IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 12, December 2016

A Better Approach for Mining High Utility Itemset from Transactional Databases

Mrs. Madhuri Zawar[1], Ashwini Barakare[2]

Department of Computer Engineering, Godavari College of Engineering, Jalgaon, India[1,2]

Abstract: Data Mining can be defined as an activity that extracts some new nontrivial information contained in large databases. Traditional data mining techniques have focused largely on detecting the statistical correlations between the items that are more frequent in the transaction databases. Also termed as frequent itemset mining, these techniques were based on the rationale that itemsets which appear more frequently must be of more importance to the user from the business perspective .In this thesis we throw light upon an emerging area called Utility Mining which not only considers the frequency of the itemsets but also considers the utility associated with the itemsets. The term utility refers to the importance or the usefulness of the appearance of the itemset in transactions quantified in terms like profit, sales or any other user preferences. In High Utility Itemset Mining the objective is to identify itemsets that have utility values above a given utility threshold. In existing system some high utility itemset mining algorithms such as Two-Phase, UP-Growth have been proposed. But there is problem like it requires more execution time and it uses more memory. The new method is memory efficient technique for mining high utility itemsets from transactional databases. This technique requires less memory space and execution time than existing algorithms.

Keywords: Data Mining, Frequent Itemset, High Utility Itemset, Utility Mining

I. INTRODUCTION

The information technology industry has huge amount of data. Just data is none of use until any knowledge can be gathered from it. Therefore any process must be applied on that data which can produced necessary knowledge from it .To acquire this knowledge applied processes have various sub processes such as Data Cleaning, Data Integration, Data Transformation, Data Mining, Pattern Evaluation and Data Presentation

A. *Data Mining*

Data mining is the important part of KDD. Data mining generally involves four classes of task; classification, clustering, regression, and association rule learning. Data Mining means to extracting or mining knowledge from large databases. The Primary goal of data mining is to searching of interesting patterns in your data. Data mining exercises utilizes blend of strategies from database advancements, statistics, and machine learning.

Over the last two decades data mining has emerged as a significant research area. This is primary due to the inter - disciplinary nature of the subject and the diverse range of application domains in which data mining based products and techniques are being employed. This incorporates medicine, education, Banking and finance, Healthcare and insurance, retail and marketing research. Data mining has been considerably used in the analysis of customer transactions in retail research where it is termed as market basket analysis. Searching of interesting patterns hidden in database is an important role in data mining tasks, such as frequent itemset mining, utility mining.

B. *Frequent Itemset Mining*

An itemset can be defined as a non-empty set of items. An itemset with k diverse items is termed as a k-itemset. For e.g. {bread, butter, milk} may denote a 3-itemset in a supermarket transaction .The notion of frequent itemsets was introduced by R. Agrawal. Frequent itemsets are the itemsets that appear frequently in the transactions. The goal of frequent itemset mining is to identify all the itemsets in a transaction dataset. Frequent itemset mining [1][2][3]plays an essential role in the theory and practice of many important data mining tasks, such as mining association rules, long patterns. The criterion of being frequent is expressed in terms of support value of the itemsets. The Support value of an itemset is the percentage of transactions that contain the itemset. Frequent pattern mining is beneficial for association rule mining.

C. *Association Rule Mining*

Association Rule Mining is well known technique for finding co-occurrences, correlations, frequent patterns, associations among set of items in the transaction database. The discovery of interesting correlation relationships among huge amount of business transaction records can help in many business decision making process, such as catalog design, customer shopping behavior analysis etc. An association rule is an expression in the form of $X \Rightarrow Y$, where X and Y are set of items called itemsets. It suggests that if a customer buys X, then he or she also buys Y. Two measures which reflect certainty of discovered association rules are support and confidence.

- Support is the percent of the transactions that contain X U Y (i.e. both X and Y) to the total number of transactions in database.
- Confidence is the percent of the transactions that contain X U Y to the total number of transactions that contain X.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
Vol. 5, Issue 12, December 2016

As an example, the information that customers who buys computers also tend to buy antivirus-software at the same time is represented in Association Rule below:

Association rules are considered useful if they satisfy both a type equation here minimum support threshold and a minimum confidence threshold that can be set by users or domain consultants.

D. *Why Utility Mining?*

The frequent itemsets identified by ARM does not reflect the impact of any other factor except frequency of the presence or absence of an item. Frequent itemsets may only contribute a small portion of the overall profit, whereas non-frequent itemsets may contribute a large portion of the profit. In reality, a retail business may be interested in identifying its most valuable customers (customers who contribute a major fraction of the profits to the company). These are the customers, who may buy full priced items, high margin items, which may be absent from a large number of transactions because most customers do not buy these items. In a traditional frequency oriented ARM, these transactions representing highly profitable customers may be left out.

Example:

Support Threshold =10%

In given example, {milk, bread} may be a frequent itemset with support 40%, contributing 4% of the total profit, and

Items	Frequent or not	Support	Profit
{Milk, bread}	Yes	40%	4%
{birthday cake, birthday card}	No	8%	8%

the corresponding consumers is Group A, whereas {birthday cake, birthday card} may be a non-frequent itemset with support 8% (assume support threshold is 10%), contributing 8% of the total profit, and the corresponding consumers is Group B. The marketing professionals must be more interested in promoting the sale of {birthday cake, birthday card} by designing promotion campaigns or coupons tailored for Group B (valuable customers), although this itemset is missed by ARM.

Frequency is not sufficient to answer questions, such as whether an itemset is highly profitable, or whether an itemset has a strong impact. The practical usefulness of the frequent itemset mining is limited by the significance of discovered itemset. So during mining process we should not be prejudiced to identify either item is frequent or not but our aim should be identify itemsets which are more utilizable to us. This leads the inception of a new approach in data mining is based on concept of itemset utility called utility mining. Hence frequency can not be only measure to detect interesting patterens.

E. *Utility Mining*

Utility mining [12] is likely to be useful in a wide range of practical applications. To address the limitation of association rule mining, utility based mining model was defined, which allows a user to conveniently express his or her perspectives concerning the usefulness of itemsets as utility and then find itemsets with high utility values higher than given threshold. In utility based mining the term utility refers to the quantitative representation of user preference i.e. the utility value of an itemset is the measurement of the importance of that itemset in the users perspective. For e.g. if a sales analyst involved in some retail research needs to find out which itemsets in the stores earn the maximum sales revenue for the stores he or she will define the utility of any itemset as the monetary profit that the store earns by selling each unit of that itemset. Formally an itemset S is useful to a user if it satisfies a utility constraint i.e. any constraint in the form u(S) >= minutil, where u(S) is the utility value of the itemset and minutil is a utility threshold defined by the user.

We start with the definition of set of terms that leads to the formal definition of utility mining problem. Consider a simple transaction database and its utility table.

TABLE 1
TRANSACTION TABLE.

ITEM / TID	A	B	C	D	E
T1	0	0	18	0	1
T2	0	6	0	1	1
T3	2	0	1	0	1
T4	1	0	0	1	1
T5	0	0	4	0	2
T6	1	1	0	0	0
T7	0	10	0	1	1
T8	3	0	25	3	1
T9	1	1	0	0	0
T10	0	6	2	0	2

Each row is transaction. The columns represent the number of items in a particular transaction. TID is the transaction identification number.

ITEM	PROFIT ($) (per unit)
A	3
B	10
C	1
D	6
E	5

TABLE 2
THE UTILITY TABLE

Each Transaction is containing diferent items. Here Zero represents that particular transaction does not contain respective item. The number represents quantity of item. For eg. Quantity of C in transaction T1 is 18.

The right column displays the profit of each item per unit in dollars. Profit can be any other measure like time, selling profit.

- I = {i_1, …, i_m} is a set of items.
- D = {T1, T2, …, Tn} be a transaction database where each transaction and each transaction has unique identifier q .
- o(i_p , T_q),local transaction utility value ,represent the quantity of item in transaction Tq. For Example, o(A,T8) = 3 in Table 1.
- s(i_p), external utility, is the value associated with item in the Utility Table. This value reflects the importance of item, which is independent of transactions. For Example, the external utility of item A, s(A) = 3 in Table 2.
- u(i_p ,T_q) is utility for item in transaction , and defined as o( ) × s( ). For Example, u(A,T8) = 3×3 = 9
- u(X), utility of an itemset X, is defined as $\sum_{T_q \in D \wedge X \leq T_q}$ u(X,T_q) . For Example, u(A) = u(A,T3) + u(A,T4) + u(A,T6) + u(A,T8) + u(A,T9) = 6 + 3 + 3 + 9 + 3 = 24.
- An itemset is called a high utility itemset if its utility is no less than a user-specified minimum utility threshold which is denoted as min-util. Otherwise, it is called a low-utility itemset. For Example, u({A, D, E}) = u({A, D, E}, T4) + u({A, D, E}, T8) = 14 + 32 = 46. If min-util = 120, {A, D, E} is a low utility itemset.
- The transaction utility of transaction Tq, denoted as TU(Tq), is the sum of the utilities of all the items in Tq :

$$Tu(Tq)= \sum_{ip \in Tq} u(ip, Tq)$$

For Example : TU(T1) = u(A,T1) + u(B,T1) + u(C,T1) + u(D,T1) + u(E,T1)
= 0 + 0 + 18 + 0 + 5 = 23

- The transaction-weighted utilization of an itemset X, denoted as TWU (X), is the sum of the transaction utilities of all the transactions containing X:

For Example, in Table 1, TWU(AD) = TU(T4) + TU(T8)
= 14 + 57 = 71.

For a given itemset X, X is a high transaction-weighted utilization itemset if TWU(X)> =min-util. For Example,
TWU(B) = TU(T2) + TU(T6) +TU(T7) + TU(T9) + TU(T10) = 71 + 13 + 111 + 13 + 72 = 280
TWU(B) >= 120 , Therefore B is high transaction weighted utilization itemset.

II. LETARATURE REVIEW

In this section we present a brief overview of the various algorithms, concepts and approaches that have been defined in various research publications.

Frequent itemset are the itemsets that occur frequently in the transaction database. The objective of frequent itemset mining is to identify all frequent itemset in a transaction database. A several frequent itemset mining algorithms have been developed with different mining efficiencies. Some of the well- known algorithms are Apriori and FP-Growth.

Apriori [1], [2], [4] is a classic algorithm for frequent itemset mining and association rule learning over transactional databases. There are two processes to finds out all the frequent itemsets from the database in Apriori algorithm. First the candidate itemsets are generated, and then the database is scanned to check the actual support count of the corresponding itemsets. During the first scanning of the database the support count of each item is calculated and the frequent -1 itemsets are generated by pruning those itemsets whose supports are below the predefined threshold. In each pass only those candidate itemsets that include the same specified number of items are generated and checked. In Second step generating association rule from frequent item set using support confidence model. Apriori Algorithm generates lot of candidate item sets and scans database every time. When a new transaction is added to the database then it should rescan the entire database again.

To overcome the problems of apriori algorithms the new method was proposed. This method is used a novel frequent pattern tree (FP-tree) structure, which is an extended prefix-tree structure for storing compressed information about frequent patterns, and develop an efficient FP-tree-based mining method, FP-Growth [4], [5] for mining the complete set of frequent patterns by pattern fragment growth. FP-Growth achieves a better performance than apriori based approach since it find frequent itemset without generating any candidate itemset and its scans database just twice. FP-growth is not able to find high utility itemsets. In Frequent itemset mining importance of item to user is not considered.

Two-Phase [7], [8] algorithm discovers high utility itemsets and uses the transaction-weighted downward closure property to maintain downward closure property in utility mining. The Two-Phase algorithms works in two phases:

A. *Phase I: Discover High Transaction Weighted Utility Itemset List Is Generated As Follows:*

- Transaction Utility: (TU) the transaction utility of an item is the sum of the utilities of all items in that transaction.
- Transaction Weighted Utility (TWU) of an item set: The weighted transaction utility of an item set is obtained by performing the addition of the transaction utility of all transactions containing that item set.
- High Transaction Weighted Utilization Itemset(HTWUIs): Only those item sets are included in the high transaction weighted utilization itemset list whose transaction weighted utility is more than the minimum utility threshold.

B. *Phase II: In This Phase Only One Database Scan Is Performed To Filter The High Utility Itemset From High B.Transaction Weighted Utilization Itemset Identified In Phase I.*

Although Two-Phase algorithm can reduce the search space by using transaction weighted downword closure property, it still generates too many candidates and requires multiple scans of database.

UP-Growth [9], [10] is one of the efficient algorithms to generate high utility itemsets depending on construction of a global UP-Tree. The tree maintains the information about the itemsets and their utilities. Each node of a tree consists of item name, utility value and support count. UP-Growth Algorithm consists of three steps:

1) Construction of Global UP-Tree:

The global UP-Tree is constructed by using two strategies: DGU (Discarding Global Unpromising Item) and DGN (Decreasing Global Node Utilities).

- DGU (Discarding Global Unpromising item) – Discarding global unpromising items and their utilities from transaction and transaction utilities of database.
- DGN (Decreasing Global Node Utility) – For any node in global UP - Tree, the utility of its descendants are discarded from the utility of node during construction of global UP-Tree.

2) Generate the potential High utility Itemsets by using UP-Growth algorithm. Generate local UP-Tree by using two strategies: DLU and DLN. For this two strategies required minimum item utility.

- DLU (Discarding Local Unpromising item) – Compute local Unpromising item and remove local unpromising item from path and recalculate path utility.
- DLN (Decreasing Local Node Utility) – Construct local UP-Tree and calculate node utility.

3) Identify High Utility Itemset.

This algorithm is complex for evaluation due to tree structures.

HUI-Miner [11] (High Utility Itemset Miner), for high utility itemset mining. HUI-Miner create the novel structure, called utility- list for each item, to store both the utility information about an itemset and the heuristic information for pruning the search space of HUI-Miner. Exact utility of an itemset is obtained by joining the utility-list of smaller itemsets.

When the number of candidates is so large that they cannot be stored in memory, the algorithms will fail or their performance will be degraded due to thrashing. By avoiding the costly generation and utility computation of numerous candidate itemsets, HUI-Miner can efficiently mine high utility itemsets from the utility lists constructed from a mined database. This algorithm is costly due to joining of utility lists.

C. *Problem with Existing System*

Performing mining process on large amount of datasets is very complex. Frequent Itemset Mining cannot meet demands arising from real life applications such as retail marketing because the frequent itemset mining treats all items with same importance.

The execution time and memory space are the main issues. The most of existing high utility itemset mining algorithms requires more execution time and memory space for large datasets.

III. PROPOSED METHOD

The existing algorithms facing some performance issues. To overcome this issues the new high utility itemset mining algorithm is proposed which is better than existing algorithms in terms of execution time and memory space.

A. *Proposed Algorithm*

- Input: D: Transaction database. minutil: user-specified threshold.
- Output: set of high utility itemset.

1) Step 1 Scan D to calculate the TWU of single items.
2) Step 2 I*🡐 each item i such that TWU (i) >= minutil
3) Step 3 Let < be the total order of TWU ascending values on I* .
4) Step 4 Scan D to built the utility –list of each item i ∈ I* and built the HMBS structure.
5) Step 5 Search (∅ ,I*,minutil,HMBS);

This new high utility itemset mining algorithm takes an input a transaction database with utility values and the minutil threshold. The algorithm first scans the database to calculate the TWU of each item. Then, the algorithm identifies the set I* of all items having a TWU no less than minutil. (Other items are ignored since they cannot be part of a high utility itemsets.) The TWU values of items are than used to establish a total order γ on items, which is the order of ascending TWU values.

A second database scan is then performed. During this database scan, items in transactions are reordered according to the total order γ, the utility-list of each item i ∈ I* is built and novel structure named HMBS (Hash Map Based Structure) is built.Building the HMBS is very fast (it is performed with a single database scan) and occupies a small amount of memory. After the construction of the HMBS, the depth-first search exploration of itemsets starts by calling the recursive procedure Search with the empty itemset $\emptyset$, the set of single items, and the HMBS structure.

The TWU measure has following important properties that are used to prune the search space.

1) Property 1 (overestimation): The TWU of an itemset X is higher than or equal to its utility, i.e. TCW(X) ≥ u(X)
2) Property 2 (antimonotonicity): The measure is anti-monotonic. Let X and Y be two itemsets. If $X \subset Y$ thenTWU(X) ≥ TWU(Y)
3) Property 3 (pruning): Let be an itemset. If TWU (X) < minutil, then the itemset X is a low-utility itemset as well as all its supersets.
4) Property 4 (sum of iutils): Let X be an itemset. If the sum of itil values in the utility-list of X is higher than or equal to minutil, then X is a high-utility itemset. Otherwise, it is a low-utility itemset.
5) Property 5 (sum of iutils and rutil): Let X be an itemset. Let the extensions of X be the itemsets that can be obtained by appending an item y > i for all item i in X . If the sum of iutil and rutil values in the utility-list of x is less than minutil, all extensions of and their transitive extensions are low-utility itemsets.

B. Pruning Procedure

- Input: P: an itemset, Extension of P: a set of extension of P, the minutil threshold, the HM structure.
- Output: the set of high utility itemsets

```
for each itemset P x € ExtensionsOf P do
If SUM($P_x$. utilitylist.iutils) ≥ minutil then
Output $P_x$;
end
If SUM(.$P_x$, utility.iutils) + SUM($P_x$.utilitylist.rutils) ≥ minutil then Extemsions0f$P_x$ ← $\emptyset$ ;
For each itemset $P_y$ ∈ ExtensionOfP such that y > x do
if ∋ (x,y,c) ∈ HMBS such that c ≥ minutil) then
$P_{xy}$ ← $P_x \cup P_y$;
$P_{xy}$.utilitylist ← Construct (P,$P_x$; $P_y$);
ExtensionsOf$P_x$ ← ExtensionsOf$P_x$ ∪ $P_{xy}$;
End
End
Search ($P_x$, ExtentionsOf$P_x$, minutil);
End
End
```

C. Construct Procedure

- Input: P: an itemset, ,P_x :the extension of P with an item x,P_y: the extension of P with an item y.
- Output: the utility – list of P_{xy}

```
For each tuple $e_x \in P_x$.utilitylist do
If ∃$e_y \in P_y$. Utilitylist and $e_x$.tid = $e_{xy}$.tid then
if P.utilitylist ≠ ∅ then
Search element e ∈ P.utilitylist such that e.tid = $e_x$.tid;
$e_{xy}$ ←($e_x.tid, e_x$ .iutil+$e_y$.iutil , $e_y$.rutil);
End
Else
$e_{xy}$ ←($e_x$.tid,$e_x$.iutil + $e_y$.iutil,$e_y$.rutil);
End
UtilityListOf$P_{xy}$ ←UtilityListOf$P_{xy}$ ∪{$e_{xy}$}
End
End
Return UtilityList$P_{xy}$;
```

The classic frequency-based framework often leads to many patterns being identified, most of which are not informative enough for business decision-making. In frequent pattern mining, a recent effort has been to incorporate utility into the pattern selection framework, so that high utility (frequent or infrequent)patterns are mined which address typical business concerns such as dollar value associated with each pattern. So we incorporate utility into sequential pattern mining, and a generic framework for high utility sequence mining is defined.

IV. EXPERIMENTAL EVALUATION

The Performance comparisons of existing and proposed algorithms are evaluated on various datasets. The experiments were performed with 4GB memory. The algorithms are implemented in java language. The real data sets are used for experiments. The real world datasets such as retails, chess are obtained from UCI Repository.

A. Execution Time

Here compare the performance of existing and proposed algorithms on retail dataset. When measuring execution time, we varied the min-util for database. Fig.1. shows the performance evaluation of existing and proposed algorithm for execution time. When minutil is 200 the execution time of proposed algorithm is 21840 ms and execution time of existing algorithm is 23425 ms.

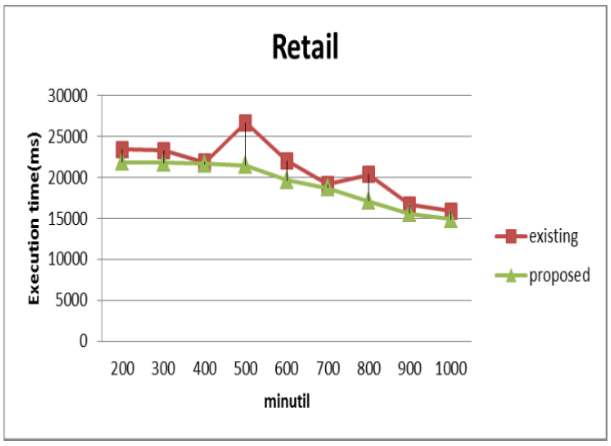

Fig. 1. Execution time on Retail dataset

Fig 2. shows the performance evaluation of existing and proposed algorithm for execution time on chess dataset. When minutil is 300 the execution time of proposed algorithm is 84742 ms and execution time of existing algorithm is 86146 ms.

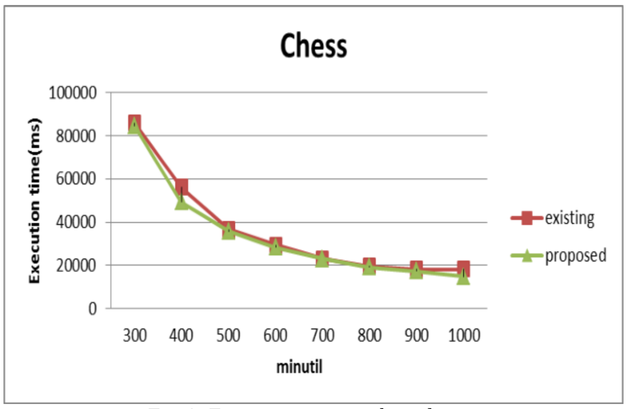

Fig. 2. Execution time on chess dataset

B. Memory Consumption

Fig 3. shows the performance evaluation of existing and proposed algorithm for memory space on retail dataset. When minutil is 200 the memory consumed by proposed algorithm is 28.15 MB and memory consumed by existing algorithm is 29.45 MB.

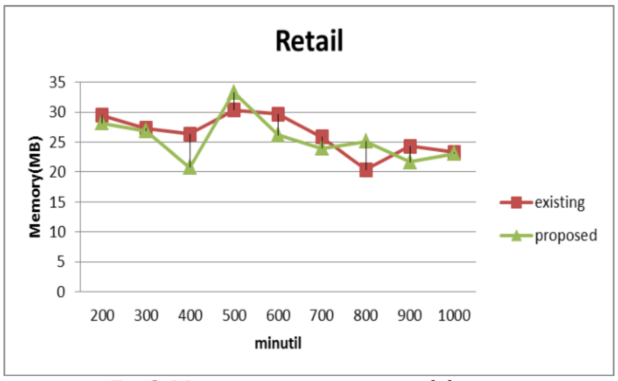

Fig. 3. Memory consumption on retail dataset

Fig 4. Shows the performance evaluation of existing and proposed algorithm for memory space on chess dataset. When minutil is 300 the memory consumed by proposed algorithm is 7.72 MB and memory consumed by existing algorithm is 7.92 MB.

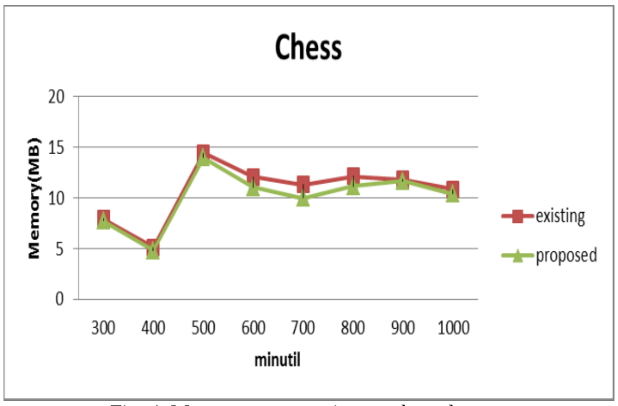

Fig. 4. Memory consumption on chess dataset

V. CONCLUSION

Frequent itemset mining is the most popular data mining algorithm. Many number of efficient techniques available for frequent itemset mining, which considers mining of frequent itemsets. But a promising technique in Data Mining is Utility Mining, which incorporates utility considerations during itemset mining. Utility Mining covers all aspects of economic utility in data mining. In existing Two-Phase, UP-Growth, HUI-Miner algorithms have been proposed, but there is problem like it requires more execution time and it uses more memory. The new technique is memory efficient technique for mining high utility itemsets from transactional databases. This technique requires less memory space and execution time than existing algorithms.

ACKNOWLEGMENT

I would first like to thank my thesis advisor **Mrs. Madhuri Zawar** of the Godavari COE, Prof. Pramod Gosavi for his guidance. I would also like to acknowledge Mr. Rahul Gaikwad, Mr. Nilesh Vani as the second reader of this thesis.

REFERENCES

[1] R.Agrawal, R.Shrikant. “Fast Algorithm for mining association rules.” Proceedings of 20th international Conference on Very Large Databases., 487-499, 1994.
[2] K.Vanitha, R.Santhi. “Evaluating the performance of association rule mining algorithm.” Journal of Global Reasearch in computer science, Vol.2, 2229-371X,June 2011.
[3] R. Agrawal, T. Imielinski, A. Swami. “Mining association rule between sets of items in large databses.” proceeding of the ACM SIGMOD International Conference of Management of data., 207-216, 1993.
[4] Gagandeep kaur, Shruti Aggrawal. “Performance analysis of association rule mining algorithm” International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3. 2277-128X, 2005.
[5] J.Han, J.Pei, Y.Yin. “Mining Frequent Patterns without Candidate generation.” Data Mining and Knowedge Discovery, 2004. pp. 53-87.
[6] Sadak Murali, Kolla Morarjee. “A survey on efficient algorithm for mining high utility itemset.” International Journal of Research in Engineering & Advanced Technology, Vol. 1. 2320-8791, Nov 2013.
[7] Liu. Y, Liao W, A. Choudhary. "A fast high utility itemset mining algorithm." Proceeding of the Utility-Based Data Mining Workshop. August 2005.
[8] Y. Liu, W-Keng Liao, A. Choudhary."Two-Phase Algorithm for Fast Discovery of High Utility Itemsets." PAKDD, Berlin, 689-695, 2005.
[9] V.S. Tseng, B-En Shie, C-W Wu, P.S. Yu. "Efficient Algorithms for Mining High Utility Itemsets from Transactional Databases." IEEE Transaction on Knowedge and Data Engineering,Vol. 25, August 2013.
[10] V.S.Tseng, C-W Wu, B-E Shie, P.S Yu." U"P-Growth: An Efficient Algorithm for High Utility Itemsets Mining. Proceeding 16th ACM SIGKDD Conf, Knowledge Discovery and Data Mining, 253-262, 2010.
[11] Liu, M.J. Q." Mining High Utility Itemsets without Candidate Generation."Proceedings CIKM12, 55–64, 2012.
[12] Sudip Bhattacharya, Deepty Dubey. "High Utility Itemset Mining." International Journal of Emerging Technology and Advanced Engineering, Vol. 2. ISSN 2250-2459,August 2012.