Mining Frequent Itemsets with Candidate Generation

Basic terms of association rule mining

Item set: A collection of one or more items.

Customers	**Transactions**
1	Milk, bread
2	Bread, butter
3	Beer
4	Milk, bread, butter
5	Bread
6	Milk, bread, butter

Support count : it represent frequency of occurrence of an item set.

Eg. Support count { milk, bread} = 3

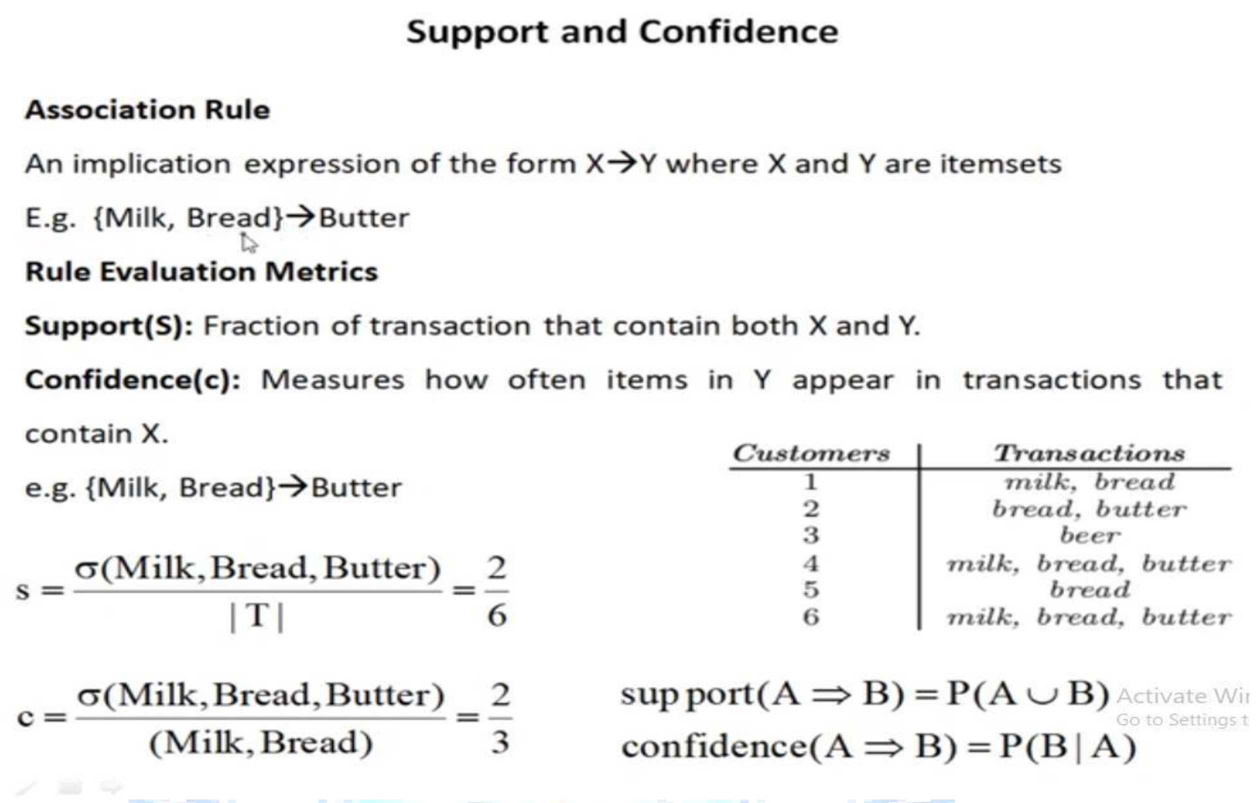
Support(S) :fraction of transaction that contains item set.

Eg. S{milk,bread} = 3/6

Frequent item sets: An item set whose support is greater than or equal to a minimum support threshold.

If minimum support threshold = 3 then {milk,bread}are frequent

If minimum support threshold = 4 then {milk,bread}are not frequent



Support and Confidence

Association Rule

An implication expression of the form X→Y where X and Y are itemsets

E.g. {Milk, Bread}→Butter

Rule Evaluation Metrics

Support(S): Fraction of transaction that contain both X and Y.

Confidence(c): Measures how often items in Y appear in transactions that contain X.

e.g. {Milk, Bread}→Butter

Customers	*Transactions*
1	*milk, bread*
2	*bread, butter*
3	*beer*
4	*milk, bread, butter*
5	*bread*
6	*milk, bread, butter*

$$s = \frac{\sigma(\text{Milk}, \text{Bread}, \text{Butter})}{|T|} = \frac{2}{6}$$

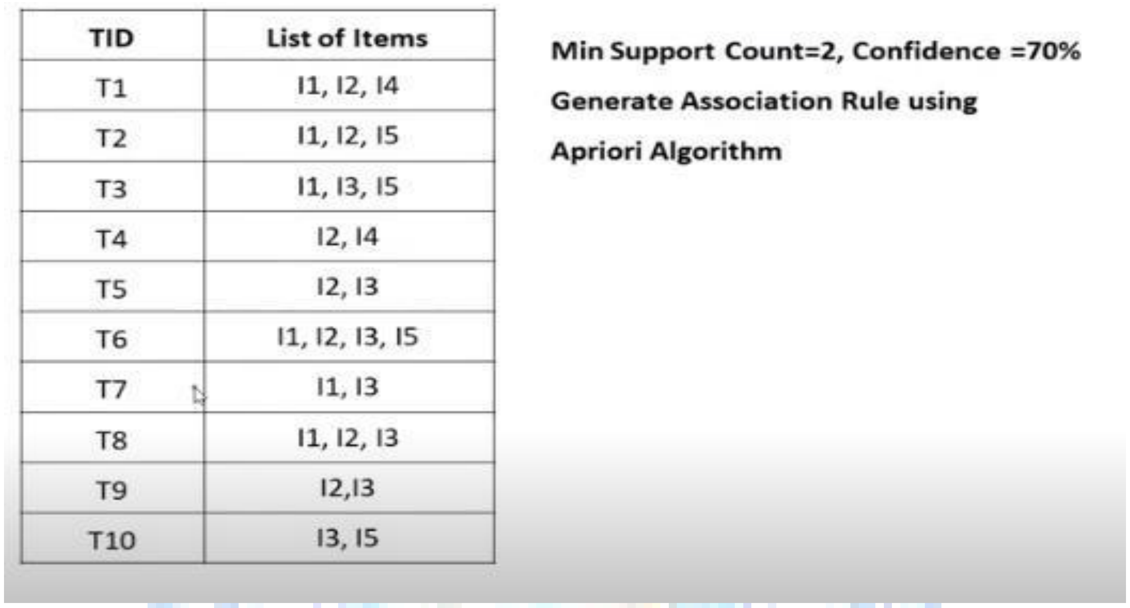
$$c = \frac{\sigma(\text{Milk}, \text{Bread}, \text{Butter})}{(\text{Milk}, \text{Bread})} = \frac{2}{3}$$

$$\text{sup port}(A \Rightarrow B) = P(A \cup B)$$

$$\text{confidence}(A \Rightarrow B) = P(B \mid A)$$

Apriori Algorithm

- Apriori Algorithm is an influential algorithm for mining frequent itemsets for boolean association rules.
- It uses **prior knowledge** of frequent itemset properties (Aprior).
- It uses K frequent itemsets to find K+1 itemsets.
- It is based on three concept: **Frequent itemset, Apriori property and join operations**

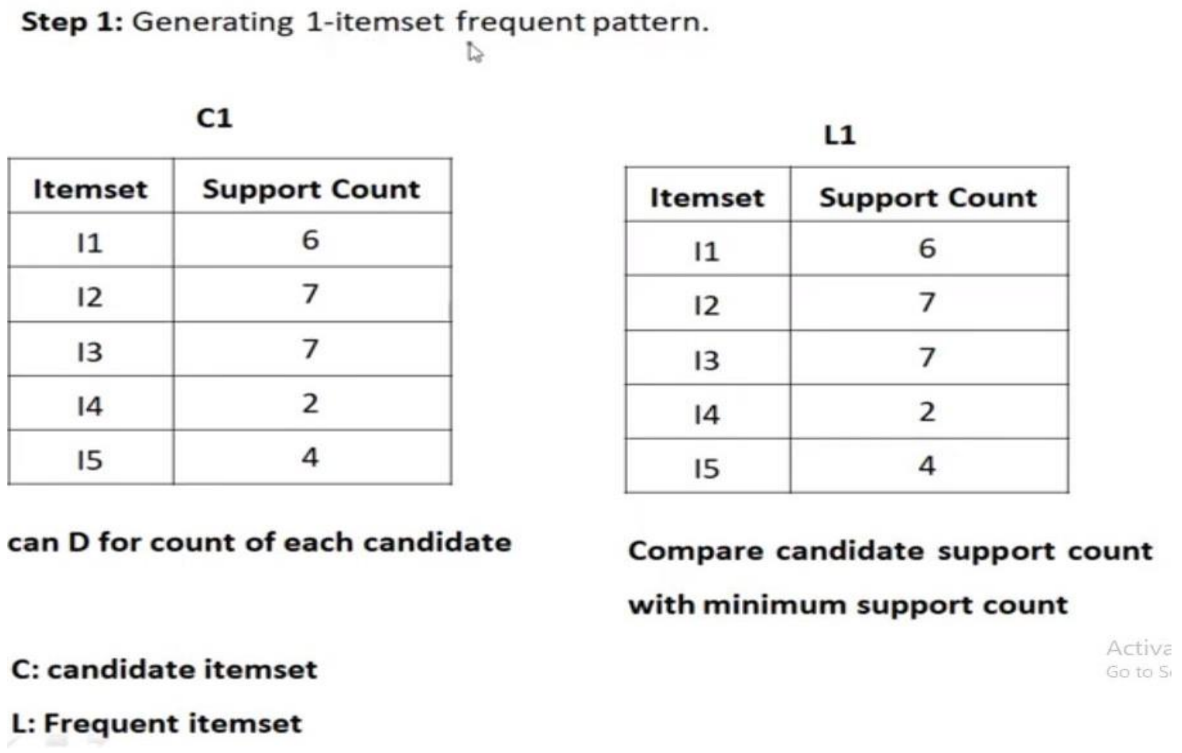


TID	List of Items
T1	I1, I2, I4
T2	I1, I2, I5
T3	I1, I3, I5
T4	I2, I4
T5	I2, I3
T6	I1, I2, I3, I5
T7	I1, I3
T8	I1, I2, I3
T9	I2,I3
T10	I3, I5

Min Support Count=2, Confidence =70%

Generate Association Rule using

Apriori Algorithm



Step 1: Generating 1-itemset frequent pattern.

C1

Itemset	Support Count
I1	6
I2	7
I3	7
I4	2
I5	4

can D for count of each candidate

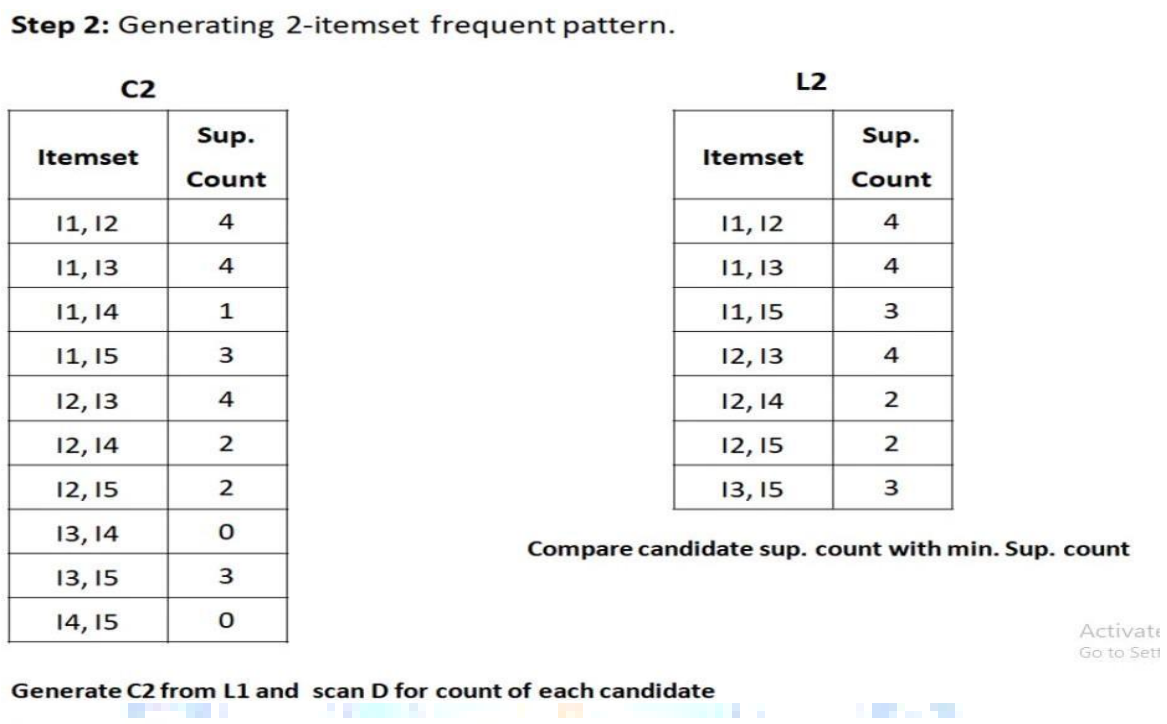
L1

Itemset	Support Count
I1	6
I2	7
I3	7
I4	2
I5	4

Compare candidate support count with minimum support count

C: candidate itemset

L: Frequent itemset



Step 2: Generating 2-itemset frequent pattern.

C2

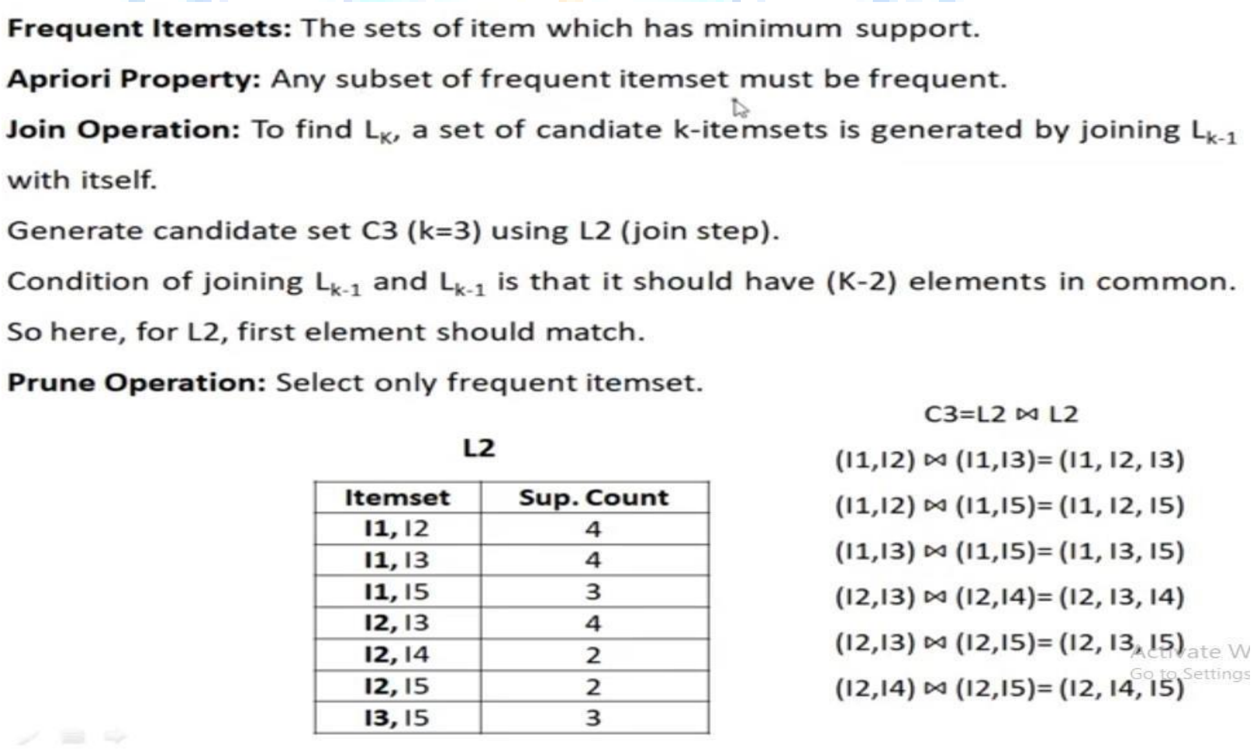
Itemset	Sup. Count
I1, I2	4
I1, I3	4
I1, I4	1
I1, I5	3
I2, I3	4
I2, I4	2
I2, I5	2
I3, I4	0
I3, I5	3
I4, I5	0

L2

Itemset	Sup. Count
I1, I2	4
I1, I3	4
I1, I5	3
I2, I3	4
I2, I4	2
I2, I5	2
I3, I5	3

Compare candidate sup. count with min. Sup. count

Generate C2 from L1 and scan D for count of each candidate



Frequent Itemsets: The sets of item which has minimum support.

Apriori Property: Any subset of frequent itemset must be frequent.

Join Operation: To find L_K, a set of candiate k-itemsets is generated by joining L_{k-1} with itself.

Generate candidate set C3 (k=3) using L2 (join step).

Condition of joining L_{k-1} and L_{k-1} is that it should have (K-2) elements in common. So here, for L2, first element should match.

Prune Operation: Select only frequent itemset.

L2

Itemset	Sup. Count
I1, I2	4
I1, I3	4
I1, I5	3
I2, I3	4
I2, I4	2
I2, I5	2
I3, I5	3

C3=L2 $\bowtie$ L2

(I1,I2) $\bowtie$ (I1,I3)= (I1, I2, I3)

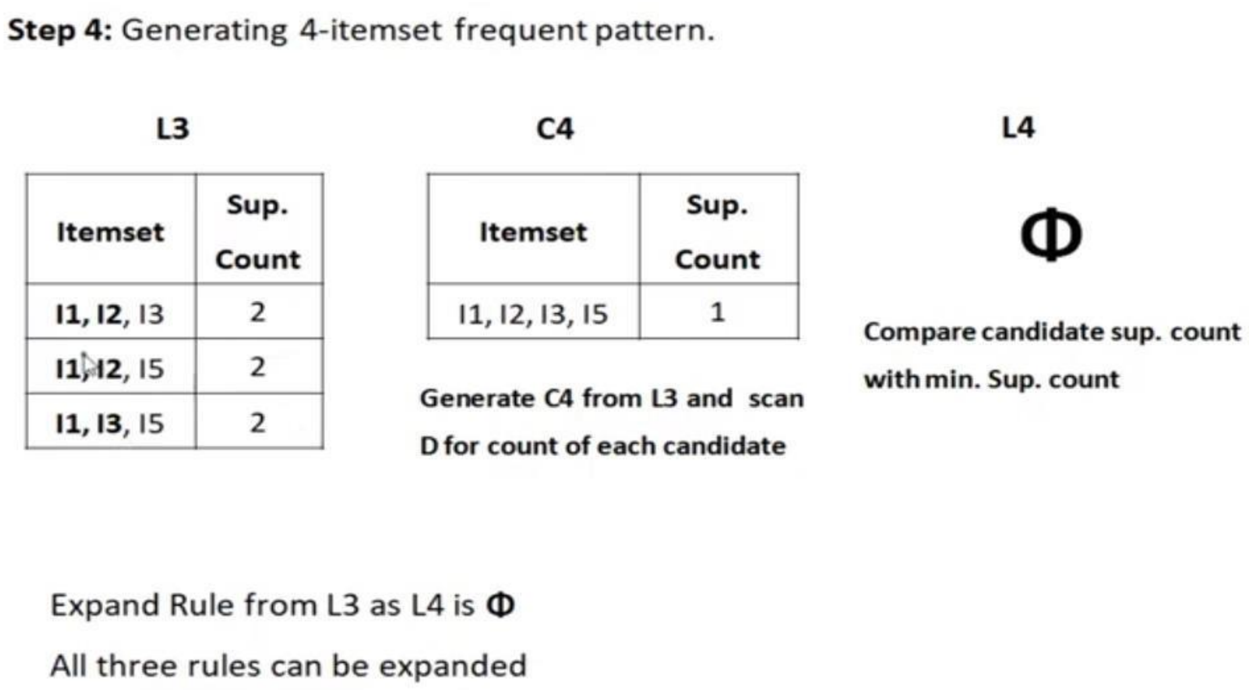
(I1,I2) $\bowtie$ (I1,I5)= (I1, I2, I5)

(I1,I3) $\bowtie$ (I1,I5)= (I1, I3, I5)

(I2,I3) $\bowtie$ (I2,I4)= (I2, I3, I4)

(I2,I3) $\bowtie$ (I2,I5)= (I2, I3, I5)

(I2,I4) $\bowtie$ (I2,I5)= (I2, I4, I5)



Step 4: Generating 4-itemset frequent pattern.

L3

Itemset	Sup. Count
I1, I2, I3	2
I1, I2, I5	2
I1, I3, I5	2

C4

Itemset	Sup. Count
I1, I2, I3, I5	1

Generate C4 from L3 and scan D for count of each candidate

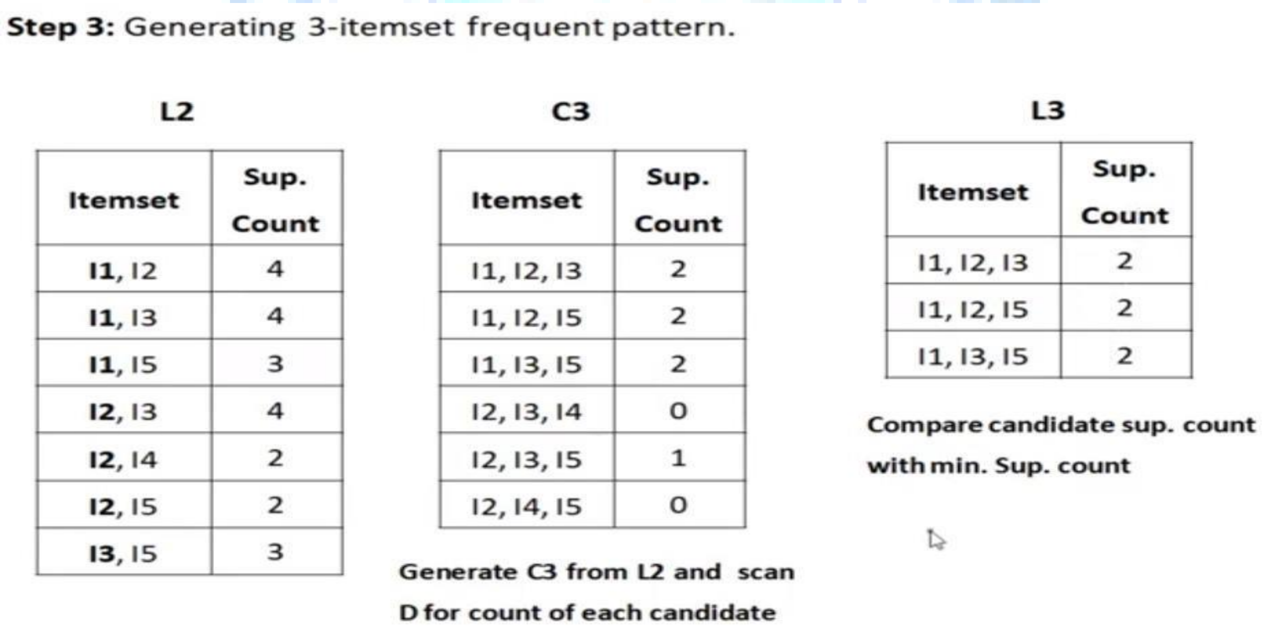
L4

Φ

Compare candidate sup. count with min. Sup. count

Expand Rule from L3 as L4 is Φ

All three rules can be expanded



Step 3: Generating 3-itemset frequent pattern.

L2

Itemset	Sup. Count
I1, I2	4
I1, I3	4
I1, I5	3
I2, I3	4
I2, I4	2
I2, I5	2
I3, I5	3

C3

Itemset	Sup. Count
I1, I2, I3	2
I1, I2, I5	2
I1, I3, I5	2
I2, I3, I4	0
I2, I3, I5	1
I2, I4, I5	0

Generate C3 from L2 and scan D for count of each candidate

L3

Itemset	Sup. Count
I1, I2, I3	2
I1, I2, I5	2
I1, I3, I5	2

Compare candidate sup. count with min. Sup. count

Step 4: Generating 4-itemset frequent pattern.

L3

Itemset	Sup. Count
I1, I2, I3	2
I1, I2, I5	2
I1, I3, I5	2

C4

Itemset	Sup. Count
I1, I2, I3, I5	1

Generate C4 from L3 and scan D for count of each candidate

L4

Φ

Compare candidate sup. count with min. Sup. count

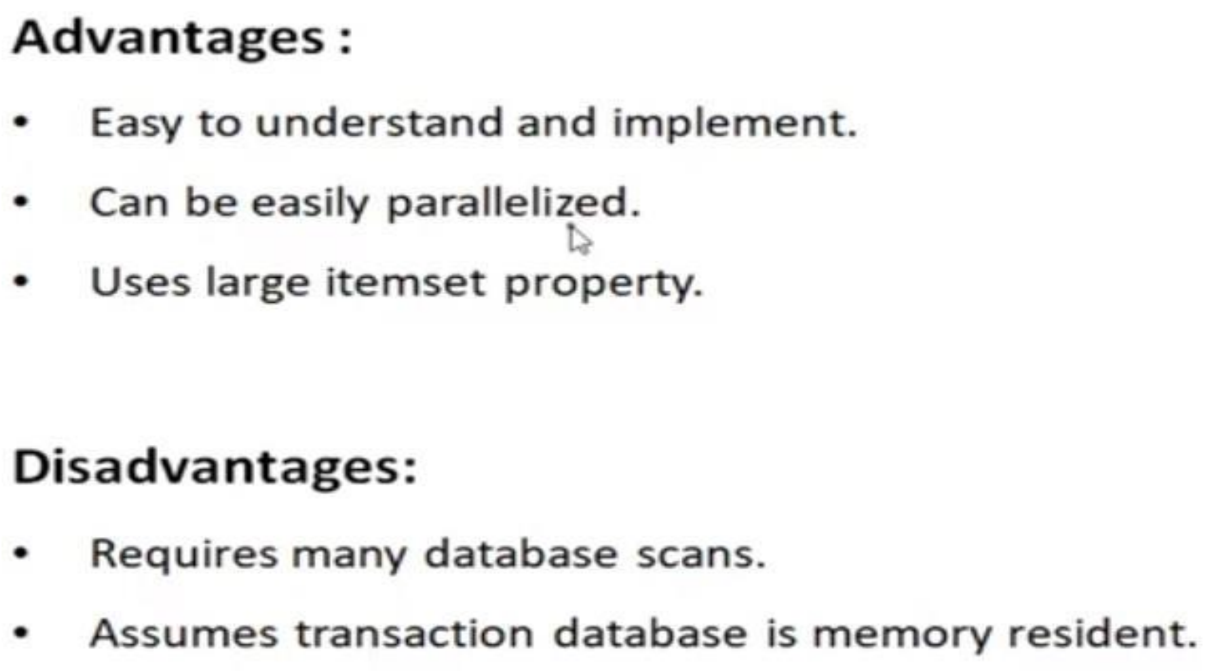
Expand Rule from L3 as L4 is **Φ**

All three rules can be expanded

We can expand any rule.
For e.g. I1, I2 and I5

Confidence =70%

Association Rule	Confidence	Confidence (%)
I1^I2→I5	C(I1,I2,I5)/C(I1,I2)=2/4	50%
I1^I5→I2	C(I1,I2,I5)/C(I1,I5)=2/2	100%
I2^I5→I1	C(I1,I2,I5)/C(I2,I5)=2/2	100%
I1→I2^I5	C(I1,I2,I5)/C(I1)=2/6	33%
I2→I1^I5	C(I1,I2,I5)/C(I2)=2/7	29%
I5→I1^I2	C(I1,I2,I5)/C(I5)=2/2	100%



Advantages :

- Easy to understand and implement.
- Can be easily parallelized.
- Uses large itemset property.

Disadvantages:

- Requires many database scans.
- Assumes transaction database is memory resident.

What is Apriori Algorithm?

- Apriori algorithm refers to an algorithm that is used in mining frequent products sets and relevant association rules.
- Generally, the apriori algorithm operates on a database containing a huge number of transactions. For example, the items customers but at a Big Bazar.
- Apriori algorithm helps the customers to buy their products with ease and increases the sales performance of the particular store

Components of Apriori algorithm

The given three components comprise the apriori algorithm.

1. Support
2. Confidence
3. Lift

Let's take an example to understand this concept. You need a huge database containing a large number of transactions. Suppose you have 4000 customers transactions in a Big Bazar. You have to calculate the Support, Confidence, and Lift for two products, and you may say Biscuits and Chocolate. This is because customers frequently buy these two items together.

Out of 4000 transactions, 400 contain Biscuits, whereas 600 contain Chocolate, and these 600 transactions include a 200 that includes Biscuits and chocolates. Using this data, we will find out the support, confidence, and lift.

Support

Support refers to the default popularity of any product. You find the support as a quotient of the division of the number of transactions comprising that product by the total number of transactions. Hence, we get

Support (Biscuits) = (Transactions relating biscuits) / (Total transactions)

= 400/4000 = 10 percent.

Confidence

Confidence refers to the possibility that the customers bought both biscuits and chocolates together. So, you need to divide the number of transactions that comprise both biscuits and chocolates by the total number of transactions to get the confidence.

Hence, Confidence = (Transactions relatingboth biscuits and Chocolate) / (Total transactions involving Biscuits)

= 200/400

= 50 percent.

It means that 50 percent of customers who bought biscuits bought chocolates **also.**

Lift

Consider the above example; lift refers to the increase in the ratio of the sale of chocolates when you sell biscuits. The mathematical equations of lift are given below.

Lift = (Confidence (Biscuits - chocolates)/ (Support (Biscuits)

= 50/10 = 5

It means that the probability of people buying both biscuits and chocolates together is five times more than that of purchasing the biscuits alone. If the lift value is below one, it requires that the people are unlikely to buy both the items together. Larger the value, the better is the combination.