*Journal of **AI** and **D**ata **M**ining*
Vol 4, No 2, 2016, 219-227

10.5829/idosi.JAIDM.2016.04.02.10

Introducing an algorithm for use to hide sensitive association rules through perturb technique

M. Sakenian Dehkordi* and M. Naderi Dehkordi

Department of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad, Isfahan, Iran.

Received 21 June 2015; Accepted 18 January 2016
**Corresponding author: sakenian91@gmail.com (M. Sakenian).*

Abstract
Due to the rapid growth of the data mining technology, obtaining private data on users through this technology has become easier. Association rules mining is one of the data mining techniques that is used to extract useful patterns in the form of association rules. One of the main problems with the application of this technique to databases is the disclosure of sensitive data, and thus endangering the security and privacy of the owners of the data. Hiding the association rules is one of the methods available to preserve privacy, and it is a main subject in the field of data mining and database security, for which several algorithms with different approaches have been presented so far. An algorithm for use to hide sensitive association rules with a heuristic approach is presented in this article, where the perturb technique based on reducing confidence or support rules is applied with an attempt to remove the considered item from a transaction with the highest weight by allocating weight to the items and transactions. The efficiency of this technique is measured by means of the failure criteria of hiding, the number of lost rules and ghost rules, and the execution time. The results obtained from this work are assessed and compared with the two known FHSAR and RRLR algorithms, which are based on the two real databases dense and sparse. The results obtained indicated that the number of lost rules in all the experiments performed decreased by 47% in comparison with the RRLR algorithm, and decreased by 23% in comparison with the FHSAR algorithm. Moreover, the other undesirable side effects in the proposed algorithm in the worst case were equal to those for the basic algorithms.

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

1. Introduction

Due to competitions in the political, military, economic, and scientific fields, and the importance of access to information in a short period of time without human intervention, the science of data analysis or data mining has defined some techniques to analyze data with the objective of finding patterns in them [1,2].
Extracting association rules is one of the main aspects of data mining that deals with discovering the correlation among the items and finding a set of frequent items from big data resources [3,4]. However, the data obtained may include sensitive personal/business information whose publishing and sharing can endanger the security and privacy of the owner of the information. For example, although sharing information about diseases is useful but releasing personal information about patients is not. Another example relates to the customers' purchasing behavior. Studying the customers' purchasing behavior can be very important and profitable for manufacturers but there exist some sensitive data that should be protected against jobbers [5]. To protect data security and to prevent the discovery of private data, the concept of privacy preserving data mining has been presented. The objective of this concept is to examine the side effects of the data mining process, which leads to protect the personal and organizational privacy. There exist many different approaches in the algorithm form. After data mining and hidden private knowledge, only insensitive data is identified in these algorithms [6].
In this paper, the new HSARWI algorithm is presented to hide the set of sensitive association rules, and to reduce the undesirable side effects.

After implementation, this algorithm will be compared with the two algorithms FHSAR [7] and RRLR [8] based on the two real databases dense and sparse.
In this paper, after studying some existing algorithms, the HSARWI algorithm will be introduced. Finally, the conclusions and suggestions for future studies will be presented.

2. Literature review

Attallah et al. [9,10] were the first to present an experimental algorithm for hiding the sensitive association rules in 1999.
In 2001, Dasseni et al. [11] introduced three algorithms for hiding the sensitive association rules. These rules should not have anything in common, and their performance in the field of controlling lost rules and ghost rules is not sufficient.
Saygin et al. [12] were the first who, in 2001, presented the use of unknown values instead of changing zero to one and vice versa in hiding the sensitive association rules. The objective of applying the unknown values was to protect the users from learning wrong rules.
Oliveira et al. [13] were the first who, in 2002, presented some manners for hiding the sensitive rules simultaneously.
Oliveira et al. [14] introduced an algorithm named SWA in 2003 with no respect to the database size and the number of sensitive rules that should be hidden. In SWA, the database is scanned only once. This algorithm is not based on memory, and so it can be applied to big databases.
Verykios et al. [15] introduced five algorithms in 2004, which reduced the support of item sets, while producing sensitive rules as long as its support was less than the minimum support threshold. The main drawback of these algorithms is that the rules should not overlap one another.
In 2007, Wang et al. [16] suggested two algorithms, where if the items are proposed, then the sensitive association rules are hidden automatically. The drawback of these two algorithms is that the sanitized database is different with respect to the order of removing the rules.
In 2007, Wang et al. [17] introduced two algorithms for hiding the predictive sensitive association rules, i.e. the rules that have sensitive items in their antecedent. Both algorithms hide these rules automatically. There is no need for data mining and manual selection of sensitive rules before the hiding process.
In 2007, Verykios et al. [18] suggested two algorithms based on weight allocation to the transactions.
By allocating weight to the transactions, the WSDA algorithm seeks to select useful transactions to remove item by considering a safety margin (SF). It hides the rules with a reduced confidence of rules less than MCT + SF.
The BBA algorithm applies the blocking technique for hiding. It also considers SF.
In 2008, Weng et al. [7] presented the FHSAR algorithm for hiding sensitive rules. This algorithm scans the database once, and consequently, reduces the execution time. This algorithm is a week selecting victim item.
In 2009, Dehkordi et al. [19] suggested a new method for maintaining privacy of the data mining association rules based on genetic algorithm, where there are no lost and ghost rules.
In 2010, Modi et al. [20] introduced an algorithm named DSRRC, which seeks to hide rules with minimum changes in the database through clustering rules based on the common item at the consequence of the rules as much as possible in a simultaneous manner. The drawback of this rule is that it only hides those rules that have one item on their right side.
In 2012, Shah et al. [8] presented two algorithms for correcting the DSRRC algorithm. The ADSRRC algorithm was presented for overcoming the restriction of multiple ordering, and the RRLR algorithm was introduced for overcoming the restriction of being a single item at the consequence of the sensitive rules.
Jain et al. [21] and Gulwani et al. [22] implemented hiding the rules as a group by applying the concept of representative rules [23] since the support of sensitive items does not change towards the original database.
In 2013, Domadiya et al. [24] proposed the MDSRRC algorithm for overcoming the DSRRC algorithm restriction. This algorithm can hide the rules that have multiple items in both their antecedent and consequent.

3. Problem definition

Association rules determine the correlations of different items in a set of input data, where these rules are selected according to the support and confidence criteria [5].
If $I = \{i_1, i_2, \ldots, i_m\}$ is a set of items, and $D = \{t_1, t_2, \ldots, t_n\}$ is a set of transactions or a database, every transaction includes a subset of I, and $t_i \subseteq I$.
The common framework of the association rules is $X \rightarrow Y$, where $X \subset I$, $Y \subset I$, $X \cap Y = \Phi$, X is the

left hand side named antecedent, and Y is the right hand side consequent of rule [25].
To calculate the support of rule $X \rightarrow Y$ and confidence, (1) and (2) are used, respectively [26].

$$\text{Support}(X \rightarrow Y) = (|X \cup Y|)/(|D|) \quad (1)$$

$$\text{Confidence}(X \rightarrow Y) = (|X \cup Y|)/(|X|) \quad (2)$$

where, $|X|$ is the number of occurrences of the item set of X in the set of transactions D, and $|D|$ is the number of transactions in D.
Association rule mining algorithms scan the database of transactions, and calculate the support and confidence of the candidate rules in order to determine whether they are significant or not. A rule is significant if its support and confidence are higher than the user specified criteria (MST and MCT), and to justify this, conditions (3) and (4) should be met at the same time [27].

$$\text{Support}(X \rightarrow Y) \geq MST \quad (3)$$

$$\text{Confidence}(X \rightarrow Y) \geq MCT \quad (4)$$

The sensitive association rule $X \rightarrow Y$ is hidden, whenever one of the following two conditions, (5) or (6), is met [27].

$$\text{Support}(X \rightarrow Y) < MST \quad (5)$$

$$\text{Confidence}(X \rightarrow Y) < MCT \quad (6)$$

Among the extracted association rules (ARs) from the original database (D), some of them are introduced as the sensitive rules from the database owner (SAR), $SAR \subseteq AR$.
The objective of the privacy preserving association rules mining algorithms is that in addition to having the basic database, MCT and MST and the set of sensitive rules or set of frequent sensitive patterns should make some changes in D. The changes prevent the extraction of sensitive rules or frequent patterns from the sanitized database (D'). The following side effects should be minimized in this process [5]:

- Execution time
- Number of hiding failure
- Number of lost rules
- Number of ghost rules

4. Proposed algorithm

The function of this algorithm is to hide the sensitive association rules through the heuristic approach, based on distorting values. The victim item and victim transaction are determined through this newly-introduced method, while it seeks to reduce the amount of support or confidence by removing the victim item. After removing any victim item, some rules whose amount of support or confidence are below the determined threshold values are added to the set of the hidden sensitive association rules.
Input: Original Database (D), SAR, MST, MCT.
Output: Sanitized Database (D'). While it goes through the association rule mining once more, the unfavorable side effects become minimized. The notations applied in this study are presented in table 1.

4.1. Calculating transaction and item weight

To calculate the transaction weight, the presented concepts are adopted as follow [7]:

$$R_{ik} = \{j \mid sar_j \subseteq t_i \text{ and } k \in t_i\} \quad (7)$$

$$MIC_i = \max(|R_{ik}|) \quad (8)$$

$$WT_i = MIC_i/2^{(|t_i|-1)} \quad (9)$$

where k is an item in t_i, and R_{ik} contains the number of sensitive association rules from SAR that is completely supported by transaction t_i. Full support means that transaction t_i should include at least all the available items in the antecedent and consequent of the sensitive association rules.
For each one of the available items in transaction t_i, the A and B sets are obtained through (10) and (11). The weight of each one of the items is calculated through (12).

$$A_{ik} = \{j \mid sar_j \subseteq t_i \text{ and } k \in RHS_j\} \quad (10)$$

$$B_{ik} = \{j \mid sar_j \subseteq t_i \text{ and } k \in LHS_j\} \quad (11)$$

$$WI_{ik} = |R_{ik}| + |A_{ik}| - |B_{ik}| \quad (12)$$

Table 1. Notations and definitions.

Notation	Definition
t_i	Transaction i of database
$\|S\|$	Number of members of set S
AR	Association rules extracted from D
AR'	Association rules extracted from D'
SAR	Set of sensitive association rules, SAR = $\{sar_1, sar_2, ..., sar_m\}$
SAR'	Set of sensitive association rules has been hidden
$Supp(sar_j)$	$Support(sar_j)$
$Conf(sar_j)$	$Confidence(sar_j)$
WT_i	Weight of t_i
WI_{ik}	Weight of item k for transaction i
VT	Victim transaction
VI	Victim item
LHS_j	An item set on the left hand side of a rule (antecedent)
RHS_j	An item set on the right hand side of a rule (consequent)
k	Determines an item in t_i

4.2. CalculateTransactionWeight(t_i) function

This function receives the number of transactions as an input parameter, and obtains the number of association rules that are completely supported by it. It obtains MIC and calculates the weight of each transaction. Its pseudo-code is shown in figure 1.

4.3. CalculateVictimItem(t_i) function

This function receives the number of victim transactions as an input parameter, and the weight of each one of the items that are repeated at least in a sensitive rule and are supported by this transaction are calculated. This is followed by the selection of an item with the highest weight as the victim item for removal from the victim transaction. The pseudo-code for this function is shown in figure 1.

4.4. CheckingNotFailure(VT, VI) Function

This function receives the victim transaction and the victim item as the input parameters, and studies whether removing this item can lead to the violation of the previous hidings, and the True or False result is returned to the main program. If only the output of this function is true, the victim item will be removed from the victim transaction, otherwise, another item will be considered for removal. The pseudo-code for this function is demonstrated in figure 1.

HSARWI Algorithm
Input: D, SAR, MST, MCT   Output: D'

```
1.  For each $t_i \in D$
2.  {
3.      For each $sar_j \in SAR$
4.      {
5.          Calculate supp($sar_j$);
6.          Calculate conf($sar_j$);
7.      }
8.      CalculateTransactionWeight($t_i$);
9.  }
10. while(SAR $\neq \Phi$)
11. {
12.     VT = Transaction with maximal weight;
13.     VI = CalculateVictimItem(VT);
14.     If (CheckingNotFailure(VT,VI) = True)
15.     {
16.         Remove VI from VT;
17.           For each $sar_j \in SAR$
18.           {
19.             Update supp($sar_j$);
20.             Update conf($sar_j$);
21.           }
22.         If (Supp($sar_j$) < MST || Conf($sar_j$) < MCT)
23.         {
24.             Remove $sar_j$ from SAR;
25.             Add $sar_j$ To SAR';
26.         }
27.         CalculateTransactionWeight($t_i$); //Update Weight
28.     }  // If (CheckingNotFailure(VT,VI) = True)
29.     Else
30.     {
31.         CalculateVictimItem(VT); // Select another Item
32.         Go to 14;
33.     }
34. } //end of while
```

Functions

```
CalculateTransactionWeight($t_i$)
{
    For each $sar_j \in SAR$
        If  $t_i$ fully support $sar_j$
            For each item $k \in sar_j$
                $|R_{ik}|$++;
    $WT_i = \max(|R_{ik}|)/2^{\wedge}(|t_i|-1)$;
}
```

```
CalculateVictimItem(ti)
{
    For each $sar_j \in SAR$
    {
        If  $t_i$ fully support $sar_j$
            For each item $k \in sar_j$
            {
                $|R_{ik}|$++;
                If ($k \in RHS_j$)
                    $|A_{ik}|$++;
                If ($k \in RHS_j$)
                    $|B_{ik}|$++;
            }
        $WI_{ik} = |R_{ik}| + |A_{ik}| - |B_{ik}|$;
    }//end of for
    Return Item with maximal
    $WI_{ik}$
}
```

```
CheckingNotFailure(VT,VI)
{
    For each $sar'_j \in SAR'$
        If (Supp($sar'_j$) $\geq$ MST
                && VI $\in LHS_j$  &&
                $sar'_j$ don't Support with $t_i$)
                  Return false;
        Else
                    Return True;
}
```

Figure 1. pseudo-code of this HSARWI algorithm.

4.5. Different levels of algorithm

In figure 1, lines 1-9 do the scanning database once, and the following cases are calculated:

- Support of each $sar_j \in SAR$
- Confidence of each $sar_j \in SAR$
- Weight of each transaction

The hiding operation begins from line 10, and a transaction with the highest weight will be selected as the victim transaction in line 12. The weight of items is calculated by calling the CalculateVictimItem(t_i) function, and an item with the highest weight, as the victim item, will be returned to the main program in line 13.

The CheckingNotFailure(VT, VI) function is called in line 14. If the value of this function is true, the victim item will be removed from the victim transaction, and in lines 17-26, the amounts of the support and confidence of all the sensitive association rules are updated, and if at least one of the amounts of the support or confidence of the rule is less than MST or MCT, the rule is hidden, removed from the set of sensitive association rules, and added to the set of hidden association rules. In line 27, the weight of transaction will be updated by calling CalculateTransactionWeight (ti).

If the CheckingNotFailure(VT, VI) function returns False, in line 31, with calling CalculateVictimItem(t_i) again, another item will be selected for removal from the transaction.

The above processes are continued until hiding all the sensitive association rules.

4.5. Example

To express the HSARWI algorithm well, an example is presented in this section with the database tabulated in table 2.

The sensitive association rules, minimum support threshold, and minimum confidence threshold are determined by the owner of the database as follow:

SAR = {(1→3),(1,3→4)}

MST = 40%, MCT = 75%

Scanning the existing transactions in the database begins from 1 in table 2.

Table 2. Sample database.

Transaction ID	Items	Transaction ID	Items
1	1,3,4	9	2,3,5,6,7
2	1,3,4,8	10	1,3,4,7
3	1,2,3,4,5,6,8	11	2,6,7
4	1,2,4,6,7	12	1,2,3,4,5
5	2,3,4,5	13	1,3,4,5,6,8
6	1,2,3	14	2,7
7	3,4,5,6,8	15	1,2,3,4,8
8	1,2,3,7		

First transaction (t_1) completely supports the first sensitive rule (1→3); therefore, Count (1→3) and Count (1) increase one. Next, the second sensitive association rule is studied; this rule is completely supported by t_1, Count (1, 3→4), and Count (1, 3) increase one. By calling the CalculateTransactionWeight (1) function, weight of t_1 is calculated. Lines 1-9 in figure 1 run to calculate the weight of all transactions. Table 3 includes the obtained data on the support and confidence of the sensitive association rules.

Table 3. Support and confidence of sensitive rules.

sar_j	Count(sar_j)	Count(LHS_j)	supp(sar_j)	conf(sar_j)
1→3	9	10	0.6	0.9
1,3→4	7	9	0.47	0.78

Table 4 shows the calculated weight for each one of the transactions in table 2.

Table 4. Calculated weights for transactions.

Transaction ID	WT(ti)	Transaction ID	WT(ti)
1	50	9	0
2	25	10	25
3	3.125	11	0
4	0	12	12.25
5	0	13	6.25
6	25	14	0
7	0	15	12.5
8	12.5		

According to table 4, t_1 has the highest weight, and it is selected as the victim transaction (VT = 1). By calling the CalculateVictimItem (1) function in line 13 of figure 1, the weight of each one of the items in t_1 is calculated according to table 5.

Table 5. Weight of each item in t_1.

Item	$\lvert R_{1k}\rvert$	$\lvert A_{1k}\rvert$	$\lvert B_{1k}\rvert$	WI_k
1	2	0	2	0
3	2	1	1	2
4	1	1	0	2

Item number 3 has the highest weight, and is selected as the victim item (VI = 3). As the set of hidden sensitive association rules has no member, the CheckingNotFailure(1, 3) function returns True to the main program, lines 15-21 in figure 1 are executed, and the new support and confidence of the sensitive association rules are calculated according to table 6. None of the sensitive association rules are hidden in lines 22-26. The new weight for t_1 is calculated in line 27 of figure 1.

Table 6. Modified support and confidence after removing item 3 from t_1.

sar_j	Count(sar_j)	Count(LHS_j)	supp(sar_j)	conf(sar_j)
1→3	8	10	0.54	0.8
1,3→4	6	8	0.4	0.75

The algorithm steps are repeated, transaction 2 with the highest weight is selected as the victim transaction (VT = t_2) according to table 4, and the calculated weight for each one of its items is shown in table 7. Item 3 has the highest weight, and so it is selected for removal. The CheckingNotFailure(2,3) function returns True, so item 3 is removed from t_2. Table 8 represents the updated support and confidence of all the sensitive rules. By reducing the confidence of the sensitive rule 1→3 with less than MCT and reducing the support of the sensitive rule 1,3→4 with less than MST, both rules are hidden.

Table 7. Weight of each item in t_2.

Item	$\lvert R_{1k}\rvert$	$\lvert A_{1k}\rvert$	$\lvert B_{1k}\rvert$	WI_k
1	2	0	2	0
3	2	1	1	2
4	1	1	0	2
8	None of the sensitive association rules is repeated, so no weight is calculated for it.			

Table 8. Modified support and confidence after removing item 3 from t_2.

sar_j	Count(sar_j)	Count(LHS_j)	supp(sar_j)	conf(sar_j)
1→3	7	10	0.46	0.7
1,3→4	5	7	0.33	0.71

6. Comparison and evaluation

To evaluate the performance and efficiency of the HSARWI algorithm, the two well-known FHSAR and RRLR algorithms are implemented on a system including Windows 8 operating system, Intel Core i7 processor, and 8 GB of main memory in visual studio environment 2012 with coding language C#.

The two real databases Mushroom and Chess are applied for the experiments; their detailed characteristics and the amounts of MST and MCT are shown in tables 9 and 10, respectively.

Table 9. Characteristics of databases.

Database Name	Number of Transactions	Number of Items	Transaction Length	Density
Mushroom	8124	119	23	19%
Chess	3194	75	37	49%

The number of sensitive association rules of both databases is considered as 2, 4, 6, and 8. Then the evaluating criteria are studied.

Table 10. Amount of applied MCT and MST.

Database Name	MST	MCT	Number of AR
Chess	0.95	0.98	303
Chess	0.88	0.92	22085
Mushroom	0.5	0.75	664
mushroom	0.4	0.6	4570

Failure: This refers to the number of sensitive rules extracted from the sanitized database with data mining after the hiding operation [28].
Due to the existence of a function to evaluate and predict failure, the HSARWI and FHSAR algorithms have no failure in any experiment. The RRLR algorithm has a failure rate of 8% in all the experiments since it makes the hiding process with inserting and removing the items. Item insertion may cause an increase in the amount of confidence, leading to a failure in hiding the rules, whose support is higher than MST.
Lost rules: This refers to the number of insensitive association rules that are extracted from the original database but are not extracted from the sanitized database after the hiding process [28]. In the HSARWI algorithm, the victim item selection manner is effective in reducing the number of lost rules. An item is selected for removal that is repeated in the sensitive rules more than the other items with respect to repetition at the consequent of the sensitive rules, and therefore, this item has the highest effect on reducing the amount of support and confidence of rules. Due to the above-mentioned reasons, the HSARWI algorithm reduces the number of removed items from the database more, in comparison with the FHSAR and RRLR algorithms, and makes the sanitized database similar to the original database. Therefore, the number of lost rules is reduced with the HSARWI algorithm. In the RRLR algorithm, due to the selection of a transaction with more sensitivity and length, more insensitive rules are being missed. Diagrams related to figures 2, 3, 4, and 5 show that the HSARWI algorithm has been more successful than the basic algorithms in reducing the number of lost rules.
Ghost rules: This refers to the number of insensitive association rules that are not extracted from the original database but are extracted from the sanitized database after the hiding process [28]. In the experiments conducted on the Chess database, no ghost rules were generated because the higher the database density, the less the generated ghost rules are, and since such databases generate many association rules, their removed element usually has a less effect on the generating ghost rules. The inserting and removing items generate the ghost rules, whose amounts of support and confidence are close to those for MST and MCT. The removing item does not always lead to the generation of ghost rules, while the inserting item is more effective in generating the ghost rules. Since the removing item always causes a decrement in the amount of support of the rules, and sometimes may cause an increment in the amount of confidence of rules, the inserting item may cause an increment in both the amounts of the support and confidence of rules. Since the hiding process is run through removing and inserting items in the RRLR algorithm, the number of ghost rules generated by this algorithm is more than those generated by the HSARWI and FHSAR algorithms. The diagrams shown in figures 6 and 7 show the number of ghost rules generated on the Mushroom database.

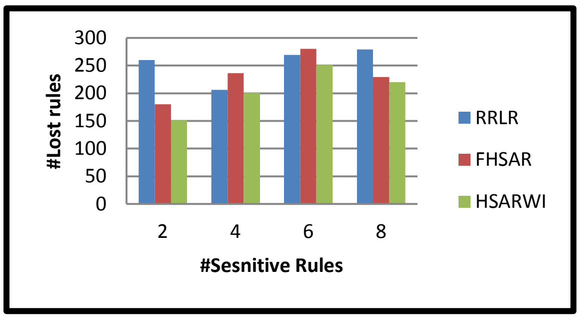

Figure 1. Lost rules in chess with MST = 0.95 and MCT = 0.98.

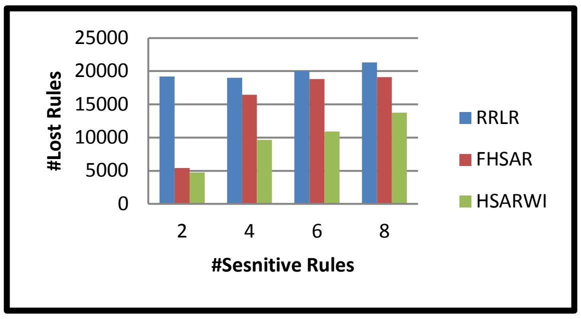

Figure 2. Lost rules in chess with MST = 0.88 and MCT = 0.92.

Execution time: This refers to the duration of executing algorithm to hide all the sensitive association rules [28]. In the FHSAR and HSARWI algorithms, scanning the database is run only once, so these two algorithms consume less time. As it is evident in figures 8, 9, 10, and 11, the execution time in HSARWI is equal to or less than the FHSAR and RRLR algorithms. Reduction in the execution time in HSARWI is directly related to the reduction in number of items removed from the database since after removal of every item, the amount of support and confidence of the rules are updated. Therefore, there is a direct relation between reduction in the number of removed items and reduction in the updating process time, and hence, a saving in time. To hide every one of the sensitive rules, the RRLR algorithm firstly removes the left hand side item and then inserts it, i.e. scanning twice for each removal and insertion. Therefore, the more the sensitive rules, the more the execution time is in the RRLR algorithm

7. Conclusion and future studies

By allocating weight to the transactions and items, the proposed algorithm has a more effective item in hiding the sensitive association rules, and removes it from a transaction with the highest weight that causes to reduce the number of removed items, the number of lost rules, and the number of ghost rules in the HSARWI algorithm. By reducing the number of removed items, the number of updates in calculating the support and confidence of rules are reduced, and this leads to a reduction in the execution time. Since the HSARWI and FHSAR algorithms have a function to predict failure, hiding failure is equal to 0 for them but the RRLR algorithm may undergo failure due to inserting item. It is possible to prevent the frequent calculation of support and confidence of rules after changing each transaction through adding the ability of calculating the number of required changes to hide the rule at the beginning of the implementation operation.

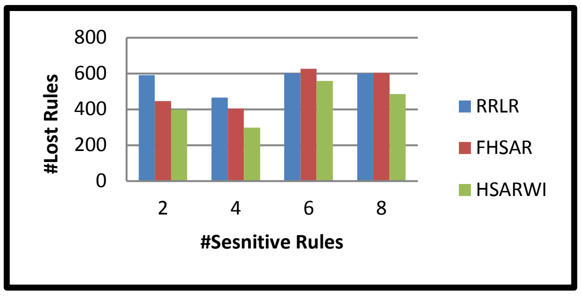

Figure 3. Lost rules in Mushroom with MST = 0.5 and MCT = 0.75.

.

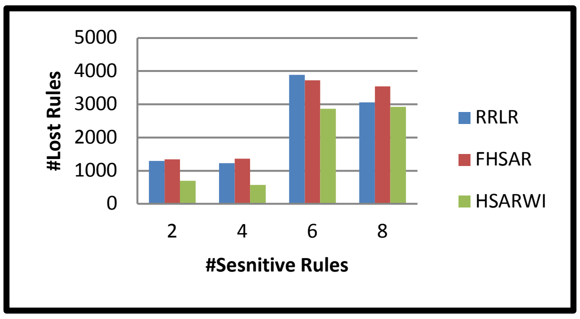

Figure 4. Lost rules in Mushroom with MST = 0.4 and MCT = 0.6.

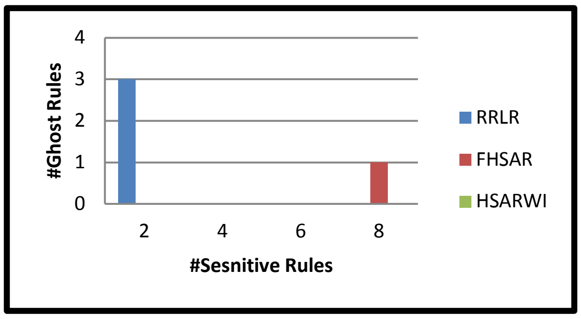

Figure 5. Ghost rules in Mushroom with MST = 0.5 and MCT = 0.75.

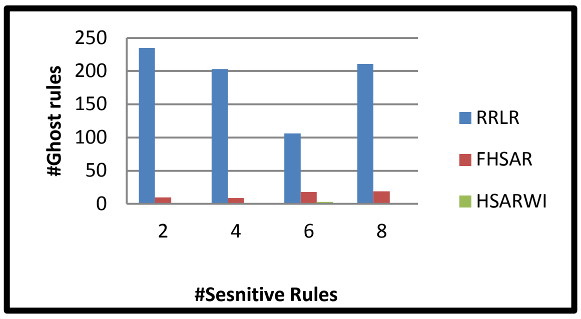

Figure 6. Ghost rule in Mushroom with MST = 0.4 and MCT = 0.6.

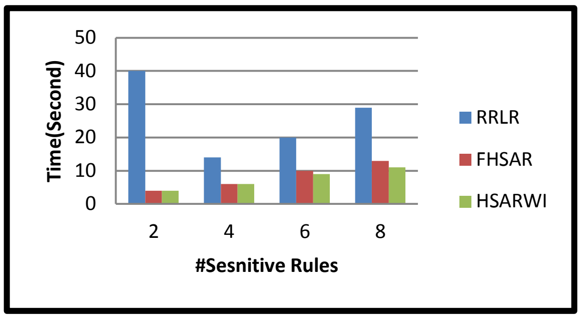

Figure 7. Execution time in chess with MST = 0.95 and MCT = 0.98.

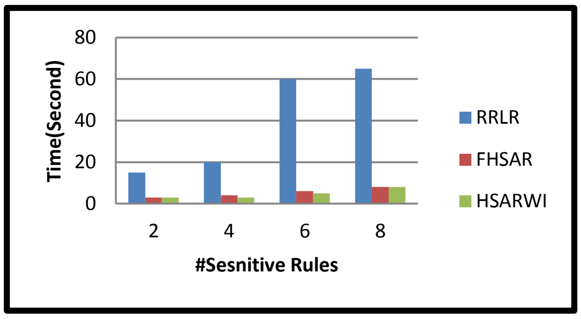

Figure 8. Execution time in chess with MST = 0.88 and MCT = 0.92.

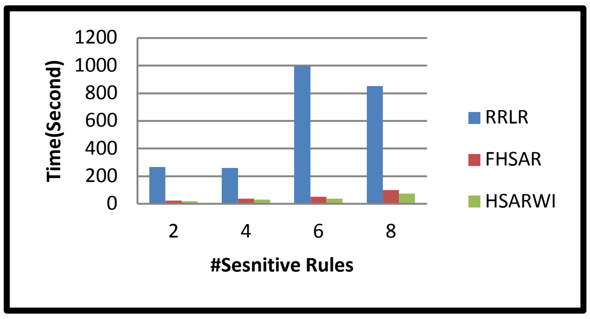

Figure 9. Execution time in Mushroom with MST = 0.5 and MCT = 0.75

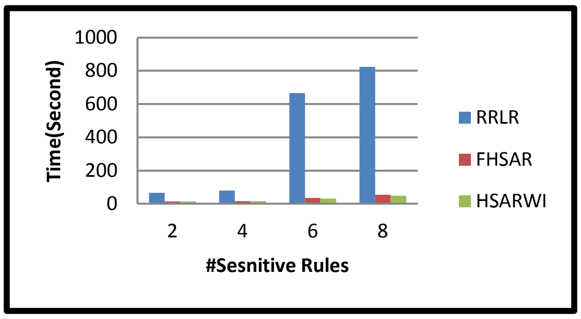

Figure 10. Execution time in Mushroom with MST = 0.4 and MCT = 0.6.

References

[1] Sathiyapriya, K. & Sadasivam, G. S. (2013). A survey on privacy preserving association rule mining. International Journal of Data Mining & Knowledge Management Process, vol. 3, no. 2, pp. 119-131.

[2] Han, J., Kamber, M. & Pei, J. (2011). Data mining: concepts and techniques: concepts and techniques. Elsevier.

[3] Gkoulalas-Divanis, A. & Verykios, V. S. (2010). Association rule hiding for data mining. Springer Science & Business Media.

[4] Gkoulalas-Divanis, A., Haritsa, J., & Kantarcioglu, M. (2014). Privacy issues in association rule mining. In Frequent Pattern Mining. Springer International Publishing.

[5] Lee, G. & Chen, Y. C. (2012). Protecting sensitive knowledge in association patterns mining. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 2, no. 1, pp. 60-68.

[6] Shah, K., Thakkar, A. & Ganatra, A. (2012). A study on association rule hiding approaches. IJEAT International Journal of Engineering and Advanced Technology, vol. 1, no. 3, pp. 72-76.

[7] Weng, C. C., Chen, S. T. & Lo, H. C. (2008). A novel algorithm for completely hiding sensitive association rules. In Intelligent Systems Design and Applications, ISDA'08. Eighth International Conference. IEEE, 2008.

[8] Shah, K., Thakkar, A. & Ganatra, A. (2012). Association Rule Hiding by Heuristic Approach to Reduce Side Effects and Hide Multiple R. H. S. Items. International Journal of Computer Applications, vol. 45, no. 1.

[9] Atallah, M., Bertino, E., Elmagarmid, A., Ibrahim, M. & Verykios, V. (1999). Disclosure limitation of sensitive rules. In Knowledge and Data Engineering Exchange, (KDEX'99) Proceedings. Workshop on. IEEE, pp. 45-52.

[10] Verykios, V. S. (2013). Association rule hiding methods. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, vol. 3, no. 1, pp. 28-36.

[11] Dasseni, E., Verykios, V. S., Elmagarmid, A. K. & Bertino, E. (2001). Hiding association rules by using confidence and support. In Information Hiding. Springer Berlin Heidelberg. pp. 369-383.

[12] 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.

[13] Oliveira, S. R. & Zaiane, O. R. (2002). Privacy preserving frequent itemset mining. In Proceedings of the IEEE international conference on Privacy, security and data mining. Australian Computer Society, 2002.

[14] Oliveira, S. R. & Zaiane, O. R. (2003). Protecting sensitive knowledge by data sanitization. IEEE 13th International Conference on Data Mining, 2003.

[15] Verykios, V. S., Elmagarmid, A. K., Bertino, E., Saygin, Y. & Dasseni, E. (2004). Association rule hiding. Knowledge and Data Engineering, IEEE Transactions on, vol. 16, no. 4, pp. 434-447.

[16] Wang, S. L., Patel, D., Jafari, A. & Hong, T. P. (2007). Hiding collaborative recommendation association rules. Applied Intelligence, vol. 27, no. 1, pp. 67-77.

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

[18] Verykios, V. S., Pontikakis, E. D., Theodoridis, Y. & Chang, L. (2007). Efficient algorithms for distortion and blocking techniques in association rule hiding. Springer Distributed and Parallel Databases, vol. 22, no. 1, pp. 85-104.

[19] Dehkordi, M. N., Badie, K. & Zadeh, A. K. (2009). A novel method for privacy preserving in association rule mining based on genetic algorithms. Journal of software, vol. 4, no. 6, pp. 555-562.

[20] Modi, C. N., Rao, U. P. & Patel, D. R. (2010). Maintaining privacy and data quality in privacy preserving association rule mining. In Computing Communication and Networking Technologies (ICCCNT), International Conference. IEEE, 2010.

[21] Jain, D., Khatri, P., Soni, R. & Chaurasia, B. K. (2012). Hiding sensitive association rules without altering the support of sensitive item (s). In Advances

in Computer Science and Information Technology. Networks and Communications. Springer Berlin Heidelberg, vol. 84, pp. 500-509.

[22] Gulwani, P. & Aloney, M. R. (2013). Securing Sensitive Rule by Changing the SC Values. Journal of Engineering, Computers & Applied Science, vol. 2, no. 7, pp. 12-17.

[23] Kryszkiewicz, M. (1998). Representative association rules. Research and Development in Knowledge Discovery and Data Mining. Springer Berlin Heidelberg, pp. 198-209.

[24] Domadiya, N. H., & Rao, U. P. (2013). Hiding sensitive association rules to maintain privacy and data quality in database. In Advance Computing Conference (IACC), IEEE 3rd International, 2013.

[25] Le, H. Q., Arch-Int, S., Nguyen, H. X. & Arch-Int, N. (2013). Association rule hiding in risk management for retail supply chain collaboration. Computers in Industry, vol. 64, no. 7, pp. 776-784.

[26] Shah, R. A. & Asghar, S. (2014). Privacy preserving in association rules using a genetic algorithm. Turkish Journal of Electrical Engineering & Computer Sciences, vol. 22, no. 2, pp. 434-450.

[27] Saygin, Y., Verykios, V. S. & Elmagarmid, A. K. (2002). Privacy preserving association rule mining. In Research Issues in Data Engineering: Engineering E-Commerce/E-Business Systems, RIDE-2EC Proceedings. Twelfth International Workshop, IEEE, pp. 151-158.

[28] Wang, H. (2013). Quality Measurements for Association Rules Hiding. AASRI Procedia, vol. 5, pp. 228-234.

نشریه هوش مصنوعی و داده کاوی

ارائه الگوریتمی جهت پنهان‌سازی قواعد وابستگی حساس با استفاده از تکنیک آشفته‌سازی

مریم ساکنیان دهکردی و محمد نادری دهکردی*

[۱] دانشکده مهندسی کامپیوتر، دانشگاه آزاد اسلامی، واحد نجف آباد، نجف آباد، ایران.

ارسال ۲۰۱۵/۰۶/۲۱؛ پذیرش ۲۰۱۶/۰۱/۱۸

چکیده:

به دلیل رشد بسیار سریع تکنولوژی داده‌کاوی به دست آوردن اطلاعات خصوصی کاربران از طریق این تکنولوژی آسان‌تر می‌شود. کاوش قواعد وابستگی یکی از تکنیک‌های داده کاویست که الگوهای مفید را در قالب قواعد وابستگی استخراج می‌کند. از مشکلات مهم اعمال این تکنیک روی پایگاه‌های داده، افشاء شدن اطلاعات حساس است که امنیت و محرمانگی آن‌ها را به خطر می‌اندازد. پنهان‌سازی قواعد وابستگی یکی از روش‌های حفظ حریم خصوصی و موضوعی مهم در زمینه داده‌کاوی و امنیت پایگاه‌داده می‌باشد که الگوریتم‌های متعددی با رویکردهای مختلف برای آن ارائه شده است. در این مقاله الگوریتمی به‌منظور پنهان‌سازی قواعد وابستگی حساس با رویکرد اکتشافی ارائه می‌گردد که از تکنیک آشفته‌سازی، مبتنی بر کاهش اطمینان یا پشتیبانی قواعد، استفاده‌کرده و سعی می‌کند با اختصاص وزن به عنصرها و تراکنش‌ها، عنصر موردنظر را از تراکنشی با بیشترین وزن حذف کند. کارایی با معیارهای شکست پنهان‌سازی، تعداد قواعد گم شده، تعداد قواعد غیرواقعی و زمان اجرا سنجیده می‌شود. نتایج بدست آمده با دو الگوریتم شناخته شده FHSAR وRRLR، بر روی دوپایگاه‌داده واقعی(متراکم و غیر متراکم) مورد ارزیابی قرار می‌گیرد، نتایج نشان می‌دهد تعداد قواعد گم شده در تمامی آزمایش‌ها کاهش یافته که این کاهش به‌طور میانگین نسبت به الگوریتم RRLR برابر ۴۷٪ و نسبت به الگوریتم FHSAR برابر ۲۳٪ است. در سایر اثرات جانبی نامطلوب نیز عملکرد الگوریتم پیشنهادی در بدترین حالت با الگوریتم‌های پایه برابر است.

کلمات کلیدی: داده‌کاوی، پنهان‌سازی قواعد وابستگی، حفظ حریم خصوصی داده‌کاوی.