Quantitative Association Rule Mining on Weighted Transactional Data

D. Sujatha and Naveen C. H.

***Abstract*—In this paper we have proposed an approach for mining quantitative association rules. The aim of association rule mining is to find interesting and useful patterns from the transactional database. Its main application is in market basket analysis to identify patterns of items that are purchased together. Mining simple association rules involves less complexity and considers only the presence or absence of an item in a transaction. Quantitative association mining denotes association with itemsets and their quantities. To find such association rules involving quantity, we partition each item into equi-spaced bins with each bin representing a quantity range. Assuming each bin as a separate bin we proceed with mining and we also take care of reducing redundancies and rules between different bins of the same item. The algorithm is capable in generating association rules more close to real life situations as it considers the strength of presence of each item implicitly in the transactional data. Also the algorithm can be applied directly to real time data repositories to find association rules.**

***Index Terms*—Association mining, quantitative association rule mining (QAR), Apriori algorithm.**

I. Introduction

Data mining represents techniques for discovering knowledge patterns hidden in large databases. Several data mining approaches are being used to extract interesting knowledge [2]. Association rule mining techniques [18][19] discover associations between itemsets, clustering techniques [3] group the unlabeled data into clusters, classification techniques [20] identify the different classes existing in categorical data.

It can be observed that most of the data mining approaches discover association rules from binary data. However, itemsets are mostly associated with quantity. In literature, it has been reported that there exists useful knowledge pertaining to quantitative association mining [5][6].

In this paper, we are investigating a simple approach to mine quantitative association rules. Association rule mining [19] finds interesting patterns and has been extensively studied [4][5]. Association rules show attributes value conditions that occur frequently together in a given dataset. A typical and widely used example of association rule mining is market basket analysis. The basic terminology about association rules is as follows: Let $I = \{i_1, i_2, \ldots i_n\}$ be a set of items and T be a set of transactions. Each transaction T_i ($i = 0,1,\ldots,m$) is a set of items such that $T_i \subseteq I$. An itemset X is a set of items $\{i_1, i_2, \ldots, i_k\}$ ($1 \leq k \leq n$) such that $X \subseteq I$. An itemset containing k number of items is called k-itemset. An association rule is an implication of the form, A => B, where $A \subset I$, $B \subset I$ and $A \cap B = \phi$. The rule A => B holds in T with support s if S% of the transactions in T contain A and B. Similarly rule A=>B holds in T with confidence c if C% of transactions in T support A also support B. Given T, the objective of association rule mining is to discover all association rules that have support and confidence greater than the user-specified minimum support *min_sup* and minimum confidence *min_conf.*

Quantitative association rule mining refers to association rule forming between frequent items.

Example 1: Consider the set of items {bread, jam, butter, eggs} selling in a super market. It can be observed that the items in the set {bread, jam} are frequently purchased items. Then the objective is to determine the quantity of bread that was taken with a particular quantity of jam[1].

In [21], one of the earliest work related to mining association rules involving weights, rules of the form $x=q_x=>y=q_y$ are discovered, where x and y are items and q denotes the quantity. However, the antecedent and the consequent are limited to a single <attribute, value> pair. This is done by partitioning the intervals and not combining later.

The problem of mining quantitative association rules was dealt by fine partitioning the attributes and combining adjacent partitions as necessary [16]. It was done by decomposing categorical attributes into unique valued attributes and quantitative attributes into ranges. However, appearance of categorical attributes in data used for market basket analysis is negligible and considering a new attribute for all the values leads to exponential amount of rules. Also it has proposed measures for preventing data loss while combining the partitions.

In the proposed algorithm, we try to find a balance between discretizing values and reducing redundancy by partitioning items into equi-width intervals and pruning intervals which are less frequent before generating the rules.

The rest of the paper is organized as follows. In section 2, we discuss the related work. In section 3, we explain the proposed approach and algorithm. Conclusion and future works are mentioned in Section 4.

II. Related Work

In this section we briefly discuss the related approaches for the extraction of quantitative association rules.

D. Sujatha is with the Department of Information Technology Aurora's Technological and Research Institute Hyderabad, India 9 (sujatha.dandu@gmail.com)

Naveen CH is with the Department of Information Technology Aurora's Technological and Research Institute Hyderabad, India (naveen.ch1231@gmail.com)

A. Apriori algorithm

The utilization of frequent itemsets or support based pruning to exponential growth of itemsets with systematic control was initiated by the Apriori algorithm [17]. The data is assumed to be represented as binary data set, with each row corresponding to a transaction, and each column corresponding to an item. The corresponding entry of any item belonging to a particular transaction is equal to 1; otherwise it is denoted as 0. The support for a set of items is the number of transactions in which each has attribute value 1.

The Apriori principle states that "If an itemset is frequent, then all of its subsets must also be frequent". The algorithm is divided into two steps. The first step is generation of candidates and second is Pruning. These two steps are applied to every iteration.

Apriori generates frequent itemsets one level at a time, from itemset of first level to the longest frequent itemset. New candidate itemsets are generated at each level by using itemsets of previous level. At each level, the database is scanned once and the support count is calculated for every itemset and those itemsets which do not satisfy the minimum support are pruned.

B. Quantitative rule mining approaches

Adaptation of the APRIORI algorithm for mining quantitative association rules was identified shortly after the introduction of APRIORI algorithm, the necessity for quantity in mining association rules was first identified in [14]. It proposed rules of the form $x=q_x=>y=q_y$ i.e. it associated a single quantity q to the antecedent and the consequent. This was done by decomposition of one quantitative attribute into several binary attributes.

In almost all works dealing with mining quantitative attributes, discretization is considered as the tool for reducing the time complexity associated with mining quantitative association rule mining algorithms as the number of quantities can be infinite. Discretization was first proposed in [16].

Mere reduction of quantitative values into Boolean values was also proposed by some authors [11][15].

In [7] it was argued that discretization leads to information loss and hence completely omitting discretization step in mining QAR was proposed. It proposed a representation of the rules based on half-spaces. But the rules generated with such method are different from the classical rules and their understandability is questioned.

A new measure of quality for mining association rules is proposed in [9]. Here a new kind of rule called ordinal association rule is used to mine QAR, it removes the step of discretization and complete disjunctive coding and aims at obtaining variable discretization of numerical attributes.

Usage of statistical values, like mean as the measure of quality for mining quantitative association rules was proposed in [8] and [12].

The time complexity of QAR mining increases exponentially as the number of possible attributes values grows. This time consumption is another important and discussed issue addressed mainly in [12] and [13].

Quantitative attributes result in lots of redundant rules, most algorithms generate rules that provide almost the same identical information. Such redundancy issue has been partially mentioned in [10], where optimized support and confidence measures are defined and used.

III. Generating the Data Set

The Data set used for our analysis has been generated using mathematical concepts to ensure it resembles a real-world data set.

The algorithm for generating the dataset uses the following two steps that were implemented on matlab:

Firstly, generate a binary dataset and next, convert the generated binary dataset to a weighted dataset.

A. Generating Binary dataset

Generate 'm' random variables that follow Poisson distribution, here 'm' represents the number of transactions of your dataset. Next, using each value of 'm', generate random numbers between 1 to 'n', where 'n' denotes the number of items in the dataset. So, the first Poisson random variable 'm_1=x' is used to generate 'x' number of random numbers between 1 to 'n'. Similarly, random numbers for all the corresponding Poisson random numbers are generated. The generated random numbers between 1 to 'n' for each Poisson random number represents the item numbers of items purchased in that transaction.

Suppose you want to generate a 500 transaction dataset with 100 items, then you first need to generate 500 Poisson random numbers. For the 1st Poisson random number, say 7, you will next generate 7 random numbers between 1 to 100. This denotes that the 1st transaction has 7 items purchased whose item numbers correspond to the 7 random numbers generated. This procedure is done for all the Poisson random numbers. Finally we get to know how many transactions contain how many items and their item numbers that were purchased.

Next, using programming tools, create a zero matrix of size 'm' rows and 'n' columns. For the first row, change the zero's to one's in the columns that have the same number as that of the random numbers generated for the first Poisson random number.

If for the first Poisson random number 7 we generate 7 random numbers say, {10,13,29,53,78,81,90}, then in the columns numbered {10,13,29,53,78,81,90} of the m x n zero matrix change '0' to '1'. The above procedure is done for the remaining rows. Finally, we get a matrix that represents a binary transactional matrix.

TABLE I. Binary Data Set

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
T_1	0	0	0	1	0	0	0	0	1	0
T_2	0	1	0	0	0	0	0	1	0	1
T_3	1	1	1	1	1	0	0	1	0	1
T_4	1	1	0	0	0	0	1	0	0	0
T_5	1	1	1	1	1	0	0	0	0	0
T_6	0	0	1	1	0	0	0	0	0	1
T_7	1	1	0	0	0	1	0	0	0	0
T_8	1	1	1	0	0	0	0	0	1	0
T_9	1	0	0	0	1	0	0	1	1	1
T_{10}	1	1	1	1	0	0	0	0	0	1

B. Converting Binar Dataset to Weighted dataset

The following procedure converts the Binary dataset

[Table I] to a non-binary or weighted dataset.

A Binary dataset of 'm' transaction and 'n' items has 'm' number of rows and 'n' number of columns. Conversion from binary to non-binary form is done column wise here. For each column of the binary matrix, we generate 'm' number of Binomially distributed random numbers and store them in a m x 1 array. We next multiply this m x 1 array values with the first column values of the binary matrix. The multiplication is a one-to-one correspondence and not a array multiplication.

The above procedure is performed for all the columns of the binary matrix converting it to a non-binary matrix. This weighted matrix is stored in an Excel sheet or a table and can be used as a non-binary dataset.

Table II represents the non-binary dataset corresponding to Table I after performing the above procedure on the later.

IV. Proposed Quantitative Association Rule Mining Algorithm

Let $I = \{i_1, i_2, i_3...i_m\}$ be the set of items. Let $T = \{t_1, t_1....t_n\}$ be the set of transactions where each transaction t_i is a set of items such that $t_i \in T$. Each item can be represented as a column in a matrix and each row corresponds to a transaction. So each row of the matrix contains weights of the corresponding item purchased and zero for items that are not purchased. It can be denoted as in Table II.

TABLE II. Non-Binary Data Set

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
T_1	0	0	0	11	0	0	0	0	7	0
T_2	0	6	0	0	0	0	0	5	0	6
T_3	10	10	4	11	13	0	0	4	0	7
T_4	4	5	0	0	0	0	10	0	0	0
T_5	4	7	6	8	11	0	0	0	0	0
T_6	0	0	5	10	0	0	0	0	0	6
T_7	5	5	0	0	0	11	0	0	0	0
T_8	2	5	4	0	0	0	0	0	7	0
T_9	5	0	0	0	11	0	0	5	7	6
T_{10}	6	8	5	11	0	0	0	0	0	5

$T_i = \{w_1I_1, w_2I_2, w_3I_3, w_4I_4, w_5I_5\}$, where T refers to a transaction, I refers to items and $\{w_1, w_2, w_3, w_4, w_5\}$ are different and variable weights.

Firstly, this non-binary dataset as shown in Table II requires new approach to calculate support, for this purpose we are going to transform the non-binary dataset into an intermediate matrix representation. Secondly, pruning will be done on the determined support values to find items of importance. Thirdly, candidates will be generated and their corresponding support will be found from the intermediate matrix representation. Lastly, the pruned candidates confidence will be calculated and we get the frequent itemset and its support and confidence, hence a rule can be generated from it.

The *confidence* of a rule is defined as

$$Conf(x=>y) = supp(X \cup Y)/supp(X)$$

A. Intermediate matrix representation

The non-binary data set as shown in Table II is represented using a matrix. Using this Table an intermediate matrix is generated that can be used to find support for item sets at any level.

Algorithm:

Initialize j=1; no=number of intervals; i = number of items; t = number of transactions; m[t,i] = non-binary dataset, n[t,i*no] = transformed matrix;

While (j < i) /* repeat for all columns of dataset matrix */

Begin

Read column 'i' of the matrix 'm';

Determine the minimum value (quantity) in the column other than zero and maximum value in the column;

Partition the column into a fixed number of equal intervals, length of each interval will be

z = (maximum − minimum)/number of intervals.

If maximum == minimum

Then z=1;

End

Initialize the intermediate matrix 'n' with zeros;

Assume each column of this matrix 'n' to represent the quantity interval of each column (item) calculated above;

Scan the column 'i' from the beginning and place the row number t_n in the corresponding interval column of the intermediate matrix.

End

Answer := n[t,i*no] /*transformed matrix*/

Example *2:*

An enterprise may map product as an item and a purchase as a transaction. Now in the non-binary search space each transaction represents the itemset of weight values of the respective-product matrix [Table II].

Applying the above algorithm to Table II, we assume the number of quantity intervals = 3. For the first item I_1, we have maximum value=10, minimum value=2, therefore z=(10−2)/3=2.6666=>3. The following Table II is generated where , $I_i = \{Iq_1 \cup Iq_2 \cup Iq_3\}$

$Iq_j = \{T_k\}$,where $(q_j.I_i) \in T_k$, q_j = quantity interval of the item.

TABLE III. Item 1 Interval's

Iq_1 (quantity={2,3,4})	Iq_2 (quantity={5,6,7})	Iq_3 (quantity={8,9,10})
T4	T7	T3
T5	T9	0
T8	T10	0
0	0	0

The three columns in Table III will be placed in the first three columns of the intermediate matrix. Repeating the above method for other items we complete the construction of intermediate matrix. So now all the intervals are stored into a matrix called the transformed matrix. The transformed matrix is shown in Table IV**.**

Transformed matrix = $\{NI_1, NI_2,...,NI_{30}\}$

where NI_i = New $item_i$ and $NI_i \in I_i$

For mapping purpose the values of minimum value, maximum value and z are stored in a three column matrix named 'map' [Table IX].

Now the transformed matrix is treated as the dataset which is to be mined. Each Item interval derived will now be considered as an individual item itself. So, we have fifteen items now.

Finding the support for these items is done by counting the number of non-zero rows in each column (new item).

B. Pruning the Items

TABLE IV. TRANSFORMED MATRIX

I_1			**I_2**			**I_3**			**I_4**			**I_5**			
New item 1	**New item 2**	**New item 3**	**New item 4**	**New item 5**	**New item 6**	**New item 7**	**New item 8**	**New item 9**	**New item 10**	**New item 11**	**New item 12**	**New item 13**	**New item 14**	**New item 15**	**......**
T_4	T_7	T_3	T_2	T_5	T_3	T_3	T_6	T_5	T_5	0	T_1	T_5	0	T_3	......
T_5	T_9	0	T_4	T_{10}	0	T_8	T_{10}	0	0	0	T_3	T_9	0	0	.....
T_8	T_{10}	0	T_7	0	0	0	0	0	0	0	T_6	0	0	0	......
0	0	0	T_8	0	0	0	0	0	0	0	T_{10}	0	0	0	......
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	......
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	......
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	......
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.....
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	......
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	......

Once the support values have been determined, we go for pruning the items; pruning is a process where the items whose support value is less than the user defined support threshold are removed.

Example *3:*

For the above transformed matrix and the support array, prune with user defined threshold=2;

We now store the results in a new matrix with two columns, the first column stores the item number and the second column stores the corresponding pruned support value.

If I_i is i^{th} item then the pruned support matrix consists of {i, support(I_i)}

C. Candidate generation

To generate the candidates we next read the first-column pruned support matrix, i.e. the item numbers of items that are qualified. We determine all the possible combinations of size two using these item numbers and store it in a matrix. So the rows of this new matrix will be the frequent itemsets of level 1.

Example *4:*

Using the *pruned support matrix as shown in Table V, we get the Candidates matrix as shown in Table VI.*

Candidates Matrix $= \{ {}^{n}C_2(NI_1, NI_2, NI_3, \ldots NI_{15}) \}$

Now, for the above candidate itemsets we determine the support using the intermediate matrix by scanning the corresponding item columns and performing join/intersection operation to get common transaction numbers i.e. to check whether the items come together in a particular transaction or not. If there exists a common entry then it implies that the new items (old item quantity interval) are related in a transaction from the original non-binary dataset.

Example *5:*

For itemset {1,2} scan the columns 1 and 2 from the transformed matrix and perform a join ($1 \cap 2$) and count the number of values returned.

$\{T2,T10\} \cap \{T1,T6,T9\} = \phi$

Similarly, for item set {2,10}

$\{T1,T6,T9\} \cap \{T3,T6,T9\} = \{T6,T9\}$=>support count=2

This way support is found for all itemsets and stored in a new column.

The new matrix now will contain items in the first two columns and their corresponding support in the third column as shown in Table VII.

Candidates Matrix with support = Candidates Matrix θ Support

Where θ is a join operation and Support = COUNT (NI_x AND NI_y being purchased together), $\forall \{NI_x, NI_y\}$ of Candidate Matrix.

Next step is to prune the candidates based on support threshold (assumed 2 earlier or dynamic support may be introduced).

After pruning, the resultant matrix called the candidates matrix with pruned support, will have itemsets and their qualified support as shown in Table VIII.

Candidate Matrix with pruned support = Candidate Matrix with Support (support > threshold).

To generate candidates for the next level, merge the item columns and generate all possible subsets and continue with the support determination as shown above. Further, candidates for the above example as shown in Table VIII are not possible so we stop here and the rules are now formulated.

D. Rule Formation

For the formation of rules the final candidate matrix with pruned support is taken and rules are written in the form

Item X => Item Y; support, confidence

i.e. purchase of item X results in purchase of item Y and has a support and confidence of some value calculated.

Example *6:*

From Table VIII. Using the pruned candidate matrix we get the following rules

New $Item_1$ => New $Item_4$; support = 2

New $Item_8$ => New $Item_{12}$; support = 2

New $Item_{23}$ => New $Item_{29}$; support = 2

i.e. purchase of new item2 results in purchase of new item4, this rule has a support of 2.

The converse is also frequent

New $Item_4$ => New $Item_1$; support = 2

New $Item_{12}$ => New $Item_8$; support = 2

New $Item_{29}$ => New $Item_{23}$; support = 2

For the above three rules, confidence is calculated using the formula,

Confidence = support (item x, item y)/support (item x)

TABLE V. PRUNED SUPPORT MATRIX

Item number	1	2	4	5	7	8	12	13	23	25	29
Support	3	3	4	2	2	2	4	2	2	3	3

TABLE VI. CANDIDATES MATRIX

New item	Possible new item for combinations after pruning									
1	2	4	5	7	8	12	13	23	25	29
2	-	4	5	7	8	12	13	23	25	29
4	-	-	5	7	8	12	13	23	25	29
5	-	-	-	7	8	12	13	23	25	29
...	...	...	...	...	...	...	...	...	...	...

TABLE VII. CANDIDATES MATRIX WITH SUPPORT

Item Set	1	1	1	...	1	1	2	2	2	...	2	2	4
	2	4	5	...	25	29	4	5	7	...	25	29	5
Support	0	2	1	...	1	0	1	1	0	...	1	1	0

TABLE VIII. CANDIDATES MATRIX WITH PRUNED SUPPORT

New Item	New Item	Support
1	4	2
8	12	2
23	29	2

Support (item x, item y) taken from the third column of the pruned candidate matrix

Support (item x) value is found from scanning transformed matrix item 'x' column and counting non-zero rows.

Now the complete rules are of the form

New $Item_1$ => New $Item_4$; support = 2, confidence = 67%

$NewItem_8$ => New $Item_{12}$; support = 2, confidence = 100%

New $Item_4$ => New $Item_1$; support = 2, confidence = 50%

New $Item_{12}$ => New $Item_8$; support = 2, confidence = 50%

Finally these assumed new items are mapped back to their respective original items where weights come back to picture and the rules involve quantity.

Exam*ple 7:*

TABLE IX..MAP MATRIX (MIN, MAX, Z)

Original item {sub-items:transformed matrix}	Minimum	Maximum	Interval size
I_1 {NI_1,NI_2,NI_3}	2	10	3
I_2 {NI_4,NI_5,NI_6}	5	10	2
I_3 {NI_6,NI_7,NI_9}	4	6	1
.....	....	....	....

Consider the rule

$NewItem_1$ => $NewItem_4$; support = 2, confidence = 67%

From the transformed matrix, $NewItem_1$ is the first quantity interval of original Item I_1 whose interval consists of quantities of value={2,3,4} [Table IV]. Similarly $NewItem_4$ corresponds to item I_2 in the first quantity interval with value={5,6} [Table IV]. This reverse mapping can be done using the 'map' matrix [Table IX].

Thus the final rules are of the form,

Item (quantity interval) => Item (quantity interval); support, confidence

Example 8:

I_1(quantity=(2,3,4)) results in purchase of I_2(quantity=(5,6)) with support value of 2 and confidence value of 67%.

This way the association rules are mined for items with their quantity intervals, For specific quantity rules can also be derived by simply taking into consideration only those transactions where that certain quantity of the item is purchased and association with other items can be determined by following the above algorithm.

V. CONCLUSION AND FUTURE SCOPE

In this paper, an approach to mine quantitative association rules over non binary data set has been proposed. The algorithm proposed can be improved further by reducing the size of quantity intervals taken during transformation or by just considering an interval for each quantity value. This would require greater memory and processing powered systems for faster computations. Our algorithm uses Apriori algorithm which is inferior to FP-Growth algorithm when the number of steps taken is considered. So the same deal of mining quantity datasets for association rules can be taken up using other mutated algorithms which will prove to be of great use to business personnel and market strategists.

REFERENCES

[1] Uday Kiran, R., Krishna reddy, P. "An Improved multiple minimum support based approach to mine rare association rules.", In: IEEE symposium on computational intelligence and data mining, 2009.

[2] Melli, G., Osmar, R. Z., and Kitts, B. "Introduction to Special Issue on Successful Real-World Data Mining Applications.", In: SIGKDD Explorations, Volume 8, Issue 1, 2006.

[3] Xu, R. "Survey of Clustering Algorithms." Proc. IEEE Transaction on Neural Networks, Volume 16, Number 3, May 2005.

[4] Rajanish, D., and Ambuj, M. "Fast Frequent Pattern Mining in Real-Time.", In: CSI, 2005, pp. 156-167.

[5] Jiawei, H., Jian, P., Yiwen, Y., and Runying, M. "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree approach*.". In: DMKD, 2004, pp . 53-87.

[6] Weiss, G. M. "Mining with Rarity: AUnifying Framework/", In: SIGKDD Explorations, 2004, Vol 6, Issue 1, pp. 7-19.

[7] U. Ruckert, L. Richter, and S. Kramer. Quantitative association rules based on half-spaces: An optimization approach. In: icdm, 00:507_510, 2004.

[8] Y. Aumann and Y. Lindell. A statistical theory for quantitative association rules. In: Journal of Intelligent Information Systems, 20:255_283, 2003.

[9] S. Guillaume. Discovery of ordinal association rules. In: Proceedings of the Sixth Pacific-Asia Conference PAKDD'02, Taiwan, 2002.

[10] R. Rastogi and K. Shim. Mining optimized association rules with categorical and numeric attributes.Proc. IEEE Trans. on KD Engineering, 14(1), 2002.

[11] S. Imberman and B. Domanski. Finding association rules from quantitative data using data booleanization. In: Proceedings of the Seventh Americas Conference on Information Systems (AMCIS 2001), 2001.

[12] G.I. Webb. Discovering associations with numeric variables .In: Proc. of ACM SIGMOD Conference on Management of Data, San Francisco, CA, 2001.

[13] J. Wijsen and R. Meersman. "On the complexity of mining quantitative association rules.". In: Data Mining and Knowledge Discovery, 2:263_281, 1998.

[14] R.J. Miller and Y. Yang." Association rules over interval data.". In: Proc. of ACM SIGMOD Conference on Management of Data, Tuscon, AZ, 1997.

[15] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama. Mining optimized association rules for numeric attributes. In: Proc. of ACM SIGMOD Conference on Management of Data, Montreal, Canada, 1996.

[16] Srikant, R., and. Agrawal, R. " Mining quantitative association rules in large relational databases". In: Proc. of ACM SIGMOD Montreal, 1996.

[17] Agrawal, R., and. Srikant, R.." Mining generalized association rules.,.

In: 21st International Conference on Very Large Databases, Taiwan, Taipei, 1995.

[18] Agarwal, R., and Srikanth, R. "Fast algorithms for mining association rules."In: VLDB, 1994.

[19] Agarwal, R., Imielinski, T., and Swami, A. "Mining association rules between sets of items in large databases." In: SIGMOD, 1993, pp. 207-216.

[20] Weiss, S., and Kulikowski, C. A. "Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems.", Morgan Kaufmann, 1991.

[21] G. Piatetsky-Shapiro. "Discovery, analysis and presentaion of strong rules." In: G. Piatetsky-Shapiro and W. J. Frawely, editors, Knowledge Discovery in Databases, pages 229-248. AAAI/MIT Press, Menlo park, CA, 1991.

D.Sujatha received B.E degree in Computer Science & Engineering from P.D.A College of Engineering, Affiliated to Gulbarga University, Gulbarga, India , in 1998.M.Tech in Computer Science & Engineering in JNTU college of Engineering, Kakinada, India in 2006. Presently pursuing Ph.D degree in Computer Science & Engineering from JNTU, Hyderabad, AP, India. Her research interest includes Data Mining and Databases

Naveen CH received B.TECH degree in Information Technology from Aurora's Technological and Research Institute, Affiliated to Jawaharlal Nehru Technological University, India, in 2011.The paper demonstrates the project done by him as a student. His interests are in the fields of Data Mining and Data processing