



## UNIT IV

### UNSUPERVISED LEARNING

#### Apriori Algorithm - Clustering- K-means

##### Apriori Algorithm

Apriori algorithm refers to the algorithm which is used to calculate the association rules between objects. It means how two or more objects are related to one another. In other words, we can say that the apriori algorithm is an association rule learning that analyzes that people who bought product A also bought product B.

The primary objective of the apriori algorithm is to create the association rule between different objects. The association rule describes how two or more objects are related to one another. Apriori algorithm is also called frequent pattern mining. Generally, you operate the Apriori algorithm on a database that consists of a huge number of transactions. Let's understand the apriori algorithm with the help of an example; suppose you go to Big Bazar and buy different products. It helps the customers buy their products with ease and increases the sales performance of the Big Bazar. In this tutorial, we will discuss the apriori algorithm with examples.

The Apriori algorithm uses frequent itemsets to generate association rules, and it is designed to work on the databases that contain transactions. With the help of these association rule, it determines how strongly or how weakly two objects are connected. This algorithm uses a **breadth-first search** and **Hash Tree** to calculate the itemset associations efficiently. It is the iterative process for finding the frequent itemsets from the large dataset.

This algorithm was given by the **R. Agrawal** and **Srikant** in the year **1994**. It is mainly used for *market basket analysis* and helps to find those products that can be bought together. It can also be used in the healthcare field to find drug reactions for patients.

##### What is Frequent Itemset?

Frequent itemsets are those items whose support is greater than the threshold value or user-specified minimum support. It means if A & B are the frequent itemsets together, then individually A and B should also be the frequent itemset.

Suppose there are the two transactions: A= {1,2,3,4,5}, and B= {2,3,7}, in these two transactions, 2 and 3 are the frequent itemsets.

##### Steps for Apriori Algorithm

Below are the steps for the apriori algorithm:

**Step-1:** Determine the support of itemsets in the transactional database, and select the minimum support and confidence.

**Step-2:** Take all supports in the transaction with higher support value than the minimum or selected support value.

**Step-3:** Find all the rules of these subsets that have higher confidence value than the threshold or minimum confidence.

**Step-4:** Sort the rules as the decreasing order of lift.

### Apriori Algorithm Working

We will understand the apriori algorithm using an example and mathematical calculation:

**Example:** Suppose we have the following dataset that has various transactions, and from this dataset, we need to find the frequent itemsets and generate the association rules using the Apriori algorithm:

TID	ITEMSETS
T1	A, B
T2	B, D
T3	B, C
T4	A, B, D
T5	A, C
T6	B, C
T7	A, C
T8	A, B, C, E
T9	A, B, C

**Given: Minimum Support= 2, Minimum Confidence= 50%**

**Solution:**

**Step-1: Calculating C1 and L1:**

In the first step, we will create a table that contains support count (The frequency of each itemset individually in the dataset) of each itemset in the given dataset. This table is called the **Candidate set** or **C1**.

Itemset	Support_Count
A	6
B	7
C	5
D	2
E	1

Now, we will take out all the itemsets that have the greater support count than the Minimum Support (2). It will give us the table for the **frequent itemset L1**. Since all the itemsets have greater or equal support count than the minimum support, except the

E, so E itemset will be removed.

Itemset	Support_Count
A	6
B	7
C	5
D	2

**Step-2: Candidate Generation C2, and L2:**

- In this step, we will generate C2 with the help of L1. In C2, we will create the pair of the itemsets of L1 in the form of subsets.
- After creating the subsets, we will again find the support count from the main transaction table of datasets, i.e., how many times these pairs have occurred together in the given dataset. So, we will get the below table for C2:

Itemset	Support_Count
{A, B}	4
{A,C}	4
{A, D}	1
{B, C}	4
{B, D}	2
{C, D}	0

- Again, we need to compare the C2 Support count with the minimum support count, and after comparing, the itemset with less support count will be eliminated from the table C2. It will give us the below table for L2

Itemset	Support_Count
{A, B}	4
{A, C}	4
{B, C}	4
{B, D}	2

**A, B, C, D**

**Step-3: Candidate generation C3, and L3:**

- For C3, we will repeat the same two processes, but now we will form the C3 table with subsets of three itemsets together, and will calculate the support count from the dataset.

It will give the below table:

Itemset	Support_Count
{A, B, C}	2
{B, C, D}	1
{A, C, D}	0
{A, B, D}	0

- Now we will create the L3 table. As we can see from the above C3 table, there is only one combination of itemset that has support count equal to the minimum support count. So, the L3 will have only one combination, i.e., **{A, B, C}**.

**Step-4: Finding the association rules for the subsets:**

To generate the association rules, first, we will create a new table with the possible rules from the occurred combination {A, B,C}. For all the rules, we will calculate the Confidence using formula  $\frac{\text{sup}(A \wedge B)}{A}$ . After calculating the confidence value for all rules, we will exclude the rules that have less confidence than the minimum threshold(50%).

Consider the below table:

Rules	Support	Confidence
$A \wedge B \rightarrow C$	2	$\text{Sup}\{(A \wedge B) \wedge C\}/\text{sup}(A \wedge B)= 2/4=0.5=50\%$
$B \wedge C \rightarrow A$	2	$\text{Sup}\{(B \wedge C) \wedge A\}/\text{sup}(B \wedge C)= 2/4=0.5=50\%$
$A \wedge C \rightarrow B$	2	$\text{Sup}\{(A \wedge C) \wedge B\}/\text{sup}(A \wedge C)= 2/4=0.5=50\%$
$C \rightarrow A \wedge B$	2	$\text{Sup}\{(C \wedge (A \wedge B))\}/\text{sup}(C)= 2/5=0.4=40\%$
$A \rightarrow B \wedge C$	2	$\text{Sup}\{(A \wedge (B \wedge C))\}/\text{sup}(A)= 2/6=0.33=33.33\%$
$B \rightarrow B \wedge C$	2	$\text{Sup}\{(B \wedge (B \wedge C))\}/\text{sup}(B)= 2/7=0.28=28\%$

As the given threshold or minimum confidence is 50%, so the first three rules  $A \wedge B \rightarrow C$ ,  $B \wedge C \rightarrow A$ , and  $A \wedge C \rightarrow B$  can be considered as the strong association rules for the given problem.

**Advantages of Apriori Algorithm**

- This is easy to understand algorithm
- The join and prune steps of the algorithm can be easily implemented on large datasets.

**Disadvantages of Apriori Algorithm**

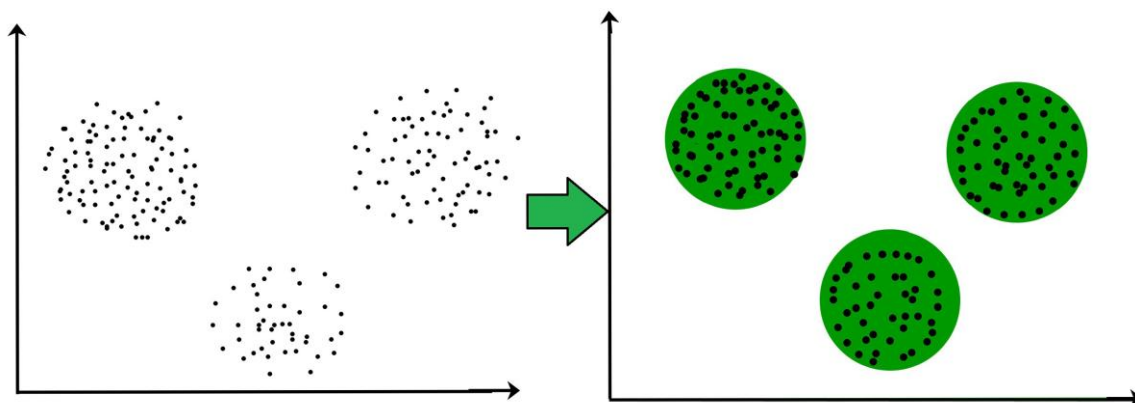
- The apriori algorithm works slow compared to other algorithms.
- The overall performance can be reduced as it scans the database for multiple times.
- The time complexity and space complexity of the apriori algorithm is  $O(2^D)$ , which is very high. Here D represents the horizontal width present in the database.

## Clustering

**Introduction to Clustering:** It is basically a type of unsupervised learning method. An unsupervised learning method is a method in which we draw references from datasets consisting of input data without labeled responses. Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples.

**Clustering** is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

**For example** The data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture.



### K means Clustering

[Unsupervised Machine Learning](#) is the process of teaching a computer to use unlabeled, unclassified data and enabling the algorithm to operate on that data without supervision. Without any previous data training, the machine's job in this case is to organize unsorted data according to parallels, patterns, and variations.

The goal of [clustering](#) is to divide the population or set of data points into a number of groups so that the data points within each group are more comparable to one another and different from the data points within the other groups. It is essentially a grouping of things based on how similar and different they are to one another.

We are given a data set of items, with certain features, and values for these features (like a vector). The task is to categorize those items into groups. To achieve this, we will use the K-means algorithm; an unsupervised learning

algorithm. 'K' in the name of the algorithm represents the number of groups/clusters we want to classify our items into.

## K-Means Clustering Algorithm

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. In this topic, we will learn what is K-means clustering algorithm, how the algorithm works, along with the Python implementation of k-means clustering.

### What is K-Means Algorithm?

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if  $K=2$ , there will be two clusters, and for  $K=3$ , there will be three clusters, and so on.

It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.

It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

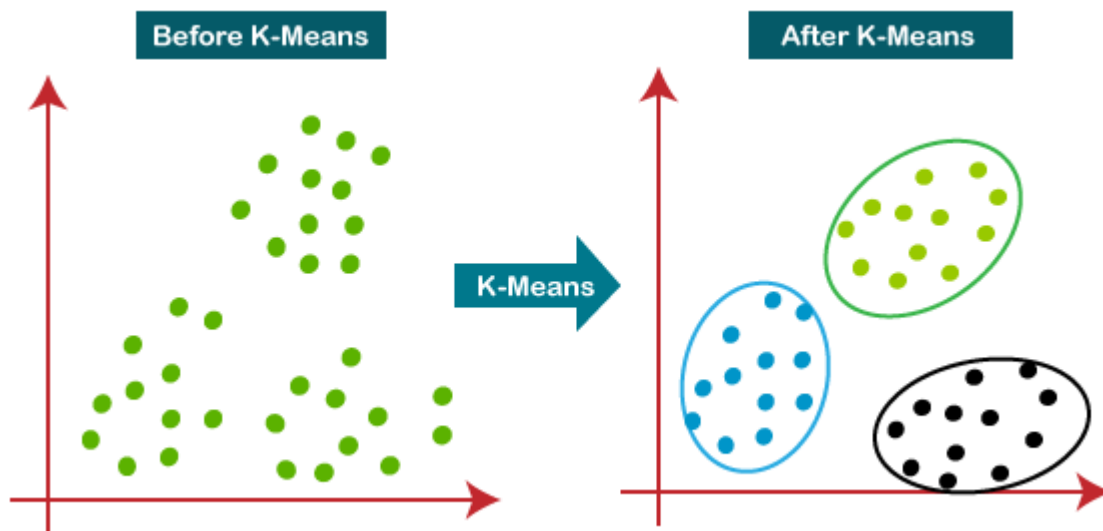
The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means clustering algorithm mainly performs two tasks:

- determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

Hence each cluster has datapoints with some commonalities, and it is away from other clusters.

The below diagram explains the working of the K-means Clustering Algorithm:



### How does the K-Means Algorithm Work?

The working of the K-Means algorithm is explained in the below steps:

**Step-1:** Select the number K to decide the number of clusters.

**Step-2:** Select random K points or centroids. (It can be other from the input dataset).

**Step-3:** Assign each data point to their closest centroid, which will form the predefined K clusters.

**Step-4:** Calculate the variance and place a new centroid of each cluster.

**Step-5:** Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

**Step-6:** If any reassignment occurs, then go to step-4 else go to FINISH.

**Step-7:** The model is ready.