iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 3 **Issue:** Issue I **Month of publication:** May 2015

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Potent Design Mining Concise and Lossless Representation of High Utility Itemsets

N.K.Anitha[1], Mrs.N.K.Priyadharsini[2]
Department of Computer Science and Engineering, P.A College of Engineering and Technology, Pollachi, India

Abstract - Mining high utility itemsets from databases is an important data mining task for discovery of itemsets with high utilities. High efficiency for mining task provides a concise mining to users. Mining closed and high utility itemsets serves as a compact and lossless representation of high utilities. Three efficient algorithms named Apriori based algorithm for mining High Utility itemsets, removing the exact utilities of itemsets, Estimated High Utility Itemset Discovery to find concise representation. Estimated High Utility Itemset Discovery is to recover all High Utility itemsets from the superset of estimated utility items without accessing the original database. Estimated High Utility Itemset Discovery algorithm includes two novel strategies named Remove Exact Utilities of items and estimated high utility itemset discovery with Global Transactional Utility Table that greatly enhance its performance. Results show Estimated High utility itemset discovery is much faster than all mining algorithms. Estimated High Utility Itemsets algorithm is used for recovering all items based on global transactional utility table and outperforms mining with High Utility Itemsets.
Keywords— Frequent itemset, High utility mining, concise and lossless representation, Global transactional utility

I. INTRODUCTION

Frequent itemset mining (FIM) is a fundamental research topic in data mining. One of its popular applications in market basket analysis refers to the discovery of items (itemsets) that are frequently purchased together by customers. The traditional model of FIM may discover a large amount of frequent but low revenue itemsets and lose the information on valuable itemsets having low selling frequencies. These problems are caused by the facts that (1) FIM treats all items as having the same importance/unit profit/weight and (2) it assumes that every item in a transaction appears in a binary form an item can be either present or absent in a transaction, that does not indicate its purchase quantity in the transaction. The utility of an itemset represents its importance that can be measured in terms of weight, profit, cost, quantity or other information depending on the user preference. An itemset is called a high utility itemset (HUI) if its utility is no less than a user-specified minimum threshold otherwise it is called a low utility itemset. Utility mining is an important task and has a wide range of applications such as website click stream analysis [2], [12], cross-marketing in retail stores, mobile commerce environment and biomedical applications. HUI mining is not an easy task, downward closure property [1], [10], [21] in FIM does not hold in utility mining.

The search space for mining HUIs cannot be directly reduced as it is done in FIM because a superset of a low utility itemset can be a high utility itemset. A very large number of high utility itemsets makes it difficult for the users to comprehend the results. It may also cause the algorithms to become inefficient in terms of time and memory requirement or even run out of memory. It is widely recognized that the more high utility itemsets algorithms generate the more processing time. The performance of the mining task decreases greatly for low minimum utility thresholds or dealing with dense databases [8], [18], [20].

II. RELATED WORKS

Utility mining (Yao & Hamilton, 2006; Yao, Hamilton, Butz, 2004) proposed to partially solve the performance problem. Utility mining would usually like to find high utility itemsets,their utility values are larger than or equal to a threshold value defined by users. The utility value of an itemset can be measured in terms of costs, profits or other measures from user preference. For example, someone may be interested in finding the itemsets with good profits and another may focus on the itemsets with low pollution while manufacturing.

Liu et al. then presented the two-phase algorithm for fast discovering all high utility itemsets (Liu, Liao, & Choudhary, 2005)

based on the downward-closure property. The property indicates that any superset of a non-frequent itemset is also non-frequent. It is called the anti-monotone property as well. The property is used to reduce the search space by pruning non-frequent itemsets early. The two-phase algorithm generates candidate high utility itemsets in a level-wise way. The database-scanning time is a bottleneck of the approach.

Han et al. [26] used the projection technique to sequential pattern mining, extended from association-rule mining with the additional consideration of the time factor. In this approach, each candidate sequence could be thought of as a query condition, and tuple in the database that contained these sequences were projected. The support count for each candidate sequence could also be easily obtained from the tuples. Researches assign different weights to items have been shown in proposed work. MINWAL [9] mines the weighted association rules in binary transaction databases based on the k-support bound property. An efficient association rules generation method, WAR [10], focuses on the generation of rules from the available frequent itemsets instead of searching for weighted frequent itemsets.

WARM [11] proposes a weighted ARM model where itemset weight is defined as the average weight value of the items comprising this itemset. [12] proposed a scheme that uniformly weights all the transactions without considering the differences among the items. These weighted ARM models are special cases of utility mining.

TABLE 1 An example database

T	Transaction Itemsets	GTU	ETU
T_1	Convex(1),smooth(1),almond(1),yellow(1)	5.3	5
T_2	Convex(1),smooth(1),almond(3)	8.2	8
T_3	Convex(1),smooth(1),white(2)	8.4	8
T_4	Almond(2),pink(1)	5.6	6
T_5	Convex(1),smooth(1),white(3)	11.4	11
T_6	Convex(1), almond(1), pink(1)	15.5	16

TABLE 2 Profit Table

Item	Convex	smooth	almond	white	pink	yellow
Prof	1	1	2	3	3	1

III. HIGH UTILITY MINING

Mining high utility itemsets from the databases is not an easy task and downward closure property is used in frequent itemset mining cannot be applied. Pruning search space for high utility itemset mining is difficult because a superset of a low utility itemset may be a high utility itemset. A naive approach for this problem is to enumerate all itemsets from the databases by the principle of exhaustion. Principle of exhaustion encounter the large search space problem and databases contain lots of long transactions. The effective prune search space and it efficiently capture all high utility itemsets with no missing items is a big challenge in utility mining.

In the table 1 Transactional itemsets are given with Global Transactional utility and values for itemsets are given in Estimated Transactional utility. The High utility itemsets are estimated with minimum threshold value and in table 2 items and profit for items are shown. Huge number of potential high utility itemsets forms challenging problem, mining performance and higher processing cost is incurred with more potential high utilityn itemsets.

The transaction utility of a transaction T_q, denoted as TU (T_q) is the sum of the utilities of all items in T_q

$$TU(T_q) = \sum_{ip \in Tq}^{n} U(i_p, T_q) \quad \text{.....(3.1)}$$

Where U(i_p ,T_q) is the utility of itemset for each transaction. Each transaction highly utilized items are stored in TWU table. The equation 3.1 and 3.2 shows the transactional weights and estimated utilities. The Transactional Weighted Utilization of an itemset X, denoted as TWU (X) is the sum of transaction utilities of all transactions containing X

$$TWU(X) = \sum_{x \in T_q}^{n} TU(T_q) \quad \ldots.(3.2)$$

The transaction weights are computed based on estimated utility values and weights are considered as high utility items.

IV. PROPOSED APPROACH

A. *Algorithms For Mining High Utility Itemsets*

We introduce mining algorithms like AprioriHC-D (AprioriHC algorithm with Discarding unpromising and isolated items), REG (Removing the Exact utilities of items from Global Utility table), Estimated High utility Itemset Discovery algorithm to find the set of items with closed utilities and check for recovery from the superset and utility unit array is removed then check for each transaction is completely recovered based on minimum threshold value.
These three algorithms are used for recovering the high utility itemsets from the superset and check that each transaction in above the minimum threshold value.

1) *APRIORIHC-D Algorithm:* AprioriHC-D algorithm is used to set the minimum threshold value based on transactional utility from the database table. The itemsets are scanned into the databases with their unit profit. Discard the global unpromising items that have low utilities and the set of closed high utility itemsets are taken.

Input: D: the database with abs_min_utility;

pCHUI: the set of Potentially Closed HUIs

Output: The complete set of Closed High Utility Itemsets.

pCHUI :=D

L1 := 1-HTWUIs in D

D1:= DGU_Strategy(D, L1)

L1:= 1-HTWUIs in D1

AprioriHC-D_Phase-I (D1, pCHUI, abs_min_utility, L1)

AprioriHC-D_Phase-II(D1, pCHUI, abs_min_utility)

Step 1: Set the threshold value based on transactional utility.

Step 2: The items are scanned in to the databases based on unit profit.

Step 3: Discarding the global unpromising items based on transaction utility are discarded.

2) *REG Algorithm:* Removing Exact utility of itemset from the Global Transactional Utility Table. Transactional itemsets are stored with Estimated Transactional utility. Then, check each item is above threshold value. Absolute utility of an item is marked. The exact utilities are removed and the items in the transactional utility table are with high utility items.

PROCEDURE: REG_Strategy

Input: g(ak): the Tidset of item ak; GTU: the global TU-Table

Output: GTU: the global TU-Table

Absolute utility itemset abs_utility_item (ak)

for each Transactional id R, g(ak) do {

remove absolute utility(ak, TR) from <R, GTU(TR)> }

Step1: All transactional itemsets are stored with the estimated transactional utility.

Step2: Check each item is above threshold value and item is stored in Global transactional utility table.

Step3: Absolute utility of an item can be removed from transactional utility of each transaction in global TU Table.

Step 4: Removing the exact utilities of items from Global TU-Table.

PROCEDURE: CalculateEstUtility

Input: g(X): the Tidset of X; TU: a TU table

Output: EstU: the estimated utility of X

EstU := 0;

for each Transactional ID Rg(X) do

{ EstU := Estimated Utility + Transactional Utility get(R) }return EstU

Step 1: Set the threshold value

Step 2: Generate apriori algorithm for itemsets and max utility is found based on threshold used.

Step 3: Based on the abs-max-utility the values, itemset are evaluated.

Step 4: Mark the closed itemsets based on the values the generated in apriori algorithm.

3) Estimated HUI Discovery Algorithm

The set of closed and high utility items for HUI Discovery are used. For each itemset the absolute utility value of an item should not be less than specified threshold value. Remove each item based on the support count with their profits. Several iterations are taken and the iteration stops when the value is less than the support count. All the items are recovered from the superset and values nearer are considered as high utility itemsets then checks with GTU-Table. Therefore recovery is done based on utility. Item that is near to the abs-utility value is also recovered. The utility unit array is removed and process is repeated for each itemset and check each transaction is completely recovered. Abs-utility is changed because transactional itemsets also differ from time to time. The process is repeated until by finding the HUI Discovery items.

Input: The maximum length of itemset in HC; abs_min_utility

Output: H: The complete set of HighUtility Items.

for (k := ML – 1; k > 0 ; k – –) do

for each k-itemset X = {a1, a2,…, ak HCk do

if (absolute_utility(X) < abs_min_utility) then

delete X from HCk . else add X and its absolute utility au(X) to H.

{ for each item ai (X) do

{ Y := X – {ai}

Absolute_ utility(Y) = au(X) – V(X, ai)

if(au (Y) abs_min_utility) then

{ if HCk-1 and Support Count (X) > Support Count (Y) then

{ SC(Y) := SC(X) }

else if HCk-1 then

{ HCk-1 := HCk-1(Y)

SC(Y) := SC(X);

Abs_utility_item (ak);

Based on the above algorithm the following steps have been followed in order to get the best results in proposed work.

Step 1: The set of items with closed utilities.

Step 2: For each itemset the absolute utility au(x) of an itemset should not be less than the minimum threshold.

Step 3: Remove each item based on the support count of X.

Step 4: Check for recovery from the superset.

Step 5: Calculate the absolute utility of an itemset and recovery based on utility.

Step 6: The utility unit array is removed and process is repeated for each itemset and check each transaction is completely recovered.

V. RESULTS AND DISSCUSSION

There are three kinds of datasets that are commonly encountered in real life scenarios (1) dense dataset, (2) sparse dataset and (3) dataset containing long transactions. In the experiments, we use real life Mushroom Dataset. The experimental results for Mushroom datasets are shown in figure. The scalability, efficiency and the memory consumption of Estimated High Utility Itemsets algorithms are shown in fig respectively.

VI. EXPERIMENTS ON MUSHROOM DATASET

The performance of the algorithms on the Mushroom da-taset shows the execution time of AprioriHC algorithm. The reason is AprioriHC simply apply the TWU-Model without using effective strategies to reduce the estimated utility of itemsets, AprioriHC-D runs faster than AprioriHC. Mushroom Datasets are used in high utility items and closed utility items are found based on minimum threshold value. AprioriHC and AprioriHC-D generate fewer High utility itemsets. This is because AprioriHC and produce items for Closed High Utility Itemsets but Apriori algorithm needs to produce items for all HUIs. AprioriHC needs to calculate the utility unit array of items and cost is not expensive. The smaller number of items generated by CHUI makes EHUI Itemsets to perform better in the execution time. Depending on the characteristics of datasets, the reduction ratios achieved by the proposed representation can be very different. For the dense dataset such as Mushroom, the proposed representation can achieve a massive reduction in the number of extracted patterns. For the sparse dataset such as Foodmart, the proposed representation achieves less reduction. The proposed representation may not achieve a massive reduction on very sparse datasets, it still has good performance in several real cases. The performance is evaluated for redundancy, loss of itemsets and efficiency based on high utility itemset and the minimum threshold value. Fig 5.1 shows the efficiency based on the execution Time. Fig 5.2 Redundancy based on Accuracy rules. Fig5.3 Minimum Loss based on distinct items.

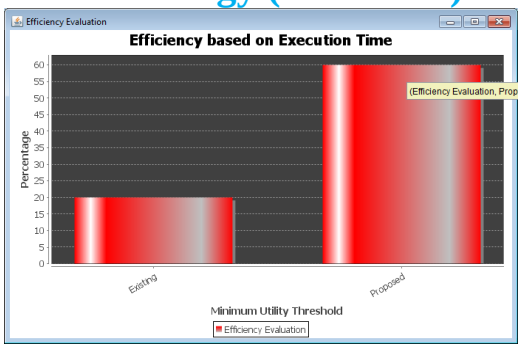


Fig 5.1-Execution Time based on min-threshold

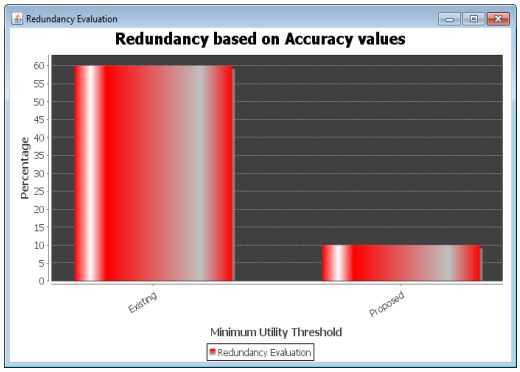


Fig 5.2 Redundancy based on Accuracy rules

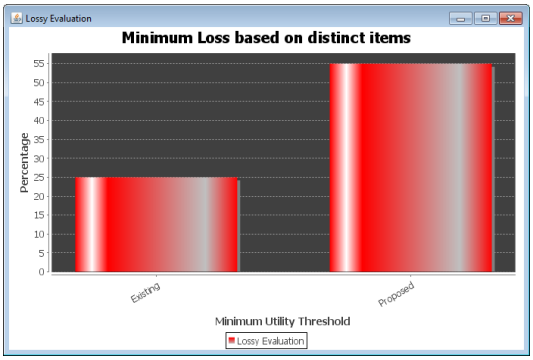


Fig 5.3 Minimum Loss based on distinct items

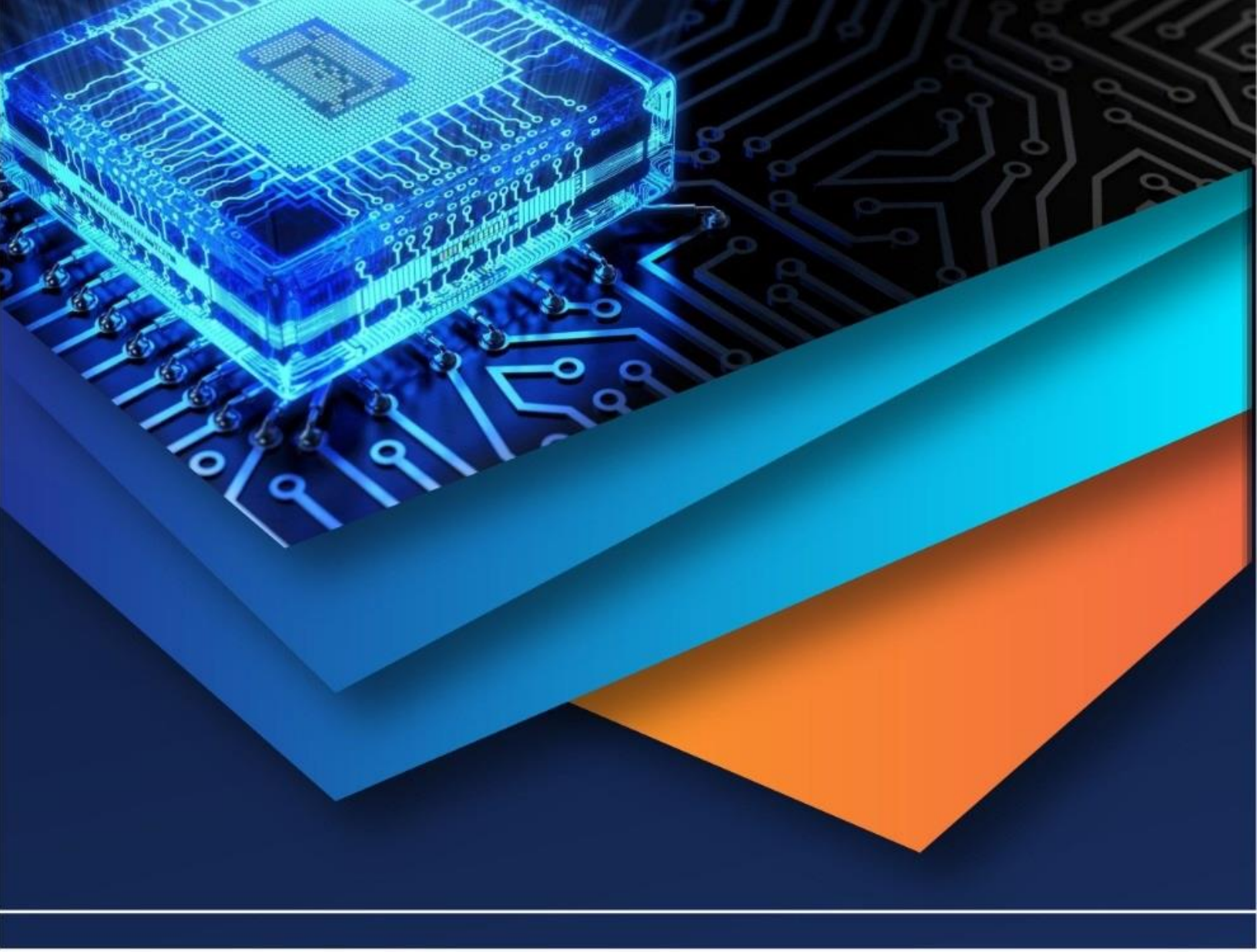
VII. CONCLUSION

The most efficient algorithm is Estimated High Utility Itemset (EHUI). The overall execution time of EHUI is always faster than UP-Growth, there are less Closed High Utility Itemsets than High Utilities. In the experiments, deriving all HUIs is inexpensive. EHUI performs better than UP-Growth, it may fail to recover all high utility itemsets due to the memory limitation there are too many HUIs in the database. EHUI performs better than the proposed two Apriori based approaches, some applications such as discovering patterns in the presence of the memory constraint, Apriori-based approach is preferred and plays an essential role. AprioriHC-D perform a breadth-first search for mining high utility itemsets from horizontal database, EHUI performs a depth first search for mining closed and high utility itemsets from vertical database. The strategies incorporated in EHUI are efficient and novel. They have never been used for vertical mining of high utility itemsets. For efficient recovery of all high utility itemsets based on high utility values efficient method named EHUI is used. Results on both

real and synthetic datasets shows that the proposed representation achieve a massive reduction in the number of high utility itemsets on all real datasets.

REFERENCES

[1] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules," in Proc. of the 20th Int'l Conf. on Very Large Data Bases, pp. 487-499, 1994.
[2] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, "Efficient Mining of Associ-ation Rules using Closed Itemset lattice," Journal of Information Systems, Vol 24, Issue 1, pp. 25–46, 1999.
[3] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," in Proc. of the ACM SIGMOD Int'l Conf. on Management of Data, pp. 1-12, 2000.
[4] K. Gouda and M. J. Zaki, "Efficiently Mining Maximal Frequent Itemsets," in Proc. of the IEEE Int'l Conf. on Data Mining, pp. 163-170, 2001.
[5] T. Calders and B. Goethals, "Mining All Non-derivable Frequent Itemsets," in Proc. of the Int'l Conf. on European Conference on Principles of Data Mining and Knowledge Discovery, pp. 74-85, 2002.
[6] R. Chan, Q. Yang, and Y. Shen, "Mining High Utility Itemsets," in Proc. of the IEEE Int'l Conf. on Data Mining, pp. 19-26, 2003.
[7] H. Li, J. Li, L. Wong, M. Feng, Y. Tan, "Relative risk and odds ratio: a data mining perspective," in Proc. of the ACM SIGACT-SIGMOD-SIGART Sympo-sium on Principles of Database Systems, pp. 368-377, 2005.
[8] C. Lucchese, S. Orlando and R. Perego, "Fast and Memory Efficient Mining of Frequent Closed Itemsets," IEEE Transactions on Knowledge and Data Engi-neering, Vol. 18, Issue 1, pp. 21-36, 2006.
[9] U. Yun, "Mining Lossless Closed Frequent Patterns with Weight Con-straints," Knowledge-Based Systems, Vol. 20, pp. 86–97, 2007
[10] A. Erwin, R. P. Gopalan, and N. R. Achuthan, "Efficient Mining of High utility Itemsets from Large Datasets," in Proc. of the Int'l Conf. on Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 554-561, 2008.
[11] K. Chuang, J. Huang, M. Chen, "Mining Top-K Frequent Patterns in the Presence of the Memory Constraint," VLDB Journal, Vol. 17, pp. 1321-1344, 2008.
[12] C. F. Ahmed, S. K. Tanbeer, B.S. Jeong, and Y.-K. Lee, "Efficient Tree Struc-tures for High utility Pattern Mining in Incremental Databases," IEEE Transactions on Knowledge and Data Engineering, Vol. 21, Issue 12, pp. 1708-1721, 2009.
[13] T. Hamrouni, S. Yahia, E. M. Nguifo, "Sweeping the Disjunctive Search Space Towards Mining New Exact Concise Representations of Frequent Itemsets," Data & Knowledge Engineering, Vol. 68, Issue 10, pp. 1091-1111, 2009.
[14] B.-E. Shie, V. S. Tseng, and P. S. Yu, "Online Mining of Temporal Maximal Utility Itemsets from Data Streams," in Proc. of Annual ACM Symposium on Applied Computing, pp. 1622-1626, 2010.
[15] V. S. Tseng, C.-W. Wu, B.-E. Shie, and P. S. Yu, "UP-Growth: An Efficient Algorithm for High Utility Itemset Mining," in Proc. of the ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, pp. 253–262, 2010.
[16] B.-E. Shie, H.-F. Hsiao, V. S. Tseng and P. S. Yu, "Mining High Utility Mobile Sequential Patterns in Mobile Commerce Environments," in Proc. of the Intl. Conf. on Database Systems for Advanced Applications and Lecture Notes in Computer Science (LNCS), Vol. 6587/2011, pp. 224-238, 2011.
[17] C.-W. Lin, T.-P. Hong, W.-H. Lu, "An Effective Tree Structure for Mining High Utility Itemsets," Expert Systems with Applications, Vol. 38 Issue 6, pp. 7419-7424, 2011.
[18] C.-W Wu, P. Fournier-Viger, P. S. Yu. and V. S. Tseng, "Efficient Mining of a Concise and Lossless Representation of High Utility Itemsets," in Proc. of the IEEE Int'l Conf. on Data Mining, pp. 824-833, 2011.
[19] C.-W. Wu, B.-E. Shie, V. S. Tseng and P. S. Yu, "Mining Top-k High Utility Itemsets," in Proc. of the ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining, pp. 78–86, 2012
[20] T. Hamrouni, "Key Roles of Closed Sets and Minimal Generators in Concise Representations of Frequent Patterns," Intelligent Data Analysis. Vol. 16, Issue 4, pp. 581-631, 2012.
[21] G.-C. Lan, T.-P. Hong, V. S. Tseng, "An Efficient Projection-based Indexing Approach for Mining High Utility Itemsets.Knowledge and Information System, Vol. 38, Issue 1, pp. 85-107, 2014.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY