A Heuristic Approach for Sensitive Pattern Hiding with Improved Data Quality

Shalini Jangra[1] and Durga Toshniowal[2]

[1] Indian Institute of Technology Roorkee, Roorkee, 247667, India
shalinijangra312@gmail.com
[2] Indian Institute of Technology Roorkee, Roorkee, 247667, India
durgatoshniwal@gmail.com

Abstract. Frequent itemset mining can be used to discover various interesting patterns present in dataset. However, this imposes a great privacy threat when data is shared with other organisations. There are some business critical frequent patterns that are considered as sensitive from organization's or individual's perspective because revealing such patterns can disclose confidential information. Privacy preserving data mining (PPDM) provides various techniques to hide sensitive patterns to make sure that they cannot be revealed by applying data mining models on shared datasets. Heuristic based sensitive pattern hiding techniques are widely adopted PPDM techniques due to their fast execution time but causes high side effects. In this paper, we propose a heuristic approach for sensitive pattern hiding based on deletion of Victim items which is named as MinMax. In the proposed algorithm, Misses Cost Impact (MCI) value of each tentative Victim item is calculated and item with minimum MCI is selected as Victim item resulting in low Misses Cost. Experimental results on benchmark datasets show that proposed algorithm achieves better data quality with less execution time as compared to existing heuristic based techniques.

Keywords: Privacy Preserving Data Mining, Data Privacy, Sensitive Patterns, Hiding Failure, Misses cost

1 Introduction

Frequent itemset mining to discover unrevealed patterns present in data benefits the businesses in their various decision making policies. Unveiling of these hidden patterns brings a threat to the privacy of sensitive and confidential information present in data [1]. For example, analysis of financial and medical records can give remarkable business and research benefits but privacy breach might allow business competitors and malicious users to misapply the information that can incur great remunerative and social loss. Privacy preserving data mining was introduced to diminish privacy issues by concealing the sensitive information while enabling data mining models to extract required information. Number of sensitive pattern hiding techniques are proposed by various researchers, which

are majorly divided into three categories i.e. heuristic based, border based and exact techniques.

Heuristic based techniques have drawn more attention of researchers due to their simplicity and fast execution time [9,10]. However these techniques experiences high side effects and provide suboptimal solution [14]. Preserving the adequate balance between data quality and data privacy has been the prominent issue because if no required information can be mined from the data, there is no use of hiding all the sensitive information. The quality of any sensitive pattern hiding technique predominantly depends on two performance metrics: Misses Cost (MC) and Hiding Failure (HF). Number of non-sensitive frequent patterns accidently concealed in order to conceal sensitive patterns accounts for Misses Cost. Number of sensitive frequent patterns that are not concealed by pattern hiding technique accounts for Hiding Failure. These two factors clearly depends on two things: *Victim Item Selection* and *Transaction Selection*. Many heuristic techniques [2,8] removes some of sensitive transactions from dataset to decrease the support of sensitive itemsets below minimum support threshold that can result into great reduction of dataset size. While other techniques delete the Victim items from sensitive transactions. Most of Victim item deletion techniques select the item on the basis of support count. For instance, MaxFIA [9] selects the item having highest support count as Victim item since it results in less probability of non-sensitive itemsets to be infrequent. Also selection of optimal transaction to delete Victim item plays a crucial role which can on the basis of transaction size [13], degree of conflict (DoC) [9], relevance of transaction with non-sensitive information (RoT) [4], etc.

This paper proposes a new heuristic based algorithm, **MinMax** that differs in the method of *Victim Item Selection*. The principle behind MinMax algorithm is to select the Victim item which appears in less number of non-sensitive patterns and first mask the sensitive itemset whose Victim item appears in more number of non-sensitive itemsets as compared to Victim item of other sensitive itemsets. The concept of RoT is chosen for selecting the transaction. Many experiments are conducted on some benchmark datasets to compare the performance of the proposed algorithm with some traditional heuristic techniques which demonstrate that MaxMin preserves a better data quality with commensurate execution time.

The remainder of this paper is organised as follows. Section 2 presents a briefing on some existing sensitive pattern hiding heuristic techniques. Section 3 provides a brief introduction of basic terminologies and problem statement. Section 4 describes proposed algorithm with demonstrating example. In section 5, the performance of the proposed algorithm is analyzed with experiments. Final conclusion is given in section 6.

2 Related Work

Numerous heuristic based sensitive pattern hiding and association rule hiding techniques exist in the literature that sanitize the data before applying data min-

ing models. The authors in paper [3] proposed a sensitive rule hiding approach in which sensitive itemset with highest support is preferred to hide first. In this paper, a graph of large itemsets is formed. This itemset graph is traversed bottom up followed by top down to mask sensitive patterns. This approach works fine on small datasets but not feasible on large datasets. Apart from hiding all sensitive patterns, other performance factors of algorithm are not evaluated. In paper [9], three itemset hiding heuristics named MaxFIA, MinFIA and IGA were proposed to maintain better data quality with data privacy. Transactions are selected on the basis of their degree of conflict. An efficient, scalable and one-scan heuristic named Sliding Window Algorithm (SWA) is proposed in the paper [10]. Transactions are selected in increasing order of their length since shorter transactions have less combinations of sensitive rules. Highest frequency item of sensitive itemsets present in selected transaction is chosen as victim item. Transactions coming under a k-sized window are sanitized once in sequential manner that imposes a scalability issue for large datasets. The paper [2] presents three heuristic approaches (*Aggregate*, *Disaggregate* and *Hybrid*) that promise better data quality at the cost of computational speed. *Aggregate* approach is based on transaction deletion while *Disaggregate* approach alter the transaction by deleting the items to reduce support of sensitive itemsets. *Hybrid* approach is a combination of the above two approaches that first identifies the transaction using *Aggregate* approach and then delete item from selected transaction using *Disaggregate* approach.

Some researchers proposed blocking based rule hiding approaches that replace sensitive information by some unknown items [13,15]. In blocking approach, apart from the addition of unknown items rest of dataset remains same. Therefore it becomes easy to restore original dataset [12]. In paper [4], item with maximum support count is selected as Victim Item like MaxFIA but transactions are selected in descending order of their relevance with non-sensitive itemsets. The paper [16] proposed an efficient distortion based rule hiding method through deletion and reinsertion of items. To reduce the misses cost, correlation between sensitive and non-sensitive rule is calculated and item with minimum influence on non-sensitive itemsets is selected as Victim item. The papers [5] and [6] proposes sanitization methods on incremental datasets. Paper [5] maintains a tree like data structure to enhance the execution speed and have less side effects. Although it is not efficient for dense datasets. In the approach, proposed in [6], sanitization process is applied only on incremented part of dataset. A dynamic itemset hiding algorithm that considers the multiple support threshold values ia proposed in the paper [11]. It uses item-deletion based sanitization approach on whole dataset hence reduces Msses Cost.

Above discussion ensures that there is great scope for researchers to explore different aspects of PPDM techniques like quality and scalability on different types of datasets i.e. static datasets, incremental datasets, dense datasets and sparse datasets etc.

3 Background

This section provides a brief introduction of basic terminologies and problem statement.

3.1 Basic Terminologies

1. **Frequent Itemsets:** Any itemset f_i having support greater than minimum support threshold value is frequent itemset i.e. $\sup(f_i) \geq \mid D \mid \times \delta$, where $\sup(f_i)$ is equal to the total number of transactions having itemset f_i, $\mid D \mid$ is the size of dataset D and δ is minimum support threshold value. For example, Table 3 shows the discovered frequent itemsets of an example database shown in Table 1 under δ=0.5.
2. **Sensitive Itemsets:** If the presence of any frequent itemset is able to discover any sensitive pattern, sequence etc. that can reveal some personal and confidential information regarding a company or an individual which they don't want to share, then it will be considered as a sensitive itemset. For example, any attribute or combination of attributes that can reveal the identity of a patient in medical records is considered as sensitive.
3. **Degree of Conflict (*DoF*):** It is defined as number of sensitive itemsets a transaction T contains. If S=$\{s_1, s_2, ..., s_n\}$ is set of sensitive itemsets, then
$$DoF(T) = \sum_{i=1}^{n} T(s_i) \tag{1}$$
where $\mathrm{T}(S_i) = 1$, when $S_i \subseteq T$, otherwise $\mathrm{T}(S_i) = 0$.
4. **Relevance of Transaction (*RoT*):** The relevance of a transaction is calculated as:
$$RoT(T) = \frac{1}{1 + NUM_{non-sens}(T)} \tag{2}$$
where $NUM_{non-sens}(T)$ is equal to number of non-sensitive itemsets transaction T supports.
5. **Misses Cost Impact (*MCI*):** Misses Cost Impact of an item 'i' equal to the total number of non-sensitive itemsets in which item 'i' appears.
6. **Victim Item:** Victim item x of a sensitive itemset s_i is an 1-itemset, such that $x \subseteq s_i$ and x is chosen for deletion in order to mask s_i.

Table 1: An example database

TID	Items
T1	B C D E F
T2	A C E F G
T3	C D F
T4	A C F G
T5	B C D F G
T6	C E G

Table 2: Projected database

TID	Items
T1	B C D E F
T2	A C E F G
T3	C D F
T4	A C F G
T5	B C D F G
T6	C E G

Table 3: Discovered frequent itemsets

1-itemset	count	2-itemset	count	3-itemset	count
a	5	ab	4		
b	8	bc	5	bce	4
c	7	be	6		
e	8	ce	6		

3.2 Problem Statement

The problem of sensitive pattern hiding is described as follows. Let D be the original source dataset and F is set of associated frequent itemsets under some minimum support threshold say δ. Let S is the subset of set of frequent itemsets having itemsets that can be helpful to derive confidential patterns hence considered as set of sensitive itemsets. The problem of sensitive frequent itemset hiding is to sanitize the data by decreasing the support count of sensitive frequent itemsets less than minimum support count value i.e $\sup(s_i) < \mid D \mid \times \delta$, so that sensitive itemsets do not appear as frequent itemsets in sanitized dataset. The problem of sensitive patten hiding mainly revolves around two things: 1) which item should be selected as Victim item for deletion to suppress a particular pattern, 2) From which transaction that selected Victim item should be deleted. Removal of Victim item results into hiding of non-sensitive frequent patterns that accounts for increasing Misses Cost. Therefore, item having less impact on non sensitive itemsets should be selected as Victim item. Transactions supporting at least one of sensitive itemsets should be considered for modification, since alteration of other transactions does not exert any impact on support of sensitive patterns. Therefore sensitive pattern hiding is transforming the original dataset D into released dataset D' such that most of non-sensitive information and none of sensitive information can be derived from D'.

4 Proposed Solution: MinMax Algorithm

The rationale behind MinMax algorithm is to select any item as Victim item if it appears in less number of non-sensitive patterns. Misses Cost Impact (MCI) of an item gives the count of number of non-sensitive itemsets in which that item appears. MCI of each 1-frequent itemsets is calculated using Algorithm 1. A list called Affinity List (AL) is maintained to have tentative Victim items and corresponding MCI values. After calculating MCI values, dataset is sanitized using Algorithm 2. For each sensitive itemset, we choose the item having lowest MCI value as Victim item (step 1-3), that contribute to Min part of algorithm MinMax. Item having lowest MCI value is picked because it will reduce Misses Cost. Then sensitive itemsets are sorted in decreasing order of their Victim item's MCI value (step 4) such that sensitive itemset whose Victim item's

Table 4: Characteristics of used datasets

Dataset	Number of Transactions	Number of Distinct Items	Average length of Transactions
Chess	3196	76	37.0
Mushroom	8124	120	23.0
BMS-1	59602	497	2.5

MCI value is largest as compared to *MCI* values of other sensitive itemset's Victim items is prefered to sanitize first. It contributes to Max part of algorithm MinMax. Item selected as Victim item for a particular sensitive itemset X may be present in other sensitive itemset Y but not selected as Victim item for Y due to its higher *MCI* value than selected Victim item. Sanitizing X first will reduce the support count of Y also. This is the main idea behind sorting of sensitive itemsets. Sensitive transactions are extracted from original dataset and stored in dataset D' (step 5).D' is dataset having transactions sorted according to their relevance value (step 6). Victim items are deleted from selected transactions to sanitize the dataset D' and support count of other affected sensitive and non-sensitive itemsets are updated (step 7-14). $\#IterToSanitize(s_i)$ is the total number of transactions from which Victim item selected for masking of s_i needs to be deleted and *TransToModify* are those selected transactions. Then sanitized dataset is returned (step 15) after removing the Victim items from selected transactions.

4.1 Example

Consider that S={D, CG, CF} is set of sensitive itemsets randomly selected from frequent itemsets shown in Table 3. Table 2 shows the projected dataset

Algorithm 1 *MCI* Calculation

Input: Set of 1-frequent itemsets, set of non-sensitive itemsets i.e NS
Output: Affinity list with items and corresponding *MCI*

1: Create an Affinity list *AL* having tentative victim item and corresponding *MCI* value.
2: **for** each 1-frequent item x **do**
3: $MCI(x)=0$
4: **for** each $ns_i \in$NS **do**
5: **if** x$\subseteq NS_i$ **then**
6: Increment $MCI(x)$ by 1.
7: **end if**
8: **end for**
9: AL.append(x, $MCI(x)$)
10: **end for**
11: **return** *AL*

Algorithm 2 MinMax algorithm

Input: S=$\{s_1, s_2, ...s_n\}$, set of sensitive frequent itemsets, NS=$\{ns_1, ns_2, ...ns_m\}$, set of non-sensitive frequent itemsets.
Output: A sanitized dataset.

1: **for** each sensitive itemset $s_i \in S$ **do**
2: Victim(S_i)← $item_v$ such that $item_v \in s_i$ and $\forall$ $item_k \in S_i$ $MCI(item_v) \leq MCI(item_k)$ {**Min-Part**}
3: **end for**
4: Sort the sensitive itemsets in decreasing order of *MCI* of their respective victim item {**Max-Part**}
5: D' ←D, where D' made up of transactions containing atleast one of sensitive itemsets.
6: Sort the transactions in D' by their relevance value in descending order
7: **for** each $s_i \in S$ **do**
8: *#IterToSanitize*(s_i)= |T[s_i]| - (| D | × δ)+1
9: *TransToModify* ← Select first *#IterToSanitize*(s_i) transactions from sorted D' that contains s_i as subset.
10: **for** each T ∈ *TransToModify* **do**
11: T ← (T - Victim(S_i))
12: Decrease the support of other affected sensitive and non sensitive itemsets
13: **end for**
14: **end for**
15: **return** sanitized dataset.

having transactions containing atleast one of these sensitive itemsets. Below are the steps to sanitize the example dataset using proposed algorithm.

1. **Misses Cost impact calculation:**
   AL={*MCI*(C)=5, *MCI*(D)=3, *MCI*(F)=5, *MCI*(G)=3}.
2. **Victim item selection:**
   Victim(D)=D, Victim(CG)=G since *MCI*(G) is less than *MCI*(C), Victim(CF)=C since *MCI*(C)=*MCI*(F), so any one of C and F can be selected.
3. **Sorting of sensitive itemsets:**
   S={CF, CG, D} is set of sensitive itemsets sorted in decreasing order of *MCI* value of their Victim item. Since *MCI* value of CF is maximum among sensitive itemsets hence selected for masking first.
4. **Sanitization:**
   #IterToSanitize(CF) =5-3+1=3 and TransToModify = {T2,T3,T4} are selected transactions according to *ROT* values of transactions.
5. **Support Reduction**
   Deleting item C from these transactions decreases the support count of CG along with complete masking of CF.

Similarly, two other sensitive itemsets are masked. Misses Cost of proposed algorithm MinMax turns out to be 5 with deletion of total 5 items while Greedy, MaxFIA and MinFIA incur Misses Cost 6, 6 and 7 with deletion of 5, 5 and

Table 5: Misses Cost with varying percentage of sensitive itemsets

Dataset	MST	Sens_Per	MinMax	Maxfia	Minfia	Greedy
Chess	0.9	1	**155**	203	179	210
		2	**271**	364	281	335
		3	**284**	335	292	313
		4	**223**	279	243	265
		5	**245**	318	252	282
Mushroom	0.4	5	**217**	363	227	252
		6	**221**	310	233	292
		7	**262**	312	269	289
		8	**220**	253	229	237
		9	**281**	305	298	310
Bms-1	0.001	1	**532**	686	623	576
		2	**839**	989	919	885
		3	**1244**	1410	1327	1326
		4	**1393**	1762	1487	1471
		5	**1388**	1560	1441	1490

6 items respectively. It indicates that MinMax preserves better data quality due to less Misses Cost as compared to Greedy, MaxFIA and MinFIA.

5 Experimental Results

The performance of the proposed algorithm is compared to some existing algorithms namely MaxFIA [9], MinFia [9] and Greedy [4]. All of these algorithms completely hide the sensitive itemsets hence value of hiding failure for all of them is zero. Effect of sanitization on quality of datasets is determined by Misses Cost. Three performance parameters are taken into consideration:

1. Misses Cost
2. Data Loss (in terms of no. of item deleted throughout sanitization process)
3. Execution Time

Three real-world benchmark datasets Chess, Mushroom and BMS-1 are used in experiments. The chess and Mushroom datasets are available on Frequent Itemset Mining Dataset Repository present at link `http://fimi.uantwerpen.be/data/`. The other dataset, BMS-1 is click-stream data from a webstore used in KDD-Cup 2000 [7] and accessed from SPMF: An Open-Source Data Mining Library through link `http://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php`. Table 4 shows the characteristics of these datasets. All the experiments are conducted on randomly selected set of sensitive itemsets. Performance of the algorithm is evaluated by varying minimum support threshold (MST) and percentage of sensitive itemsets (Sens_Per). For each combination of Sens_Per and MST, five samples of sensitive itemsets are randomly drawn. Average value of each performance factor on these five samples is considered for result comparison. In the experiments, the value of MST and Sens_Per

Table 6: Number of item deleted with varying percentage of sensitive itemsets

Dataset	MST	Sens_Per	MinMax	Maxfia	Minfia	Greedy
Chess	0.9	1	200	202	205	203
		2	465	434	466	406
		3	514	428	517	399
		4	413	386	402	365
		5	462	431	433	378
Mushroom	0.4	5	5553	4315	5525	4448
		6	5771	6226	5588	5968
		7	6795	5833	6803	5997
		8	5267	5351	5076	5115
		9	7655	6240	7494	6759
Bms-1	0.001	1	2587	2564	2585	2555
		2	4930	4914	4924	4900
		3	7692	7613	7707	7586
		4	8553	7677	8573	8486
		5	10406	10450	10418	10475

parameters are different for each dataset, adjusted based on each dataset's characteristics.

5.1 Varying Percentage of Sensitive Itemsets

The performance of the proposed algorithm is evaluated on the datasets by varying percentage of sensitive itemsets. It is shown in Table 5 that Misses Cost incurred by proposed algorithm on used datasets is less than the other algorithms, hence proposed algorithm, MinMax ensures better quality of data while preserving its privacy. Table 6 shows the number of deleted items by different algorithms to lower the support of sensitive itemsets which concludes that Data Loss by MinMax is less than MinFIA and slightly greater than MaxFIA and Greedy algorithm. Here, Data Loss is measured in terms of item deletion hence it will not result in much higher dropping of data as compared to other algorithms. Fig. 1.(a) shows execution time taken by algorithms on all the three datasets. Execution time of MinMax algorithm is less than MaxFIA and MinFIA algorithms and commensurate to Greedy algorithm. This is due to sorting of dataset and selection of Victim items according to non-sensitive patterns.

5.2 Varying Minimum Support Threshold

To analyze the influence of different minimum support threshold values on proposed algorithm's performance, further experiments are carried out on the selected datasets. It can be concluded from Table 7 that Misses Cost incurred by MinMax algorithm is less than all other three algorithms that promises better data quality. Table 7 also shows that on increasing the MST value, Misses Cost of algorithms decreases. But this is due to less number of frequent itemsets

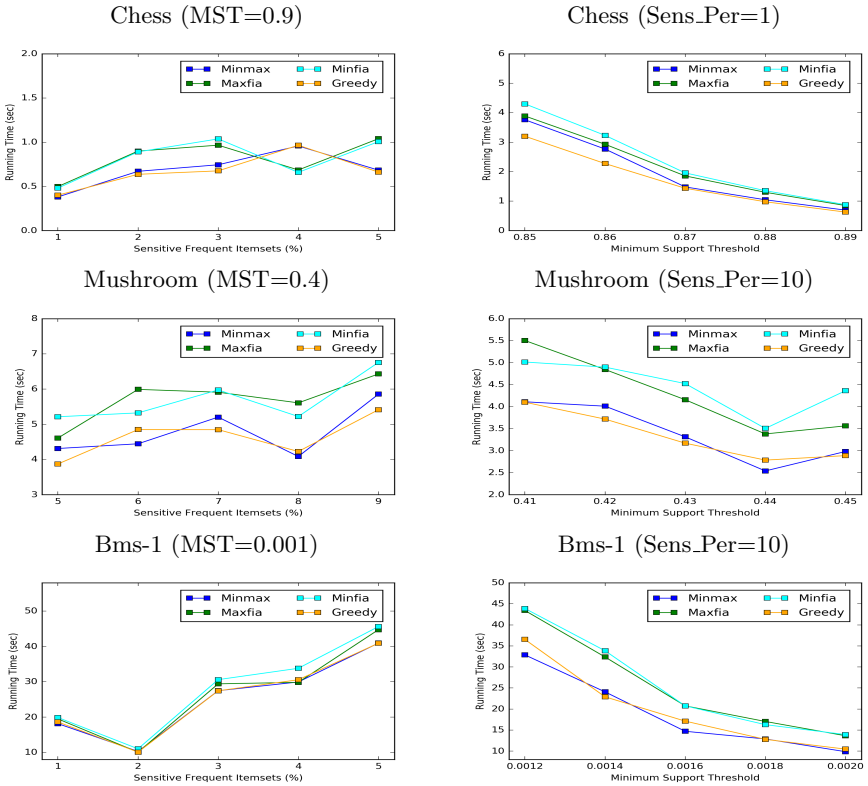
Table 7: Misses Cost with varying minimum support threshold

Dataset	MST	Sens_Per	MinMax	Maxfia	Minfia	Greedy
Chess	0.85	1	**1545**	1756	2367	1712
	0.86		**1081**	1227	1102	1161
	0.87		**635**	816	680	781
	0.88		**466**	622	560	546
	0.89		**287**	389	314	340
Mushroom	0.41	10	**186**	220	186	203
	0.42		**181**	280	187	187
	0.43		**158**	173	161	165
	0.44		**115**	121	117	122
	0.45		**125**	136	155	131
Bms-1	0.0012	10	**933**	977	962	951
	0.0014		**553**	587	593	572
	0.0016		**348**	395	356	378
	0.0018		**258**	270	270	260
	0.0020		**192**	195	198	206

generated on increased MST. It is indicated from Table 8 that number of items deleted by proposed algorithm is less than the MinFIA and greater than MaxFIA and Greedy which concludes that Misses Cost is majorly affected by selection of Victim item & not by total number items deleted during sanitization. Execution time taken by proposed algorithm is commensurate to Greedy and less than MaxFIA and MinFIA which is shown in Fig. 1.(b) for used datasets. BMS-1 took more time for execution because of its largest dataset size among three.

Table 8: Number of item deleted with varying minimum support threshold

Dataset	MST	Sens_Per	MinMax	Maxfia	Minfia	Greedy
Chess	0.85	1	1029	870	1017	846
	0.86		915	762	931	704
	0.87		564	513	571	515
	0.88		484	436	498	419
	0.89		364	329	370	297
Mushroom	0.41	10	5863	5939	5541	5843
	0.42		6062	5571	6198	5665
	0.43		5952	4927	5815	5014
	0.44		4389	3941	4346	4123
	0.45		5542	3504	5647	3294
Bms-1	0.0012	10	17388	17310	17525	17257
	0.0014		17880	17631	18122	14360
	0.0016		14083	14460	14007	14428
	0.0018		14541	14902	14583	14892
	0.0020		14585	14465	14431	14446



(a) With varying percentage of sensitive itemsets

(b) With varying minimum support threshold value

Fig. 1: Execution Time

6 Conclusion

Various heuristic based techniques for sensitive pattern hiding have been proposed by the researchers. In these techniques, maintaining the balance between data quality and data privacy has been the biggest challenge. This paper has proposed a new efficient heuristic technique which preserves better data quality as compared to existing knowledge hiding heuristics. Proposed algorithm considers the impact of Victim item deletion on non-sensitive knowledge while selecting the Victim item and corresponding transaction. This heuristic can conceal all of sensitive itemsets with less Misses Cost as compared to some of existing heuristic based techniques. Experiments show that the proposed technique performs well in terms of execution time on small datasets. It incurs high computational cost for large datasets due to its sequential nature. Future research will intend to improve the proposed algorithm so that data privacy along with good data quality can be achieved on big datasets within real execution time.

References

1. Aggarwal, C.C., Philip, S.Y.: A general survey of privacy-preserving data mining models and algorithms. In: Privacy-preserving data mining, pp. 11–52. Springer (2008)
2. Amiri, A.: Dare to share: Protecting sensitive knowledge with data sanitization. Decision Support Systems **43**(1), 181–191 (2007)
3. Atallah, M., Bertino, E., Elmagarmid, A., Ibrahim, M., Verykios, V.: Disclosure limitation of sensitive rules. In: Proceedings 1999 Workshop on Knowledge and Data Engineering Exchange (KDEX'99)(Cat. No. PR00453). pp. 45–52. IEEE (1999)
4. Cheng, P., Roddick, J.F., Chu, S.C., Lin, C.W.: Privacy preservation through a greedy, distortion-based rule-hiding method. Applied Intelligence **44**(2), 295–306 (2016)
5. Dai, B.R., Chiang, L.H.: Hiding frequent patterns in the updated database. In: 2010 International Conference on Information Science and Applications. pp. 1–8. IEEE (2010)
6. Jadav, K.B., Vania, J., Patel, D.: Efficient hiding of sensitive association rules for incremental datasets. International Journal of Innovations & Advancement in Computer Science (IJIACS) (2014)
7. Kohavi, R., Brodley, C.E., Frasca, B., Mason, L., Zheng, Z.: Kdd-cup 2000 organizers' report: Peeling the onion. SIGKDD explorations **2**(2), 86–98 (2000)
8. Lin, C.W., Hong, T.P., Hsu, H.C.: Reducing side effects of hiding sensitive itemsets in privacy preserving data mining. The Scientific World Journal **2014** (2014)
9. Oliveira, S.R., Zaiane, O.R.: Privacy preserving frequent itemset mining. In: Proceedings of the IEEE international conference on Privacy, security and data mining-Volume 14. pp. 43–54. Australian Computer Society, Inc. (2002)
10. Oliveira, S.R., Zaïane, O.R.: Protecting sensitive knowledge by data sanitization. In: Third IEEE International Conference on Data Mining. pp. 613–616. IEEE (2003)
11. Öztürk, A.C., Ergenç, B.: Dynamic itemset hiding algorithm for multiple sensitive support thresholds. International Journal of Data Warehousing and Mining (IJDWM) **14**(2), 37–59 (2018)
12. Pontikakis, E.D., Theodoridis, Y., Tsitsonis, A.A., Chang, L., Verykios, V.S.: A quantitative and qualitative analysis of blocking in association rule hiding. In: Proceedings of the 2004 ACM workshop on Privacy in the electronic society. pp. 29–30. ACM (2004)
13. Saygin, Y., Verykios, V.S., Elmagarmid, A.K.: Privacy preserving association rule mining. In: Proceedings Twelfth International Workshop on Research Issues in Data Engineering: Engineering E-Commerce/E-Business Systems RIDE-2EC 2002. pp. 151–158. IEEE (2002)
14. Sharma, S., Toshniwal, D.: Mr-i maxmin-scalable two-phase border based knowledge hiding technique using mapreduce. Future Generation Computer Systems (2018)
15. Wang, S.L., Jafari, A.: Using unknowns for hiding sensitive predictive association rules. In: IRI-2005 IEEE International Conference on Information Reuse and Integration, Conf, 2005. pp. 223–228. IEEE (2005)
16. Zamani Boroujeni, F., Hossein Afshari, D.: An efficient rule-hiding method for privacy preserving in transactional databases. Journal of computing and information technology **25**(4), 279–290 (2017)