IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 3, Ver. I (May-Jun. 2014), PP 83-90
www.iosrjournals.org

Minimization of Side Effects of Knowledge Hiding Based On Association Rule Mining

Ms. Snehal V. Waghchaure [1], Mr. Nitin J. Khapale[2],Ms.Badhitala Thulasi[3]
[1]*(Computer Department, SRESCOE Kopargaon, India)*
[2]*(Electronics and telecommunication Department, KBP Polytechnic Kopargaon, India)*
[3]*(Computer Department, SRESCOE Kopargaon, India)*

Abstract: *Privacy is very important in Data Mining. Association rule mining, is a important technique, has already been applied in a wide range of areas. Large databases cannot be secured by encryption or other techniques because of complexities. These security issues can be divided into two categories: data hiding and knowledge hiding. This paper deals with the problem of association rule mining which preserves the confidentiality of each database i.e. Knowledge hiding. Knowledge hiding is concerned with the sanitization of confidential knowledge from the data. There are many methods to solve this problem. The one, presented in the paper is called support-based and confidence-based blocking schemes. Apriori algorithm is used for mining rule from database. Knowledge hiding is achieved using ISL and DSR. The improvement over sensitive rule hiding is proposed to offer more accuracy and security with smaller negative impact.*

Keywords: *Association rule mining, rule hiding with minimum side effects, ISL, DSR.*

I. INTRODUCTION

In ever changing business environment like enterprises e-businesses, the old fashioned disclosure and database inference protection techniques are not capable to ensure complete data privacy. Privacy is important for the on line disclosure of private information Organizations should be able to evaluate the risk of disclosing information and should adopt new more efficient approaches for information disclosure control, in order to maintain their competitive edge in the market. Not only the data but also the hidden knowledge in the data should be made secure. The sensitive information can be extracted in the form of association rules with association rule mining tools. But this can jeopardize the privacy of the customers. Some other rule could be very critical for the company itself such as buying patterns of very rich customers. For example, an Internet based company may give up selling hardware and may concentrate on selling books and videos. Therefore a rule relating the customer buying patterns of hardware may no longer be sensitive for that company. It is desirable to apply a trivial algorithm that hides data by deleting it randomly.

Data Mining is the process of discovering new patterns from large data sets involving methods from statistics and artificial intelligence but also database management. Privacy has become an important issue in Data Mining. Many methods have been brought out to solve this problem. This paper focuses on the problem of association rule mining, which hides the knowledge from each database. This paper reviews the major method of knowledge hiding. Association rule mining, as a very important technique, has already been applied in a wide range of areas. Knowledge hiding is concerned with the sanitization of confidential knowledge from the data. As a result of association rule mining, many useful association rules will be discovered, but at the same time, many privacy rules will also be exposed which do not want others to know. To solve this, limit the mining process, in order to keep these sensitive rules being hidden. There are so many methods to solve this problem. The one that is covered in the paper is just one kind of them, called support-based and confidence-based blocking schemes. Algorithm used For Knowledge Hiding is ISL algorithm or DSR algorithm [2]. The main disadvantage of a block in algorithm is the fact that the dataset, apart from the blocked values, is not distorted. Thus, an adversary can disclose the hidden rules by identifying those generating item sets that contain question marks and lead to rules with a maximum confidence that lies above the minimum confidence threshold. If the number of these rules is small then the probability of identifying the sensitive ones among them becomes high. To avoid this problem, a method for selectively removing individual values from a database is introduced to prevent the discovery of a set of rules, while preserving the data for other applications. The problem of building privacy preserving algorithms is considered for one category of data mining techniques, the association rule mining. New metrics is introduced in order to demonstrate how security issues can be taken into consideration in the general framework of association rule mining [1].

II. RELATED WORK

Successful applications of data mining have been demonstrated in marketing, business, medical analysis, product control, engineering design and scientific exploration, among others. While all of these applications of data mining can benefit commercial, social and human activities, there is also a negative side to this technology, the threat to data privacy. Knowledge hiding is concerned with the sanitization of confidential knowledge from the data. As a result of association rule mining, many useful association rules will be discovered, but at the same time, many privacy rules will also be exposed which do not want others to know. To solve this, limit the mining process, in order to keep these sensitive rules being hidden. There are so many methods to solve this problem. The one that is covered in the report is just one kind of them, called support-based and confidence-based blocking schemes.

Algorithm used for Knowledge Hiding is ISL algorithm or DSR algorithm [2]. The main disadvantage of a blocking algorithm is the fact that the dataset, apart from the blocked values, is not distorted. Thus, an adversary can disclose the hidden rules by identifying those generating item sets that contain question marks and lead to rules with a maximum confidence that lies above the mini- mum confidence threshold. If the number of these rules is small then the probability of identifying the sensitive ones among them becomes high.

Chris Clifton presents a method to avoid this problem. A method for selectively removing individual values from a database is introduced to prevent the discovery of a set of rules, while preserving the data for other applications. The problem of build-in privacy preserving algorithms is considered for one category of data mining techniques, the association rule mining. New metrics is introduced in order to demonstrate how security issues can be taken into consideration in the general framework of association rule mining [1].

To overcome the side effects of ISL such as which are false rule generation and useful Rules are lost, enhancements in existing system are explained in detail in paper [6] by Yi-Hung Wu, Chia-Ming Chiang, and Arbee L.P. Chen, where the minimum number of modifications to be done to the database can be calculated and applied to the database, which will produce minimum distortion in original database.

III. PROGRAMMER'S DESIGN

The breakdown structure mainly focuses on following areas-

1. Module1: Rule Mining and Rule Minimization
2. Module 2: Implementation of ISL and DSR Algorithms
3. Module 3: Enhancement in existing system

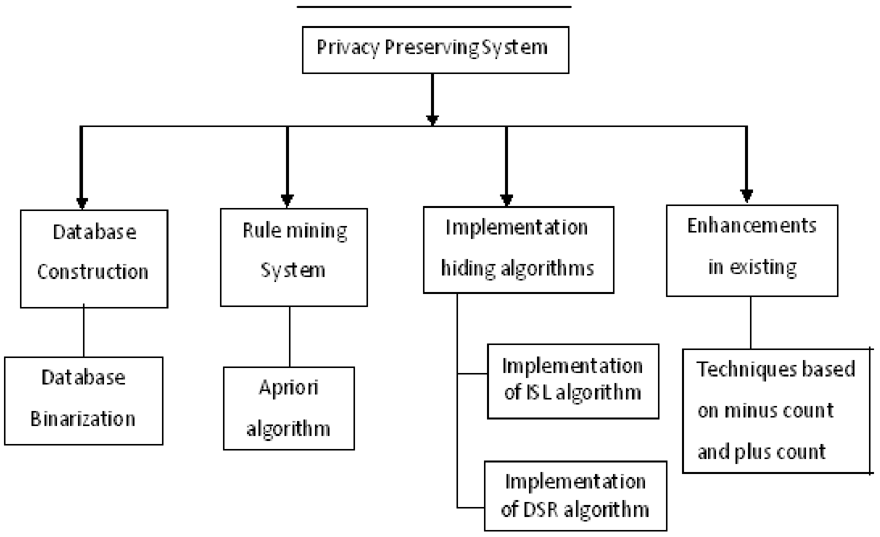
Privacy Preserving System
Database Construction
Rule mining System
Implementation hiding algorithms
Enhancements in existing
Database Binarization
Apriori algorithm
Implementation of ISL algorithm
Implementation of DSR algorithm
Techniques based on minus count and plus count

Fig. 1 Work Breakdown Structure

3.1. Rule Mining and Rule Minimization

Mining frequent item sets and association rules is a popular and well researched method for discovering interest in relations between variables in large databases. Based on the concept of strong rules, Agrawal, Imielinski, and Swami (1993) introduced the problem of mining association rules from transaction data as follows:

Let I = i1, i2 . . . in be a set of items. Let D = t1, t2 ... tm 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. The sets of items for short itemsets X and Y is called antecedent i.e left hand side or LHS and consequent i.e. right hand side or RHS of the rule. To select interest- in rules from the set of all possible rules, constraints on various measures of significance and interest can be used. The best known constraints are minimum thresh olds on support and confidence. The support supp(X) of an itemset X is defined as the proportion of transactions in the data set which contain the itemset. The confidence of a rule is defined as,

$$\text{Confidence } (X \rightarrow Y) = \text{support}(X \cup Y) / \text{support}(X)$$

In terms of Association rule mining, any rule will be discovered, as long as satisfy these two conditions:
1. Support $(A \rightarrow B) \geq$ Minsup.
2. Confidence $(X \rightarrow Y) = (\text{Support}(X \cup Y)) / (\text{Support}(X)) \geq$ Minconf.

A frequent itemset is an itemset whose support is greater than some user-specified minimum support. Since the definition of support enforces that all subsets of a frequent itemset have to be also frequent, it is sufficient to only mine all maximal frequent itemsets, defined as frequent itemsets which are not proper subsets of any other frequent itemset.[5] Large amounts of data have been collected routinely in the course of day-to-day management in business, administration, banking, the delivery of social and health services, environmental protection, security and in politics. Such data is primarily used for accounting and for management of the customer base. One requires robust, simple and computationally efficient tools to extract information from such data sets. The development and understanding of such tools is the core business of data mining. The Apriori Algorithm is an influential algorithm for mining frequent itemsets for boolean association rules.

Key Concepts of Apriori:
1. Frequent Itemsets: The sets of item which has minimum support (denoted by Li for ith-Itemset).
2. Apriori Property: Any subset of frequent itemset must be frequent.
3. Join Operation: To find Lk, a set of candidate k- itemsets is generated by joining Lk-1with itself.

Uses a Level-wise search, where k-itemsets (An itemset that contains k items is a k-itemset) are used to explore (k+1)-itemsets, to mine frequent itemsets from transactional database for Boolean association rules. First, the set of frequent 1-itemsets is found. This set is denoted L1. L1 is used to find L2, the set of frequent 2-itemsets, which is used to fine L3, and so on, until no more frequent k item sets can be found.

- Find all frequent itemsets: Each support S of these frequent itemsets will at least equal to a pre-determined min-sup (An itemsetis a subset of items in I, like A)
- Generate strong association rules from the frequent itemsets: These rules must be the frequent itemsets and must satisfy min-sup and min-conf [5].

The Apriori Algorithm: Pseudo code

```
1. Scan the transaction database to get the support count of all the items.
2. Compare the support of each item with minimum support and generate frequent-1 itemset L1.
3. Join Lk-1 with Lk-1to generate next candidate itemset.
4. Use the Apriori property to prune the itemsets, which are not frequent.
5. Scan the transaction database to get the support count of candidate k-items and compare the support
   of each item with minimum support and generate frequent-k itemset Lk.
6. Check whether Candidate itemset is null. If no goto step3.
7. Generate strong association rules from the frequent itemsets.
```

Table 1 Sample Database

TID	Items
T1	ABC
T2	ABC
T3	ABC
T4	AB
T5	A
T6	AC

Table 2 Support Count of each itemset

TID	Items
A	66
B	66
C	66
AB	66
BC	50
AC	66
ABC	50

Table 3 The rules derived from the Table 2

Rules	Confidence	Support
B→A	100%	66%
B→C	75%	50%
C→A	100%	66%
C→B	75%	50%
B→AC	75%	50%
C→AB	75%	50%
AB→C	75%	50%
AC→B	75%	50%
BC→A	100%	50%

3.2. **ISL and DSR algorithms:**

Wang and Jafari have proposed two modification schemes that incorporate unknowns and aim at the hiding of predictive association rules, i.e. rules containing the sensitive items on their LHS. Both algorithms rely on the distortion of a portion of the database transactions to lower the confidence of the association rules. In order to hide a rule A → B, one can either decrease the support of the itemset AB below the minimum sup- port threshold, or one can decrease the confidence below the minimum confidence threshold. Let I = i1... in be a set of literals, called items. Let D be a set of transactions which is the database that is going to be disclosed. Each transaction $t \subset D$ is an itemset such that $t \subseteq I$. A unique identifier, called as TID, is associated with each transaction. It is said that a transaction t supports X, a set of items in I, if $X \subseteq t$. It is assumed that the items in a transaction or an itemset, are sorted in lexicographic order. Each row in the table represents a transaction. There are 3 items, and 6 transactions. AB is an itemset, and transaction T1 supports that itemset. An itemset X has support s if s% of the transactions support X. Support of X is denoted as Supp (X). All possible itemsets and their supports obtained from the database in Table 2 are listed in Table 3. For example, itemset BC is supported by 3 transactions out of 6, and therefore has 50% supports [3, 4]

The Hiding Strategies: The hiding strategies, that are proposed, heavily depend on finding transactions that fully or partially support the generating itemsets of a rule. The reason for this is that if a rule is to be hidden, one need to change the support of some part of the rule (i.e. decrease the support of the generating itemset). Another issue is that the changes in the database introduced by the hiding process should be limited. The decrease in the support of an itemset S can be done by selecting a transaction t, that supports S and by setting to 0 at least one of the non-zero values of t values of items that represent items in S. The increase in the support of an itemset S can be accomplished by selecting a transaction t that partially supports it and setting to 1 the values of all the items of S in t values of items. In order to be able to identify some viable ways for reducing either the support or the confidence of a rule, it needs to analyze the formulae that have already presented for the confidence and the support. Both the confidence and the support are expressed as ratios of supports of itemsets that support the two parts of a rule or its generating itemset. In this way to lower the value of a ratio, there is either one of the following options:

(a) Decrease the numerator, while keeping the denominator fixed, or

(b) Increase the denominator while keeping the numerator fixed.

First consider which one of the above option (a) can be adopted for decreasing the support of a rule $X \rightarrow Y$. The support of this rule is $((X \cup Y)*100)/N \geq N$ where N is the number of transactions in D. Since N is constant, this means that only numerator can be changed. For this reason one can only adopt option (a) from above, which is to decrease the numerator (support) of the generating itemset of the rule. As an example, consider the database in Table 1 and the rules in Table 3 that are obtained from this database. Given that min supp=33% and min conf=70% the interest is to hide the rule AC→B, with support = 50%, and confidence = 75%. In order to decrease the support of the rule AC→B, select the transaction t= (T1, [111], 3) and turn to 0 one of the elements in the list of items that corresponds to A or to B or C. The element corresponding to C is set to 0, obtaining t= (T1, [110], 2). The rule AC→B has been hidden (support=33%, confidence=66%). Following one need to investigate which of the three options can be adopted for decreasing the confidence of a rule X→Y. The confidence of this rule is written as$((X \cup Y)*100)/(X)$.In order to see whether any of the options that were mentioned previously can be applied to this case or not, one need to investigate if any of the pairs of the actions stated in each option are conflicting, based on the relationship between the numerator and the denominator in the confidence ratio. In option (a) we need to decrease the support of numerator, while the support of the itemset in the left hand side of the rule remains fixed. It can be achieved by modifying the transactions that support this itemset. Again, let's consider the rule AC→B in Table 3. In order to decrease the confidence, select the transaction t = (T1, [111], 3) and turn to 0 the element of the list of items that corresponds to B. The transaction becomes t= (T1, [101], 2) and AC→B with support=33% and confidence=50% is obtained, which means that the rule is hidden. Option (b) implies that one need to increase the denominator (Which is the support of the itemset in the antecedent) of the rule, while the support of the generating itemset of the rule remains fixed. This option (b) is also applicable, since the support of the rule antecedent can be increased while keeping the support of the generating itemset fixed by modifying the transactions that partially support the itemset in the antecedent of the rule but do not fully support the itemset in the consequent.

Let's consider the rule AC→B in Table 3 one more time. In order to decrease the confidence, select the transaction t = (T5, [100], 1) and turn to 1 the element of the list of item that corresponds to C. It results into t = (T5, [101], 2). Now, the rule AC→B has support=50% and confidence=60%, which means that the rule has been hidden since its confidence is below the min_conf threshold. [3] The strategies that are developed, based on the observations below, can be summarized as follows:

1. Decrease the confidence of the rule:

(a) By increasing the support of the rule antecedent X through transactions that partially support it.

(b) By decreasing the support of the rule consequent Y in transactions that support both X and Y.

2. Decrease the support of the rule by decreasing the support of either the rule antecedent X, or the rule consequent Y, through transactions that fully support the rule

Algorithm ISL:

The first algorithm hides the sensitive rules according to the first strategy. For each selected rule, it increases the support of the rule's antecedent until the rule confidence decreases below the min_conf threshold. [2]

INPUT: A set RH of rules to hide, the source database D, the number of transactions in D, the min_conf threshold, and the min_supp threshold.
OUTPUT: The database D transformed so that the rules in RH cannot be mined.

```
Begin
For each rule r in RH do
1. T’lr= t in D / t partially supports lr
2. For each transaction of T’lr count the number of items of lr in it.
3. Sort the transactions in T’lr in descending order of the number of items of lr supported.
4. Repeat until Conf(r) ≤ min conf
{
5. Choose the transaction t∈ T’lr with the highest number of items of lr supported (t is the first transaction in T’lr).
6. Modify t to support lr
7. Increase the support of lr by 1
8. Recomputed the confidence of r
9. Remove t from T’lr
}
10. Remove r from RH
}
```

3.2.1. Algorithm DSR:

This algorithm decreases the support of the sensitive rules until either their confidence is below the min_conf threshold or their support is below the min supp thresh- old. [2]
INPUT: A set RH of rules to hide, the source database D, the size of the database, the min conf threshold, the min supp threshold
OUTPUT: The database D transformed so that the rules in RH cannot be mined

```
Begin
For each rule r in RH do
1. T’r= t in D / t fully supports r
2. For each transaction of T’r count the number of items of r in it.
3. Sort the transactions in T’r in ascending order of the number of items of lr supported.
4. Repeat until Conf(r) ≤ min conf
{
5. Choose the transaction t ∈ T’r with the lowest num- ber of items of lr supported (t is the first transaction in T’r).
6. Modify t to support lr
7. Decrease the support of lr by 1
8. Recomputed the confidence of r
9. Delete j from t
10. Remove t from T’r
}
11. Remove r from RH
}
```

3.3. **Enhancements in the existing system**

Undesired side effects e.g. non-sensitive rules falsely hidden and spurious rules falsely generated, may be produced in the rule hiding process. This module focuses on development of system that strategically modifies a few transactions in the transaction database to decrease the supports or confidences of sensitive rules without producing the side effects. Since the correlation among rules can make it impossible to achieve this goal. Heuristic methods are considered for increasing the number of hidden sensitive rules and reducing the number of modified entries. In order to apply sensitive rule optimization, we need to try permutations of ISL and DSR algorithms on all records. This will take so much time if we do not know how many minimum records to modify that will yield proper result. So we shall make use of the other paper [6] that gives us formulae to calculate Plus Count (PC) and Minus Count (MC) representing mini- mum modifications required to keep the damage at the minimum Undesired side effects, e.g. non-sensitive rules falsely hidden and spurious rules falsely generated, may be produced in the rule hiding process. This module focuses on development of system that strategically modifies a few transactions in the transaction database to decrease the supports or confidences of sensitive rules without producing the side effects. Since the correlation among rules can make it impossible to achieve this goal. Heuristic methods are considered for increasing the number of hidden sensitive rules and

reducing the number of modified entries. In order to apply sensitive rule optimization, we need to try permutations of ISL and DSR algorithms on all records. This will take so much time if we do not know how many minimum records to modify that will yield proper result. So we shall make use of formulas to calculate Plus Count (PC) and Minus Count (MC) representing minimum modifications required to keep the damage at the minimum. The properties are given that will be used to identify the valid modifications. The following properties indicate the minimal number of transactions that should be modified.
Modification Schemes for Rule Hiding Scheme 1: Modify entries from 1s to 0s. If an item in X∪Y is deleted from a transaction containing X∪Y, Sup(X→Y) and Conf(X→Y) will be decreased. X→Y is hidden if we repeat this operation until one of the conditions for minimum support and threshold holds.

Scheme 2: Modify entries from 0s to 1s.
Conf(X→Y) Will be decreased if we insert an item i∈X into a transaction that contains X but does not contain Y. X→Y is hidden if we repeat this operation until condition for minimum confidence holds.

Properties for Rule Hiding:
Property 1: Let EX∪Y be the set of all transactions containing X∪Y. To hide X → Y by removing items from the transactions in EX∪Y, the minimal number of transactions that should be modified, called the minus support count, is computed as:

$$MSC(x \to y) \ = C(x \cup y) \ - \ [|D| \ \times \ MST] + 1$$

Property 2: To hide X→Y by removing items in Y from the transactions in EX∪Y, the minimal number of transactions that should be modified, called the minus consequent confidence count, is computed as:

$$MCCC(x \to y) \ = C(x \cup y) \ - \ [\, C_x \times \ MCT] + 1$$

Property 3: To hide X→Y by removing items in X from the transactions in EX∪Y, the minimal number of transactions that should be modified, called the minus precedent confidence count, is computed as:

$$MPCC(x \to y) = \ C(x \cup y) - C_x \times \frac{MCT}{1 - MCT} + 1$$

Definition 1:
Minus count: The minimum of MCCC(x→y) and MPCC(x→y) is called the minus confidence count of X→Y. The minimum of MSC(X→Y) and MCC(X→Y) is called the minus count of X→Y, MC(X→Y), indicating the minimal number of transactions to be modified.

Definition 2: Plus count: Compared with Scheme 1, Scheme 2 only uses the support of the precedent in a rule to decrease the confidence. Therefore, we consider PPCC(x→y) as the minimal number of transactions to be modified by Scheme 2 for hiding X→Y, and call it the plus count of rule X→Y, PC(X→Y).

For each sensitive rule, based on the minus count and the plus count, this approach can determine the minimal number of transactions that should be modified. In a similar way, the maximal number of transactions that can be modified without hiding a non-sensitive rule can be estimated. On the other hand, to avoid generating a spurious rule, we have the following properties to estimate the maximal number of transactions that can be modified by Schemes 1 and 2, respectively [6].

IV. RESULT AND DISCUSSION

In order to better understand the characteristics of the implemented algorithms numerically, we have performed a series of experiments to measure various effects. The following effects are considered: time effects, side effects, database effects. For time effects, we have measured the running time required to hide a particular rule for three methods Hide using No False Rules, Hide using No Lost Rules and hide using both methods. For side effects, We have measured the hiding failures, number of new rules generated and lost rules. The hiding failure side effect measures the number of sensitive association rules that cannot be hidden. The new rule side effect measures the number of new rules appeared in the transformed database but is not in the original database. The lost rule side effect measures the number of rules that are in the original database but not in the transformed database. The database effects measure the percentage of altered transactions in the database.

For each data set, various sets of association rules are generated under various minimum supports and minimum confidences. The minimum support range is from 20% to 70%. The minimum confidence range is from 50% to 70%. Total number of association rules is from 4 to 951. The Percentage of hidden rules ranges from 0 to 100. In Figure 2 Effect of Hiding with no False Rule is shown. From that we can say that Hiding with no False Rule method shows 0% False Rules, near about 36% rules are lost while hiding is being processed and 94% Rules can be hidden using this method. If Owner of database wants to prevent the false rules and want to achieve more hiding chances, this approach is best suited for this purpose.

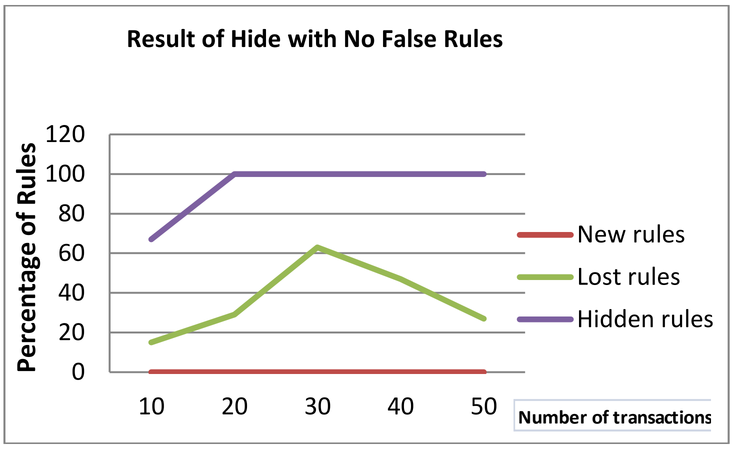

Fig. 2 Effect of Hiding with no False Rule

In Figure 3 Effect of Hiding with no Lost Rule is shown. From that we can say that Hiding with no Lost Rule method shows 0% Lost Rules, few i.e. 3% new rules are generated while hiding is being processed and 48% Rules can be hidden using this method. If Owner of database wants to prevent the loss of original rules that is less distortion in original database then this approach is used for that purpose. Graph for percentage of rules hidden for different number of transactions shows so much variation in percentage because it depends on user's selection of sensitive rule. When a system encounter that while hiding a particular sensitive rule, if original rules are being lost then it undoes all the modifications and thus rule cannot be hidden. Thus, this percentage may vary according to selection of sensitive rule.

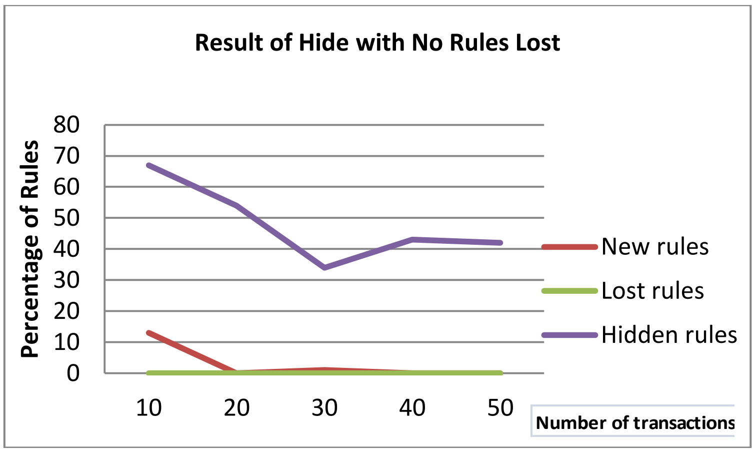

Fig.3 Effect of Hiding with no Lost Rule

In Figure 4 Effect of Hiding with no Lost Rule is shown. From that we can say that Hiding with no False rules and Lost Rule method shows 0% Lost Rules, 0% new rules while hiding is being processed and 54% Rules can be hidden using this method. If Owner of database want to prevent the loss of original rules as well as want to prevent the further new rules generation then this approach is used for that purpose.

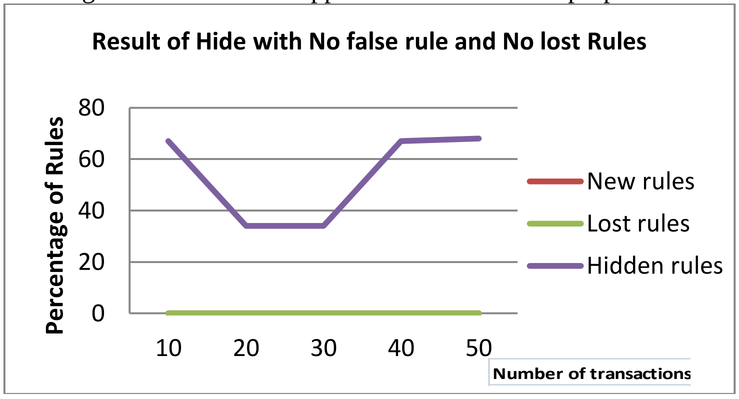

Fig.4 Effect of Hiding with Both

This section analyzes some of the features of the implemented algorithms based on my experimental results. The first feature observed is the time effects. On all the datasets we tested, Hide with no lost rules method

(Method-I) requires less running times than Hide with No False rules method (Method-II). This is because in hide with no lost rules method as soon as it discovers the rule is going to lost, it exit from the program. Of course rule can not be hidden in this case. The second feature observed is the side effects of both methods. The two methods has opposite side effects. Hide with no false rules method shows less hiding failure (6%), No new rules (0%) are generated and some rules (37%) are lost. However, Hide with no lost rules method (Method-II) shows hiding failure (52%), few new rules (3%) are generated and no rule (0%) is lost during hiding process. The new rules generated in second method include some rules that were previously hidden, but got generated again when trying to hide other rules. For example, if no hiding failure is required, Method-I should be adopted. However, if no lost rule is required, Method-II is a better choice. The third feature observed is the database effects. Method-I alters more transactions than that of method-II more that once because the changes observed in above tables are more for Method-I that is why time required for its execution is more. Hide with both methods (Method-III) shows 0% new rules and lost rules. It has 46% hiding failure. But the time required for this method is more as compared to Method-II.

V. CONCLUSION

Sharing of data is often beneficial, but is often prevented because of privacy and security concerns. This paper presents a technique to obscure a specific set of association rules, while minimizing the effect on the usefulness of the data for purposes other than learning those rules. Some approaches are implemented about privacy preserving while doing association rule mining. Here, Database is designed and serialized. Apriori algorithm is implemented in order to generate frequent itemsets and thus strong association rules from the input transactional database. Sensitive Rule hiding is implemented using ISL and DSR algorithms and optimization technique is implemented to get a better result, with smaller negative impact. Two methods i.e. Hide with No False Rules and Hide with No Lost Rules are implemented using ISL and DSR algorithms. Method-I shows false rules are not generated (0%) and Method-II ensures no rule is lost (0%).Method-I has much (94%) hiding success but requires more time as compared to Method-II because it alters more transaction. If no hiding failure is required, Method-I should be adopted. However, if no lost rule is required, Method-II is a better choice.

ACKNOWLEDGEMENT

I am very much thankful to my respected project guide Prof. P. N. Kalavadekar, for his ideas and help proved to be valuable and helpful during the creation of this dissertation work. I would like to thank all the faculties who have helped me during my research work. Lastly, I am thankful to my friends who shared their knowledge in this field with me.

REFERENCES

[1] Tinghuai Ma, Sainan Wang, Zhong Liu, "Privacy Pre serving Based on Association Rule Mining", 3rd Inter- national Conference on Advanced Computer Theory and Engineering, 2010
[2] S.L.Wang and A. Jafari. "Hiding informative association rule sets". Expert Systems with Applications 33 (2007) 316-323
[3] Y Saygin, V. S. Verykios, and A. K. Elmagarmid. "Privacy preserving association rule mining". In Proceedings of the 2002 International Workshop on Research Issues in Data Engineering: Engineering Ecommerce E-Business Systems (RIDE 2002).
[4] V.S.Verykios, A.K. Elmagarmid, E.Bertino,Y. Saygin and E. Dasseni, "Association Rule Hiding", Jan 7, 2003.
[5] J.Han, M. Camber, "Data Mining Concepts and Techinques", Morgan Kaufmann Publishers, 2001.
[6] Yi-Hung Wu, Chia-Ming Chiang and Arber L.P. Chen, Senior Member, IEEE Computer Society, "Hiding Sensitive Association Rules with Limited Side Effects."
[7] Markus He gland, CMA, Australian National University, "The Apriori Algorithm-a Tutorial".
[8] Apriori Algorithm, www.wikipedia.org.
[9] Rakesh Agrawal, Ramkrishnan Shrikant, IBM Research Center, Privacy-Preserving Data Mining