌سازی قوانین انجمنی، مجموعه فقره‌های تکرارشونده، کاوش قوانین انجمنی، حفاظت از حریم‌خصوصی در داده‌کاوی.