Association Rules:

1) Association rules are the ones that help us analyze the relationship between the different things in a particularly large data set.
2) They help us with understanding correlation between the different data items on the basis of the frequency of occurrence, patterns, etc.
3) They can also be thought of as the if- then statements that can be useful in determining the probability of relationship between two distinct data items.
4) An association rule as an implication can be shown as given below:
   X => Y, i.e, X implies Y.
   LHS = X = antecedent
   RHS = Y = consequent
5) A very naive example of an Association rule can be {onion, potato} => {tomato}
   The above example tells us that : if people buy onion and potatoes together then, they also purchase tomatoes. In such a way we can find correlation between the the sales and transaction data.
6) 'If' part in the association rule = antecedent
   'then' part in the association rule = consequent
7) **How are the association rules created ?**

The association rules created by searching and gathering if - then patterns using criterias like:

support = indication of how frequently the items appear in the data.
confidence = gives us the number of if - the associations that are actually true.
lift = it is a metric that can be used to compare confidence with the expected confidence

---

Rule	Support	Confidence	Lift
$A \Rightarrow D$	2/5	2/3	10/9
$C \Rightarrow A$	2/5	2/4	5/6
$A \Rightarrow C$	2/5	2/3	5/6
$B\&C \Rightarrow D$	1/5	1/3	5/9

The image above shows us how exactly support, confidence and lift are calculated using the following formulae:

For n association rule : X => Y

Support = freq(X, Y) / N ; N being the no. of transactions or the no. of occurrence of that particular event / item

Confidence = freq(X, Y) / freq(X)

lift = Support / (support(X) * support(Y))

- if the denominator of lift is more then, it tells us that the occurrence of randomness is more rather than the occurrence due to any association.

8) Association rules are quite popularly used in the market basket analysis.
9) According to me, In the the context of phrase mining, the association rules will help us to predict what is the set of words that is most likely to appear if a particular group of words has appeared priorly in a sentence.

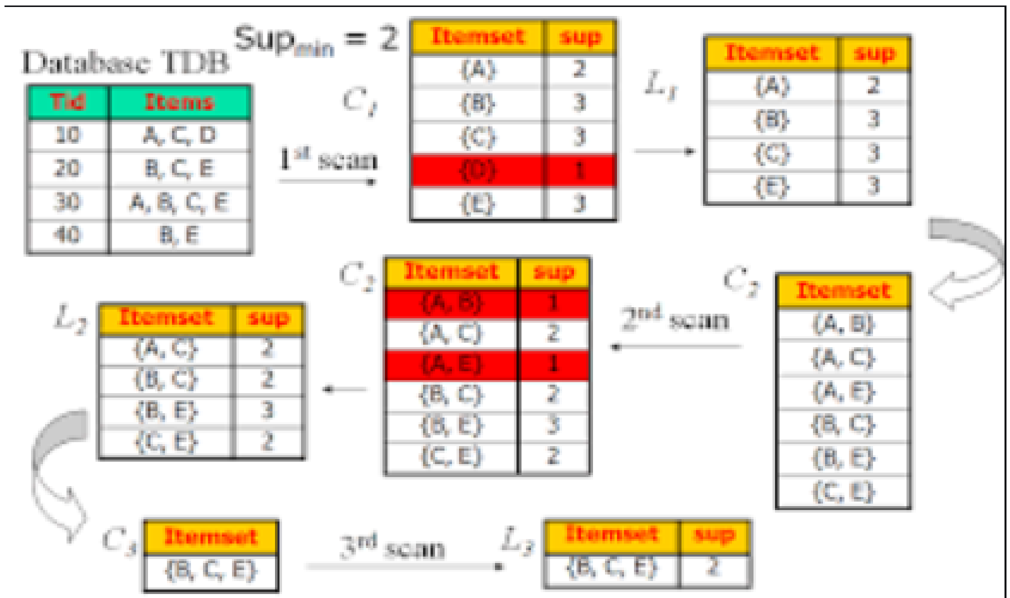
By doing this our machine can learn to capture or anticipate the phrases that are most likely to come, beforehand.

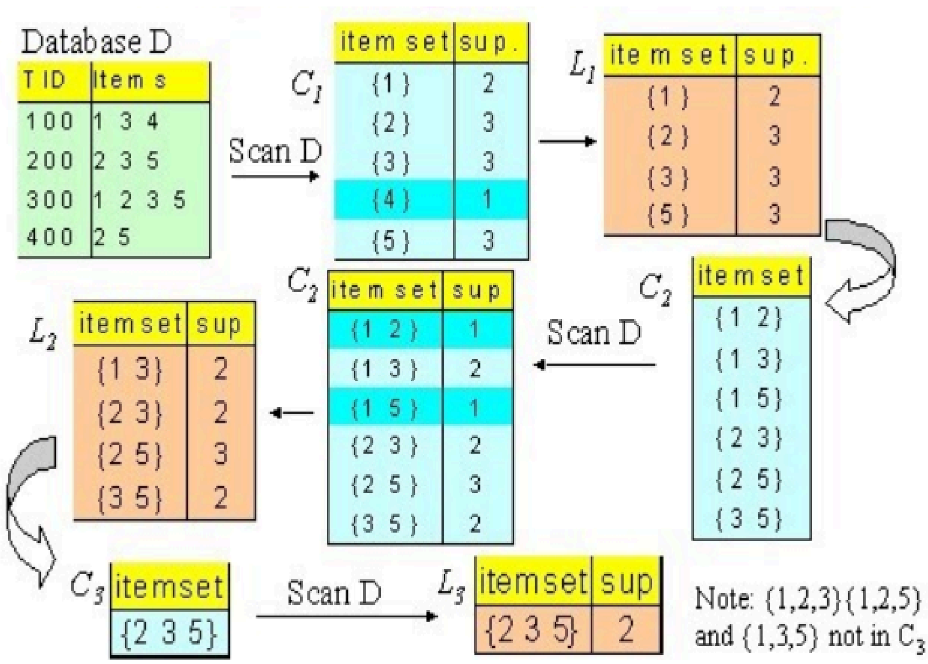
Apriori Algorithm:

1) It's an algorithm that is used for generating association rules.
2) **Why use the Apriori Algorithm ?**

   Suppose, we have an association rule :
   X => Y, then it means that if event X has occurred then there are some chances that Y will occur. Now, a relationship between X and Y is called single cardinality. But, what if 2 or more events are present (for eg, (X, Y, Z) => A), in such cases the cardinality increases and the possible combination of all the events and the number of association rules also increases. Let's say we have 10k events, imagine the number of association rules we will end up creating.
   For this reason, the association rules mining is extremely important so that we do not end up creating tens of thousands of rules. This is where the Apriori Algorithm comes into action. It helps us to limit the number of association rules.
3) Apriori Algorithm uses the frequent item sets to generate association rules. It is based on the concept that a subset of a frequent itemset must also be a frequent itemset.

4) Support and Confidence, each of them have a fixed threshold value. So, a frequent itemset is the one which has support value to be greater than the threshold.
5) Apriori algo example with the support threshold value = 2
6) We can see that the algo eliminates the association rules that have a support value less than 2.

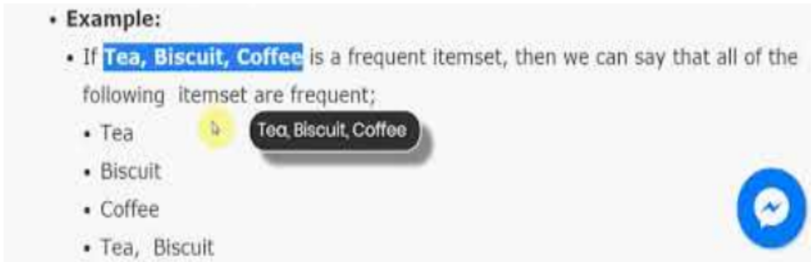




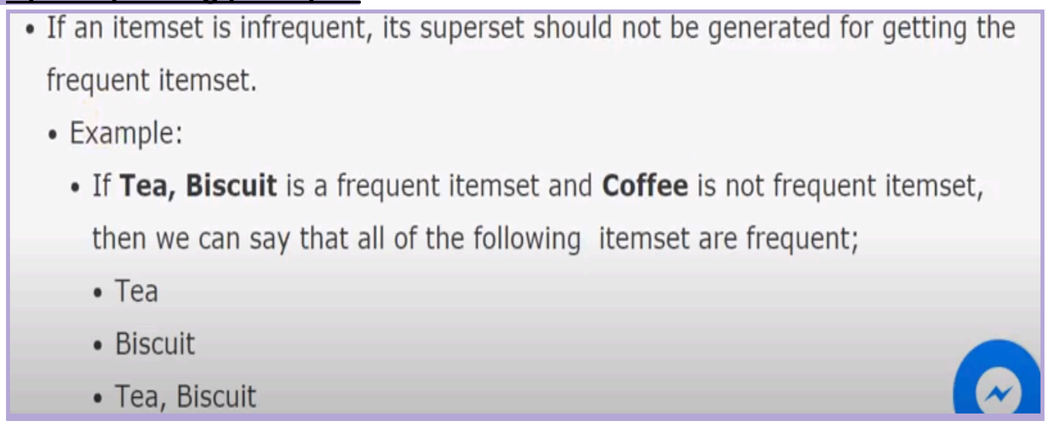
7) **Apriori Principles:**

- **Downward closure property of frequent patterns:**

  All the subsets of any frequent itemset must also be frequent



- **Apriori pruning principle:**



For getting the basics of Association Rules and Apriori Algo:
WATCH THIS VIDEO!!!
Or copy the Link : https://youtu.be/guVvtZ7ZClw

N - grams And Unigram:

1) N - grams are a set of co-occurring words within a given window say N = 2.
   - For eg, in the sentence “The quick brown fox jumps over the lazy dog”, if **N = 2** (referred to as bi-gram), the ngrams are:
     1. The quick
     2. quick brown
     3. brown fox
     4. fox jumps
     5. jumps over
     6. over the
     7. the lazy
     8. lazy dog

We have 8 n-grams in this case. (here, we considered moving only 1 word forward)

- For **N = 3**(referred as tri-gram), ngrams are:
    1. The quick brown
    2. quick brown fox
    3. brown fox jumps
    4. fox jumps over
    5. jumps over the
    6. over the lazy
    7. the lazy dog

We have & n grams in this case.

- For ***N = 1*** , this case is referred to as Unigrams
  The unigrams essentially returns the individual words of the sentences. Like in our case it returns:
  The, quick, brown, fox, jumps, over, the, lazy, dog

3) **How to calculate the number of N-grams in a sentence ?**

If X = total number of words in a sentence,

N = window for N-gram

Number of n-grams = X - (N - 1)

- If we assign a probability to the occurrence of an N-gram or probability of a word occurring next in a sequence of words, it can be very useful for phrase mining because I guess then we can restrict our search for a particular phrase to only that region in a sentence.

[contd]

Association Rules:

Positive association rules : association rules consider only items enumerated in transactions

Negative association rules : consider the same items, but in addition consider negated items (i.e. absent from transactions). Knowing the relationship between the absence of an item and the presence of another in the basket can be very important in some applications. For example, the association bread implies milk indicates the behaviour of buying milk and bread together. There might be a behaviour pattern where customers who buy tea do not buy coffee, or customers who buy juice do not buy cold drinks.

This can be denoted by ~ X => Y,   X => ~Y, ~X => ~Y

Further read: https://doi.org/10.1155/2014/973750

Apriori Algorithm:

We apply an iterative approach or level-wise search where k-frequent itemsets are used to find k+1 itemsets.

The parameters support and confidence are used. Support refers to items' frequency of occurrence; confidence is a conditional probability (p(A|B))

The algorithm begins by identifying frequent, individual items (items with a frequency greater than or equal to the given support threshold) in the database and continues to extend them to larger, frequent itemsets.

The following are the main steps of the algorithm:

1. Calculate the support of item sets (of size k = 1) in the transactional database. This is called generating the candidate set.

2. Prune the candidate set by eliminating items with a support less than the given threshold.
3. Join the frequent itemsets to form sets of size $k + 1$, and repeat the above sets until no more itemsets can be formed. This will happen when the sets formed have a support less than the given support.

For X->Y, confidence = Support(X and Y) / Support(X).
We can use this to find which items are bought together etc.

[Hash Trees can be used for the pruning step-check it out]

Further read: https://arxiv.org/ftp/arxiv/papers/1403/1403.3948.pdf
Set and Lattice Theory [interesting!]

N-gram:

n-gram is a contiguous sequence of n items from a given sample of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application.
We can use n-gram models to derive a probability of the sentence ,W, as the joint probability of each individual word in the sentence
P(W) = P(w1, w2, ..., wn)

This can be reduced to a sequence of n-grams (using conditional probability)
P(x1, x2, ..., xn) = P(x1)P(x2|x1)...P(xn|x1,...xn-1)
(check how)

For eg: There was heavy traffic

P('There was heavy traffic') = P('There', 'was', 'heavy', 'traffic')
P('There was heavy traffic') = P('There')P('was'|'There')P('heavy'|'There was')P('traffic'|'There was heavy')

But this will take too long to compute, we can instead use Markov Property
(check more!)
Some examples include auto completion of sentences, auto spell check, and check for grammar in a given sentence.

Further Read: https://web.stanford.edu/~jurafsky/slp3/3.pdf