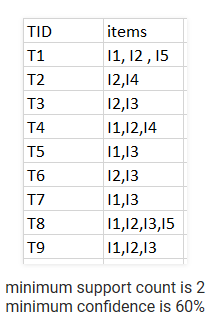
**Apriori Algorithm**

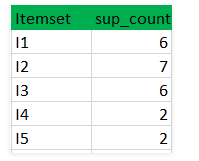
Prerequisite – Frequent Item set in Data set (Association Rule Mining) Apriori algorithm is given by R. Agrawal and R. Srikant in 1994 for finding frequent item sets in a dataset for Boolean association rule. Name of algorithm is Apriori is because it uses prior knowledge of frequent itemset properties. We apply a iterative approach or level-wise search where k-frequent itemsets are used to find k+1 itemsets.

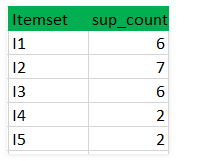
To improve the efficiency of level-wise generation of frequent itemsets an important property is used called Apriori property which helps by reducing the search space.

**Apriori Property**

All nonempty subset of frequent itemset must be frequent. The key concept of Apriori algorithm is its anti-monotonicity of support measure. Apriori assumes that..

Before we start understanding algorithm go through some definitions which are explained in my previous post.  
Consider the following dataset and we will find frequent itemsets and generate association rules on this.

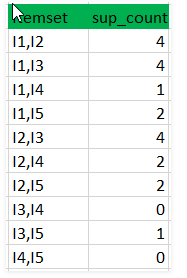
**Step-1:** K=1  
(I) Create a table containing support count of each item present in dataset – Called **C1 (candidate set)**

(II) compare candidate set item’s support count with minimum support count(here min\_support=2 if support\_count of candidate set items is less than min\_support then remove those items) this gives us itemset L1.

Step-2: K=2

Generate candidate set C2 using L1 (this is called join step). Condition of joining is Lk-1 and Lk-1 is that it should have (K-2) elements in common.

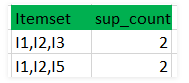
Check all subsets of a itemset are frequent or not and if not frequent remove that itemset.(Example subset of{I1, I2} are {I1}, {I2} they are frequent. Check for each itemset)

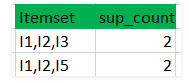
 Now find support count of these itemsets by searching in dataset.

**Step-3:**

Generate candidate set C3 using L2 (join step). Condition of joining Lk-1 and Lk-1 is it should have (K-2) elements in common. So here for L2 first element should match.  
So itemset generated by joining L2 is {I1, I2, I3} {I1, I2, I5} {I1, I3, i5} {I2, I3, I4} {I2, I4, I5} {I2, I3, I5}

Check all subsets of these itemsets are frequent or not and if not remove that itemset. (Here subset of {I1, I2, I3} are {I1, I2} {I2, I3} {I1, I3} which are frequent. For {I2, I3, I4} subset {I3, I4} is not frequent so remove this. Similarly check for every itemset)

Find support count of these remaining itemset by searching in dataset.

(II) Compare candidate (C3) support count with minimum support count(here min\_support=2 if support\_count of candidate set item is less than min\_support then remove those items) this gives us itemset L3

**Step-4:**

Generate candidate set C4 using L3 (join step). Condition of joining Lk-1 and Lk-1 (K=4) is these should have (K-2) elements in common. So here for L3 first 2 element (items) should match.

Check all subsets of these itemsets are frequent or not (Here itemset formed by joining L3 is {I1, I2, I3, I5} so its subset contain {I1, I3, I5} which is not frequent). So no itemset in C4

We stop here because no frequent itemset are found frequent further

Thus we discovered all frequent item-sets now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule.

Confidence –

A confidence of 60% means that 60% of the customers who purchased a milk and bread also bought the butter.

So here by taking example of any frequent itemset we will show rule generation.  
Itemset {I1, I2, I3} //from L3  
SO rules can be  
[I1^I2]=>[I3] //confidence = sup(I1^I2^I3)/sup(I1^I2) = 2/4\*100=50%  
[I1^I3]=>[I2] //confidence = sup(I1^I2^I3)/sup(I1^I3) = 2/4\*100=50%  
[I2^I3]=>[I1] //confidence = sup(I1^I2^I3)/sup(I2^I3) = 2/4\*100=50%  
[I1]=>[I2^I3] //confidence = sup(I1^I2^I3)/sup(I1) = 2/6\*100=33%  
[I2]=>[I1^I3] //confidence = sup(I1^I2^I3)/sup(I2) = 2/7\*100=28%  
[I3]=>[I1^I2] //confidence = sup(I1^I2^I3)/sup(I3) = 2/6\*100=33%

So if minimum confidence is 50 % first 3 rules can be considered strong association rules