Modified Apriori Algorithm to find out Association Rules using Tree based Approach

Abhijit Sarkar[1], Apurba Paul[2], Sainik Kumar Mahata[3], Deepak Kumar[4]
[1, 2, 3 and 4]: Assistant Professor, Dept. of CSE, JIS College of Engineering, Kalyani, Nadia, West Bengal, India.

ABSTRACT

In the modern world, where data is in abundant, the problem of segregating data that leads to a specific purpose is the order of the hour. Traditionally we use data mining to achieve the purpose. A good example of the above mentioned problem can be found in a store. Generally, we use market basket analysis to find out which groups of products have a high chance of selling together. In this paper we present a new approach of segregating data by modifying the traditional Apriori algorithm. Since it is based on the traditional Apriori algorithm, the presented algorithm can be used for any number of data.

Keywords*:* Apriori algorithm, association rules, itemset, tree, support count.

1. INTRODUCTION

Data mining is the process of extracting patterns for a specific purpose from huge amount of data. With the help of data mining we can transform raw data into business intelligence. The very concept of data mining can be used to predict future trends and behaviors in business. In the general Apriori algorithm, we find out the 1-itemset, 2-itemset and so on to derive frequent grouping of data. After applying the steps of Apriori algorithm, we get the desired frequent grouping and then we apply association rules to find out desired rules that help us in developing marketing strategies. But, one disadvantage of this is that we get huge number of tables, and we require huge amount of space to store these tables, if the number of data is high. In this paper, we present an approach that, according to Apriori algorithm, help us in finding the frequent item sets but with less amount of tables. Thus the amount of space required to store the tables are considerably reduces. In our approach, we find out only the 1-itemset table and then we find out the frequent items using minimum support. Then we construct a tree using the 1-itemset table. The tree will contain a root node. The root node will consist of the frequent itemsets that was derived from the 1-itemset table. Then, we break the tree in to child nodes. The level of the tree is found out using the formula level=n-1, where n is the number of items in the root node of the tree. A child node will contain all possible combination of

(n-1)itemsets. We will traverse the tree using bottom-up approach. If a child node is infrequent then its parent node is also infrequent and rejected.

2. APRIORI ALGORITHM

1. k = 1
2. Find frequent set L_k from C_k of all candidate itemsets
3. Form C_{k+1} from L_k; k = k + 1
4. Repeat 2-3 until C_k is empty

- Details about steps 2 and 3

Step 2: scan *D* and count each itemset in C_k , if it's greater than minSup, it is frequent

Step 3:

- For k=1, C_1 = all 1-itemsets.
- For k>1, generate C_k from L_{k-1} as follows:
  - *The join step*

C_k = join of L_{k-1} with itself(itemset of size k)

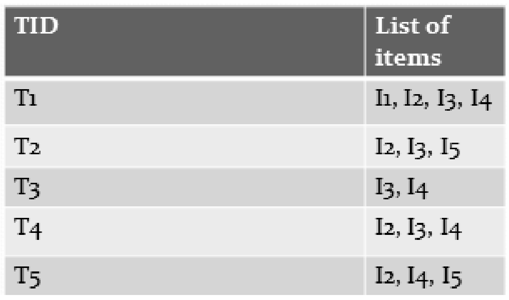
If both $\{a_1, \ldots, a_{k-2}, a_{k-1}\}$ & $\{a_1, \ldots, a_{k-2}, a_k\}$ are in L_{k-1}, then add $\{a_1, \ldots, a_{k-2}, a_{k-1}, a_k\}$ to C_k

(We keep items **sorted**).

  - *The prune step*

Remove $\{a_1, \ldots, a_{k-2}, a_{k-1}, a_k\}$ if it contains a non-frequent (k-1) subset

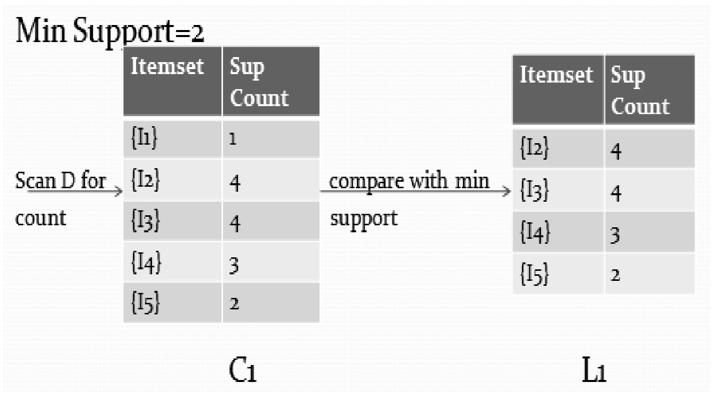
3. ILLUSTRATION OF APRIORI ALGORITHM



TID	List of items
T1	I1, I2, I3, I4
T2	I2, I3, I5
T3	I3, I4
T4	I2, I3, I4
T5	I2, I4, I5

Figure 1: Transaction database

Figure 1 shows the transaction database. It contains five different transactions and the items purchased.

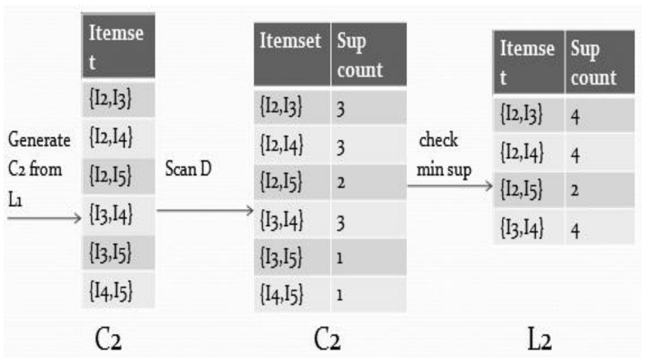


Itemset	Sup Count
{I1}	1
{I2}	4
{I3}	4
{I4}	3
{I5}	2

Itemset	Sup Count
{I2}	4
{I3}	4
{I4}	3
{I5}	2

Figure 2: Generation of candidate itemset and frequent 1-itemset

Figure 2 shows the support count of different 1-itemsets and the selected 1-itemsets based on minimum support 2.


Generate C2 from L1
Scan D
check min sup
C2
C2
L2

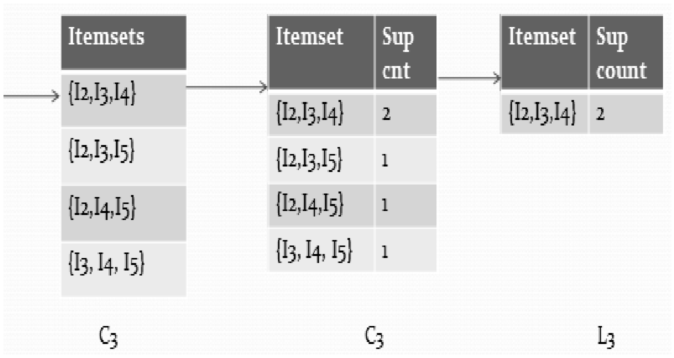
Itemset
{I2,I3}
{I2,I4}
{I2,I5}
{I3,I4}
{I3,I5}
{I4,I5}

Itemset	Sup count
{I2,I3}	3
{I2,I4}	3
{I2,I5}	2
{I3,I4}	3
{I3,I5}	1
{I4,I5}	1

Itemset	Sup count
{I2,I3}	4
{I2,I4}	4
{I2,I5}	2
{I3,I4}	4

Figure 3: Generation of candidate itemset and frequent 2-itemset

Above figure shows the support count of different 2-itemsets and the selected 2-itemsets based on minimum support 2.



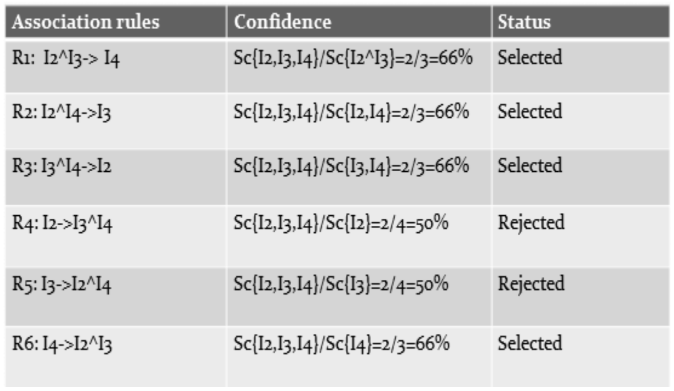
Itemsets
{I2,I3,I4}
{I2,I3,I5}
{I2,I4,I5}
{I3, I4, I5}

Itemset	Sup cnt
{I2,I3,I4}	2
{I2,I3,I5}	1
{I2,I4,I5}	1
{I3, I4, I5}	1

Itemset	Sup count
{I2,I3,I4}	2

Figure 4: Generation of candidate itemset and frequent 3-itemset

Above figure shows the support count of different 3-itemsets and the selected 3-itemsets based on minimum support 2.



Association rules	Confidence	Status
R1: I2^I3-> I4	Sc{I2,I3,I4}/Sc{I2^I3}=2/3=66%	Selected
R2: I2^I4->I3	Sc{I2,I3,I4}/Sc{I2,I4}=2/3=66%	Selected
R3: I3^I4->I2	Sc{I2,I3,I4}/Sc{I3,I4}=2/3=66%	Selected
R4: I2->I3^I4	Sc{I2,I3,I4}/Sc{I2}=2/4=50%	Rejected
R5: I3->I2^I4	Sc{I2,I3,I4}/Sc{I3}=2/4=50%	Rejected
R6: I4->I2^I3	Sc{I2,I3,I4}/Sc{I4}=2/3=66%	Selected

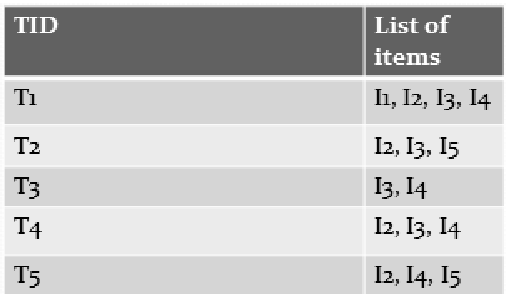
Figure 5: List of selected association rules based on confidence = 60%

Above figure shows the different possible association rules and selection among them based on confidence = 60%

4. TREE BASED APPROACH ALGORITHM

1. Support count of 1-itemset are checked and based on the minimum support threshold, some of the itemsets are selected. The selected itemsets are taken together to form a set S, of length n, where n is the number of items.
2. The set S is considered as the root node of the tree.
3. The child nodes contain all possible (n-1) length combinations, where n is the length of immediate parent node.
4. Following the bottom-up approach, we calculate the frequency of the itemsets in the leaf nodes.
5. If the frequency of the itemsets in the leaf node is less than the specified minimum support threshold, then the leaf node as well as its immediate parent node is rejected.
6. After completing the above steps, we get the final tree from which the selected nodes give the possible association rules.

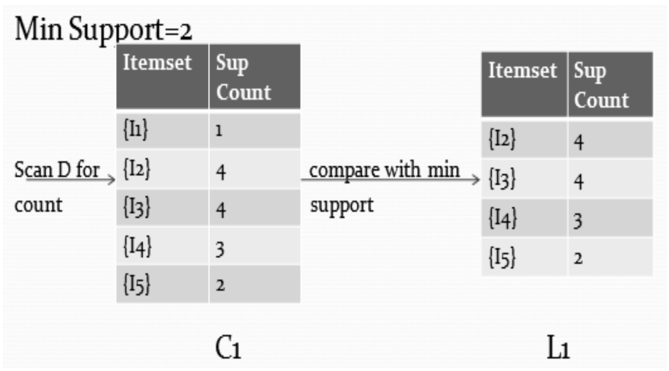
5. TREE BASED APPROACH ALGORITHM ILLUSTRATION



TID	List of items
T1	I1, I2, I3, I4
T2	I2, I3, I5
T3	I3, I4
T4	I2, I3, I4
T5	I2, I4, I5

Figure 6: Transaction database

Fig 6 shows the transaction database. It contains five different transactions and the items purchased.


Min Support=2
Scan D for count
compare with min support
C1
L1

Itemset	Sup Count
{I1}	1
{I2}	4
{I3}	4
{I4}	3
{I5}	2

Itemset	Sup Count
{I2}	4
{I3}	4
{I4}	3
{I5}	2

Figure 7: Generation of candidate itemset and frequent 1-itemset

Figure 7 shows the support count of different 1-itemsets and the selected 1-itemsets based on minimum support 2.

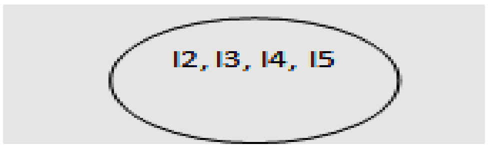


Figure 8: Root of the tree using set S.

The above figure shows the root of the tree. This node is created using the rule 1 and 2 of the Tree Based Approach Algorithm.

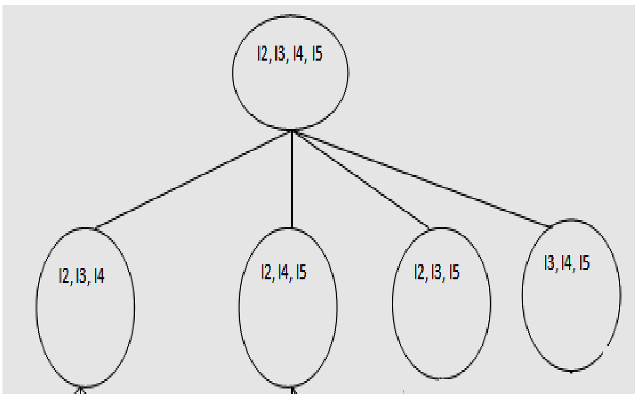


Figure 9: Two levels of the tree.

The above figure shows the child nodes of the root. This is done in accordance with steps 3 of the Tree Based Approach Algorithm.

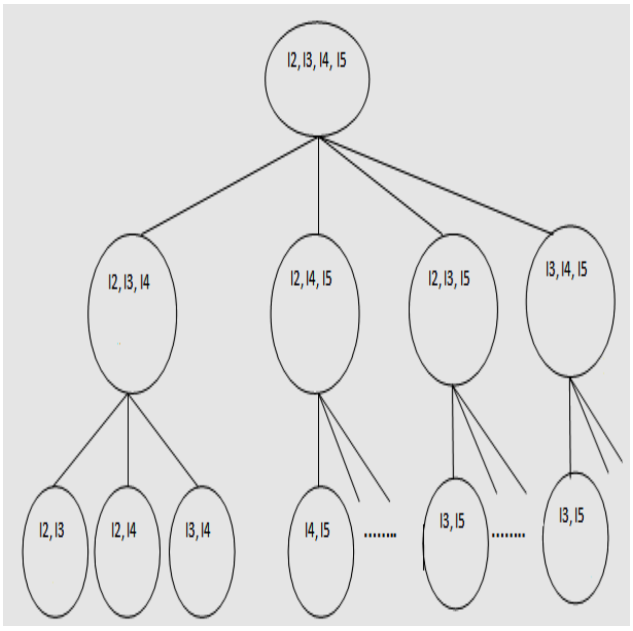


Figure 10: Three levels of the tree.

The above figure shows the child nodes of the root. This is done in accordance with steps 3 of the Tree Based Approach Algorithm

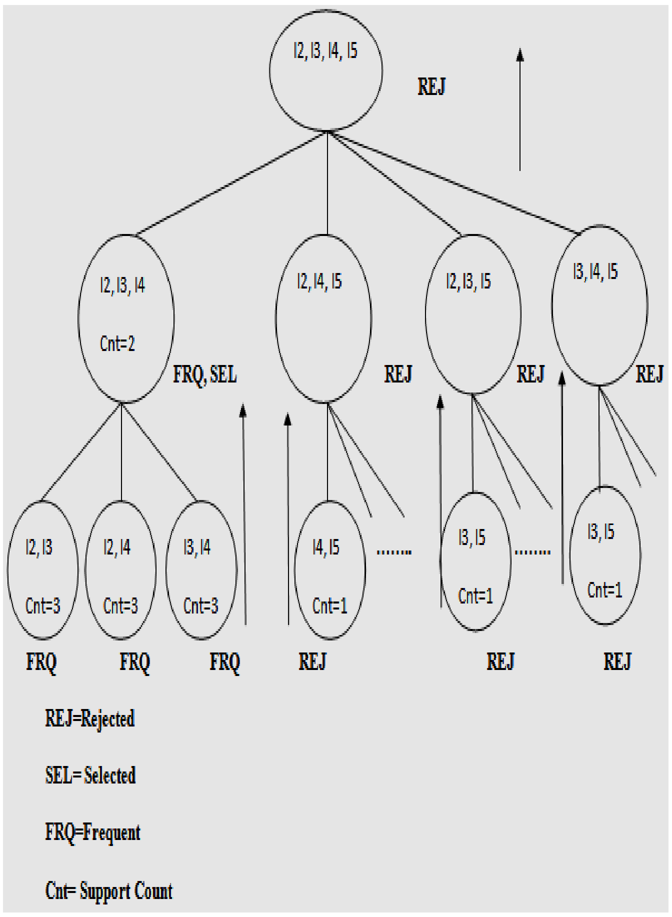


Figure 11: Tree based final selection and rejection of frequent itemset

When we consider the child nodes with the items (I2,I3), (I2,I4), (I3,I4), we see that the frequency of these itemsets exceeds the specified minimum support threshold 2. So all of them are frequent and as a result, its immediate parent node is selected.

When we consider the child nodes (I4,I5), (I3,I5), (I3,I5) of three different immediate parents, we see that these child nodes are not frequent with respect to the minimum support threshold value. So the immediate parents are rejected.

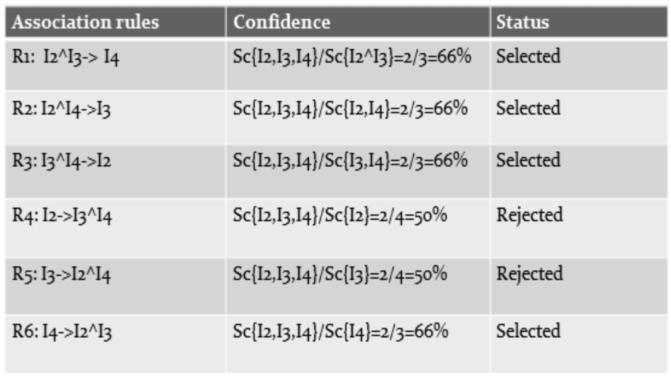
As the child nodes (I2,I4,I5), (I2,I3,I5) and (I3,I4,I5) are also rejected, the immediate parent node (I2,I3,I4,I5), of this child is also rejected.

The only selected itemset is (I2,I3,I4) This is shown in the figure 11.

So based on the above found result, we create the association rules. Based on one arbitrary confidence threshold value = 60%, we get four different association rules.

Compared with the conventional approach, we are not counting the support count of (I2,I4,I5), (I2,I3,I5), (I3,I4,I5). So there will be less database scanning compared to the existing Apriori algorithm.

The association rules, which are selected and rejected are shown in figure 12, given below.



Association rules	Confidence	Status
R1: I2^I3-> I4	Sc{I2,I3,I4}/Sc{I2^I3}=2/3=66%	Selected
R2: I2^I4->I3	Sc{I2,I3,I4}/Sc{I2,I4}=2/3=66%	Selected
R3: I3^I4->I2	Sc{I2,I3,I4}/Sc{I3,I4}=2/3=66%	Selected
R4: I2->I3^I4	Sc{I2,I3,I4}/Sc{I2}=2/4=50%	Rejected
R5: I3->I2^I4	Sc{I2,I3,I4}/Sc{I3}=2/4=50%	Rejected
R6: I4->I2^I3	Sc{I2,I3,I4}/Sc{I4}=2/3=66%	Selected

Figure 12: List of selected association rules based on confidence = 60%

6. CONCLUSION

In the Apriori algorithm one of the major disadvantages is that we have to access the database very frequently, which increases the time complexity of the algorithm. Keeping this thing in mind we have developed the algorithm to minimize the time complexity, which gives us a good result.

7. REFERENCES

[1] Han J,Kamber M.Data Mining:Concepts and Techniques.Higher Education Press,2001.

[2] Jong S P,Ming S C,Philip S Y.An effective hash based algorithm for mining association rules.In Proceedings of the 2005 ACM SIGMOD International Conference On Management of Data.2005,24(2): 175-186.

[3] Han, Pei,Y Yin and R Mao.Mining Frequent Patterns without Candidate Generation:A Frequent-Pattern Tree Approach. Data Mining and Knowledge Discovery,2004,8:53-87.

[4] Kong Fang , Qian Xue-zhong,Research of improved apriori algorithm in mining association rules,Computer Engineering and Design, 2008,v29, n16, p4220-4223.

[5] Li Qingzhong , Wang Haiyang, Yan Zhongmin, Efficient mining of association rules by reducing the number of passes over the database, Computer Science and Technology,2008,p 182-188.