HURI – A Novel Algorithm for Mining High Utility Rare Itemsets

Jyothi Pillai[1], O.P. Vyas[2], and Maybin Muyeba[3]

[1] Associate Professor, Bhilai Institute of Technology, Durg-491001, Chhattisgarh, India
jyothi_rpillai@rediffmail.com
[2] Professor, Indian Institute of Information Technology, Allahabad, Uttar Pradesh, India
dropvyas@gmail.com
[3] Senior Lecturer, Deptt. of Computing and Mathematics,
Manchester Metropolitan Univ., U.K.
M.Muyeba@mmu.ac.uk

Abstract. In Data mining field, the primary task is to mine frequent itemsets from a transaction database using Association Rule Mining (ARM). Utility Mining aims to identify itemsets with high utilities by considering profit, quantity, cost or other user preferences. In market basket analysis, high consideration should be given to utility of item in a transaction, since items having low selling frequencies may have high profits. As a result, High Utility Itemset Mining emerged as a revolutionary field in Data Mining. Rare itemsets provide useful information in different decision-making domains. High Utility Rare Itemset Mining, HURI algorithm proposed in [12], generate high utility rare itemsets of users' interest. HURI is a two-phase algorithm, phase 1 generates rare itemsets and phase 2 generates high utility rare itemsets, according to users' interest. In this paper, performance evaluation and complexity analysis of HURI algorithm, based on different parameters have been discussed which indicates the efficiency of HURI.

Keywords: Association Rule Mining, Utility Mining, Rare itemset, High Utility Rare itemset Mining.

1 Introduction

The most important assets of any corporation might be data. Data Mining extracts diamonds of knowledge from historical data and predicts outcomes of future situations, in the form of patterns and associations from large databases. One of the most widely used areas of data mining for retail industry is marketing. 'Market basket analysis' is a marketing method used by many retailers to promote products. Data Mining uses information about products purchased by customers to predict which products they would buy if given special offers [13]. Apriori algorithm, the first ARM algorithm developed by Rakesh Agrawal [10], firstly identifies those itemsets from sales dataset, having frequencies above a specified threshold and then generates association rules. Itemset Mining is done using Association Rule Mining (ARM). Most classical association rule mining algorithms consider utilities of the itemsets to be

N. Meghanathan et al. (Eds.): Advances in Computing & Inform. Technology, AISC 177, pp. 531–540.
springerlink.com

equal [15]. In real life retail marketing or business other utility factors such as quantity, cost, revenue or profit of an item should also be considered. Yao et al defined Utility as a measure of how useful or profitable an itemset is [15]. In many practical situations rare ones are of higher interest (e.g., in medical databases, rare combinations of symptoms might provide useful insights for the physicians)[6]. Rare itemset mining is a challenging topic; the rare combinations of items in the itemset with high utilities provide very useful insights to the user. For example, a sales manager may not be interested in frequent itemsets that do not generate significant profit. In many real-life applications, high-utility itemsets consist of rare items, i.e. itemsets that occur infrequently in the transaction data set but may contribute a large portion of the profit; for eg, customers purchase microwave ovens or LEDs rarely as compared to bread, butter, etc. The former may yield more profit for the supermarket than the latter.

Jyothi et al proposed High Utility Rare Itemset Mining (HURI) algorithm [12], which finds high profitable rare itemsets according to user's perspective in two phases. In first phase, rare itemsets having support value less than the maximum support threshold are generated. Second phase finds high utility rare itemsets having utility value greater than the minimum utility threshold. The novel contribution of HURI is to effectively find rare itemsets, which are of high utility according to users' preferences. In this paper, performance evaluation of HURI based on different values of parameters such as support, utility, etc. have been described. The rest of paper is organized as follows. In section 2, we discuss some related works: section 3 presents the HURI algorithm. Section 4 performs the evaluation of HURI and section 5 presents conclusion and future work.

2 Related Work

The basic bottleneck in association rule mining is the rare itemset problem. In many applications, some items appear more frequently in the data, while others rarely appear. If frequencies of items vary, two problems may be encountered – (1) If minsup is set too high, then rules of rare items will not be found (2) To find rules that involve both frequent and rare items, minsup has to be set very low, where minsup is the minimum support of an item. This may cause combinatorial explosion in the number of itemsets.

Utility mining is now an important association rule-mining paradigm. Yao et al focuses on the measures used for utility-based itemset mining [8]. A unified framework is proposed for incorporating utility based measures into the data mining process via a unified utility function. Ying et al proposed a Two-Phase algorithm [14] that discovers high utility itemsets highly efficiently. Rare itemsets provide very useful information in real-life applications such as security, business strategies, biology, medicine and super market shelf management. For example [5] shows that normal behavior is very frequent whereas abnormal or suspicious behavior is less frequent. For example, from a marketing strategy perspective, it is important to identify product combinations that have a significant impact on company's bottom line, having highest revenue generating power [7].

S. Shankar et al presents a novel algorithm Fast Utility Mining (FUM) in [4], which finds all high utility itemsets within the given utility constraint threshold. The

authors also suggest a novel method of generating different types of itemsets such as High Utility and High Frequency itemsets (HUHF), High Utility and Low Frequency itemsets (HULF), Low Utility and High Frequency itemsets (LUHF) and Low Utility and Low Frequency itemsets (LULF) using a combination of FUM and Fast Utility Frequent mining (FUFM) algorithms.

A different approach known as Apriori Inverse [9], involves use of maximum support measure to generate candidate itemsets, i.e., only items with a lower support than a given threshold are considered. Then rules are generated by an Apriori approach. In [6], L. Szathmary et al presented a novel algorithm for computing all rare itemsets by splitting the rare itemset mining task into two algorithms , (i) a naïve one that relies on an Apriori-style enumeration, Apriori-rare and (ii) an optimized method that limits the exploration to frequent generators only. Apriori-rare generates a set of all minimal rare generators, also called MRM. To retrieve all rare itemsets from minimal rare itemset, a prototype algorithm called "A Rare Itemset Miner Algorithm (ARIMA)" was proposed. ARIMA generates the set of all rare itemsets, splits into two sets: the set of rare itemsets having a zero support and the set of rare itemsets with non-zero support.

A totally different approach to all these algorithms presented demands developing new algorithms to tackle new challenges. Apriori-inverse [9], is a more intricate variation of the traditional Apriori algorithm. The main idea is that given a user-specified maximum support threshold, MaxSup and a derived MinAbsSup value, a rule X is rare if Sup(X)<MaxSup and Sup(X)>MinAbsSup. M. Adda et al [5] proposed a framework to represent different categories of interesting patterns and then instantiate it to the specific case of rare patterns. A generic framework called AfRIM for Apriori Rare itemset was presented to mine patterns based on the Apriori approach. The generalized Apriori framework was instantiated to mine rare itemsets. In [11], Lan et al proposed rare-utility itemset, by considering profit and quantity of each item in a transaction and an algorithm TP-RUIMD (Two-Phase Algorithm for Mining Rare Utility Itemsets in Multiple Databases), to find rare-utility itemsets in a multi-database environment. HURI algorithm proposed in [12] generates high-utility rare-itemsets, by considering the utility of itemsets other than the frequency of items in the transaction set. The utility of items is decided by considering factors such as profit, sale, temporal aspects, etc. of items. By using HURI, high-utility rare itemsets can be generated based on minimum threshold values and user preferences.

3 HURI Algorithm

3.1 Definitions

In this section, HURI algorithm is presented with formal definitions and examples to illustrate the approach.

Utility Mining. Let D1 (Table 1) be a given transaction database with a set of transactions {T1,T2,…,Tn} and a set of quantities of items I={i1, i2, i3,… , im} where each item i ε I has a set of utilities defined as U={u1, u2, u3,… , uk} (Table II). For example in transaction T29, the quantities of items A001, B002, C003, D004, E005… are 1,3, 0,1,1… respectively. The utility of an itemset X, i.e., u(X), is the sum of

utilities of itemset X in all transactions containing X. An itemset X is called a high utility itemset if and only if u(X) >= min_utility, where min_utility is a user-defined minimum utility threshold [8]. Identification of the itemsets with high utilities is called as Utility Mining [5].

Utility Table. A utility Table UT (Table 2) contains items and their utility values where each item i has some utility value uj in U={u1, u2, u3,… , uk } for some k > 0. For example utility of item A001 from D1 is u(A001) = 4 in (Table 2).

Internal Utility. Internal utility value of item ip in a transaction T_q, denoted $o(i_p, T_q)$ is the value of an item ip in a transaction T_q (Table 2), reflecting the occurrence of item. For eg., internal utility of item A0001 in transaction T1 is o(A001, T1) = 1, while internal utility of item A0001 in D1 is o(A001, D1) = 21(Table 1).

External Utility. External utility value of an item is a numerical value s(ip) associated with an item ip such that $s(i_p) = u(i_p)$, where u is a utility function, a function relating specific values in a domain according to user preferences (Table 2). From Table 3, external utility of item A0001 in D1 is s(A0001) = u(A0001) = 4.

Item Utility. The utility of an item i_p in a transaction T_q, denoted $U(i_p, T_q)$ is product of $o(i_p, T_q)$ and $s(i_p)$, i.e. internal and external utility respectively . For eg., total utility of item A0001 in D1 is U(A001) = s(A001)*o(A001)=4*21=84.

Transaction Utility. The transaction utility value of a transaction, denoted as U(Tq) is the sum of utility values of all items in a transaction Tq (Table I, Table II). From Table I and Table 3, the transaction utility of the transaction T1 from D1, U(T1) = U(A001)+U(B002)+U(C003)+ … + U(T020)=39.

Rare Itemset. Mining Rare itemsets are itemsets that occur infrequently in the transaction data set.

The HURI algorithm(Fig. 1) can be best understood by transactional dataset D1 (Table 1) and Item utility table (Table 2). Given a user-specified maximum support threshold maxsup, and a generated minabssup value, we are interested in a rule X if sup(X) < maxsup and sup(X) > minabssup. Rare rules are generated in the same manner as in apriori rule generation. Apriori-Inverse produces rare rules that do not consider any itemsets above maxsup. In Apriori inverse algorithm, rare itemsets are itemsets which fall below maxsup value. In HURI Algorithm (Fig. 1), high utility rare itemsets are generated in two phases:-

(i)In first phase, rare itemsets are generated by considering those itemsets which have support value less than the maximum support threshold (using apriori-inverse concept). Table III lists rare itemsets generated from dataset D1(Table 1).

(ii)In second phase, high utility rare itemsets having utility value greater than the minimum utility threshold are generated (Table 4).

By applying HURI algorithm [12] on Transaction dataset described in Table 1 and by setting the value of maximum support threshold to 40%, the rare itemsets generated are listed in Table 3. Then HURI algorithm generates high utility rare itemsets which fall below a maximum support value but above a user provided high utility threshold. For example by setting high utility threshold as 45, the high utility rare itemsets

generated from D1 are listed in Table 4. Both HURI and apriori inverse algorithm considers utility values of all items in transaction set in addition to frequency. But apriori inverse produces only rare itemsets whereas HURI produces high utility rare itemsets according to users' interest.

Table 1. Transaction Dataset D

T_ID	A 001	B 002	C 003	D 004	E 005	F 006	G 007	H 008	I 009	J 010	K 011	L 012	M 013	N 014	O 015	P 016	Q 017	R 018	S 019	T 020
T1	1	2	2	0	0	1	1	0	2	0	1	0	5	0	0	1	4	0	1	0
T2	1	0	1	1	1	0	0	0	0	3	1	1	0	4	0	1	0	3	0	1
T3	1	3	2	0	0	0	0	0	2	0	1	0	3	2	1	0	0	2	0	1
T4	0	0	0	0	1	3	0	0	0	4	1	1	0	1	0	1	0	1	1	0
T5	0	1	0	0	1	0	1	0	1	0	0	0	1	0	1	0	1	1	0	0
T6	0	2	0	0	0	0	0	1	0	0	0	0	0	1	0	1	3	0	1	0
T7	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	1	1	5	1	1
T8	1	0	1	1	1	0	0	0	3	0	1	0	4	4	0	1	0	0	0	1
T9	0	0	1	0	2	4	0	2	0	0	0	1	0	0	1	1	0	4	1	1
T10	2	3	1	1	1	0	0	0	5	0	0	1	6	2	1	1	6	0	0	0
T11	1	1	0	0	0	1	0	0	0	3	1	0	0	0	1	0	0	3	0	1
T12	1	0	1	0	1	0	1	1	1	1	0	0	1	5	1	0	0	0	0	1
T13	0	2	0	1	0	0	0	1	3	1	1	1	0	0	1	1	1	1	1	0
T14	0	0	1	0	2	3	1	0	1	5	0	0	3	2	0	0	5	0	0	1
T15	1	1	1	0	1	0	0	0	1	1	1	1	0	0	1	1	1	2	0	1
T16	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0
T17	0	0	0	0	0	0	0	0	2	1	1	1	0	4	0	1	0	2	0	1
T18	1	3	0	0	1	4	0	0	0	0	0	0	5	0	0	1	0	0	1	0
T19	0	0	0	1	2	0	0	1	0	0	0	1	0	2	0	1	1	1	0	1
T20	0	0	2	0	0	0	0	1	2	0	0	0	1	0	0	1	5	0	1	0
T21	2	0	1	0	0	3	0	1	0	2	0	1	1	1	0	0	0	3	0	1
T22	0	0	2	0	0	0	0	1	0	0	0	0	0	1	1	0	0	1	0	0
T23	0	0	0	0	2	1	1	0	1	0	0	1	1	0	0	0	1	2	1	1
T24	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	4	1	0	0
T25	2	2	1	1	1	0	1	0	1	1	2	1	4	1	0	0	1	1	0	0
T26	0	0	0	2	1	0	0	0	0	0	0	0	1	1	1	0	1	0	1	0
T27	1	0	0	0	0	0	1	0	1	0	0	0	1	5	0	0	2	5	0	1
T28	0	0	0	0	0	4	0	1	2	0	0	0	2	0	1	0	1	0	1	1
T29	1	3	0	1	1	2	1	0	1	2	1	0	0	1	0	0	2	2	0	0
T30	0	0	2	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	1
T31	2	1	0	1	1	0	1	0	1	0	0	0	2	1	0	1	0	2	0	1
T32	0	1	0	0	1	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0
T33	1	1	0	1	0	1	1	0	1	1	1	1	5	2	0	0	1	1	0	0
T34	0	1	0	0	0	0	0	1	1	1	0	0	1	0	0	1	0	0	0	0
T35	0	1	1	1	2	0	0	0	0	1	1	1	1	0	0	1	1	1	0	0

Table 2. Item Utility Table

Items	External Utility	Internal Utility	Total Utility
A 001	4	21	84
B 002	1	28	28
C 003	3	21	63
D 004	2	12	24
E 005	7	23	161
F 006	5	27	135
G 007	6	10	60
H 008	1	13	13
I 009	1	34	34
J 010	4	27	108
K 011	3	15	45
L 012	1	14	14
M 013	1	50	50
N 014	2	40	80
O 015	3	14	42
P 016	1	18	18
Q 017	1	42	42
R 018	1	44	44
S 019	1	11	11
T 020	0	17	0

Table 3. Rare Itemset Table

Rare Itemsets	List of rare itemsets	Itemset Utility
1-itemset	{D004}	24
	{G007}	60
	{H008}	13
	{S019}	11
2-itemset	{D004,G007}	84
	{D0004,H008}	37
	{D004,S019}	35
	{G007,H008}	73
	{G007,S 019}	71
	{H008,S019}	24
3-itemset	{D004,G007,H008}	97
	{D004,G007,S019}	95
	{G0007,H0008,S0019}	84
	{D004,H008,S019}	48
4-itemset	{D004,G007, H0008,S019}	108

Table 4. High Utility Rare Itemset Table

High Utility Rare Itemsets	List of high utility rare Itemsets	Utility
1-itemset	{G007}	60
2-itemset	{D004,G007}	84
	{G007,H008}	73
	{G007,S 019}	71
3-itemset	{D004,G007,H008}	97
	{D004,G007,S019}	95
	{S007,H008,S019}	84
4-itemset	{D004,G007, H008,S019}	108

3.2 Algorithm HURI

Description: Finding High Utility Rare Itemsets of users interest
Ck: Candidate itemset of size k **Lk: Rare itemset of size k**
For each transaction t in database
begin
increment support for each item i present in t
End
L1= {Rare 1-itemset with support less than user provided max_sup}
for(k= 1; Lk!=Ø; k++)
begin
C k+1= candidates generated from Lk;
//loop to calculate total utility of each item
For each transaction t in database

```
begin
Calculate total quantity of each item i in t
Find total utility for item i using following formula:-
      u(i,t) = quantity[i] * user_provided_utility for i
End
//loop to find rare itemsets and their utility
For each transaction t in database
begin
increment the count of all candidates in Ck+1 that are contained in t
      Lk+1 = candidates in Ck+1 less than min_support
    Add Lk+1 to the Itemset_Utility Table by calculating rare itemset utility using formula:
          Utility(R,t) = Σfor each individual item i in R (u(i,t));
End
//loop to find high utility rare itemset
For each itemset iset in rare itemset Table R
begin
If (Utility(iset) > user_provided_threshold_for_high_utility_rare_itemset)
then iset is a rare_itemset that is of user interest i.e.high_utility_rare_itemset
else iset is a rare itemset but is not of user interest
End
Return high_utility_rare_itemsets
END
```

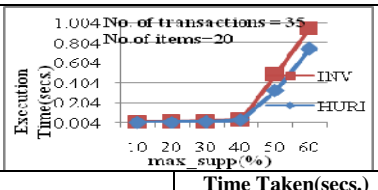
Fig. 1. Pseudo Code for HURI

4 Performance Evaluation of HURI

4.1 Comparative Analysis

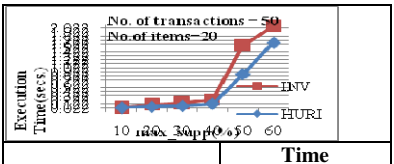
Apriori Inverse and HURI algorithms were compared using a transactional datasets of different sizes. Java as front end and MS Access as backend tool were used to evaluate HURI algorithm. We study the impact of different values of minimum support, number of transactions, number of items, on processing time etc. for comparing Apriori Inverse and HURI. The different comparative parameters are:-

(i) Number of rare itemsets generated.
(ii) Total execution time taken for generation of rare itemsets.



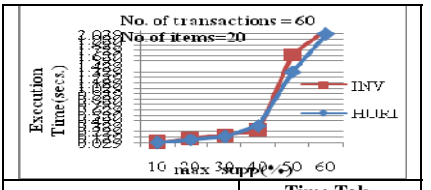
max_supp	Time Taken(secs.)	
	HURI	INV
10	0.004	0.006
20	0.009	0.011
30	0.014	0.015
40	0.031	0.031
50	0.325	0.484
60	0.741	0.954

Fig. 2. Execution Time on Database D1



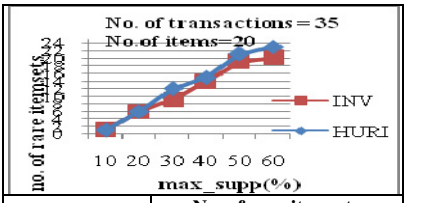
max_supp	Time Taken(secs.)	
	HURI	INV
10	0.022	0.024
20	0.053	0.099
30	0.076	0.143
40	0.123	0.234
50	0.867	1.597
60	1.65	2.053

Fig. 3. ExecutionTime on Database D2



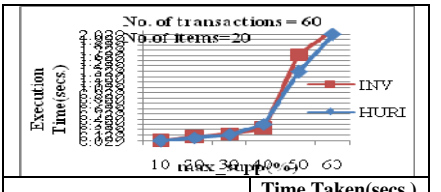
max_supp	Time Taken(secs.)	
	HURI	INV
10	0.029	0.031
20	0.089	0.099
30	0.145	0.168
40	0.342	0.255
50	1.324	1.654
60	2.031	2.112

Fig. 4. Execution Time on Database D3



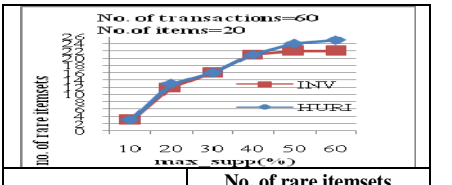
max_supp	No. of rare itemsets	
	HURI	INV
10	1	1
20	6	6
30	12	9
40	15	14
50	21	19
60	23	20

Fig. 5. Number of rare itemsets from D1



max_supp	Time Taken(secs.)	
	HURI	INV
10	0.029	0.031
20	0.089	0.099
30	0.145	0.168
40	0.342	0.255
50	1.324	1.654
60	2.031	2.112

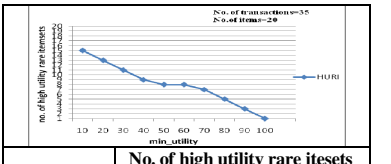
Fig. 6. Number of rare itemsets from D2



max_supp	No. of rare itemsets	
	HURI	INV
10	3	3
20	13	12
30	16	16
40	21	21
50	24	22
60	25	22

Fig. 7. Number of rare itemsets from D3

In item utility Table (Table 2), each item is assigned an external utility and internal utility is calculated from database D1. We considered three transaction sets, D1 (number of transactions is 50, number of items is 20), D2 (number of transactions is 50, number of items is 20) and D3 (number of transactions is 60, number of items is 20). Fig.2, Fig.3 and Fig.4 shows execution time of algorithms to generate rare itemsets from datasets D1, D2 and D3 respectively, by varying the support threshold. Fig.5, Fig.6 and Fig.7 shows number of rare itemsets generated from datasets D1, D2 and D3. The results show that HURI algorithm yields more rare itemsets with less execution time as compared to Apriori inverse. Apriori inverse produces only



Min.utility	No. of high utility rare itesets
	HURI
10	15
20	13
30	11
40	9
50	8
60	8
70	7
80	5
90	3
100	1
110	0

Fig. 8. Effect of utility threshold on high utility rare itemsets from dataset D1

rare itemsets whereas HURI produces high utility rare itemsets according to users' interest.

HURI was evaluated on dataset D1 under varied minimum utility thresholds, for generation of high utility rare itemsets (Fig.8). The experimental result shows that number of high utility rare itemsets decreases as minimum utility threshold increases, as desired, which indicates effectiveness of HURI.

4.2 Computational Complexity

The computational complexity of HURI can be affected by following factors –

4.2.1 Support Threshold

Increasing the support threshold results in more itemsets declared as rare. This has an adverse effect on the computational complexity of the algorithm because more candidate itemsets must be generated and counted as shown in Figure VIII, Figure IX and Figure I0. The maximum size of rare itemsets also tends to increase with higher support thresholds.

4.2.2 Number of Items (Dimensionality)

As number of items increases, more space will be needed to store support counts of items. If number of rare items also grows with dimensionality of the data, the computation and I/O costs will increase because of the larger number of candidate itemsets generated by the algorithm.

4.2.3 Number of Transactions

Since HURI algorithm makes repeated passes over the dataset, its run time increases with a large number of transactions.

4.2.4 Average Transaction Width

For dense datasets, the average transaction width can be very large. The maximum size of rare itemsets tends to increase as the average transaction width increases. As a result, more candidate itemsets must be examined during candidate generation and support counting.

A detailed analysis of time complexity for HURI algorithm is presented –

4.2.5 Generation of Rare 1-Itemsets

For each transaction, support count of every item presented in transaction is updated. If n is total number of transactions and m is the average transaction width, the time required for this operation is O(n*m).

4.2.6 Candidate Generation

To generate candidate k-itemsets, pairs of rare (k-1)-itemsets are merged. In the best-case scenario, every merging step produces a viable k-itemset. In the worst-case scenario, the algorithm must merge every pair o0f rare (k-1)-itemsets found in the previous iteration.

Therefore the overall cost of merging rare itemsets is $O(\sum_{k=2}^{m}(k-2)\,|C_k|)$

4.2.7 Support Counting

Each transaction of length x produces tCk itemsets of size k. If m is the maximum transaction width and αk is cost for updating support count of candidate k-itemset then cost for support counting is $O(n \cdot \sum_k ({}^mC_k)\,\alpha_k)$

4.2.8 High Utility Rare Itemsets

If m is maximum transaction width then time required to generate high utility rare itemsets is O(m2), which is negligible as compared to time taken for other operations for finding rare itemsets. Hence time taken in calculating Utilities like Internal utility, total item utility, Itemset Utility, Dataset Utility, etc. does not affect time taken by HURI algorithm to generate high utility rare itemsets. This proves the effectiveness and efficiency of HURI.

5 Conclusions and Future Work

Our paper proposes an innovative algorithm, HURI, for making business data mining more realistic and usable to business analyst. Data mining can identify products that are often purchased together, which can help product bundles that are more likely to be successful [13]. Frequency of item is not sufficient to answer a product combination i.e. whether it is highly profitable or whether it has a strong impact. Marketers are interested in knowing how various marketing programs affect the discovery of subtle relationships. The novelty of HURI is the ability to discover high utility rare itemsets. HURI algorithm may have practical meaning to real-world marketing strategies.The high utility rare itemsets are generated according to the users' preference.

HURI may be more beneficial on application to transactional data set. The high utility rare itemsets are generated based on transactional database information and external information about utilities. HURI uses the concept of Apriori inverse which produces only rare itemsets having support less than maximum support value where as HURI can produce high utility rare itemsets based on users' interest, support utility thresholds. The outcome of HURI would enable the top management or business analyst in crucial decision-making such as providing credit facility, finalizing discount policy, analyzing consumers' buying behaviour, organizing shelf space, quality improvement in supermarket scenario. Also the time complexity of HURI is almost the same as other algorithms. The future work includes the incorporation of temporal and fuzzy concept in HURI.

References

[1] Pillai, J., Vyas, O.P.: User centric approach to itemset utility mining in Market Basket Analysis. IJCSE 3(1), 393–400 (2011) ISSN : 0975-3397

[2] Pillai, J., Vyas, O.P.: Overview of Itemset Utility Mining and its Applications. IJCA (0975–8887) 5(1), 9–13 (2010)

[3] Pillai, J., Vyas, O.P., Soni, S., Muyeba, M.: A Conceptual Approach to Temporal Weighted Itemset Utility Mining. IJCA (0975-8887) 1(28) (2010)

[4] Shankar, S., Purusothoman, T.P., Jayanthi, S., Babu, N.: A Fast Algorithm for Mining High Utility Itemsets. In: Proceedings of IEEE IACC 2009, India, pp. 1459–1464 (2009)

[5] Adda, M., Wu, L., Feng, Y.: Rare Itemset Mining. In: Sixth International Conference on Machine Learning and Applications, pp. 73–80 (2007)

[6] Szathmary, L., Napoli, A., Valtchev, P.: Towards Rare Itemset Mining. In: Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence, vol. 1, pp. 305–312 (2007) ISSN: 1082-3409 , 0-7695-3015-X

[7] Erwin, A., Gopalan, R.P., Achuthan, N.R.: A Bottom-up Projection based Algorithm for mining high utility itemsets. In: Proceedings of AIDM 2007, Australia, Conferences in Research and Practice in Information Technology, CRPIT, vol. 84 (2007)

[8] Yao, H., Hamilton, H., Geng, L.: A Unified Framework for Utilty-Based Measures for Mining Itemsets. In: Proceedings of UBDM, pp. 28–37 (2006)

[9] Sun, X., Orlowska, M.E., Li, X.: Finding Temporal Features of Event-Oriented Patterns. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 778–784. Springer, Heidelberg (2005)

[10] Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large databases. In: Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, pp. 207–216 (1993)

[11] Lan, G.C., Hong, T.P., Tseng, V.S.: A Novel Algorithm for Mining Rare-Utility Itemsets in a Multi-Database Environment. In: 26th Workshop on Combinatorial Mathematics and Computation Theory, pp. 293–302

[12] Pillai, J., Vyas, O.P.: High Utility Rare Itemset Mining (HURI): An approach for extracting high-utility rare itemsets. Journal on Future Engineering and Technology 7(1) (October 1, 2011)

[13] Data Mining: A Competitive Tool in the Bankingand Retail Industries, http://icai.org/resource_file/9935588-594.pdf

[14] Liu, Y., Liao, W.-k., Choudhary, A.K.: A Two-Phase Algorithm for Fast Discovery of High Utility Itemsets. In: Ho, T.-B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 689–695. Springer, Heidelberg (2005)