

Mining of High Value Item Sets with Negative Utility

Aatif Jamshed, Bhawna Mallick

Abstract: The patterns generated by frequent pattern mining aims to find the frequent items without considering the utilities of the different items. The traditional association rule mining treats all items to be of equal utility. This is not always the case for a real world application. Utility based data mining is a new area of research and is complementing the frequency based approach. The main objective of Utility Mining is to identify the item sets with highest utilities, by considering profit, quantity, cost or other user preferences as the Utility of the item. Recent approaches developed so far considers the utilities of items to be same over a particular period of time. In our approach we have proposed that the utility of items vary over a period of time. Our work also proposed that the utility of items may also assume negative values. Our work thus treats the data mining in more realistic manner

Index Terms: Mining, High value Item set, Utility mining, Negative value.

I. INTRODUCTION

Structured knowledge can be extracted from large datasets with the help of data mining. Data mining tasks like Classification and Clustering also use frequent pattern mining and it also forms the premise for association rule mining. Clustering also use frequent pattern mining and it also forms the premise for association rule mining. Relationships of the form $X \square Y$ (amongst item sets that occur together in a database where X and Y are disjoint item sets) can be found using association rule mining. But frequent pattern mining algorithm has a drawback that it works for a limited set of applications where the items are treated to be of equal utility to the business. It did not consider quantity and weight that are significant for addressing real world decision problems that require maximizing the utility in an organization. Here comes the role of utility based data mining that aims to find the item sets that are of high utility to the business. It treats every item to be of different significance and utility of different items can express as function of cost, profit etc. For e.g. selling a microwave may occur less frequently than a loaf of bread and butter in a superstore, but the former gives a much higher profit per unit sold. The high utility item set mining problem is to find all item sets that have utility larger than a user specified value of $min_utility$. Hence frequency of item sets is not a sufficient indicator of item's interestingness. Other measures like utility should also be taken into consideration.

Revised Manuscript Received on July 10, 2019.

Aatif Jamshed, Information Technology, ABES Engineering College, Ghaziabad, India

Bhawna Mallick, Computer Science, Maverick Quality Advisory Services Pvt.Ltd Ghaziabad, India

Most of the algorithms developed so far for utility mining either followed a priority based approach which relied on the technique of candidate generate and test method. These approaches suffered from high time costs more over they always run under the jeopardy of combinatorial explosion. Moreover the algorithms did not take into account into this kind of activity. dynamic nature of utility and possibility of the negative value of utility. As more constraints on the utility factors are applied the computational complexity for the algorithms also increases therefore in a nutshell we can say that not only simplification in the calculation of utility is the challenge but also candidate size restriction is also an issue that most of the algorithms tried to address. Before HUIVIV [5] no other work was taking into consideration that utility can also take negative values, when wophas algorithm [3] was applied to the scenario of negative values we saw that many of the high utility item sets which should have been there in the results were missing but since HUIVIV was the only work done in the area of negative values therefore there was an inherent need to improvise upon the proposed algorithm. Henceforth a challenge to improve the existing HUIVIV was confronting us. Later section of our work describes in detail that how it met these challenges.

II. RELATED WORK

High utility item set mining is nothing but an augmentation of frequent pattern mining. Perhaps it can be viewed as a more generalized approach of the later. Business/Fraudulent transaction, retail communities, security, bio-informatics, are some of the all life applications where utility mining can be employed. Sometimes Rare item sets are the result of utility mining. These item sets provide a very useful insight to decision makers. For instance, a supermarket's customers purchase microwave or refrigerator rarely as compared to bread, baking powder, sauce. But profit for the supermarket in case of later transactions is less than the former. Researches that define various interestingness measures of discovered patterns is proposed by Liqiang Geng and H.J. Hamilton in [1]. Their work intended to come up with some parameters that allow reduced time and space costs mining process. In this work utility is defined as one of the interestingness measures of discovered patterns. A foundational approach to mine item set utilities from

Mining of High Value Item Sets with Negative Utility

databases have been proposed in [2] by HONGYAO et al. This paper delves into the theoretical analysis of the resulting utility mining problem and it laid the foundation for various future utility mining algorithms. This work introduces utility bound and support bound property and thereafter it shares a model christened as MEU (mining using expected utility) that mined the high utility item sets. Then a two-phase algorithm [5] by Ying Liu et al was proposed that claimed to efficiently cut down the quantum of candidates and hence a complete set of high utility item sets come out more readily. The proposed two-phase algorithm solves the issue of maintaining downward closure property. Besides several other measures like Transaction Weighted Utilization (TWU) were introduced which form the premise and works like the benchmark for several other algorithms. Then an algorithm HUI NIV was proposed by Chun-Jung Chu et al [6] which mines the high utility item sets from database and also takes into account the item with negative utility values. Their work was authentic in the way that it matches very nearly to the real life shopping basket analytics situation where some items may also appear with negative utility values. Temporal association mining or more precisely calendar based mining by Keshri Verma et al [4] was suggested which can mine the item set which are frequent in a user defined time period T. These methods address the problem to excavate items that are frequent for only some time of the database periodically.

III. METHODOLOGY

This chapter will be delving into minutiae of the approach that we comply to reach to our motive of mining high utility item set. First of all this portion will expound various measures that are omnipresent throughout the text. It will define various terms and explain in that how are they calculated taking an example.

Table 3.1

TID	A	B	C	D	E	Date	TU
T1	5	3	10	2	2	02/2 1/05	159
T2	4	1	8	1	10	09/1 1/06	324
T3	5	6	8	1	2	04/0 9/05	151
T4	6	4	9	2	1	08/2 2/07	174
T5	6	2	12	1	1	11/1 1/02	1256
T6	2	4	15	3	15	05/2 2/02	492
Sum	28	20	62	10	31	—	2546

The above table denotes various transactions of the retail store. Each transaction is explicitly given a unique transaction id as T1, T2 etc. Business store is presumed to have five items like A, B, C, D, and E. Time

stamping of transaction place is on the 7th column and on 8th column transaction utility as defined later is mentioned. The numbers represent the quantum of items that were bought in the corresponding transaction.

Table 3.2

Utility (ITEMS ↓ TIME →)	Quarter 1	Quarter 2	Quarter 3	Quarter 4
A	5	5	8	5
B	2	2	-2	-2
C	10	10	10	100
D	4	8	12	20
E	10	20	20	10

Table 3.2 describes the utility values of the items of the retail store during the four quarters of the year.

A. Definitions

Definition 1. The utility of an item i in the transaction T is denoted as $u(i, T)$ and defined as $p(i) \times q(i, T)$. For example, in Table 3.1, $u(\{A\}, T1, q1) = 5 \times 5 = 25$. [5]

Definition 2. The utility of an item set X in T is denoted as $u(X, T)$ and defined as $\sum(X, T)$. For example, $u(\{AC\}, T3, q2) = u(\{A\}, T3, q2) + u(\{C\}, T3, q2) = 25 + 80 = 105$. [5]

Definition 3. The utility of an item set X in Dis is denoted as $u(X)$ and defined as $\sum(X, T)$. For example, $u(\{AD\}) = u(\{AD\}, T1) + u(\{AD\}, T2) + u(\{AD\}, T3) + u(\{AD\}, T4) + u(\{AD\}, T5) + u(\{AD\}, T6) = 33 + 44 + 33 + 72 + 50 + 34 = 266$. [5]

Definition 4. The transaction utility of a transaction T is denoted as $TU(T)$ and defined as $u(T)$. For example, $TU(T4) = u(\{ABCDE\}, T4) = 48 - 8 + 90 + 24 + 20 = 174$. [5]

Definition 5. The transaction-weighted utilization of an item set X is the sum of the transaction utilities of all the transactions containing X , which is denoted as $TWU(X)$. For example, $TWU(AE) = TU(T1) + TU(T2) + TU(T3) + TU(T4) + TU(T5) + TU(T6) = 159 + 324 + 151 + 174 + 1259 + 492 = 2546$ [5]

If any of the transaction would not have contained A and C simultaneously then the transaction utility of that transaction would not have been taken into account.

B. U-tree data structure

For the sake of simplicity to understand the construction of U-tree we consider on more sampled database and its corresponding utility value.



Table 3.3

TID	A	B	C	D	E	F	Transaction Utility (\$)
t_1	2	0	1	1	0	0	80
t_2	2	1	1	0	0	0	195
t_3	0	0	1	1	10	0	110
t_4	0	1	0	0	15	0	225
t_5	1	0	1	0	0	1	37
t_6	2	0	0	1	10	0	105
t_7	2	0	0	0	8	1	62
t_8	1	1	0	1	2	0	205
t_9	1	0	0	1	10	0	95
t_{10}	1	1	0	0	5	0	185
total	12	4	4	5	60	2	1299

Table 3.4

ITEM	ITEM ID	UTILITY
A	1	10
B	2	150
C	3	25
D	4	35
E	5	5
F	6	2

I have constituted the U-Tree data structure in a way to compactly represent the transaction data required for high utility mining. It consists of an item header table and a U-tree.

C. Item Header Table

The Item Header table contains index, itemid, quantity, TWU, utility, PST where TWU and PST are transaction weighted utilization and pointer to subtree respectively.

D. Construction of Item Header Table

Initially no items except the row headings are there in the table. Thereafter the database of transactions is scanned once. As the transactions are read from the database, the item header table is updated using the following steps.

- For any item which is absent in the item header table but present in transaction, a new column in the table is generated and the entries are initialized, like itemid, utility/unit, its quantity, and the TWU. At this stage mappedid and the pointer remain null.
- For every item whose entry is there in the item header table as well as the transaction, update in the corresponding column entries are done i.e. total quantity of the item is added to existing detail also TU is added to existing entry.
- Items in the item header table are stored in increasing order of their TWU and their mappedid is nothing but their corresponding ranks when placed in ascending order. The item which does not cross the minimum utility is not included in the item header table. The item with the ITEMID 6 i.e. F is eliminated as it was below the minimum utility of 120. The original items that were 3, 4, 2 and 1, 5 are now have a new mappedid

i.e. indexes 1, 2, 3, 4, 5.

E. Construction of U-Tree

Since this work is on temporal mining different tree will be constructed for different time specifications but since the procedure of construction will remain same, the following example will talk only about construction of tree taking only the utility values of only one time period. Now the very first phase of the tree construction which is initialization of tree comes into picture. The process of the

Table 3.5 item header table

INDEX	1	2	3	4	5
ITEMID	3	4	2	1	5
UTILITY	25	35	150	10	5
QUANT.	4	5	4	12	60
TWU	422	595	810	964	987
PST	0	0	0	0	0

initialization of tree is as follows

The tree is initialized with root node of index 1. A path of N-1 nodes with index from 2 to N is created such that any node i is the successor of (i-1). A pointer to the root of the subtree is attached to every item present in the item header table. i-item set is attached to every node and its TWU is initialized as 0. Any node i will represent pattern like i, i-1, i-2, i-3...1. Each pattern also has a pointer attached with the array of total quantity of item in the transaction. This is the procedure of tree initialization. The full tree construction is the next step in our proposed methodology. The database is scanned once again to update the U-tree. Scanning of every transaction is done in increasing order of item's TWU. If a pattern is absent in current tree then node is created and the appropriate pattern and corresponding TWU are recorded. Also array of total quantity is updated. If the pattern is present then only the update of TWU and total quantity is done in the present U-tree. One important thing to be kept in mind while updating the tree is that in the item header table the items which were absent, their contribution to the TWU is not considered and excluded. As the database is scanned once the tree is constructed. The tree corresponding to the above example is shown in above figure.

F. Mining Algorithm

Description of four proposed algorithms to mine the required set of high utility patterns is given in this section. The algorithm is explained using the example dataset/database as described earlier. Item header table's each item serves as commencement point to mine HUI. This is done by first inspecting the TWU of every item in the item header table in its associated subtree.

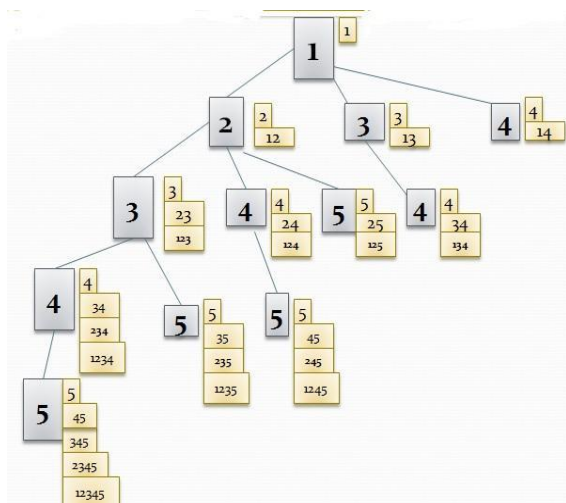


Figure 3.1

Algorithm starts with subtree that has item with index or mapped id 1 as its root, then the corresponding item's actual utility is checked whether it lies above or below the threshold. This results in pattern 1 with the actual utility of 100 which certainly lies below the threshold. Hence this pattern 1 is not considered among the HUI. Now the current node's child subtrees are checked for the items that are left in the item header table starting from rightmost item and up to current item's second right neighbor. At the same time an array to store the intermediate results of TWU while traversing the tree is initialized. This array is termed as `sum_TWU`. Now we start from the rightmost item in the item header table which is 5. All the occurrence of 5 is inspected under 1's subtree and every pattern's TWU is added to the array, this is how the TWU of the pattern 15 is checked. There are 4 occurrences of 5 under 1's subtree but each occurrence apart from 125 is neglected because those patterns are not present in the database itself, i.e. they have the TWU of 0. The TWU of 125 is 110 which is less than minimum utility threshold. And hence its real utility calculation will not be done. Also 5 being the rightmost item, the augmentation to pattern 15 is not required.

occurrence of 4 under 1's subtree is considered and which results in pattern 14, 124, 134, 1234. But since 1234 is absent in the database itself it is not considered. The patterns 14, 134 and 124 occur with the corresponding TWU of 35, 195 and 80 which add up to give a TWU value of 310 which is > threshold and therefore being the high TWU item we'll continue to find the actual utility contribution of 14, 124, 134 which is 35, 45 and 45 respectively which

add up to 125 and less than the threshold value of 150. Now the longer pattern with 14 as the prefix are explored but then there is no pattern present hence the algorithm again iterates to mine the pattern of 13. There is only presence of 13 with TWU of 195 and since the value lies above the utility threshold therefore the actual utility calculation of this pattern 13 is done and it comes up to 175 which is also the high utility item set so it is among the HUI. Again longer patterns like 134 and 135 are to be searched and mined with 13 as the prefix and we see that the pattern 135 is absent and 134 results in the HUI with utility of 195. The original id of pattern 134 is item set 123. Again 134 is further extended to mine for patterns like 1345 but no pattern like this is present in the database therefore the iteration stops here, and since the 3 is second right neighbor of 1 therefore path 12 is not mined and is left for later computation. Moreover since there is a path 13 in the tree therefore this pattern TWU and quantity is attached to the corresponding node in the leftmost branch. This completes the mining of HUI items with subtree having root node 1. Now the subtree of 2 is checked. Patterns like 2 and 12 are derived from the root. It should be noted that from this node pattern 12 can be derived. And now the program will check for the occurrence of rightmost item under not only 2 but also 12. The patterns like 25, 125 and 24 and 124 are checked for high TWU. And if $TWU > \text{threshold}$ then actual utility calculation of the pattern is done. From this point 24 and 25 comes out to be of HUI and they map to original item set 14 and 45 respectively. The extension of 25 is not possible but 24 is extended to check for 245 and 1245. And algorithm checks for both of these item sets and finds 245 to be a HUI with utility of 255. Also it is noted that both 24 and 245 exist in the tree therefore their quantity value and TWU value is added to corresponding node occurrence at the leftmost branch of the tree. Again the algorithm loops for checking the subtree with root 3. In this case the rightmost item's occurrences are checked under the occurrence of root 123 and 23 and 3. The whole algorithm follows the same procedure as described above and find the HUI of 35 and 235. Similarly subtree with root 4 will give 234 and 34 as HUI. And 5 will give 5, 45, 345, 2345 as high utility item sets. After this is done mapping of index to original item is done which gives the result of our proposed approach, i.e. set of High utility item set, they are as follows:



Table 3.6

Item sets	Utility values	Item Sets	Utility Values
B	600	BE	560
E	300	DE	300
AB	490	ABC	195
AD	200	ABD	195
AE	245	ABE	355
BC	175	ADE	255
BD	185	BDE	195
RESULTS		ABDE	205

IV. PERFORMANCE STUDY

In this section the performance of our proposed algorithm is empirically studied. The algorithm is developed in Java and tested on Intel Pentium core i3 processor, 3GB RAM on transaction was 50000. We generate utility value randomly following a lognormal distribution as in [5]. The algorithm took more time in execution when utility threshold were higher as compared to the case when utility threshold are lower and it outperformed many other previous algorithms when utility thresholds are further lowered.

V. CONCLUSION.

In this paper, a novel algorithm that mines the high utility item sets from a transaction temporal database is proposed which mines the complete set of high utility item set over a user defined time period. The algorithm most closely relates to the real world scenario where the item utilities are not constant and keep on changing. This algorithm addresses many problems like maintaining downward closure property and also addressing the problem to consider negative values. This algorithm uses a tree based approach and does not follow candidate generation and test method which efficiently mines the required set of items.

REFERENCES

1. Geng, L., & Hamilton, H. J. (2006). Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)*, 38(3), 9.
2. Yao, H., Hamilton, H. J., & Butz, C. J. (2004, April). A foundational approach to mining itemset utilities from databases. In *Proceedings of the 2004 SIAM International Conference on Data Mining* (pp. 482-486). Society for Industrial and Applied Mathematics.

3. Li, Y., Ning, P., Wang, X. S., & Jajodia, S. (2003). Discovering calendar-based temporal association rules. *Data & Knowledge Engineering*, 44(2), 193-218.
4. Keshari Verma, O.P., Vyas, Efficient calendar based temporal Association Rule. *ACM SIGMOD*, ISSN No. 0163-5808, vol 134, No 3 September 2005.
5. Liu, Y., Liao, W. K., & Choudhary, A. (2005, May). A two-phase algorithm for fast discovery of high utility itemsets. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 689-695). Springer, Berlin, Heidelberg.
6. Chu, C. J., Tseng, V. S., & Liang, T. (2009). An efficient algorithm for mining high utility itemsets with negative item values in large databases. *Applied Mathematics and Computation*, 215(2), 767-778. *Computation Volume 215, issue 2*, 15 September 2009
7. Pillai, J., Soni, S., Vyas, O. P., & Muyebe, M. (2010). A Conceptual Approach to Temporal Rare Item set Utility Mining. *International Journal of Computer Applications*, 1, 65-72.

AUTHORS PROFILE



Mr. Aatif Jamshed is pursuing Ph.D from Uttarakhand Technical University (A State Govt), Dehradun. He has completed his M.Tech in Computer science & Engineering from Jamia Hamdard University. He has experience in real life projects from leading IT company in India with proficiency in Java, J2EE. Mr. Jamshed has about 10 years of Teaching and Industrial experience across multiple institutes like Galgotias, Vidya, Beganto Inc. etc. He has published more than 20 Research papers in reputed International Refereed Journals/ Conferences. He has published 5 Books, 2 out of them are available on Amazon. He has contributed as Session Chair, organizing Committee, Member of Reviewer Committee and Technical Program Committee for multiple International Conferences (IEEE). He is the member of UACEE, IAENG, IACSIT, CSTA etc. His



Dr Bhawna Mallick is working as senior consultant at Maverick Quality Advisory Services Pvt. Ltd Ghaziabad, India. She was Professor and Head of the Department at Galgotia College of engineering and Technology. She is having 20+ years' experience. Degree(s) PhD, M.Tech and B.E (Computer Technology). She has published more than 30 Research papers in reputed International Refereed Journals/ Conferences. She has organized 2 International Conferences and many workshops & Seminars.