International Journal of Engineering Research and Development
e-ISSN: 2278-067X, p-ISSN: 2278-800X, www.ijerd.com
Volume 9, Issue 5 (December 2013), PP. 51-59

BSS Homomorphic Encryption: A Privacy model for large transactional database

R.Sakthi[1], V.Umarani[2]
[1]*Mphil Scholar Department of Computer Science Sri Ramakrishna College of Arts and Science for Women Coimbatore, Tamilnadu, India.*
[2]*Assistant Professor Department of Computer Science Sri Ramakrishna College of Arts and Science for Women Coimbatore, Tamilnadu, India.*

Abstract:- Data mining is the extraction of interesting patterns or knowledge from huge amount of data. In recent years, privacy issues in data mining have been increased enormously especially when internet is booming with social networks, e-commerce, forums, blocks, etc. Because of privacy issues the personal information collected from the users are used in unethical way that leads to information insecurity. Hence Privacy Preserving Data Mining is a research area concerned with the privacy driven from personally identifiable information when considered for data mining. The Rob Frugal method is introduced to overcome the privacy vulnerabilities of outsourced data. It is an encryption scheme, based on one to one substitution ciphers for items and adding fake patterns for database. Here the attackers/hackers can find data by guessing attack. However, it contains a number of fake patterns which leads to security issue in privacy preserving. To overcome this problem, the proposed strategy encompasses Blind Source Separation Homomorphic encryption in order to reduce the number of fake patterns and to improve the security level for outsourced data with less complexity. To avoid guessing attack from unauthorized users, the encrypted data are converted into matrix format. Our comprehensive experiments on a very large and real transaction database demonstrate that our techniques are effective, scalable, and protect privacy.

Keywords:- Association rule mining, privacy-preserving, fake pattern partitioning, grouping.

I. INTRODUCTION

Spurred by developments such as cloud computing, there has been considerable recent interest in the paradigm of data mining-as-a-service [15]. The scope of information technologies and the internet in the past two decades has carried a wealth of individual information into the hands of commercial companies and government agencies. Data owners constantly seek to make better use of the data they possess, and utilize data mining tools to extract useful knowledge and patterns from the data. Data mining is one of the top upward fields during the computer industry that deals with discovering the patterns from large data sets [2]. It is the major part of the knowledge discovery procedure and is used to mine human comprehensible information[16]. Mining is favorably used for a huge amount of data [6] [7] and associated with algorithms frequently require large data sets to generate quality models [1].

It is advantageous to achieve sophisticated analysis on tremendous volumes of data in a cost-effective way; there exist several serious security issues of the data-mining as- a-service paradigm. One of the main security issues is that the server has access to valuable data of the owner and may learn sensitive information from it. For example, by looking at the transactions, the server (or an intruder who gains access to the server) can learn which items are always co-purchased. However, both the transactions and the mined patterns are the property of the data owner and should remain safe from the server. This problem of protecting important private information of companies is referred to as corporate privacy [3]. Unlike personal privacy, which only considers the protection of the personal information recorded about individuals, corporate privacy requires that both the individual items and the patterns of the collection of data items are regarded as corporate assets and thus must be protected.

In this paper, we study the problem of outsourcing the association rule mining task within a corporate privacy-preserving framework. A substantial body of work has been done on privacy-preserving data mining in a variety of contexts. A common characteristic of most of the previously studied frameworks is that the patterns mined from the data (which may be distorted, encrypted, anonym-zed, or otherwise transformed) are intended to be shared with parties other than the data owner. The key distinction between such bodies of work and our problem is that, in the latter, both the underlying data and the mined results are not intended for sharing and must remain private to the data owner.

We adopt a conservative frequency-based attack model in which the server knows the exact set of items in the owner's data and additionally, it also knows the exact support of every item in the original data. Wong et

al. [14] was one of the early works on defending against the frequency-based attack in the data mining outsourcing scenario. They introduced the idea of using fake items to defend against the frequency-based attack; however, it was lacking a formal theoretical analysis of privacy guarantees, and has been shown to be flawed very recently in [12], where a method for breaking the proposed encryption is given. Therefore, in our previous and preliminary work [5], we proposed to solve this problem by using k-privacy, i.e., each item in the outsourced dataset should be indistinguishable from at least $k - 1$ items regarding their support.

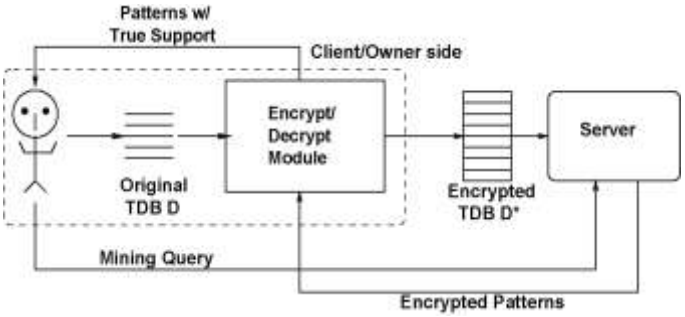

Fig.1. Architecture of mining-as-service paradigm.

In this paper, our goal is to devise an encryption scheme which enables formal privacy guarantees to be proved, and to validate this model over large-scale, real-life transaction databases. The architecture behind our model is illustrated in Figure 1. The client/owner encrypts its data using encrypt/decrypt (E/D) module, which can be essentially treated as a "black box" from its perspective. While the details of this module will be explained in Sec. V, it is responsible for transforming the input data into an encrypted database. The server conducts data mining and sends the (encrypted) patterns to the owner. Our encryption scheme has the property that the returned supports are not true supports. The E/D module recovers the true identity of the returned patterns as well their true supports. It is trivial to show that if the data is encrypted using 1-1 substitution ciphers (without using fake transactions), many ciphers and hence the transactions and patterns can be broken by the server with a high probability by launching the frequency-based attack. Thus, the major focus of this paper is to devise encryption schemes such that formal privacy guarantees can be proven against attacks conducted by the server using background knowledge, while keeping the resource requirements under control.

First we develop an encryption scheme, called BSS (Blind Source Separation) Homomorphic that the E/D module can employ to transform client data before it is shipped to the server.

Second, the E/D module partitions the transaction database into groups and adds fake transaction to it. Then the cipher items are transferred to the server. The E/D module recovers the true patterns and their correct support. Then it creates a matrix format for avoiding guessing attack.

Last but not least, we conduct experimental analysis of our schema using a large real data set. Our results show that our encryption schema is effective, scalable, and achieve the desired level of privacy.

Related work is described in the next section. The background on frequent pattern mining task is quickly reviewed in Sec. III. Privacy model are given in Sec. IV. Sec. V develops the encryption/decryption scheme we use. Sec. VI discusses the results of a comprehensive set of experiments conducted using real data sets. Finally, we conclude the paper and discuss directions for future research in Sec. VII.

II. RELATED WORK

Privacy-Preserving Data Mining is not applicable in a data stream environment which requires dynamic updating. For example, for a massive amount of income data, the execution efficiency of traditional methods can no longer respond to user demand. Furthermore, the potential infinite number of data streams plus limited memory space has constrained the traditional methods from obtaining the mining result with accuracy. In view of the above-mentioned issues, studies on Privacy- Preserving Data Stream Mining have become one of the important issues in the field of data mining. The research of privacy-preserving data mining (PPDM) has caught much attention recently.

The main model here is that private data is collected from a number of sources by a collector for the purpose of consolidating the data and conducting mining. The collector is not trusted with protecting the privacy, so data is subjected to a random perturbation as it is collected. Techniques have been developed for perturbing the data so as to preserve privacy while ensuring the mined patterns or other analytical properties are sufficiently close to the patterns mined from original data. This body of work was pioneered by [8] and has been followed up by several papers since [13]. This approach is not suited for corporate privacy, in that some analytical properties are disclosed.

Another related issue is secure multiparty mining over distributed datasets (SMPM). Data on which mining is to be performed is partitioned, horizontally or vertically, and distributed among several parties. The partitioned data cannot be shared and must remain private but the results of mining on the "union" of the data are shared among the participants, by means of multiparty secure protocols [10], [9], [11]. They do not consider third parties. This approach partially implements corporate privacy, as local databases are kept private, but it is too weak for our outsourcing problem, as the resulting patterns are disclosed to multiple parties.

The particular problem attacked in our paper is outsourcing of pattern mining within a corporate privacy-preserving framework. A key distinction between this problem and the above mentioned PPDM problems is that, in our setting, not only the underlying data but also the mined results are not intended for sharing and must remain private. In particular, when the server possesses background knowledge and conducts attacks on that basis, it should not be able to guess the correct candidate item or itemset corresponding to a given cipher item or itemset with a probability above a given threshold.

The works that are most related to ours are [14] and [4]. Similar to our work, they assume that the adversary possesses prior knowledge of the frequency of items or item sets, which can be used to try to re-identify the encrypted items. The work [14] utilizes a one-to-n item mapping together with non-deterministic addition of cipher items to protect the identification of individual items. A recent paper [12] has formally proven that the encoding system in [14] can be broken without using context-specific information. The success of the attacks in [12] mainly relies on the existence of unique, common and fake items, defined in [14]; our scheme does not create any such items, and the attacks in [12] are not applicable to our scheme. Tai et al. [4] assume the attacker knows exact frequency of single items, similarly to us. They use a similar privacy model as ours, which requires that each real item must have the same frequency count as k – 1 other item in the outsourced dataset. They show that their outsourced data set satisfies k-support anonymity. However, they do not offer any theoretical analysis of anonymity of item sets. Instead they confine themselves to an empirical analysis.

In privacy preserving of vertically partitioned data Alan et al [17] proposed a linear-algebra-based protocol for computing secure matrix products for conducting secure regressions and similar analyses on vertically partitioned data with identical records but disjoint sets of attributes. This protocol allows data owners to estimate coefficients and standard errors of linear regressions, and to examine regression model diagnostics, without disclosing the values of their attributes to each other. No third parties are involved. The linear algebra-based approach is possible for statistical agencies and other data holders to obtain matrix products in vertically partitioned data settings. This enables agencies with vertically partitioned data to perform linear regressions without sharing their data values. Here the drawback is that, the protocols cannot be shared secure in nonlinear analyses.

The Frugal method consists of grouping together cipher items into groups of k adjacent items in the item support table in decreasing order of support. To fix the privacy vulnerabilities of Frugal, Giannotti et al [15] introduce the RobFrugal grouping method, which modifies Frugal. This enables formal privacy guarantees to be proven against attacks, conducted by the server using background knowledge, while keeping the resource requirements under control. The drawback is that, the attackers can find data by guessing attack.

Compared with these works, we have formal analysis to that our scheme can always achieve provable privacy guarantee. In general, it is prohibitively expensive to achieve perfect secrecy of outsourced frequent itemset mining [13]. We show that with less strict privacy models, we can achieve practical privacy-preserving methods that provide reasonable privacy guarantee. Our empirical study also shows that in practice, due to specific characteristics of the real transaction datasets the privacy-preserving methods for less-strict privacy models can enjoy a relatively high level of privacy in practice.

III. THE PATTERN MINING TASK

With the basics of association rule mining, let I = $i_1,\ldots..i_n$ be the set of items and D =$t_1,\ldots..t_m$ a TDB of transactions, each of which is a set of items. The support of an itemset is denoted by $\mathrm{supp}_D(S)$ and the frequency by $\mathrm{freq}_D(S)$. Recall that $\mathrm{freq}_D(S) = \mathrm{supp}_D(S) / |D|$. For each item *i*, $\mathrm{supp}_D(i)$ and $\mathrm{freq}_D(i)$ denote, respectively, the individual support and frequency of *i*. The function $\mathrm{supp}_D(.)$, projected over items, is also called the item support table of D represented in tabular form [see, the support value in Table 4.2].

IV. PRIVACY MODEL

Let D denote the original TDB that the owner has. To protect the identification of individual items, the owner applies an encryption function called BSS Homomorphic to D and transforms it to D*, the encrypted database. The items in D referred as plain items and items in D* as cipher items. The term item shall mean plain item by default. The notions of plain item sets, plain transactions, plain patterns, and their cipher counterparts are defined in the obvious way. Let I denotes the set of plain items and E to refer to the set of cipher items.

A. Adversary Knowledge

The server or an intruder who gains access to it may possess some background knowledge using which they can conduct attacks on the encrypted database *D**. Generically these agents are referred as an attacker. A conservative model was adopted and assumes that the attacker knows exactly the set of (plain) items *I* in the original TDB D and their true supports in D, i.e., $\mathrm{supp}_D(i)$, $\forall\ i \in I$. The attacker may have access to similar data from a competing company, may read published reports, etc. In reality, the attacker may possess approximate knowledge of the supports or may know the exact/approximate supports of a subset of items in D. However, to make the analysis robust, the conservative assumption is adopted that one who knows the exact support of every item.

Note that as the attacker has access to the encrypted database D*, one who also knows the supports $supp_D\ *(e)$, $e \in$ E, where E is the set of cipher items in the encrypted database D*. The encryption schema is based on adding fake transactions to the database. In particular, no new items are added and the assumption can be made that the attacker knows this and thus one who knows that |E| = |I|. To avoid guessing attack the transaction database is converted into matrix format.

V. ENCRYPTION/DECRYPTION SCHEME

In this section, we discuss the details of the E/D module.

A. Encryption

In this section a proposed encryption scheme called BSS homomorphic encryption scheme which transforms a TDB D into its encrypted version D*. The proposed scheme is parametric with respect to k>0 and consist of two main steps; 1) k-grouping method; 2) adding new fake transactions for achieving k-privacy. The constructed fake transactions are added to D to form D*, and transmitted to the server. The records of the fake transactions are stored in matrix format.

B. Decryption

When the client requests the execution of a pattern mining query to the server, specifying a minimum support threshold, the support returns the computed frequent pattern from D*.

C. Grouping items for k-privacy

Given the items support table, several strategies can be adopted to cluster the items into groups of size k. This method consists of grouping cipher items into groups of k adjacent items in the item support table in decreasing order of support, starting from the most frequent item $e1$. Assume $e_1, e_2, \ldots, e_n$ is the list of cipher items in descending order of support (with respect to D), the groups created by BSS homomorphic encryption are $\{e_1, \ldots, e_k\}$, $\{e_{k+1}, \ldots, e_{2k}\}$ and so on.

Table 4.1: Transaction database

Transaction database (TDB)
Bread Beer
Milk Bread
Bread Milk
Water Milk
Bread Beer
Bread eggs
Water

BSS homomorphic encryption grouping is optimal among all the groupings with the item support table sorted in descending order of support. This means, it minimizes ||G||, the size of the fake transactions added, and hence the size ||D*||. But is BSS homomorphic encryption a robust grouping, i.e., it will guarantee that itemsets (or transactions) cannot be cracked.

Consider the TDB and item support table in Table 4.1 and 4.2. The first group created by BSS homomorphic for k = 2, {e2, e4, e1} is supported in D, because {e2, e4, e1} occur together in a transaction of D. The first group created by BSS homomorphic encryption consists of the candidate itemset of {e2, e5}. The second group consist the candidate itemset of {e4, e1}. The third group consist the candidate itemset of {e3}, where e1 is Bread, e2 is Beer, e3 is Eggs, e4 is Milk and e5 is Water. The item support table listed in decreasing order of support, starting from the most frequent item Bread, as shown in Table 4.3.

The BSS Homomorphic encryption scheme is proposed to improve the security for the original transactional database. To make the transactional database secure the following encryption scheme is used, by which the data can be saved from the attackers/hackers.

Table 4.2 Item Support Table

Item	Support
Bread	5
Beer	2
Eggs	1
Milk	3
Water	2

Table 4.3 Support Values in Decreasing Order

Items	Support
Bread	5
Milk	3
Water	2
Beer	2
Eggs	1

When new data is added to the transactional database, the encryption scheme is performed for that particular group instead of whole transaction database. The encryption and decryption scheme are performed. It consumes time by partitioning the data in to groups. Here with the help of background knowledge the data cannot be hacked. The BSS Homomorphic encryption scheme avoids guessing attack from hackers by partitioning the data into groups as shown in Table 4.5.

Table 4.5 BSS Homomorphic Grouping with k=2

Item	Support
e_2	5
e_5	3
e_4	2
e_1	2
e_3	1

D. Constructing Fake Patterns

To fix the privacy vulnerabilities of BSS Homomorphic, a noise table specifying the noise N(e) needed for each cipher item e, the fake transactions are generated as shown in Table 4.6. First, the rows with zero noise are dropped, corresponding to the most frequent items of each group or to other items with support equal to the maximum support of a group as shown in table 4.6. Second, the remaining rows are sorted in descending order of noise.

Table 4.6 Item Support Table with Noise Values

Item	Support	Noise
e_2	5	0
e_5	3	3
e_4	2	0
e_1	2	0
e_3	1	2

Table 4.7 IST without Zero Noise Values

Item	Support	Noise
e_5	3	3
e_3	1	2

In the BSS encryption scheme, the grouping can be represented as the noise table. It extends the item support table with an extra column “Noise” indicating in Table 4.7, for each cipher item e, the difference among the support of the most frequent cipher item in e’s group and the support of e itself, as reported in the item support table. The noise of a cipher item *e* is denoted as N(e). The encrypted data are transferred to server for generating the fake transactions to obtain D*. The D* encrypted data from the server is used for decryption to compute the true support of a pattern and represents the fake transactions.

Continuing the example, the noise table obtained with BSS is reported in Table 4.7. The output of grouping is encrypted. It can be shown that the BSS Homomorphic encryption yields a minimum number of different types of fake transactions that equal the number of cipher items with distinct noise.

The actual data of every pattern are converted into matrix format by using the formula c = $(m_1 + m_2)^e$(mod Ø(n)) for avoid guessing attacks. The matrix format of the transaction database is shown in the Table 4.8. The purpose of using a matrix format is to reduce the storage overhead at the side of the data owner who may be equipped with sufficient computational resources and storage, which is common in the outsourcing data model.

Table 4.8 Matrix Format

Matrix Format
10000
11000
11000
01100
10010
10001

As the data owner outsources the encrypted database (including the fake transactions) one does not need to maintain the fake transactions in its own storage. Instead the data owner only has to maintain a matrix, which stores all the information needed on the fake transactions, for later recovery of real supports of item sets. The size of the matrix is linear in the number of items and is much smaller than that of the fake transactions.

VI. EXPERIMENTS

In this section, we report our empirical evaluation to assess the encryption/decryption overhead and the overhead at the server side incurred by the proposed schema.

A. Data Sets

To evaluate the performance of proposed technique, real data sets are used in the experiments. Real world data sets Retail and Chess are obtained from FIMI Repository [FIMI]. Finally, the results are evaluated by using a real life dataset (medical) is collected from medical shop. The performance of proposed algorithm was compared with the existing RobFrugal encryption scheme.

B. Experimental Evaluation

We implemented the BSS Homomorphic encryption scheme, as well as the decryption scheme, as described in Sec. V, in Java. The experiments were performed on a 2.80 GHz Intel Pentium D Processor with 3.5 GB memory. The operating system is Microsoft Windows 7.

Encryption Overhead: The performance comparison on real data sets, dense data set Chess and sparse data sets Retail and Medical are discussed. The chess dataset is an extremely dense dataset. Dense datasets have very long frequent as well as high patterns. Because the probability of an item's occurrence is very high in every transaction, for comparatively higher thresholds and dense datasets have too many candidate patterns. Here, first comparing the encryption overhead for the total time needed to encrypt the database (grouping, creation of fake transactions). Figure 5.1 clearly shows the runtime for chess dataset. It is compared with the different values of k (support values). The results show that the encryption time is small even for the biggest chess dataset TDB. The retail dataset is provided by Tom Brijs (FIMI), and contains the retail market basket data from an anonymous Belgian retail store. It is an extreme sparse dataset. Sparse datasets normally have too many distinct items. Although in the average case their transaction length is small, they normally have many transactions. Figure 5.2 clearly shows the running time comparsion on Retail Data Set. It shows that the runtime of Rob frugal is the worst,where as BSS Homomorphic is the best. Since BSS Homomorphic encryption efficiently encrypts the database. The medical dataset is a real dataset, which is a sparse dataset like Retail data. Here the performance is measured between the existing work and the BSS Homomorphic encryption scheme. When comparing to existing work, the BSS Homomorphic encryption scheme consumes time. Figure 5.3 clearly shows the run time of medical dataset of BSS Homomorphic encryption scheme and Rob Frugal encryption scheme. Overall, the runtime of Rob Frugal is the worst, followed by BSS Homorphic encryption.

Mining Overhead: The mining overhead for pattern mining task is done in chess dataset. Instead of measuring performance in run time, here proposed work measures the increase in number of frequent patterns obtained from mining the encrypted TDB, considering different support thresholds. From Figure 5.4 it is clear that the number of frequent patterns, at a given support threshold, increases with k, as expected. However, mining over chess dataset exhibits a small overhead even for very small support thresholds. Hence, a small support could make harder in discovering frequent patterns in chess dataset as there are large number of distinct items. The performance analysis on Retail data set is shown in Figure 5.5, which clearly shows the number of frequent patterns at a given support threshold, increases with k, as expected. From Figure 5.5 it is

clear that the Rob Frugal encryption scheme is the worst, when compared to the BSS Homomorphic Encryption scheme. This is because when the support value increases, runtime for identifying frequent patterns also increases. The performance analysis BSS Homomorphic and Rob Frugal encryption scheme on medical dataset is shown in Figure 5.6. It is clear that the frequent pattern generated for the given support threshold decreases, while increasing with k. For small values of support threshold, the incurred overhead at server side is kept under control. Furthermore there exists a tradeoff between the level of privacy, which increases with k, and the minimum affordable support threshold for mining, which also increases with k.

Fake Patterns: The size of fake transactions added to the database after encryption. For example, in encrypted chess data for k = 30, is only 80% larger than chess dataset. It observe that the size of fake transactions increases linearly with k. From Figure 5.7 reports the sizes of fake transactions for different values of k in encrypted chess dataset. It is clear that the BSS Homomorphic encryption scheme generates more fake transactions than the existing encryption scheme, to provide more security to the transactional database.Here the fake pattern increases when compared to retail dataset, due to large number of distinct items. The size of fake transactions added to the database after encryption. Where retail dataset is a sparse dataset. The fake transactions are added to each group of items through which the data cannot be hacked. Figure 5.8 reports the sizes of fake transactions for different values of k in encrypted retail dataset. For example, in encrypted retail data for k = 30, is only 8% larger than retail dataset. It is clear that the BSS Homomorphic encryption scheme generates more fake transactions than the existing encryption scheme, to provide more security to the transactional database. The performance results under different number of support value on real medical dataset for generating fake patterns. From Figure 5.9 it is clear that the number of fake patterns genereated by BSS Homomorphic encryption scheme is the smallest. This shows that BSS Homomorphic encryption can efficiently generates fake patterns even for large transactional database.

VIII. SUMMARY AND FUTURE WORK

In this research, a new BSS Homomorphic encryption scheme is proposed for the privacy preserving mining. The proposed encryption scheme partition the data into three groups to generate the noise values. The zero noise values are removed from each groups and encrypted. The fake transactions are efficiently added to the encrypted data to provide security to the original transactional database. To be more secure the encrypted original transactional database are converted into matrix format to avoid guessing attack. In the experiments the real dataset were used for performance evaluation. Result reveals that the encryption scheme gives more security for the transactional database. Moreover the proposed, BSS Homomorphic encryption scheme outperform the state-of-the-art of the encryption/decryption scheme especially, when database contains lot of large transaction.

A.) ENCRYPTION OVERHEAD

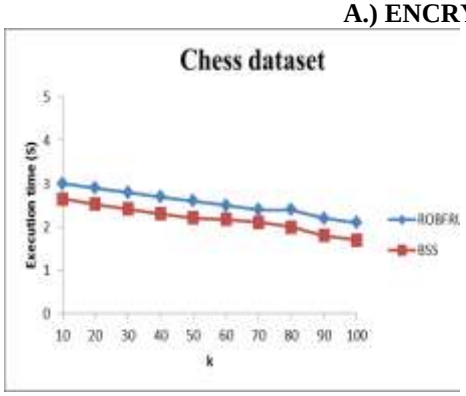

Figure 5.1 Encryption Overhead on Chess Data Set

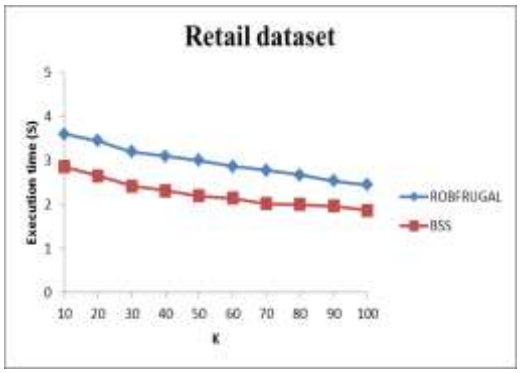

Figure 5.2 Encryption Overhead on Retail Data Set

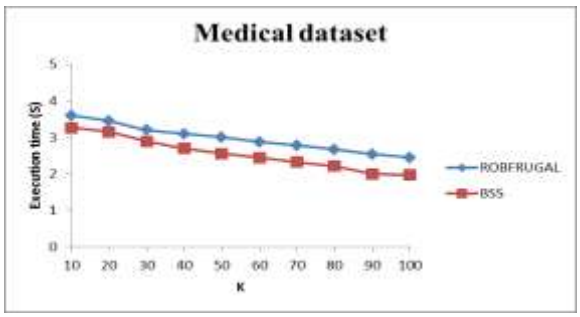

Figure 5.3 Encryption Overhead on Medical Data Set

B.) MINING OVERHEAD

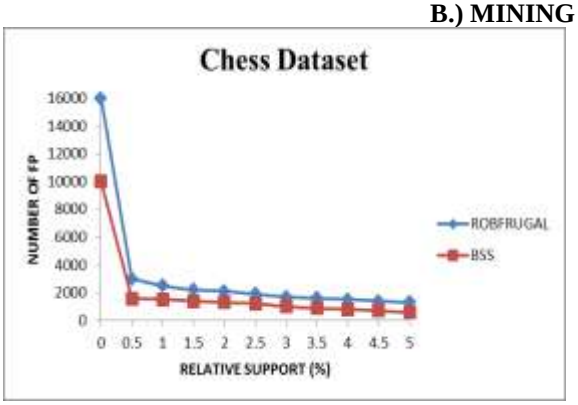

Figure 5.4 Mining Overhead on Chess Data Set

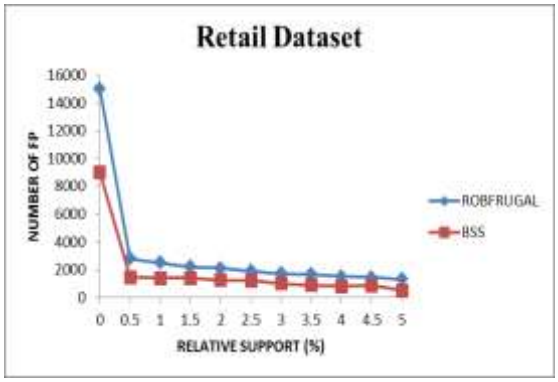

Figure 5.5 Mining Overhead on Retail Data Set

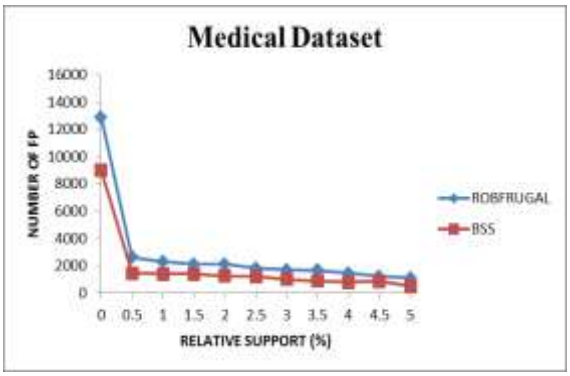

Figure 5.6 Mining Overhead on Medical Data Set

C.) FAKE PATTERNS

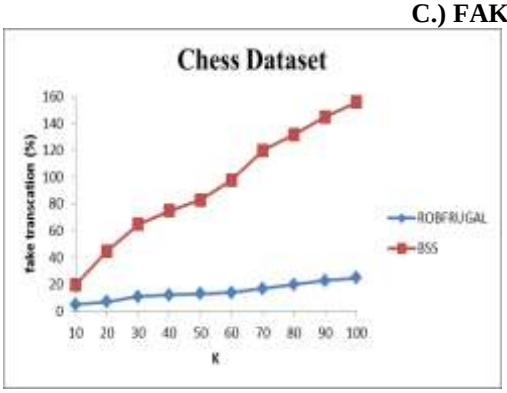

Figure 5.7 Fake Patterns for Chess Data Set

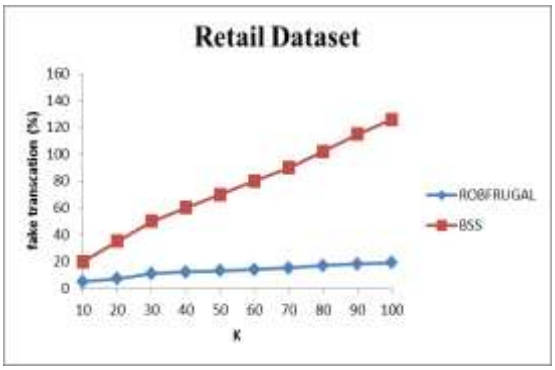

Figure 5.8 Fake Patterns for Retail Data Set

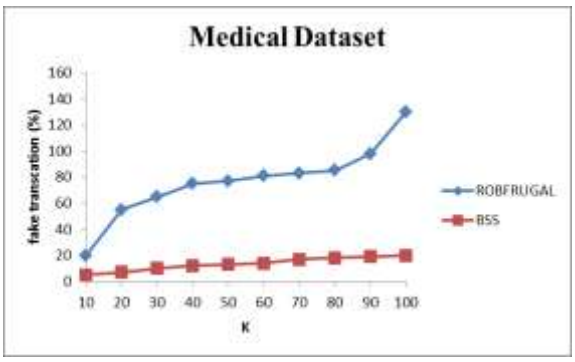

Figure 5.9 Fake Patterns for Medical Data Set

FUTURE WORK

Following are the future enhancements of the research:

(i)Privacy-preserving tools for individuals:

The privacy preserving techniques in research is proposed only for data holders, but individual record owners should also have the rights and responsibilities to protect their own private information.

(ii)Incorporating privacy protection in engineering process:

The privacy issue should be considered as a primary requirement in the engineering process of developing new technology. This involves formal specification of privacy requirements and formal verification tools to prove the correctness of a privacy-preserving system.

(iii)Health Monitoring:

The proposed work can be applied in health monitor for sensor measurements of people's health data which should be kept private and hidden from other people during transmission with aggregation to the sink node.

(iv)Military Surveillance:

In military communications, proposed work can be pragmatic with WSNs to replace guards and sentries around defensive perimeters, keeping soldiers out of harm's way, to locate and identify targets for potential attacks and to support attacks by locating friendly troops and unmanned vehicles. Therefore, the privacy of the sensor data is always critical and it should be preserved during aggregation.

REFERENCES

[1]. R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka, and J. Molina. Controlling data in the cloud: Outsourcing computation without outsourcing control. Pages 85–90, 2009.

[2]. M. J. Shaw, C. Subramaniam, G. W. Tan, and M. E. Welge. Knowledge management and data mining for marketing. Decis. Support Syst., 31(1):127–137, 2001.

[3]. G. M. Weiss. Data mining in the real world: Experiences, challenges, and recommendations. In DMIN, pages 124–130, 2009.

[4]. C. Tai, P. S. Yu, and M. Chen. K-support anonymity based on pseudo taxonomy for outsourcing of frequent itemset mining. In *KDD*, pages 473–482, 2010.

[5]. Fosca Giannotti, Laks V.S. Lakshmanan, Anna Monreale, Dino Pedreschi, and Hui Wang, "Privacy-preserving data mining from outsourced databases", In SPCC2010, in conjunction with CPDP, 2010.

[6]. L. Torgo. Data Mining with R: Learning with Case Studies.Chapman & Hall/CRC, 2010. L.Van Wel and L. Royakkers, Ethical issues in web data mining. Ethics and Inf. Technol., 6:129–140, 2004.

[7]. Rakesh Agrawal and Ramakrishnan Srikant, Privacy-preserving data mining. In *SIGMOD*, pages 439–450, 2000.

[8]. Gilburd B, Schuste A, and Wolff R. k-ttp: A new privacy model for large scale distributed environments. In *VLDB*, pages 563 – 568, 2005.

[9]. Murat Kantarcioglu and Chris Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *TKDE*, 16(9):1026–1037, 2004.

[10]. P. Krishna Prasad and C. Pandu Rangan, Privacy preserving birch algorithm for clustering over arbitrarily partitioned databases. In Advanced Data Mining and Applications, pages 146–157, 2007.

[11]. Ian Molloy, Ninghui Li, and Tiancheng Li. On the (in)security and (im)practicality of outsourcing precise association rule mining. In ICDM, pages 872–877, 2009.

[12]. Shariq J. Rizvi and Jayant R. Haritsa, Maintaining data privacy in association rule mining. In VLDB, pages 682–693, 2002.

[13]. W. K. Wong, David W. Cheung, Edward Hung, Ben Kao, and Nikos Mamoulis. Security in outsourcing of association rule mining. In VLDB, pages 111–122, 2007.

[14]. Fosca Giannotti, Laks V. S. Lakshmanan, Anna Monreale, Dino Pedreschi, and Hui (Wendy) Wang "Privacy-Preserving Mining of Association Rules From Outsourced Transaction Databases", IEEE systems journal, April 26, 2012.(20)

[15]. G. M. Weiss. Data mining in the real world: Experiences, challenges, and recommendations. In DMIN, pages 124–130, 2009.

[16]. Alan F. Karr, Xiaodong Lin, Ashish P. Sanil and Jerome P. Reiter "Privacy-Preserving Analysis of Vertically Partitioned Data Using Secure Matrix Products" Journal of Official Statistics, Vol. 25, No. 1, 2009, pp. 125–138.