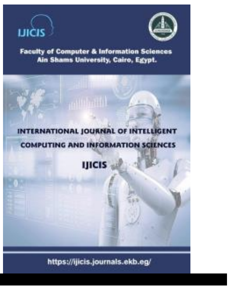
IJICIS, Vol.23, No.1, 69-83
DOI: 10.21608/ijicis.2022.142694.1191

International Journal of Intelligent Computing and Information Sciences

https://ijicis.journals.ekb.eg/


IJICIS
Faculty of Computer & Information Sciences Ain Shams University, Cairo, Egypt.
INTERNATIONAL JOURNAL OF INTELLIGENT COMPUTING AND INFORMATION SCIENCES
IJICIS
https://ijicis.journals.ekb.eg/

AN EFFICIENT HIDING METHOD FOR PRIVACY PRESERVING UTILITY MINING

Mohamed Ashraf*
Information Systems Deptartment,
Faculty of Computer and Information Science, Ain Shams University,
Cairo, Egypt
mohamed.a.a.h.is@gmail.com

Sherine Rady
Information Systems Deptartment,
Faculty of Computer and Information Science, Ain Shams University,
Cairo, Egypt
srady@cis.asu.edu.eg

Tamer Abdelkader
Information Systems Deptartment,
Faculty of Computer and Information Science, Ain Shams University,
Cairo, Egypt
tammabde@cis.asu.edu.eg

Tarek F. Gharib
Information Systems Deptartment,
Faculty of Computer and Information Science, Ain Shams University,
Cairo, Egypt
tfgharib@cis.asu.edu.eg

Received 2022-06-15; Revised 2022-12-22; Accepted 2022-12-24

Abstract: *Due to the rapid evolution of data saved in electronic form, data mining technologies have become critical and indispensable in looking for nontrivial, implicit, hidden, and possibly beneficial information in enormous volumes of data. High Utility Pattern Mining (HUPM), among the most intriguing data mining techniques, is broadly leveraged to analyze business interactions in market data based on the notion of economic utilities. These economic utilities can be used to examine the factors influencing a customer's purchasing behavior or to come up with new tailored selling and promotion tactics. This in turn has made utility-driven techniques an essential operation and vital activity for many data analysts since they can lead to proper decision-making processes. Nevertheless, such techniques can also lead to major threats regarding privacy and information security if they were misused. Privacy-Preserving Utility Mining (PPUM), also known as High Utility Pattern Hiding (HUPH), has recently emerged to mitigate the security and privacy issues that could happen in the utility framework. In this paper, we propose a heuristic PPUM method, named* ***HUP-Hiding****, to protect the results when mining sensitive data using a utility mining algorithm. The proposed method employs a dataset projection mechanism and a new victim item selection technique to efficiently perform the sanitization process. Experiments were performed to verify the reliability of the suggested algorithm. Our experimental results on different datasets confirm that HUP-Hiding has reasonable performance and fewer side effects compared to existing approaches.*

Keywords: *Privacy preserving, Data science, Utility mining, Sensitive pattern, Sanitization process*

1. Introduction

Data mining, or so-called Knowledge Discovery in Databases (KDD), refers often to the systematic extraction of actionable and beneficial patterns from massive data repositories, where the meaning of interest and usefulness varies depending on the issue description and context. Common data mining

***Corresponding Author**: Mohamed Ashraf

Information Systems Department, Faculty of Computer and Information Science, Ain Shams University, Cairo, Egypt

Email address: mohamed.a.a.h.is@gmail.com

tasks include classification, clustering, and pattern mining. Pattern mining is a significant data mining task utilized mainly to reveal specific patterns in large collections of data. Based on the different needs in diverse fields and systems, the extracted patterns may be classed as association rules (ARs) [1], sequential patterns [2], frequent patterns [3], or high utility patterns [4]. Among these patterns, Frequent Pattern Mining (FPM) [3] has been a popular research area for many years as it can reveal frequent associations between objects and items in binary databases. The basic idea behind FPM is that frequency equals significance. Patterns that have high frequency in the database are referred to as frequent or interesting patterns. Nevertheless, for a variety of real-world situations (e.g., business analytics and web mining), some attributes are more important than support factors like the profitability of the pattern. As a result, if typical methods for mining frequent patterns are applied in these settings, several essential patterns with low frequency but great value may be overlooked, whereas several insignificant high-frequency patterns may be discovered. To handle these disadvantages, **H**igh **U**tility **P**attern **M**ining (**HUPM**) [5] has appeared as one of the most significant pattern mining directions.

In economics, utility is a crucial notion. By incorporating utility theory from economics [6], the birth of a new mining and computing framework, known as HUPM, can be accomplished. HUPM is under the purview of both information systems and businesses. Due to only having restricted interestingness metrics, such as frequency or support, a slew of scholars has rushed to enhance data mining techniques in recent decades. However, these are ineffective since, in the end, each object or item is unequal. Other preferences, such as profit, cost, risk, and weight, are progressively being researched, and they permit the discovery of more beneficial knowledge than prior frequency-based mining methods. Given this, HUPM has arisen as a generalization for the Frequent Pattern Mining problem. HUPM [19-23] considers both subjective measures (i.e., profits) and objective measures (i.e., quantities) when evaluating the importance of items; so that more valuable and interesting patterns can be found. In general, the task of HUPM involves discovering high utility patterns (HUPs) possessing utility values no lower than a pre-determined minimal utility threshold value. This in turn encouraged many researchers and practitioners to apply the utility mining framework in several applications such as the Customer Relationship Management (CRM) process [16], analyzing tourists places to find the most profitable ones [17], and finding the best routes that can maximize the profits of taxi drivers [18].

Data mining algorithms have tremendous advantages, but they also pose a significant privacy danger [7, 8]. With the expansion of digital technologies, a lot of privacy concerns have arisen. The volume of data created and shared by businesses, governmental institutions, and other entities has skyrocketed in recent years. Many businesses and sectors are now engaging in collaborative computing and data sharing to discover more fascinating and impactful patterns that yield better profitability and decisions. Meanwhile, adversaries might exploit confidential patterns to enhance their company decisions and policies, causing data supplier organizations to suffer large losses [9, 10]. This is particularly disturbing when corporations make their data publicly available. These technical challenges illustrate the critical need to reconsider techniques for addressing some privacy and accuracy concerns when data is shared or disseminated prior to mining. Maintaining privacy when data is released for mining is a tricky matter [11]. Existing technologies can be divided into two key groups: data concealing and knowledge concealing. Data concealing is also called **P**rivacy **P**reserving **D**ata **P**ublishing (**PPDP**) [12]. It includes many fundamental security techniques, including encrypting, randomizing, and anonymizing to turn raw data into modified forms and obscure the record owner. Nonetheless, they may limit data usefulness and result in erroneous or unrecoverable knowledge for mining methods. Moreover, using these techniques is often impractical because data mining algorithms may still reveal some sensitive insights. Knowledge concealing is also called **P**rivacy **P**reserving **D**ata **M**ining (**PPDM**) [13, 14], which is the act of separating the confidential knowledge in datasets from data mining methods. This confidential knowledge can be association rules, knowledge patterns, clusters, classifications…etc. The PPDM framework involves techniques that are utilized to perturb the original database to sanitize it. It is significant and widely used in many fields, including medical, marketing, and statistics. Due to the unique properties of HUPM, **P**rivacy **P**reserving **U**tility **M**ining (**PPUM**) [15] has arisen as a crucial problem in the last few years. The basic objective of PPUM is to provide a concealing method that maximizes privacy while preserving data usefulness as much as possible.

In this paper, we consider the task of PPUM and suggest a heuristic method, named *HUP-Hiding*, for masking the sensitive patterns in transaction datasets. The proposed approach employs a few novel ideas to enhance sanitization performance. The experiments show promising results with regard to reducing side effects and runtime of the sanitization process compared to existing competitors.

The remaining content of this paper is structured as next. Section 2 gives a brief review of the existing HUPM and PPUM techniques. Section 3 presents the PPUM framework and its problem statement. Section 4 introduces the proposed hiding approach. The experiments are demonstrated in section 5. Eventually, section 6 draws the conclusion.

2. Related Works

2.1. Overview of high utility pattern mining

Since the proposal of the Two-Phase model [19], High Utility Itemset Mining (HUIM) has received a great deal of attention. A wide range of efficient algorithms have been designed during the last decade to mine high utility itemsets [20-23]. In 2005, Liu et al. [19] devised an Aproiri-based approach, named two-phase, to explore the entire set of HUIs in transactional datasets. The two-phase method applies the breadth-first search and a pruning property to identify the candidate HUIs. In 2012, Tseng et al. [20] introduced two FP-growth-inspired approaches; named UP-Growth and UP-Growth+, to reduce the aggressive dataset scans required in the two-phase method. In the same year, Liu et al. [21] presented a depth-first search approach to find the required HUIs using only two database scans. The authors designed a revolutionary data structure called (Utility List) and a moderately tighter upper bound (U-prune) to avoid the limitations of Liu et al's two-phase model. As users' and applications' requirements are different in real life, multiple variations have been proposed to expand the main concept of HUIM, examples may include mining top-k HUIs [22] and closed HUIs [23].

2.2. Overview of privacy preserving utility mining

Privacy-preserving utility mining (PPUM) has received a considerable attention in the recent years, especially in the context of quantitative transactional datasets. Several strategies have been developed over the last years for the mission of PPUM based on the assumption that selected record alteration or removal is NP-Hard difficulty [24], and as a result, heuristics may be used to handle the complexity issue [25 - 32]. The data utilization, by the completion of the privacy-preserving procedure, is a crucial problem as in order to hide the private knowledge the dataset is basically transformed by inserting fake information whether by reducing the actual utility of items (item-based sanitization), or deleting some sensitive transactions (transaction-based sanitization). In [25], the problem of PPUM was first introduced to conceal the sensitive high utility itemsets that can be found by an unauthorized third party if the data was shared or published. Two hiding methods were presented, namely HHUIF and MSCIF, to sanitize the dataset from confidential information. Both methods are item-based sanitization techniques and they have similar working ways except that they choose the victim items based on different criteria. HHUIF opts for the item with the maximum utility value, whereas MSCIF chooses the items having the highest conflict value among the sensitive itemsets to be modified. In 2015, Yun and Kim [26] argued that the aforementioned two algorithms require massive time to complete the sanitization task and thus they are not suitable for real-world databases, which tend to be huge. They proposed FPUTT algorithm which employs the same sanitization technique as HHUIF but has a notably better runtime performance. In [27], the concept of maximum and minimum utility has been put forward to reduce the side effects that can be emerged as a reaction to the sanitization process. The authors reported that their adopted ideas have significantly reduced the negative effects of the sanitization process compared to HHUIF and MSCIF. In [28], a transaction-based sanitization technique was developed to conceal sensitive patterns. The authors employed the pre-large concept and genetic algorithm to quickly identify the sensitive transactions that must be removed in order to secure the dataset. Unlike previous approaches which target an approximate solution, Li et al. [29] employed the concept of integer linear programming in an attempt to find an exact solution to the problem of PPUM. In case no exact solution was found, a relaxation method is used to guarantee reaching a semi-optimal solution. In [30], three PPUM algorithms were proposed to protect the sensitive knowledge in transactional databases. The three methods employed the same data structure to avoid reading the dataset multiple times and to hold the required information for the hiding operation in memory. Recently, Jangra et al. [31] introduced three variations from two algorithms, called MinMax and Weighted. They introduced three dataset sorting techniques, specifically, (1) RoT_DoC, (2) DoC_RoT and (3) sorting by transaction length, and tried every one of them with the two aforementioned algorithms in a dedicated variation. The idea behind these sorting techniques is that they should reduce the negative impacts of the sanitization process by giving some sensitive transactions higher priority for sanitization than others. Lastly, Ashraf et al. [32] proposed another PPUM approach, which is called SB2VF, to address the issue of finding the most appropriate victim items for sanitization. SB2VF adopts a complex sorting technique to identify the best sensitive transactions for sanitization. The algorithm also optimizes the victim-items selection process by applying the concepts of positive and negative coverage before the sanitization process to choose two specific victim items for each sensitive itemset.

3. The HUPM and PPUM frameworks

To better explain the PPUM and HUPM problems, we follow the definitions and notions introduced in prior studies [20, 21], [25, 32]. Let $I = \{i_1, i_2, i_3, \ldots. i_m\}$ be a finite set of m distinct items. A transaction dataset is a set of transactions that consist of items. Table 1 shows an original transaction dataset *DS* which will be used as a running example. This original dataset includes four transactions $DS = \{T_1, T_2, T_3, T_4\}$. Each transaction T_n is attached with a transaction id and has a different set of items such that $T_n \subseteq I$. Each item x_i in a transaction has an internal utility value (*IU*) which is usually referred to as the purchase quantity or frequency of the item in the transaction, and has an external utility value (*EU*) which is usually referred to as the profit, weight or benefit of the item. Table 2 shows the unique profits of items. For example, in the dataset *DS*, transaction T_1 indicates that items a, c and d were purchased in this transaction with quantities 3, 5, 2 respectively. To calculate the utility of item x_i in a transaction T_n such that $x_i \in T_n$ we use the following formula: $U(x_i, T_n) = IU(x_i, T_n) * EU(x_i)$. Considering the dataset *DS*, $U(a, T_1) = IU(a, T_1) * EU(a) = 3 * 2 = 6$. Assume an itemset $X \subseteq I$, $X \subseteq T_n$ is true if X occurred in T_n. The utility of X in a transaction T_n can be computed using the following formula: $U(X, T_n) = \sum_{x_i \in X \wedge X \subseteq T_n} U(x_i, T_n)$. For example, in the dataset *DS*, the utility of the itemset {ad} in transaction T_1 is computed as: $U(\{ad\}, T_1) = (3 * 2) + (2 * 3) = 12$. The utility of the itemset X in the dataset *DS* can be calculated as next: $U(X, DS) = \sum_{X \subseteq T_n \wedge T_n \in DS} U(X, T_n)$. As an example, the total utility of the itemset {ab} in the dataset *DS* is calculated as next: $U(\{ab\}, DS) = U(\{ab\}, T_2) + U(\{ab\}, T_4) = [(2 * 2) + (1 * 1)] + [(4 * 2) + (3 * 1)] = 16$. As could be observed from the previous examples, the significance of itemsets and patterns is measured based on their utility values. Consider a minimum utility threshold δ, which is a user-given value and identified according to his preference, we call an itemset X a High Utility Itemset (HUI) if its utility value exceeds the utility threshold value, to be specific, $U(X) \geq \delta$. For generalization purposes, the set of high utility itemsets hidden in the dataset is denoted as follows: $HUIs \leftarrow \{X \mid U(X) \geq \delta\}$. Table 3 outlines the set of HUIs when the utility threshold is equal to 21. Until now, we have introduced the main notions and definitions of the HUPM framework. Next, we demonstrate the problem from the PPUM perspective.

Table 1. Original transaction dataset.

Transaction ID	Transaction (item, internal utility)
T_1	(a, 3), (c, 5), (d, 2)
T_2	(a, 2), (b, 1), (d, 4)
T_3	(a, 1), (d, 2)
T_4	(a, 4), (b, 3)

Table 2. Unit profits of items.

Item	a	b	c	d
Unit Profit	2	1	4	3

Table 3. The set of HUIs when δ = 21.

Itemset	Utility
{ad}	36
{acd}	**32**
{cd}	26
{ac}	**26**
{d}	24

3.1. Sensitive high utility itemset

A high utility itemset can be seen as a sensitive pattern if its presence in the mining result can lead to the exposure of some sort of secret information. This type of itemsets should be removed from the mining results to enable data owners from sharing and publishing their data. Considering the running example, Table 3 displays the sensitive itemsets (the Bold ones) in the current dataset *DS*.

3.2. Problem statement

Given some sensitive HUIs, the task of PPUM [25] is to entirely conceal the sensitive itemsets in such a manner that their utility values fall below the user-defined minimal-security threshold. This should be done while mitigating the expected negative impacts of the hiding process such that all the non-sensitive high utility itemsets should still be discoverable by any utility mining algorithm. Considering the running example, as can be seen in Table 3, the target is to prevent the sensitive itemsets, {acd} and {ac}, from showing up when mining the dataset *DS* using a utility threshold equal to 21.

3.3. Negative coverage of sensitive items

Consider a sensitive itemset *S* and a sensitive item $i_s \in S$, the negative coverage of i_s represents the number of non-sensitive itemsets that comprise i_s. Considering the current example, according to Table 3, the sensitive items are a, c and d. And the non-sensitive itemsets are {ad}, {cd} and {d}. Item a and c have appeared in only one non-sensitive itemset, while item d has appeared in three non-sensitive itemsets. Therefore, the **N**egative **C**overage (***NC***) of these items is as follows, *NC*(a) = *NC*(c) = 1 and *NC*(d) = 3.

3.4. Negative coverage of sensitive transactions

Consider a sensitive transaction T_s and a sensitive itemset *S* such that $S \subseteq ST$, the negative coverage of T_s represents the number of non-sensitive itemsets that appeared in T_s. Considering the current example, according to Tables 1 & 3, there is only one sensitive transaction which is T_1 and it contains three non-sensitive itemsets. Therefore, the negative coverage of transaction T_1 is 3, more formally, $NC(T_1) = 3$.

3.5. Projected sensitive transactions

Let be a sensitive itemset *S*, the projected transactions of *S* can be represented as a hash map notated as *pTable*. To be specific, $pTable \leftarrow \{T_s \mid S \subseteq T_s \wedge T_s \in DS\}$. In other words, this table will contain all the transactions of every sensitive itemset in the original dataset. This should cancel the need to traverse through the whole sensitive transactions for each sensitive itemset as their transactions were already defined and saved in a table. Table 4 depicts the index table of the current sensitive itemsets.

Table 4. The *pTable* of the sensitive itemsets

Sensitive Itemset	List of Transactions IDs
{acd}	T_1
{ac}	T_1

3.5. The weight of a sensitive transaction

Let be a sensitive itemset S and a sensitive transaction T_s such that $S \subseteq T_s$, the weight of the transaction T_S with respect to the itemset *S* can be estimated using the next equation:

$$W(T_s, S) = \frac{U(S,\ T_S)}{1+NC(T_S)} \qquad (1)$$

The estimated weight should reflect the significance of the transaction for the sensitive itemset. The higher the value of the numerator the faster the itemset can be hidden as more utility from the sensitive itemset will be reduced. Whereas, the lower the denominator the fewer non-sensitive itemsets will be lost. Thus, we give the sensitive transactions with higher weight, a higher priority for sanitization.

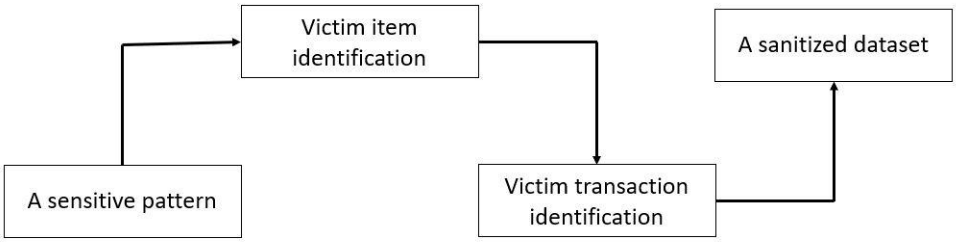
3.6. The order of sensitive transactions

It is assumed in this paper that the projected transactions of each sensitive itemset in the *pTable* are ordered as per the descending order of their weights. More precisely, transactions with higher weights will be sanitized first.

4. The proposed hiding method

In this section, a sanitization method called *HUP-Hiding* is introduced. Figure 1 gives a brief view of the sanitization process working way. Modifying the dataset for hiding sensitive HUIs causes negative impacts on non-sensitive HUIs and the dataset. The method that produces fewer side effects is more efficient. The proposed method follows the item-based sanitization approach [25] and involves three main stages. (1) The determination of victim items; (2) the determination of victim transactions; (3) the modification of the victim item; whether by deleting it or reducing its utility value in the victim transaction. Also, there are some optimizations utilized to decrease the side effects such as the processing order of the sensitive patterns and the sorted transactions in the *pTable*. Algorithm 1 presents the pseudo-code of the suggested sanitization approach. At first, the data owner applies a utility mining algorithm to the dataset to extract the collection of patterns that satisfy the owner-preferred minimal utility threshold δ. Then, the sensitive and non-sensitive itemsets are identified. In order to be able to release the dataset, the data owner will need to apply the proposed *HUP-Hiding* to sanitize the dataset from the sensitive itemsets. The proposed hiding method requires four parameters to run: (1) the original dataset; (2) the set of sensitive itemsets; (3) the set of non-sensitive itemsets; (4) the minimum utility threshold. The proposed method first scans the dataset once to catch the sensitive transactions and calculate their negative coverage. Also, the non-sensitive itemsets are scanned for each sensitive item to calculate the negative coverage of sensitive itemsets. After that, a definite victim item is selected for each sensitive itemset in advance. This victim item will be the only one modified during the whole sanitizing process. For the determination of victim items, the items with the least negative coverage value in each sensitive itemset will be chosen for sanitization. If there is only one item with the least coverage value, this item will be the only one chosen. Otherwise, if there is more than one item with the same minimum coverage, these items will be candidate victim items and will be stored in a list. The reason why we choose based on the negative coverage is that the less the negative coverage the smaller the number of non-sensitive itemsets that will be affected by sanitization. Subsequently, the *pTable* is built to associate each sensitive itemset with its transactions. This should help in reducing the computational overhead of finding the sensitive transactions of the itemset being processed. Before the sanitization process, the sensitive itemsets are ordered in descending order of their utilities such that the itemsets having the highest utilities are sanitized first. This is because itemsets having high utilities will most likely affect the other sensitive itemsets. Thus, more than one sensitive itemset can be hidden at once. The sanitization process then begins by computing the *dif_to_hide* value which reflects how much utility needs to be reduced from the sensitive itemset so that it becomes hidden. In another word, we need to reduce this utility from the sensitive high utility itemset to turn it into a low utility itemset. If the *dif_to_hide* value is greater than 0, then the sensitive transactions of the current sensitive itemset are restored from the *pTable* and sorted in descending order of their weight which is calculated using Eq. (1). Next, a loop is carried out over the ordered sensitive transactions until the sensitive itemset being processed is completely hidden. In each iteration, a victim item is retrieved from the candidate victim items. If there is more than one candidate item, then the one with the lowest utility value in the victim transaction is selected for modification. The reason behind this is that this item should have the least negative effect on the total utility of the dataset. After the victim item selection, there are two scenarios for modification that could happen to the victim item. The first scenario is when the utility of the victim item in the victim transaction is less than the *dif_to_hide* value. In this case, the victim item will be deleted from the transaction and the utility of the sensitive itemset will be reduced accordingly. The second scenario is when the victim item's utility is above the *dif_to_hide* value. In such a case, the internal utility of the victim item will be reduced and *dif_to_hide* will be assigned a 0 value, which means that the sensitive itemset has finally become a low utility itemset. In both the previous cases, the *pTable* is updated for all sensitive itemsets containing the same victim item and existing in the same victim transactions. Besides, the non-sensitive itemsets that have the same victim item and reside in the same transaction will be also affected negatively and could be over hidden. Each time a transaction has been sanitized; the original dataset will be updated. Eventually, a check is made after each modification on the *dif_to_hide* value to see whether the itemset has been already concealed or not. The same previous steps will also be applied to the rest of the sensitive itemsets until they all have been masked.

Figure 1. The purification procedure for concealing the sensitive patterns.



Algorithm

```
Input: $DS$, the original transactional dataset; $SI$, the set of sensitive HUIs; $NI$, the set of non-sensitive
    HUIs; $\delta$, the security threshold.
Output: $DS^{'}$, a purified dataset.
Scan $DS$ once to determine the sensitive transactions and the negative coverage of each transaction.
Scan the $NI$ to calculate the negative coverage of all sensitive items.
For each sensitive itemset $S_j \in SI$
    $CandI_{vic}(S_j) \leftarrow z_k$ such that $z_k \in S_j \wedge \forall z_r \in S_j \mid NC(z_k) \leq NC(z_r)$
End for
Build the $pTable$ to associate each sensitive itemset with its transactions.
Sort the sensitive itemsets $S_j \in SI$ as per the decreasing order of their utility values.
For each $S_j$ in sorted $SI$ do
    $dif\_to\_hide = U(S_j) - \delta + 1$
    **If** $dif\_to\_hide > 0$ **then**
        Retrieve transactions of $S_j$ from the $pTable$.
        Calculate the weight of each transaction using Eq. (1)
        Sort the transactions as per the descending order of their weight.
        **For each** transaction $T_k \in pTable(S_j)$ **do**
            $T_{vic}(S_j) \leftarrow T_k$
            **If** $|CandI_{vic}(S_j) = 1|$ **then**
                $I_{vic}(S_j) \leftarrow z_k : z_k \in CandI_{vic}(S_j)$
            **Else**
                $I_{vic}(S_j) \leftarrow z_k : z_k \in CandI_{vic}(S_j) \wedge \forall z_r \in S_j \mid U(z_k, T_k) \leq U(z_r, T_k)$
            **End if**
            **If** $dif\_to\_hide \geq U(I_{vic}(S_j), T_{vic}(S_j))$ **then**
                Remove $I_{vic}(S_j)$ from $T_{vic}(S_j)$
                $dif\_to\_hide = dif\_to\_hide - U(I_{vic}(S_i), T_{vic}(S_i))$
                Update $pTable$ $\forall S_h \in SI$ such that $S_h \subseteq T_{vic}(S_j) \wedge I_{vic}(S_j) \in S_h$
            **Else**
                $IU(I_{vic}(S_j), T_{vic}(S_j)) = IU(I_{vic}(S_j), T_{vic}(S_j)) - \lceil \boldsymbol{dif\_to\_hide / EU(Ivic(Sj))} \rceil$
                $dif\_to\_hide = 0$
                Update $pTable$ $\forall S_h \in SI$ such that $S_h \subseteq T_{vic}(S_j) \wedge I_{vic}(S_j) \in S_h$
            **End if**
            $DS' \leftarrow T_{vic}(S_j)$
            **If** $dif\_to\_hide \leq 0$
                break
            **End if**
        **End for**
    **End if**
**End for**
**Return** purified dataset $DS'$
```

5. Experimental results

In this section, experiments were performed to test the reliability of the suggested hiding method against the currently existing PPUM algorithms. The proposed HUP-Hiding was evaluated on two benchmark datasets and compared against the following algorithms: HHUIF [25], MSICF [25], MAU-MSU [27], MAU-MIU [27], SMAU [30] and Weighted_RoT_DoC [31]. As can be seen in Table 5, the adopted datasets have different natures (dense and sparse). Thus, they express to some extent the diversity of the real-life databases. To better show the influences of the proposed ideas two cases were tested in each studied dataset. The first case is when raising the number of sensitive itemsets SIs while the value of the minimum utility threshold δ is fixed. The second case is when increasing the value of δ while the number of SIs required to be concealed is not changed. Experiments were executed on a machine with a third-generation Intel processor, 6 GB Ram, and Windows 7 OS. Four quality measures, that are widely used in the PPUM literature, were employed in the performed experiments to assess the efficacy of the sanitization way, which are the Runtime RT [26], Hiding Failure HF [25] (becomes 0 if all the sensitive patterns are concealed in the purified dataset), Artificial Cost AC [25] (becomes 0 if no fake patterns were produced in the sanitized), and Missing Cost MC [25] (becomes 0 if no non-sensitive pattern is missed in the sanitized dataset). Since all the compared algorithms follow the item-based sanitization model [25-27], [30, 32], they can always hide all the sensitive patterns without producing any fake patterns in the sanitized dataset. Hence, the results of the HF and AC metrics are not displayed in this paper since they are always 0 for all the compared algorithms.

Table 5. The features of the benchmark datasets.

Dataset	#Trans	#Items	Avg. length	Max. length	Type	Density
chess	3,196	75	37	37	Dense	49.3
retail	88,162	16,470	10.3	76	Sparse	0.062

5.1. Runtime

The sanitization time of the compared algorithms can be viewed in Figures 2 & 3. It can be observed that as the number of sensitive itemsets rises, so usually does the sanitization time. This is logical since the more the number of sensitive patterns the more time is required to hide them. However, we can observe the opposite in the case of increasing the utility threshold. This is because as the threshold gets higher, the number of resultant high utility itemsets gets lower. Thus, less time is required to hide the sensitive patterns. In chess dataset, we can see that the compared algorithms needed a very short time to finish the perturbation process. This is because in dense datasets, like chess, hiding one sensitive itemset can affect multiple others; as transactions and items overlapping is high among the sensitive itemsets and the victim item in one sensitive itemset can be the same in other itemsets. Nevertheless, this is not the case in retail dataset as the time required to complete the sanitization process is notably higher. This is because the items and transactions overlapping are lower in sparse datasets. Thus, there is less chance to hide more than one sensitive itemset at once. We can also note that the proposed *HUP-Hiding* needs far less time than HHUIF and MSICF algorithms. This is something expected given that *HUP-Hiding* projects the transactions of each sensitive itemset to avoid scanning the dataset multiple times. However, the proposed *HUP-Hiding* needed more time in retail dataset compared to SMAU, MSU-MIU, and MSU-MAU. This is because *HUP-Hiding* puts into perspective the side effects that could happen to the non-sensitive patterns during the sanitizing process, unlike the previously mentioned algorithms.

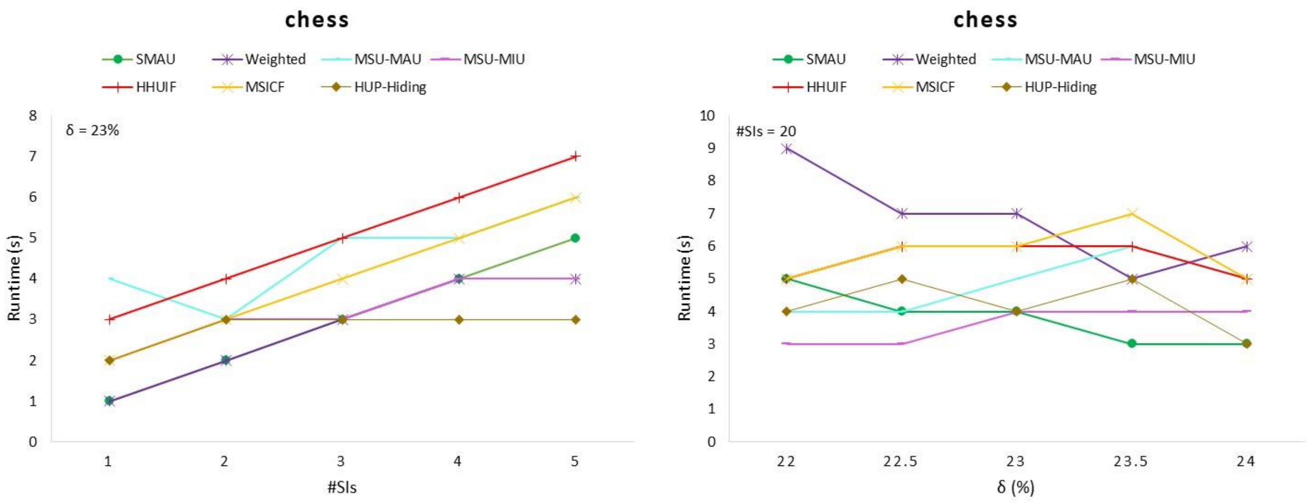


Figure 2. The runtime results on chess dataset.

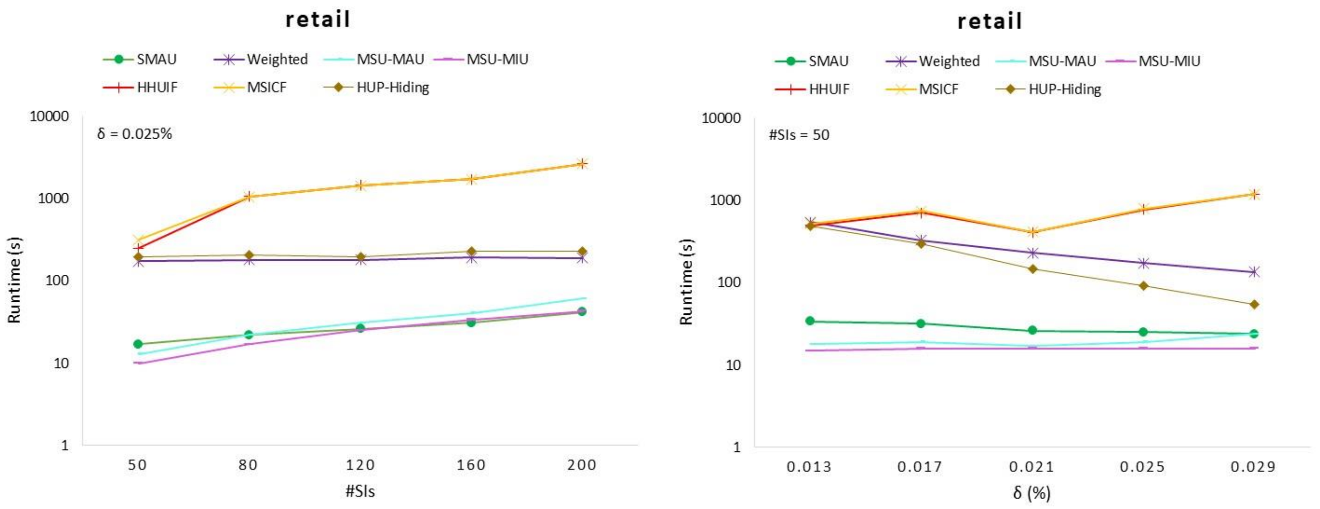


Figure 3. The runtime results on retail dataset.

5.2. Missing Cost

The missing cost MC measure is calculated using the following formula:

$$\mathrm{MC} = \frac{NI(D)-NI(D')}{NI(D)}$$

Where $NI(D)$ is the number of non-sensitive itemsets before sanitization and $NI(D')$ is the number of non-sensitive itemsets after the sanitization process has been completed. The target value of the MC measure for any PPUM algorithm is 0 which means that no non-sensitive itemsets were lost in the purified dataset. The missing cost of the compared algorithms can be viewed in Figures 4 & 5.

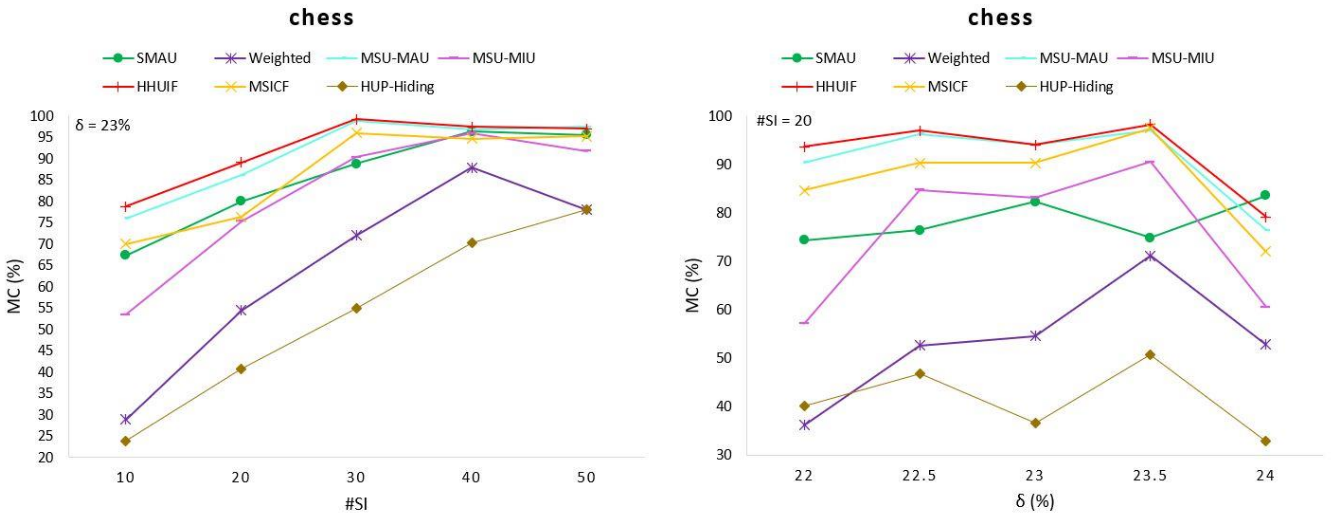


Figure 4. The missing cost results on chess dataset.

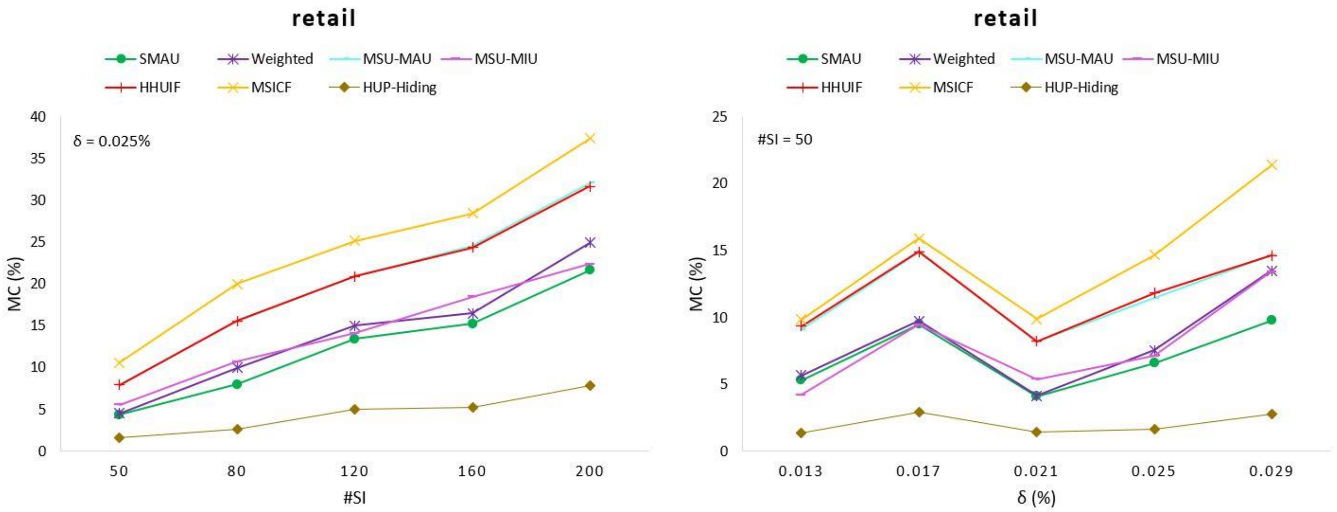


Figure 5. The missing cost results on retail dataset.

It is evident from the results that *HUP-Hiding* has the least missing cost compared to the other competitors. The percentages of the missing cost reduction achieved by *HUP-Hiding* are tabulated in Table 6. The cause behind this superior performance is twofold. First, the proposed method chooses the item that exists in the least number of non-sensitive itemsets as the victim item, which greatly reduces the negative effects on the non-sensitive itemsets. Second, the proposed method gives priority to transactions to be sanitized as per their estimated weights, which guarantees that transactions that contain the least number of non-sensitive itemsets and have the highest utility value for the sensitive itemset being processed are sanitized first. From Figures 4 & 5, we can also observe that the missing cost tends to increase as the number of sensitive itemsets increases. This is reasonable because hiding more sensitive itemsets increases the chance of reducing the utilities of the non-sensitive itemsets due to items overlapping. In general, we can conclude that *HUP-Hiding* has fair results when it comes to the sanitization speed, and the best results when it comes to reducing side effects which is more important in PPUM than reducing the runtime.

Table 6. Enhancements of *HUP-Hiding* over the compared algorithms in terms of reducing the missing cost.

Dataset	SMAU	Weighted	MSU-MAU	MSU-MIU	HHUIF	MSICF
	The best Missing Cost reduction percentages achieved by *HUP-Hiding*					
chess	43.5%	17.57%	52%	35.39%	54.91%	46.18%
retail	13.79%	17.08%	24.22%	14.49%	23.79%	29.53%

6. Conclusion

Privacy Preserving Utility Mining (PPUM) is a crucial and popular topic in the field of PPDM, where it is an integration of both Utility Pattern Mining (UPM) and Privacy Preserving Utility Mining (PPDM). UPM is used to discover valuable and useful patterns on transactional data by considering the utility of the patterns, which mainly reflects a business objective (e.g., profit, weight, risk, user's interest). PPDM is used to modify the data before the mining process in such a way that we can safely and effectively apply the data mining services without losing the benefits of mining or compromising the privacy and security of the sensitive information residing in the data. Aligning with this, the PPUM framework is mainly concerned with providing privacy control methods based on the concept of utility. In this paper, we introduced HUP-Hiding method to protect confidential knowledge from utility mining techniques by hiding the sensitive high utility itemsets that can be extracted. In the proposed method, a projection mechanism was adopted to reduce the time required to scan the sensitive dataset. Moreover, an efficient victim item selection technique was utilized to enhance the hiding process. Furthermore, an effective transaction sorting technique was proposed to reduce the side effects to the maximum limit. To highlight the efficiency of the proposed method, extensive comparisons were conducted using dense and sparse datasets. The acquired results clearly show the superiority of HUP-Hiding over the existing PPUM algorithms in terms of reducing the side effects of the sanitization process. In future work, we are planning to extend the proposed hiding method so that it can hide the sensitive patterns in more complicated datasets such as sequential datasets.

References

1. Diaz-Garcia, J.A., Ruiz, M.D. and Martin-Bautista, M.J., 2022. A survey on the use of association rules mining techniques in textual social media. *Artificial Intelligence Review*, pp.1-26.
2. Li, Y., Zhang, S., Guo, L., Liu, J., Wu, Y. and Wu, X., 2022. NetNMSP: Nonoverlapping maximal sequential pattern mining. *Applied Intelligence*, pp.1-24.
3. Agrawal, R. and Srikant, R., 1994, September. Fast algorithms for mining association rules. In *Proc. 20th int. conf. very large data bases, VLDB* (Vol. 1215, pp. 487-499).
4. Kumar, R. and Singh, K., 2022. A survey on soft computing-based high-utility itemsets mining. *Soft Computing*, pp.1-46.
5. Zhang, C., Han, M., Sun, R., Du, S. and Shen, M., 2020. A survey of key technologies for high utility patterns mining. *IEEE Access*, *8*, pp.55798-55814.
6. Fishburn, P.C., 1968. Utility theory. *Management science*, *14*(5), pp.335-378.

7. Amiri, A., 2007. Dare to share: Protecting sensitive knowledge with data sanitization. *Decision Support Systems*, *43*(1), pp.181-191.
8. Oliveira, S.R. and Zaïane, O.R., 2003, November. Protecting sensitive knowledge by data sanitization. In *Third IEEE International conference on data mining* (pp. 613-616). IEEE.
9. O'Leary, D.E., 1991. Knowledge Discovery as a Threat to Database Security. *Knowledge discovery in databases*, *9*, pp.507-516.
10. O'Leary, D.E., Bonorris, S., Klosgen, W., Khaw, Y.T., Lee, H.Y. and Ziarko, W., 1995. Some privacy issues in knowledge discovery: the OECD personal privacy guidelines. *IEEE Expert*, *10*(2), pp.48-59.
11. Gkoulalas-Divanis, A. and Verykios, V.S., 2009. Hiding sensitive knowledge without side effects. *Knowledge and Information Systems*, *20*(3), pp.263-299.
12. Fung, B.C., Wang, K., Chen, R. and Yu, P.S., 2010. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys (Csur)*, *42*(4), pp.1-53.
13. Agrawal, R. and Srikant, R., 2000, May. Privacy-preserving data mining. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data* (pp. 439-450).
14. Verykios, V.S., Bertino, E., Fovino, I.N., Provenza, L.P., Saygin, Y. and Theodoridis, Y., 2004. State-of-the-art in privacy preserving data mining. *ACM Sigmod Record*, *33*(1), pp.50-57.
15. Dinh, D.T., Huynh, V.N., Le, B., Fournier-Viger, P., Huynh, U. and Nguyen, Q.M., 2019. A survey of privacy preserving utility mining. In *High-Utility Pattern Mining* (pp. 207-232). Springer, Cham.
16. Krishna, G.J. and Ravi, V., 2021. High utility itemset mining using binary differential evolution: An application to customer segmentation. *Expert Systems with Applications*, *181*, p.115122.
17. Vu, H.Q., Li, G. and Law, R., 2020. Discovering highly profitable travel patterns by high-utility pattern mining. *Tourism Management*, *77*, p.104008.
18. Liu, C. and Guo, C., 2021. Mining top-N high-utility operation patterns for taxi drivers. *Expert Systems with Applications*, *170*, p.114546.
19. Liu, Y., Liao, W.K. and Choudhary, A., 2005, August. A fast high utility itemsets mining algorithm. In *Proceedings of the 1st international workshop on Utility-based data mining* (pp. 90-99).
20. Tseng, V.S., Shie, B.E., Wu, C.W. and Philip, S.Y., 2012. Efficient algorithms for mining high utility itemsets from transactional databases. *IEEE transactions on knowledge and data engineering*, *25*(8), pp.1772-1786.
21. Liu, M. and Qu, J., 2012, October. Mining high utility itemsets without candidate generation. In *Proceedings of the 21st ACM international conference on Information and knowledge management* (pp. 55-64).
22. Ashraf, M., Abdelkader, T., Rady, S. and Gharib, T.F., 2022. TKN: An efficient approach for discovering top-k high utility itemsets with positive or negative profits. *Information Sciences*, *587*, pp.654-678
23. Lin, J.C.W., Djenouri, Y. and Srivastava, G., 2021. Efficient closed high-utility pattern fusion model in large-scale databases. *Information Fusion*, *76*, pp.122-132.
24. Atallah, M., Bertino, E., Elmagarmid, A., Ibrahim, M. and Verykios, V., 1999, November. Disclosure limitation of sensitive rules. In *Proceedings 1999 Workshop on Knowledge and Data Engineering Exchange (KDEX'99)(Cat. No. PR00453)* (pp. 45-52). IEEE.

25. Yeh, J.S. and Hsu, P.C., 2010. HHUIF and MSICF: Novel algorithms for privacy preserving utility mining. *Expert Systems with Applications*, *37*(7), pp.4779-4786.
26. Yun, U. and Kim, J., 2015. A fast perturbation algorithm using tree structure for privacy preserving utility mining. *Expert Systems with Applications*, *42*(3), pp.1149-1165.
27. Lin, J.C.W., Wu, T.Y., Fournier-Viger, P., Lin, G., Zhan, J. and Voznak, M., 2016. Fast algorithms for hiding sensitive high-utility itemsets in privacy-preserving utility mining. *Engineering Applications of Artificial Intelligence*, *55*, pp.269-284.
28. Lin, J.C.W., Hong, T.P., Fournier-Viger, P., Liu, Q., Wong, J.W. and Zhan, J., 2017. Efficient hiding of confidential high-utility itemsets with minimal side effects. *Journal of Experimental & Theoretical Artificial Intelligence*, *29*(6), pp.1225-1245.
29. Li, S., Mu, N., Le, J. and Liao, X., 2019. A novel algorithm for privacy preserving utility mining based on integer linear programming. *Engineering Applications of Artificial Intelligence*, *81*, pp.300-312.
30. Liu, X., Wen, S. and Zuo, W., 2020. Effective sanitization approaches to protect sensitive knowledge in high-utility itemset mining. *Applied Intelligence*, *50*(1), pp.169-191.
31. Jangra, S. and Toshniwal, D., 2022. Efficient algorithms for victim item selection in privacy-preserving utility mining. *Future Generation Computer Systems*, *128*, pp.219-234.
32. Ashraf, M., Rady, S., Abdelkader, T. and Gharib, T.F., 2023. A Robust Privacy Preserving Approach for Sanitizing Transaction Databases from Sensitive High Utility Patterns. In International Conference on Advanced Intelligent Systems and Informatics (pp. 381-394). Springer, Cham.