Novel Algorithms for Privacy Preserving Utility Mining

Jieh-Shan Yeh, Po-Chiang Hsu and Ming-Hsun Wen
Department of Computer Science and Information Management, Providence University
{jsyeh, g9471055, g9571005}@pu.edu.tw

Abstract

Privacy Preserving Data Mining (PPDM) has become a popular topic in the research community. How to strike a balance between privacy protection and knowledge discovery in the sharing process is an important issue. This study focuses on Privacy Preserving Utility Mining (PPUM) and presents two novel algorithms, HHUIF and MSICF, to achieve the goal of hiding sensitive itemsets so that the adversaries can not mine them from the modified database. In addition, we minimize the impact on the sanitized database in the process of hiding sensitive itemsets. The experimental results show that HHUIF achieves the lower miss costs than MSICF does on two synthetic datasets. On the other hand, MSICF generally has the lower difference between the original and sanitized databases than HHUIF does.

1. Introduction

The traditional association rule mining, one of the most important methodologies in data mining, discovers all itemsets which support values are greater than a given threshold. There are lots of algorithms proposed for discovering the frequent itemsets in literature. The Apriori algorithm [1, 2, 13] is considered as the most famous one. In order to measure how "useful" an itemset is in the database, utility mining is proposed [22]. It overcomes the shortcomings of traditional association rule mining, which ignores the sale quantity and price (or profitability) among items in a transaction.

On the other hand, Privacy Preserving Data Mining (PPDM) [20] becomes a popular research direction in data mining in the past few years. In 1996, Clifton et al. [5] analyzed that data mining can bring about threat against databases and addressed possible solutions to achieve privacy protection of data mining. In 2002, Rizvi et al. discussed the privacy preserving mining of association rules [17, 18]. However, to the best of our knowledge, there is no Privacy Preserving Utility Mining (PPUM) discussed in any related literature. Therefore, this study focuses on PPUM and presents two novel algorithms, HHUIF and MSICF, to achieve the goal of hiding sensitive itemsets so that the adversaries can not mine them from the modified database. In addition, we minimize the impact on the sanitized database in the process of hiding sensitive itemsets.

The rest of this paper is organized as follows. Related works are reviewed in Section 2. Then, Section 3 proposes the HHUIF and MSICF algorithms to improve the balance between privacy protection and knowledge discovery. Section 4 presents the experimental results and evaluates the performance of the proposed algorithms. Finally, we present the conclusions in Section 5.

2. Related Works

2.1. Utility Mining

Utility mining discovers all itemsets whose utility values are equal or greater than a user specified threshold in a transaction database. However, the utility value of a itemset does not satisfy the "downward closure property". That is, a subset of a high utility itemset may not be a high utility itemset. The challenge of utility mining is in restricting the size of the candidate set and simplifying the computation for calculating the utility. Recently, Li et al. developed some efficient approaches, including the FSM, SuFSM, ShFSM, and DCG methods for share mining [8, 9, 10]. Under appropriate adjustments on item count and external utility of items, share mining is equivalent to utility mining. In the meanwhile, Liu et al. [12] also presented the Two-Phase (TP) algorithm for fast discovering all high utility itemsets. By using isolated items discarding strategy, Li et al. [11] proposed an efficient algorithm for discovering high utility itemsets.

The following set of terms are defined and given in [11] for the utility mining problem.

978-0-7695-3382-7/08 $25.00 © 2008 IEEE
DOI 10.1109/ISDA.2008.89

Table 1. An example of transaction database [9]

(a) Transaction table.

TID	A	B	C	D	E
T1	0	0	18	0	1
T2	0	6	0	1	1
T3	2	0	1	0	1
T4	1	0	0	1	1
T5	0	0	4	0	2
T6	1	1	0	0	0
T7	0	10	0	1	1
T8	3	0	25	3	1
T9	1	1	0	0	0
T10	0	6	2	0	2

(b) The external utility table.

ITEM	PROFIT ($) (per unit)
A	3
B	10
C	1
D	6
E	5

Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of items, where m is the total number of items. Let $DB = \{T_1, T_2, \ldots, T_n\}$, the task-relevant database, be a set of transactions where each transaction T_q is a set of items, that is, $T_q \subseteq I$. A set of items is also referred as an *itemset*. An itemset that contains *k*-items is called a *k-itemset*.

- The *item count* of item $i_p \subseteq I$ in transaction T_q, $c(i_p, T_q)$, is the number of item i_p purchased in transaction T_q. For example, $c(A, T_1) = 0$, $c(B, T_1) = 0$, and $c(C, T_1) = 18$, in Table 1 (a).
- Each item i_p has an associated set of transactions $T_{i_p} = \{T_q \in DB \mid i_p \in T_q\}$.
- A *k*-itemset $X = \{x_1, x_2, \ldots, x_k\}$ is a subset of I, where $1 \le k \le m$.
- Each *k*-itemset X has an associated set of transactions $T_X = \{T_q \in DB \mid X \subseteq T_q\}$.
- The *external utility* of item $i_p \subseteq I$, $eu(i_p)$, is the value associated with item i_p in the external utility table. This value reflects the importance of an item, which is independent of transactions. For example, in Table 1 (b), the external utility of item A, $eu(A)$, is 3.
- The utility of item $i_p \subseteq I$ in transaction T_q, $u(i_p, T_q)$, is the quantitative measure of utility for item i_p in transaction T_q, defined as $eu(i_p) \times c(i_p, T_q)$.
- The utility of itemset X in transaction T_q, $u(X, T_q)$, is $\sum_{i_p \in X} u(i_p, T_q)$, where $X \subseteq T_q$.
- The utility of itemset X, $u(X)$, is defined as $\sum_{X \subseteq T_q \in DB} u(X, T_q)$.

Utility mining is to find all the itemsets whose utility values are beyond a user specified threshold. An itemset X is a *high utility itemset,* if $u(X) \ge \varepsilon$, where ε is the minimum utility threshold. In Table 1, $u(\{A, D\}) = u(\{A, D\}, T_4) + u(\{A, D\}, T_8)$ =9+27=36, and $u(\{A, D, E\}) = u(\{A, D, E\}, T_4) + u(\{A, D, E\}, T_8)$ = 14+32 = 46. If ε is set to be 40, $\{A, D\}$ is a low utility itemset and $\{A, D, E\}$ is a high utility itemset. That is, the "downward closure property" does not hold in the utility mining model.

2.2 Privacy Preserving Mining on Association Rules

The sanitizing algorithms for the privacy preserving mining on association rules can be divided into two categories: **(1) Data-Sharing approach** and **(2) Pattern-Sharing approach**.

(1) Data-Sharing approach: the sanitization process acts on the data to remove or hide the group of restrictive association rules that contain sensitive knowledge. Among the algorithms of the Data-Sharing approach, they are classified the following sub-categories [18]: (a) Item Restriction-Based [21], (b) Item Addition-Based [21], and (c) Item Obfuscation-Based [19, 20].

(2) **Pattern-Sharing approach:** the sanitizing algorithm acts on the rules mined from a database instead of the data itself. Regarding pattern-sharing techniques, the only known approach that falls into this category was introduced in [21].

Rule Restriction-Based: This approach blocks some inference channels to ensure that an adversary cannot reconstruct restrictive rules from the non-restrictive ones. In doing so, we can reduce the inference channels and minimize the side effect. For example, DSA Algorithm proposed by Oliveira et al. [16]. MINSS, MINNS, SIMBLK, and BINFCH by Wang et al. [23, 24].

3 Proposed Algorithms

In this section, we present two algorithms for the privacy preserving utility mining: (1) Hiding High Utility Item First Algorithm (**HHUIF**) and (2) Maximum Sensitive Itemsets Conflict First Algorithm (**MSICF**). Frist of all, we give the definitions of some notations which will be used in the rest of the paper.

Definition 1: (***Sensitive itemset***) Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of items, where *m* is the total number of items, $DB = \{T_1, T_2, \ldots, T_n\}$ be a set of transactions, ε be the minimum utility threshold, and L be a set of all high utility itemsets for ε. Let $U = \{S_1, S_2, \ldots, S_l\}$ be a subset of L, where S_i, called a ***sensitive itemset***, is a itemset which needs to be hidden according to some security policies.

Definition 2: ***(Count of conflict)*** The count of conflict of an item i_p in the sensitive itemsets U denotes as $Icount_{i_p}(U)$, where $Icount_{i_p}(U) = |\{S_i \in \mathrm{U} \mid i_p \in S_i\}|$.

3.1 Hiding High Utility Item First Algorithm (HHUIF)

For each sensitive itemset, the sanitization process decreases the utility value of the sensitive itemset by modifying the quantity value of an item with the highest utility value in some transaction containing the sensitive itemset. The process repeats until the utility values of all sensitive itemsets are below the minimum utility threshold. The pseudo-code of the HHUIF algorithm is as follows:

Algorithm HHUIF

Input: the original database DB; the minimum utility threshold ε; the sensitive itemsets $U = \{S_1, S_2, \ldots, S_l\}$.

Output: the sanitized database DB' so that S_i cannot be mined.

1 **For each** sensitive itemset $S_i \in U$

2 $diff = u(S_i) - \varepsilon$

3 **While** ($diff > 0$) {

4 $(i_p, T_j) = \arg\max_{(i \in S_i, S_j \subseteq T)}(u(i,T))$

5 modify $o(i_p, T_j)$ with

$$o(i_p, \mathrm{T_j}) = \begin{cases} 0 & , if\ \mathrm{u}(i_p, \mathrm{T_j}) < diff \\ o(i_p, \mathrm{T_j}) - \left\lceil \frac{diff}{s(i_p)} \right\rceil & , if\ \mathrm{u}(i_p, \mathrm{T_j}) > diff \end{cases}$$

6

$$diff = \begin{cases} diff - u(i_p, \mathrm{T_j}) & , if\ \mathrm{u}(i_p, \mathrm{T_j}) < diff \\ 0 & , if\ \mathrm{u}(i_p, \mathrm{T_j}) > diff \end{cases}$$

}

7 **Return** the sanitized database DB'

End

Line 2 calculates the difference between the utility of an itemset S_i and the minimum utility threshold ε. In Lines 3 to 6, **HHUIF** sanitizes the transactions containing the sensitive itemsets repeatedly until $diff \le 0$. Line 4 selects item $i_p \in S_i$ and transaction $T_j \supseteq S_i$ such that the utility value of i_p in T_j is maximal for a given sensitive itemset S_i. Line 5 modifies the quantity of item i_p in transaction T_j. If $\mathrm{u}(i_p, \mathrm{T_j}) < diff$, that is, the utility of i_p on T_j is less than $diff$, we reduce the quantity of item i_p in T_j to *0* and continue to modify the quantity of next item until $diff \le 0$. If $\mathrm{u}(i_p, \mathrm{T_j}) > diff$, the quantity of i_p in T_j does not have to reduced to *0*. The quantity value of i_p in T_j is set to be $o(i_p, \mathrm{T_j}) - \left\lceil \frac{diff}{s(i_p)} \right\rceil$. Then, we continue the process until the utility values of each sensitive itemsets are below the minimum utility threshold.

3.2 Maximum Sensitive Itemsets Conflict First Algorithm (MSICF)

To reduce the number of the modified items from the original database, **MSICF** selects an appropriate item which has the maximum count of conflict among items in the sensitive itemsets. First, **MSICF** calculates $Icount_{i_p}(U)$ and sort i_p in decreasing order of $Icount_{i_p}(U)$. Second, it finds a transaction T_j, such that $u(i_p, T)$ is maximal. Then, it modifies $o(i_p, T_j)$. **HSICF** sanitizes the transactions containing the sensitive itemsets repeatedly until $diff \le 0$. Due to the paper page limit, we omit the detail of **HSICF** here.

4. Experimental Results

To measure the effectiveness of **HHUIF** and **MSICF** algorithms, the experiments were conducted on two synthetic datasets. All experiments were performed on a Dell workstation with 3.40 GHz Intel Pentium 4 processor and 2 GB of main memory, running the Windows XP Professional. We first applied ShFSM algorithm [8] to extract all high utility itemsets from the datasets. From the high utility itemsets found in each dataset, we randomly selected two sets of sensitive itemsets with size of five and ten.

Next, we sanitized sensitive itemsets in the Microsoft SQL server 2005 database.

In most cases, the data receiver may choose different thresholds for mining high utility itemesets on the released database. Oliveira et al. first proposed the concept of the privacy threshold ψ in [14]. The proportion of restrictive patterns that are still discovered from the sanitized database can be controlled by users with the privacy threshold ψ, and this proportion ranges from 0% to 100%. When ψ = 0%, no restrictive patterns are allowed to be discovered. When ψ = 100%, there are no restrictions on the restrictive patterns. In other words, all restrictive patterns can be discovered. The advantage of having this threshold is that between privacy and the disclosure of information can be balanced. In our experiments, we adopt the minimum utility threshold (MinUtility) ε for the data deliverer and introduce another parameter called ***Expecting Minimum Utility Threshold*** δ for the data receiver.

4.1 Datasets

We used the IBM synthetic data generator [7] to generate two synthetic datasets: *data.ntrans1.nitems0.1* and *data.ntrans1.nitems0.05*. The first one contains 1,000 transactions with 100 distinct items, and the average length of transactions is 9 by default. The second one contains 1,000 transactions with 50 different items, and the average length of transactions is 9 by default.

4.2 Effectiveness Measurement

To measure the effectiveness of our proposed algorithms, we adopt the performance measures introduced by Oliveira and Zaïane [20]. We summarize the performance measures as follows:

(a) Hiding Failure (HF): the ratio of sensitive itemsets that are disclosed before and after the sanitization process. The hiding failure is calculated as follows:

$$HF = \frac{|U(DB')|}{|U(DB)|}, \qquad (1)$$

where $U(DB)$ and $U(DB')$ denote the sensitive itemsets discovered from the original database DB and the sanitized database DB', respectively. The cardinality of a set S is denoted as $|S|$.

(b) Miss Cost (MC): the ratio of legitimate itemsets that are hidden by accident after the sanitization. The misses cost is measured as follows:

$$MC = \frac{|\sim U(DB) - \sim U(DB')|}{|\sim U(DB)|}, \qquad (2)$$

where $\sim U(DB)$ and $\sim U(DB')$ denote the non-sensitive itemsets discovered from the original database DB and the sanitized database DB', respectively.

(c) Difference between the Original and Sanitized Databases: the difference between the original database DB and the sanitized database DB', denoted by $diff(DB, DB')$, is given by:

$$diff(DB, DB') = \frac{|DB - DB'|}{|DB|} \qquad (3)$$

4.3 Comparison of Hiding Failure

Table 2 and Table 3 show the performance of **HHUIF** and **MSICF** algorithms on the hiding failure over various MinUtility and Expecting Minimum Utility Threshold δ applied to *data.ntrans1.nitems0.1* and *data.ntrans1.nitems0.05* with 5 sensitive itemsets, respectively. When MinUtility is less than or equal to δ, the hiding failure of two algorithms must be 0%. When MinUtility is greater than δ, in our experiment, the hiding failure of two algorithms is 100%, due to the chosen δ is far less than MinUtility.

Table 2. Hiding Failure (HF) for *data.ntrans1.nitems0.1* with |*U*| = 5

	MinUtility=3000		MinUtility=4000		MinUtility=5000	
	δ	**HF**	**δ**	**HF**	**δ**	**HF**
HHUIF	4000	0.00%	5000	0.00%	6000	0.00%
	3000	0.00%	4000	0.00%	5000	0.00%
	2000	100.00%	3000	100.00%	4000	100.00%
	MinUtility=3000		**MinUtility=4000**		**MinUtility=5000**	
	δ	**HF**	**δ**	**HF**	**δ**	**HF**
MSICF	4000	0.00%	5000	0.00%	6000	0.00%
	3000	0.00%	4000	0.00%	5000	0.00%
	2000	100.00%	3000	100.00%	4000	100.00%

Table 3. Hiding Failure (HF) for data.ntrans1.nitems0.05 with |*U*| = 5

	MinUtility=6000		MinUtility=7000		MinUtility=8000	
	δ	**HF**	**δ**	**HF**	**δ**	**HF**
HHUIF	7000	0.00%	8000	0.00%	9000	0.00%
	6000	0.00%	7000	0.00%	8000	0.00%
	5000	100.00%	6000	100.00%	7000	100.00%
	MinUtility=6000		**MinUtility=7000**		**MinUtility=8000**	
	δ	**HF**	**δ**	**HF**	**δ**	**HF**
MSICF	7000	0.00%	8000	0.00%	9000	0.00%
	6000	0.00%	7000	0.00%	8000	0.00%
	5000	100.00%	6000	100.00%	7000	100.00%

4.4 Comparison of Miss Cost

Table 4 and Table 5 show the performance of **HHUIF** and **MSICF** algorithms on the misses cost over various MinUtility and Expecting Minimum Utility Threshold δ applied to *data.ntrans1.nitems0.1*

and *data.ntrans1.nitems0.05* with 5 sensitive itemsets, respectively. We explained three different cases of between MinUtility and δ as follows：

(1) If MinUtility is greater than δ, the miss cost must be 0%. That is, all sensitive itemsets will not be mined.

(2) If MinUtility is equal to δ, the miss cost may occur. The reason of occurring misses cost is that the sanitized items may be contained in other high utility itemsets. For example, in *data.ntrans1.nitems0.1* with MinUtility = 7000, the utility value of high utility itemset {3, 36, 48} is 7037. While we hided sensitive itemset {29, 36} with deleting the quantity of item 36, it led to the utility value of high utility itemset {3, 36, 48} below 7000.

(3) If MinUtility is less than δ, the miss cost usually occur.

Table 4. Miss costs (MC) for *data.ntrans1.nitems0.1* with |*U*| = 5

	MinUtility=3000		MinUtility=4000		MinUtility=5000	
	δ	MC	δ	MCs	δ	MC
HHUIF	4000	62.04%	5000	50.00%	6000	35.71%
	3000	0.93%	4000	0.00%	5000	0.00%
	2000	0.00%	3000	0.00%	4000	0.00%
	MinUtility=3000		MinUtility=4000		MinUtility=5000	
	δ	MC	δ	MCs	δ	MC
MSICF	4000	70.37%	5000	55.26%	6000	35.71%
	3000	12.96%	4000	13.16%	5000	0.00%
	2000	0.00%	3000	0.00%	4000	0.00%

Table 5. Miss costs (MC) for *data.ntrans1.nitems0.05* with |*U*| = 5

	MinUtility=6000		MinUtility=7000		MinUtility=8000	
	δ	MC	δ	MC	δ	MC
HHUIF	7000	39.38%	8000	46.74%	9000	32.61%
	6000	0.00%	7000	2.17%	8000	2.17%
	5000	0.00%	6000	0.00%	7000	0.00%
	MinUtility=6000		MinUtility=7000		MinUtility=8000	
	δ	MC	δ	MC	δ	MC
MSICF	7000	39.38%	8000	47.83%	9000	30.43%
	6000	0.00%	7000	2.17%	8000	2.17%
	5000	0.00%	6000	0.00%	7000	0.00%

4.5 Comparison of Difference between the Original and Sanitized Databases

Table 6 and Table 7 show the performance of differences between *DB* and *DB'* with **HHUIF** and **MSICF** algorithms over various MinUtility applied to *data.ntrans1.nitems0.1* and *data.ntrans1.nitems0.05* with 5 sensitive itemsets, respectively.

Table 6. Differences between D and D' for *data.ntrans1.nitems0.1* with |*U*| = 5

MinUtility	3000	4000	5000
HHUIF	1.93%	0.81%	2.95%
MSICF	1.93%	0.81%	2.54%

Table 7. Difference between *DB* and *DB'* for *data.ntrans1.nitems0.05* with |*U*| = 5

MinUtility	6000	7000	8000
HHUIF	0.61%	1.32%	1.83%
MSICF	0.51%	1.32%	1.73%

5. Conclusions

Data quality plays a quite important role in the mining process. Accurate input data bring meaningful mining results. When users provide false data to protect their privacy will produce other problems at the same time. In strategic alliance cases, companies need to share information with others and protect their own business confidential as well. In this study, we present **HHUIF** and **MSICF** algorithms to reduce the impact on the source database for the Privacy Preserving Utility Mining. These algorithms are based on modifying the database transactions containing the sensitive itemsets so that the utility value can be reduced below the given threshold. There is no possible way to reconstruct the original database from the sanitized one. The experimental results show that **HHUIF** has the lower miss costs than **MSICF** does in two synthetic datasets. On the other hand, **MSICF** has the lower difference between the original and sanitized databases than **HHUIF** has, except in the case where MinUtility = 4000.

6. References

[1] R. Agrawal, T. Imielinski, and A. N. Swami, "Mining association rules between sets of items in large databases," in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pp. 207-216, 1993.

[2] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proceedings of 20th International Conference on Very Large Data Bases*, pp. 487-499, 1994.

[3] M. Atallah, E. Bertino, A. Elmagarmid, M. Ibrahim, and V. Verykios, "Disclosure limitation of sensitive rules," in *Proceedings of IEEE Knowledge and Data Engineering Workshop*, pp. 45-52, 1999.

[4] R. Chan, Q. Yang, and Y. D. Shen, "Mining high utility itemsets," in *Proceedings of the 2003 IEEE International Conference on Data Mining (ICDM 2003)*, pp. 19-26, 2003.

[5] C. Clifton and D. Marks, “Security and privacy implications of data mining,” in *Proceedings of the 1996 ACM SIGMOD Workshop on Data Mining and Knowledge Discovery*, pp.15-19, 1996.
[6] E. Dasseni, V. S. Verykios, A. K. Elmagarmid, and E. Bertino, “Hiding association rules by using confidence and support,” in *Proceedings of the 4th Information Hiding Workshop*, pp. 369-383, 2001.
[7] IBM Almaden Research Center. Synthetic data generation code for associations and sequential patterns. http://www.almaden.ibm.com/cs/projects/iis/hdb/Projects/data_mining/mining.shtml.
[8] Y. C. Li, J. S. Yeh, and C. C. Chang, "Efficient algorithms for mining share-frequent itemsets," in *Proceedings of Fuzzy Logic, Soft Computing and Computational Intelligence - 11th World Congress of International Fuzzy Systems Association (IFSA 2005),* pp. 534-539, 2005.
[9] Y. C. Li, J. S. Yeh, and C. C. Chang, “A fast algorithm for mining share-frequent itemsets,” *Lecture Notes in Computer Science 3399*, pp. 417-428, 2005.
[10] Y. C. Li, J. S. Yeh, and C. C. Chang, “Direct candidates generation: a novel algorithm for discovering complete share-frequent itemsets,” *Lecture Notes in Artificial Intelligence 3614*, pp. 551-560, 2005.
[11] Y. C. Li, J. S. Yeh, and C. C. Chang, "Isolated items discarding strategy for discovering high utility itemsets," *Data & Knowledge Engineering*, vol. 64, no. 1, pp. 198-217, 2008.

[12] Y. Liu, W. K. Liao, and A. Choudhary, “A two-phase algorithm for fast discovery of high utility itemsets,” *Lecture Notes in Computer Science 3518 (PAKDD 2005),* pp. 689-695, 2005.
[13] H. Mannila, H. Toivonen, and A. I. Verkamo, “Efficient algorithms for discovering association rules,” in *Proceedings of AAAI Workshop on Knowledge Discovery in Databases (KDD’94)*, pp. 181-192, 1994.
[14] S. R. M. Oliveira and O. R. Zaïne, “A framework for enforcing privacy in mining frequent patterns,” Technical Report, TR02-13, Computer Science Department, University of Alberta, Canada, June 2000.
[15] S. R. M. Oliveira and O. R Zaïane, ”Privacy preserving frequent itemset mining,” in *Proceedings of the IEEE ICDM Workshop on Privacy, Security, and Data Mining*, pp. 43–54, 2002.
[16] S. R. M. Oliveira, O. R. Zaïane, and Y. Saygin, “Secure association rule sharing,” in *Proceedings of 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD’04)*, pp. 74-85, 2004.
[17] S. Rizvi, and J. Haritsa, “Maintaining data privacy in association rule mining,” in *Proceedings of 28th Intl. Conf. on Very Large Databases (VLDB)*, 2002.
[18] S. J. Rizvi and J. R. Haritsa, “Privacy-preserving association rule mining”, in *Proceedings of the 28th Int’l Conference on Very Large Databases*, 2002.
[19] Y. Saygin, V. S. Verykios, and C. Clifton, “Using unknowns to prevent discovery of association rules,” *SIGMOD Record*, vol. 30, no. 4, pp.45-54, 2001.
[20] V. Verykios, E. Bertino, I. G. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis, “State-of-the-art in privacy preserving data mining”, *SIGMOD Record*, vol. 33, no. 1, pp. 50-57, 2004.
[21] V. Verykios, A. Elmagarmid, E. Bertino, Y. Saygin, and E. Dasseni, “Association rules hiding,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 4, pp.434-447, 2004.
[22] H. Yao, H. J. Hamilton, and C. J. Butz, “A foundational approach to mining itemset utilities from Databases,” in *Proceedings of the 4th SIAM International Conference on Data Mining*, pp. 482-486, 2004.
[23] Z. Wang, W. Wang, B. Shi, and S. H. Boey, "Preserving private knowledge in frequent pattern mining," in *Proceedings of 6th IEEE International Conference on Data Mining - Workshops (ICDMW'06)*, pp.530-534, 2006.
[24] Z. Wang, W. Wang, and B. Shi, "Blocking inference channels in frequent pattern sharing," in *Proceedings of 2007 IEEE 23rd International Conference on Data Engineering*, pp. 1425-1429, 2007.