International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 02 Issue: 05 | Aug-2015 www.irjet.net p-ISSN: 2395-0072

TENET PRODUCTION WITH APRIORI AND GENETIC ALGORITHMS

Vankara Ramesh Babu[1], Appala Raju Samanthula [2]

[1] *M.Tech Final Year, Dept of Computer Science & Engg, Raghu Engineering College, Andhra Pradesh, India*
[2] *Assistant Professor, Dept of Computer Science & Engg, Raghu Engineering College, Andhra Pradesh, India*

Abstract - ***Association rule mining is performed in production of frequent item sets and rule generation. Mining association rules cannot be rewarded completely, unless it is utilized to improve decision making process in a firm. Here, we are concerned with discovering positive and negative association rules. An apriori algorithm can find all valid positive and negative association rules and overcome some limitations of the previous mining methods, some of the practitioners had motivated to optimize the rule for analysis purpose because of the size and complexities of rules observed in previous mining methods. We can predict the rules by using genetic algorithm. The major advantage of genetic algorithm is, they perform global search and its complexity is less compared to other algorithms. This work is proposed to find all the possible optimized rules from given data set using genetic algorithms.***

Key Words: *Production, Item sets, Tenet generation, Mining association rules, Practitioners, Genetic Algorithms.*

1. INTRODUCTION

1.1 Data Mining Algorithms

A data mining [1][5][19] algorithm is a set of heuristics and calculations that creates a data mining model [6] from data. To create a model, the algorithm first analyzes the data which is provided, considering specific types of patterns or trends. The algorithm uses the results of this analysis to define the optimal parameters for creating the mining model [8]. These parameters are then applied across the entire data set to extract actionable patterns and detailed statistics.

The mining model [13] that an algorithm creates from the data can take various forms, including:

- A set of clusters [11] that describe how the cases in a dataset are related.
- A decision tree that predicts an outcome.
- A decision tree describes how different criteria affect that outcome.
- A mathematical model that forecasts sales.
- A set of rules that describe how items are grouped together in a transaction

1.2 Apriori Algorithm

Apriori is an algorithm [2][3][4][9] for frequent item set mining(I) and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and extending them to larger " I " as long as those " I " appear sufficiently often in the database. The frequent " I " determined by apriori can be used to determine association rules which highlight general trends in the database. Apriori is designed to operate on databases containing transactions for finding association rules in data which has no transactions or no timestamps. Each transaction is seen as a set of items, given a threshold " C", the apriori algorithm identifies the " I " which are subsets of at least "C" transactions in the database . Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time , and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.Apriori uses breadth-first search and a has tree structure to count candidate " I " efficiently. It generates candidate " I " of length "K" from " I " of length "K-1". Then it prunes the candidates which have an infrequent sub pattern. According to the downward closure lemma, the candidate set contains all frequent K-length " I ". After that, it scans the transaction database to determine frequent " I " among the candidates.

The pseudo code for the algorithm is given below for a transaction database "T" and a support threshold of "€".

- Usual set theoretic notation is employed; though note that "T" is a multi set.
- "C_k" is the candidate set for level "K".
- for each step, the algorithm is assumed to generate the candidate sets from the large "I" of the preceding level, heeding the downward closure lemma.
- do
  - ❖ Count(C) accesses a field of the data structure that represents candidate set C, where C==0.
  - ❖ Implementing the data structure used for storing the candidate sets.
- Done

1.3 Genetic Algorithms

A GA is a search heuristic that mimics the process of natural selection. This heuristic is routinely used to generate useful solutions to optimization and problems. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover.In a genetic algorithm [10][12][20], a population of candidate solutions, has a set of mutated, altered, traditional, solutions which are represented in binary strings of 0's and 1's. The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a production. In each production, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function [14] in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when maximum number of generations has been produced, or a satisfactory fitness [15][21] level has been reached for the population.

1.4 Mining Association Rules

Mining association rules [7][13] are intended to identify strong rules discovered in databases using different measures of interest. Based on the concept of strong rules introduced, association rules for discovering regularities between items in large-scale transaction data recorded by point-of-sale (POS). Analysis association rules are employed today in many application areas including web usage mining [16], intrusion detection [17], Continuous production bioinformatics. In contrast with sequence mining, association rule learning [18] typically does not consider the order of items either within a transaction or across transactions. The problem of association rule mining is defined as: Let I= { i_1, i_2,..... i_n} be a set of "n" binary attributes called items. Let D= { t_1, t_2,..... t_m} be a set of transactions called the database. Each transaction in "D" has a unique transaction ID and contains a subset of the items in "I". A rule is defined as an implication of the form X=>Y where X, Y ⊆ I and $X \cap Y = \emptyset$. The sets of items "X" and "Y" are called antecedent and consequent of the rule respectively.

Process

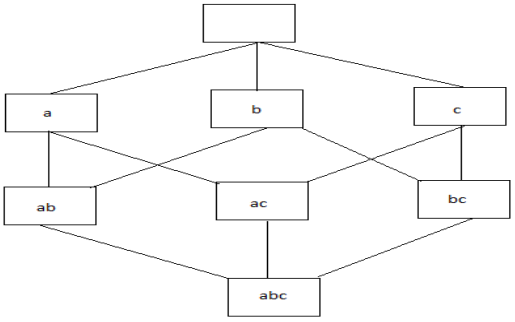

Fig - 1: Frequent item set lattice

Frequent item set lattice, where the colour of the box indicates how many transactions contain the combination of items. Note that lower levels of the lattice can contain at most the minimum number of their parent items; e.g. {ac} can have only at most min(a,c) items. This is called the downward-closure property. Association rules are usually required to satisfy a user-specified minimum support(MINSUP) and a user - specified minimum confidence (MINCONF) at the same time. Association rule generation is usually split up into two separate steps:

1. First, minimum support is applied to find all frequent " I " in a database.
2. Second, these frequent " I " and the minimum confidence constraint are used to form rules.

While the second step is straightforward, the first step needs more attention.Finding all frequent " I " in a database is difficult since it involves searching all possible " I ". The set of possible " I " is the power set over "I" and has size "$2^n - 1$". Although the size of the power set grows exponentially in the number of items n in "I", efficient search is possible using the downward-closure property of support, which guarantees that for a frequent item set, all its subsets are also frequent and thus for an infrequent item set, all its supersets must also be infrequent. Exploiting this property, efficient algorithms can find all frequent " I ".

2. PROPOSED SYSTEM

Let I= { i_1, i_2,..... i_m} be a set of literals, called items. Let D be a set of transactions, where each transaction T is a set of items such that T ⊆ I, associated with each transaction is a unique identifier, called its TID. We say that a transaction T contains X, a set of some items in I, if X⊆ T. An association rule is an implication of the form X⇒ Y, where X⊂ I, Y⊂ I, and X∩Y= ∅.The rule X⇒Y holds in the transaction set D with confidence C, if C% of transactions in D that contain X also contain Y. The rule X⇒Y has support S in the transaction set D if S% of transactions in

D contains X∪Y.Given a set of transactions D, the problem of mining association rules is to generate all association rules that have support and confidence greater than the user-specified MINSUP and MINCONF respectively.The problem is usually decomposed into two sub problems, one is to find those " I " whose occurrences exceed a predefined threshold in the database; those "I" are called frequent or large "I". The second problem is to generate association rules from those large "I" with the constraints of minimal confidence. Suppose one of the large " I " is L_K, $L_K = \{I_1, I_2, \dots, I_K\}$, association rules with this " I " are generated in the following way: the first rule is $\{I_1, I_2, \dots, I_{K-1}\} \Rightarrow \{I_K\}$, by checking the confidence this rule can be determined as interesting or not. Then other rule are generated by deleting the last items in the antecedent and inserting it to the consequent, further the confidences of the new rules are checked to determine the interestingness of them. Those processes iterated until the antecedent becomes empty. Since the second sub problem is quite straight forward, most of the researches focus on the first sub problem. The apriori algorithm finds the frequent sets L in Database D.

Let $X,Y \subseteq I$ be any two " I ". Observe that if $X \subseteq Y$, then $sup(X) \geq sup(Y)$, which leads to the following two corollaries:

- If X is frequent, then any subset $Y \subseteq X$ is also frequent.
- If X is not frequent, then any superset $Y \supseteq X$ cannot be frequent.

Based on the above observations, we can significantly improve the item set mining algorithm by reducing the number of candidates we generate, by limiting the candidates to be only those that will potentially be frequent. First, we can stop generating supersets of a candidate once we determine that it is infrequent, since no superset of an infrequent item set can be frequent. Second, we can avoid any candidate that has an infrequent subset. These two observations can result in significant pruning of the search space.

- Find frequent set L_{K-1}
- Join Step.
  - ❖ C_K is generated by joining L_{K-1} with itself
- Prune Step.
  - ❖ Any (K-1) item set that is not frequent cannot be a subset of a frequent K-item set, hence it should be removed.

where

- (C_K: Candidate item set of size K)
- (L_K: frequent item set of size K)

GAs simulates the survival of the fittest among individuals over consecutive generation for solving a problem. GAs are based on an analogy with the genetic structure and behaviour of chromosomes within a population of individuals using the following fundamentals:

- Individuals in a population compete for resources and mates.
- Those individuals most successful in each 'competition' will produce more offspring than those individuals that perform poorly.
- Genes from `good' individuals propagate throughout the population so that two good parents will sometimes produce offspring that are better than either parent.
- Thus each successive generation will become more suited to their environment.

Support:

Support = (X ∪ Y).count / n

Here the support count of item set X is denoted by X.count in a data set T and it has a number of n transactions

Horizontal database:

The transaction id (TID) item set format that is {TID : item set } and the item set is the set of items bought is transaction TID is known as horizontal data format

Table -1: Horizontal database view

Transaction id or TID	*Item sets*
1	*a,b,c,d,e*
2	*a,b*
3	*a,b,c,d*
4	*a,b,c*
5	*b,c*
6	*c,d,e*
7	*e*
8	*d,e*
9	*a,c,e*
10	*a,b,d*

3. IMPLEMENTATION

Based on Natural Selection

After an initial population is randomly generated, the algorithm evolves the through three operators:

1. **Selection** which equates to survival of the fitness;
2. **Crossover** which represents mating between individuals;
3. **Mutation** which introduces random modifications.

3.1 Selection Operator

- Key idea: give preference to better individuals, allowing them to pass on their genes to the next generation.
- The goodness of each individual depends on its fitness.
- Fitness may be determined by an objective function or by a subjective judgement.

3.2 Crossover Operator

- Prime distinguished factor of GA from other optimization techniques
- Two individuals are chosen from the population using the selection operator
- A crossover site along the bit strings is randomly chosen
- The values of the two strings are exchanged up to this point
- If S1=000000 and s2=111111 and the crossover point is 2 then S1'=110000 and s2'=001111
- The two new offspring created from this mating are put into the next generation of the population
- By recombining portions of good individuals, this process is likely to create even better individuals

3.3 Mutation Operator

- With some low probability, a portion of the new individuals will have some of their bits flipped.
- Its purpose is to maintain diversity within the population and inhibit premature convergence.
- Mutation alone induces a random walk through the search space
- Mutation and selection create a parallel, noise-tolerant, hill-climbing algorithms

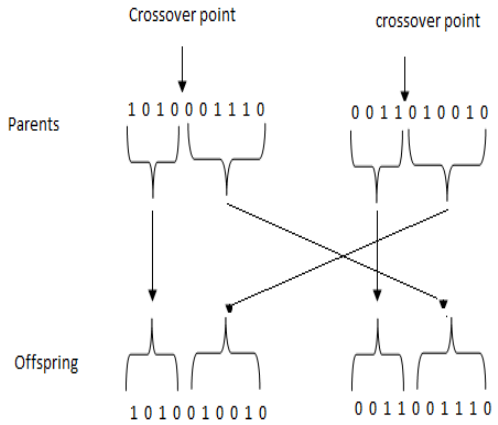

Fig -2: Crossover operation in between two bit springs

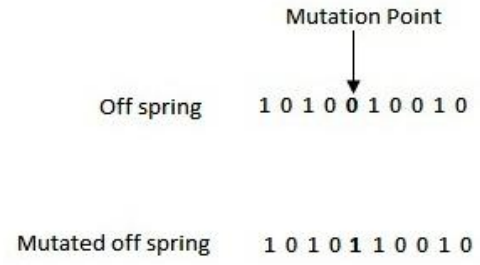

Fig -3: Single Point Crossover

4. USER INTERFACE DIAGRAM

Initially the user or administrator login. Then it checks whether the login process is valid or not. If valid, then assign a valid data set which has the item sets is displayed. Then enter the MINSUP count manually and the item sets are generated by using apriori algorithm and the most frequent item sets with the user defined MINSUP are displayed. Then for optimizing the generated frequent item sets, we first generate or select the chromosomes and apply cross over and mutation operation, then performs mutation on those child's to give the combinations in the form of positive and negative association rules

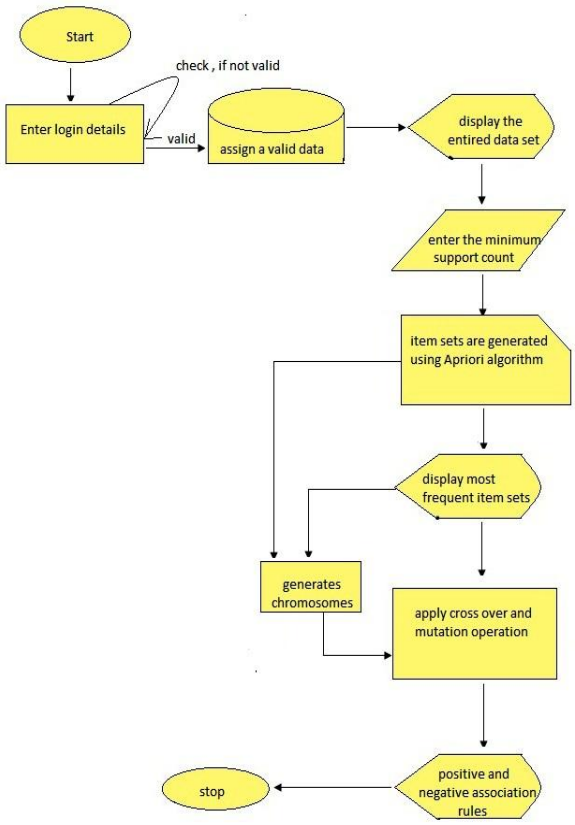

Fig - 4: User Interface Diagram for Association Mining Rules

4.RESULTS AND EXPLANATION

Table -2: Table 2: SUP COUNT and SUP values of L1 for item sets(a,b,c,d,e)

L1

	0.1		0.2		0.3		0.4		0.5		0.6	
	SUP COUNT	SUP	SUP COUNT	SUP	SUP COUNT	SUP	SUP COUNT	SUP	SUP COUNT	SUP	SUP COUNT	SUP
a	6	0.6	6	0.6	6	0.6	6	0.6	6	0.6	6	0.6
b	6	0.6	6	0.6	6	0.6	6	0.6	6	0.6	6	0.6
c	6	0.6	6	0.6	6	0.6	6	0.6	6	0.6	6	0.6
d	5	0.5	5	0.5	5	0.5	5	0.5	5	0.5	***	***
e	5	0.5	5	0.5	5	0.5	5	0.5	5	0.5	***	***

From table(2), let a,b,c,d,e are resources produced in data set and 0.1,0.2,0.3,0.4,0.5 and 0.6 are minimum support values which are obtained by using apriori and genetic algorithms. Now, the SUP COUNT , SUP values are 6,0.6 for minimum support value 0.1 which is shown above. Similarly all the resource values considered as the same. But in the last column, the values of both d,e are null represented by "***" because its SUP values are less than its minimum support value.

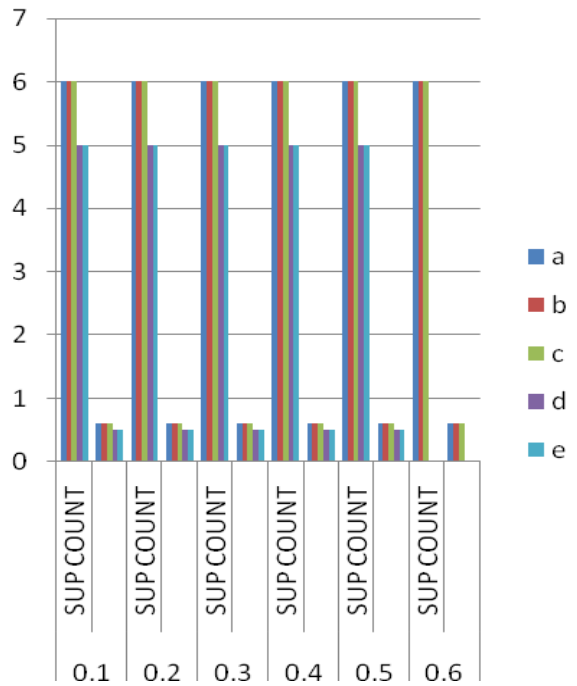

Chart -1: Varying in SUP count and SUP level values @ different ranges say 0.1 to 0.6

From chart (1), let 0.1, 0.2, 0.3, 0.4, 0.5, 0.6 are minimum support values. For 0.1, the SUP value is 0.6 which is shown on y-axis, similarly all the values considered the same and at 0.6, the SUP values of both d,e are less than the minimum support value, so it has no bar graphs.

Table -3: SUP COUNT and SUP values of L2 for item sets(a,b,c,d,e)

L2

	0.1		0.2		0.3		0.4		0.5	
	SUP COUNT	SUP	SUP COUNT	SUP	SUP COUNT	SUP	SUP COUNT	SUP	SUP COUNT	SUP
a,b	5	0.5	5	0.5	5	0.5	5	0.5	a,b	5
a,c	4	0.4	4	0.4	4	0.4	4	0.4	***	***
a,d	3	0.3	3	0.3	3	0.3	***	***	***	***
a,e	2	0.2	2	0.2	***	***	***	***	***	***
b,c	4	0.4	4	0.4	4	0.4	4	0.4	***	***
b,d	3	0.3	3	0.3	3	0.3	***	***	***	***
b,e	1	0.1	***	***	***	***	***	***	***	***
c,d	3	0.3	3	0.3	3	0.3	***	***	***	***
c,e	3	0.3	3	0.3	3	0.3	***	***	***	***
d,e	3	0.3	3	0.3	3	0.3	***	***	***	***

From table(3), the resources are {(a,b)(a,c) (a,d) (a,e)(b,c)(b,d)(b,e)(c,d)(c,e)(d,e)} and 0.1,0.2,0.3,0.4,0.5 and 0.6 are minimum support values. Here we didn't

mention 0.6 because its value is larger than SUP value of (a,b,c)

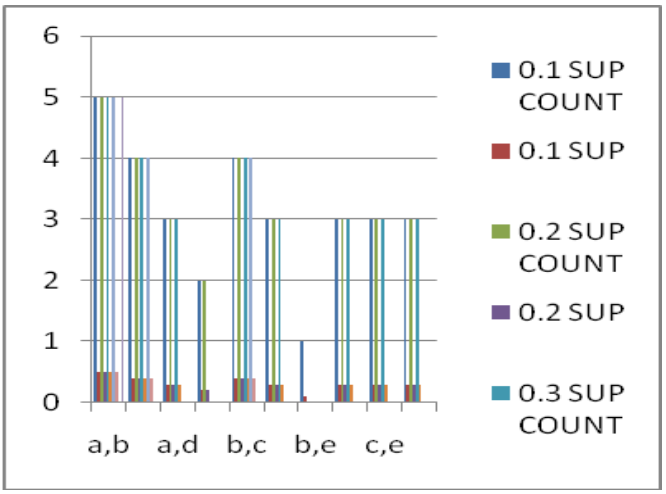

Chart -2: Varying in SUP count and SUP level values @ different ranges say 0.1 to 0.5

Let{(a,b)(a,c) (a,d) (a,e) (b,c) (b,d) (b,e) (c,d) (c,e) (d,e)} are resources. For resource (a,b), the SUP COUNT value is 5 and SUP value is 0.5. Similarly all values are considered like this only. At 0.2, for resource (b,e), the SUP value is 0.1 which is less than minimum support value i.e., 0.2, has no bar chart. Similarly same as remaining all the resources which are shown in graph as different colours.

Table -4: SUP COUNT and SUP values of L3 for item sets(a,b,c,d,e)

L3

	0.1		0.2		0.3	
	SUP COUNT	SUP	SUP COUNT	SUP	SUP COUNT	SUP
a,b,c	3	0.3	3	0.3	3	0.3
a,b,d	3	0.3	3	0.3	3	0.3
a,b,e	1	0.1	***	***	***	***
a,c,d	2	0.2	2	0.2	***	***
a,c,e	2	0.2	2	0.2	***	***
a,d,e	1	0.1	***	***	***	***

For L3, here {(a,b,c)(a,b,d)(a,b,e)(a,c,b)(a,c,e)(a,d,e)} are resource produced in data set and for resource "(a,b,c)" the SUP COUNT is 3 and SUP is 0.3. Here the "(a,b,c)" are repeated in three data set items. Similarly all values are considered like this.

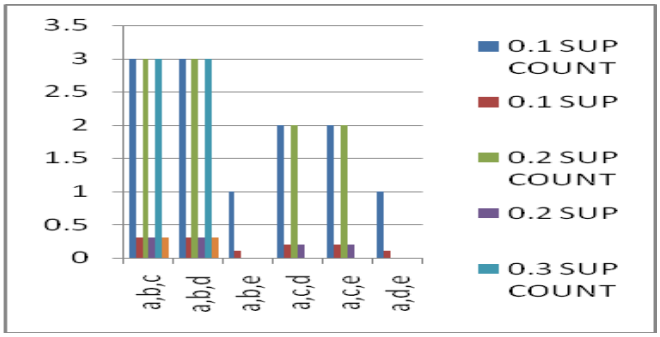

Chart -3: Varying in SUP count and SUP level values @ different ranges say 0.1 to 0.3

From chart (3), {(a,b,c) (a,b,d) (a,b,e) (a,c,b) (a,c,e) (a,d,e)} are resource produced in data set. The SUP COUNT ,SUP values are 3, 0.3 for "0.1" which is shown in graph as a bars on Y-axis,and same as remaining resources.

Table -5: SUP COUNT and SUP values of L4 for item sets(a,b,c,d,e)

L4

	0.1		0.2	
	SUP COUNT	SUP	SUP COUNT	SUP
a,b,c,d	2	0.2	a,b,c,d	2
a,b,c,e	1	0.1	***	***
a,b,d,e	1	0.1	***	***

For L4, here {(a,b,c,d)(a,b,c,e)(a,b,d,e)} are resources. For resource "(a,b,c,d)" the values of SUP COUNT is 2 and SUP is 0.2 are obtained. Similarly all the values are considered the same.

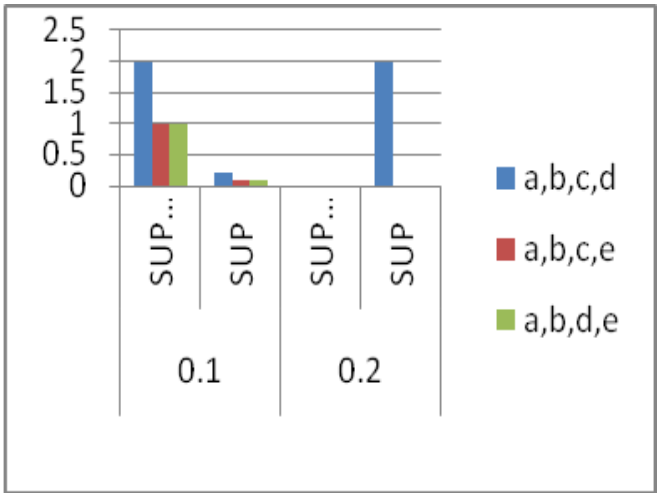

Chart -4: Varying in SUP count and SUP level values @ different ranges say 0.1 to 0.2

From the chart (4) let {(a,b,c,d)(a,b,c,e)(a,b,d,e)} are resources produced. Here clearly the lines noticed that at

0.1 the SUP COUNT for (a,b,c,d) is 2,1,1 and SUP is 0.2,0.1,0.1. Similarly for 0.2 , here it shows only single line that is, the SUP COUNT value is 2 for (a,b,c,d).

Table -6: SUP COUNT and SUP values of L5 for item sets(a,b,c,d,e)

L5

	0.1	
	SUP COUNT	**SUP**
a,b,c,d,e	1	

For L5, only the resource (a,b,c,d,e) is produced only once, so the SUP value is 1

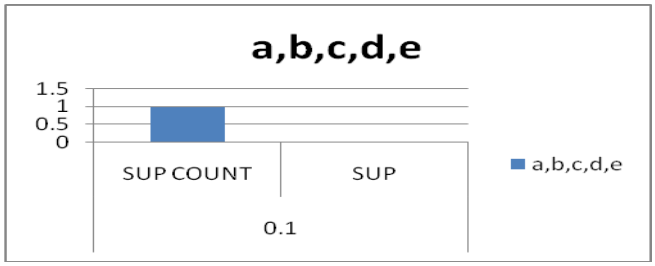

Chart -5: Varying in SUP count and SUP level values say 0.1

From chart (5) let 0.1 be a single point on X-axis. Here the SUP COUNT value is 1, hence it is shown as a single line only.

6. CONCLUSION

We designed efficient mining method by obtaining positive and negative association rules in database and optimization of positive and negative association rule using genetic algorithm, the design of pruning strategies for reducing the search space and improving the usability of mining rules are used to correlate to association with mining methods. The approach is effective, efficient and promising.

REFERENCES

[1] Sanat Jain, Swati Kabra," Mining & Optimization of Association Rules Using Effective Algorithm", International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, Volume 2, Issue 4, April 2012

[2] Paresh Tanna and Dr. Yogesh Ghodasara," Using Apriori with WEKA for Frequent Pattern Mining", International Journal of Engineering Trends and Technology (IJETT) – Volume 12 Number 3 – June ISSN: 2231-5381, 2014

[3] Aanum Shaikh , " Web Usage Mining Using Apriori and FP Growth Alogrithm", International Journal of Computer Science and Information Technologies, Vol. 6 (1) , ISSN:0975-9646,354-357, 2015

[4] J Karthikeyan and Dr. Udaykumar , " Modifed Bit-Apriori Algorithm for Frequent Item-Sets in Data Mining", Proc. of Int. Conf. on Advances in Information Technology and Mobile Communication, 2013

[5] Neelamadhab Padhy, Dr. Pragnyaban Mishra, and Rasmita Panigrahi," The Survey of Data Mining Applications And Feature Scope ", International Journal of Computer Science, Engineering and Information Technology (IJCSEIT), Vol.2, No.3, June 2012

[6] Rachna Somkunwar," A study on Various Data Mining Approaches of Association Rules", International Journal of Advanced Research in Computer Science and Software Engineering, pp. 141-144, Volume 2, Issue 9, ISSN: 2277 128X, September 2012

[7] T. Karthikeyan and N. Ravikumar," ASurvey on Association Rule Mining", International Journal of Advanced Research in Computer and Communication Engineering , Vol. 3, Issue 1, ISSN (Print) : 2319-5940, ISSN (Online) : 2278-1021 , January 2014

[8] Vipul Mangla, Chandni Sarda, SarthakMadra," Improving the efficiency of Apriori Algorithm in Data Mining", International Journal of Engineering and Innovative Technology (IJEIT), Volume 3, Issue 3, ISSN: 2277-3754 ISO 9001:2008, September 2013

[9] Jaishree Singh, Hari Ram, Dr. J.S. Sodhi ," Improving Efficiency of Apriori Algorithm Using Transaction Reduction ", International Journal of Scientific and Research Publications, Volume 3, Issue 1, ISSN 2250-3153, January 2013

[10] Richa Garg , Saurabh mittal Optimization by Genetic Algorithm", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 4, ISSN: 2277, April 2014.

[11] "Information Clustering Based Upon Rough Sets", International Journal of Scientific Engineering and Technology Research (IJSETR)", Volume 3 , Issue 41, Pages :.8330-8333,ISSN: 2319-8885, November 2014.

[12] Gunjan Verma, Vineeta Verma ," Role and Applications of Genetic Algorithm in Data Mining", International Journal of Computer Applications (0975 – 888), Volume 48– No.17, June 2012

[13] Sotiris Kotsiantis, Dimitris Kanellopoulos," Association Rules Mining: A Recent Overview", GESTS

International Transactions on Computer Science and Engineering, Vol.32 (1), pp. 71-82, 2006

[14] Adugna Fita," Multiobjective Programming With Continuous Genetic Algorithm", international journal of scientific & technology research volume 3, issue 9, issn 2277-8616, september 2014.

[15] Saheeda A, Shanavas K A, Sandra G, "A Method Based on Genetic Algorithm for Discrimination Discovery", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 7, ISSN: 2277 128X, July 2014.

[16] Shaily G.Langhnoja, Mehul P. Barot, Darshak B. Mehta , "Web Usage Mining Using Association Rule Mining on Clustered Data for Pattern Discovery ", International Journal of Data Mining Techniques and Applications, Volume 2, Issue 1, ISSN: 2278-2419, June 2013.

[17] Flora S. Tsai, "Network Intrusion Detection Using Association Rules", International Journal of Recent Trends in Engineering, Vol 2, No. 2, November 2009

[18] Manisha Thool, Prof. Preeti Voditel, "Association Rule Generation in Streams", International Journal of Advanced Research in Computer and Communication Engineering,Vol. 2, Issue 5, ISSN : 2278-102,1,May 2013.

[19] Prof. Paresh Tanna, Dr. Yogesh Ghodasara," Foundation for Frequent Pattern Mining Algorithms', International Journal of Computer Trends and Technology (IJCTT) – Volume 4 Issue 7 - July 2013, ISSN: 2231-2803.

[20] MasoumehVali ," Sub- Dividing Genetic Method for Optimization Problems.

[21] Ruby Yadav, Waseem Ahmad," Benchmark Function Optimization using Genetic Algorithm", Journal of Engineering, Computers & Applied Sciences (JEC&AS) ISSN No: 2319-5606, Volume 2, No.6, June 2013.

BIOGRAPHIES

Mr V Ramesh Babu, present he is pursuing M.Tech in Raghu Engineering College affiliated by JNTUK. He completed his B.Tech in Chaitanya Engineering College affiliated by JNTUk in 2008. Attended workshops and seminars on Data Mining

Mr Appala Raju Samanthula, working as an Assistant Professor in Raghu Engineering College since 3 years and fetching a total experience of 8 years. He had guided many UG and PG projects which are used in college/small firm levels also. He completed B.Tech in 2005 and M.Tech in 2010 and the certifications are from JNTU.