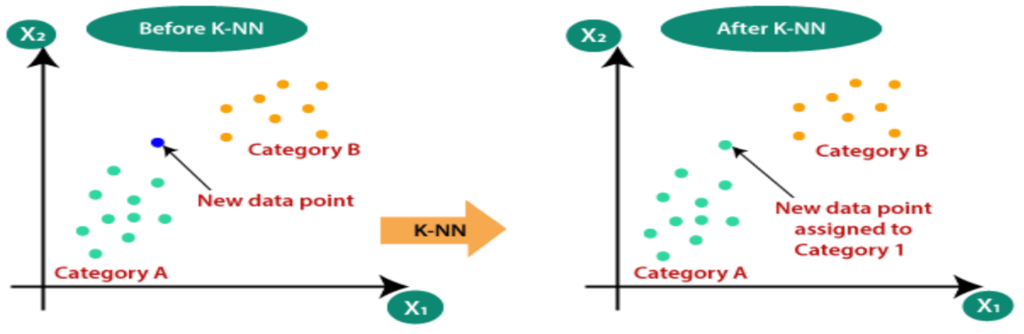
Chapter 3: Machine Learning Algorithms

1. K-Nearest Neighbors Algorithm (KNN)

The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other. The algorithm's learning is:

- ***Instance-based learning:*** Here we do not learn weights from training data to predict output (as in model-based algorithms) but use entire training instances to predict output for unseen data.
- ***Lazy Learning:*** Model is not learned using training data prior and the learning process is postponed to a time when prediction is requested on the new instance.
- ***Non -Parametric:*** In KNN, there is no predefined form of the mapping function.

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



Determine your distance metrics

In order to determine which data points are closest to a given query point, the distance between the query point and the other data points will need to be calculated. These distance metrics help to form decision boundaries, which partitions query points into different regions. You commonly will see decision boundaries visualized with diagrams.

- ***Euclidean distance*:** This is the most commonly used distance measure, and it is limited to real-valued vectors. It measures a straight line between the query point and the other point being measured.
- ***Manhattan distance*:** This is also another popular distance metric, which measures the absolute value between two points. It is also referred to as taxicab distance or city block distance as it is commonly visualized with a grid, illustrating how one might navigate from one address to another via city streets.
- ***Minkowski distance*:** This distance measure is the generalized form of Euclidean and Manhattan distance metrics. The parameter, p, in the formula below, allows for the creation of other distance

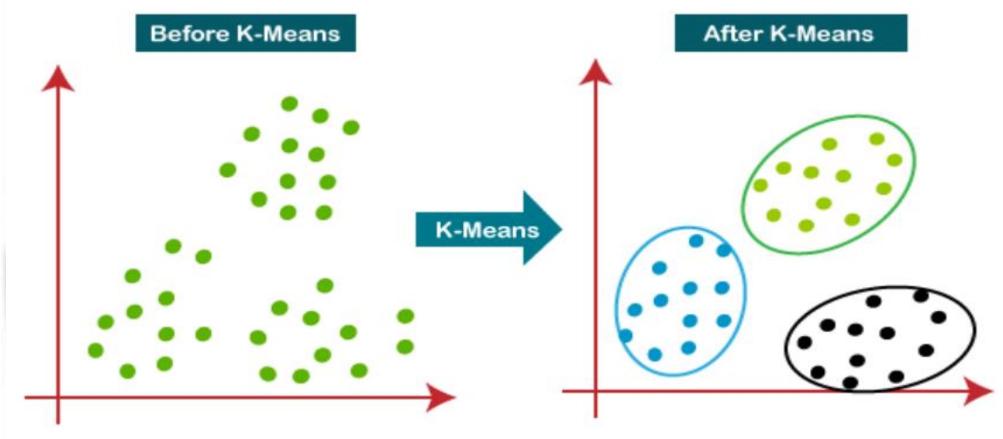
metrics. Euclidean distance is represented by this formula when p is equal to two, and Manhattan distance is denoted with p equal to one.

- ***Hamming distance*:** This technique is used typically used with Boolean or string vectors, identifying the points where the vectors do not match. As a result, it has also been referred to as the overlap metric.

2. K-Means Algorithm

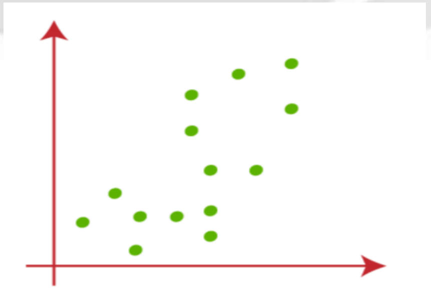
K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. The k-means clustering algorithm mainly performs two tasks:

- 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.



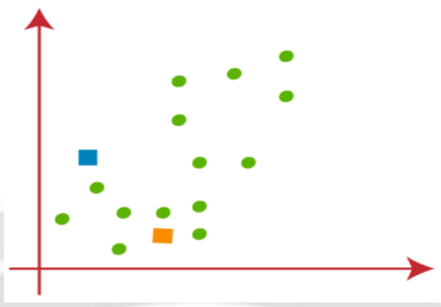
Let’s understand it with an example.

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

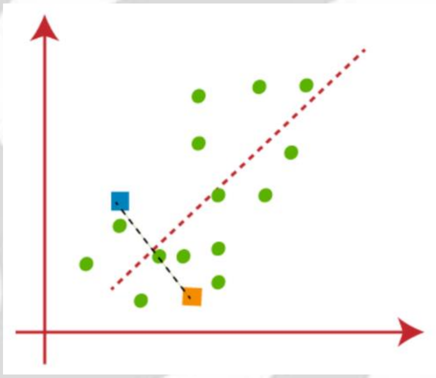


- Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.

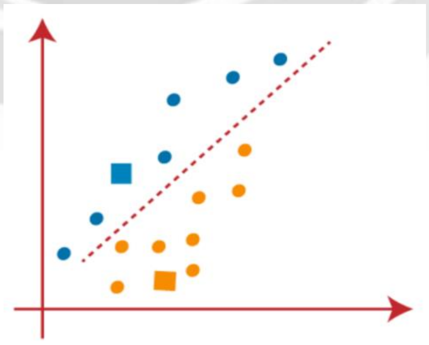
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the below image



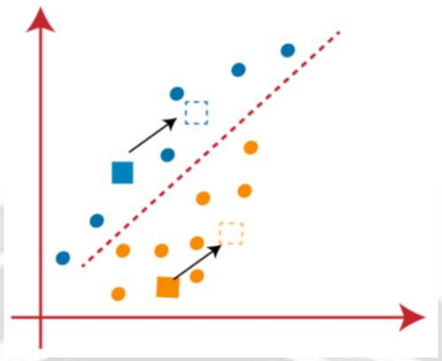
- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids. Consider the below image:



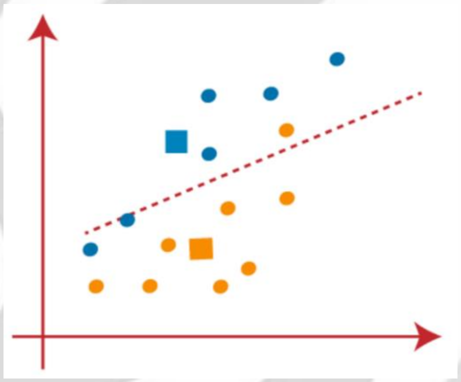
- From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.



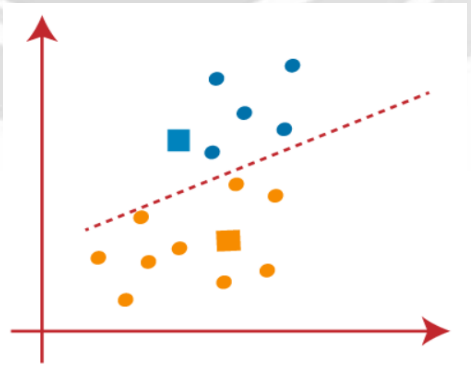
- As we need to find the closest cluster, so we will repeat the process by choosing a new centroid. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below



- Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image

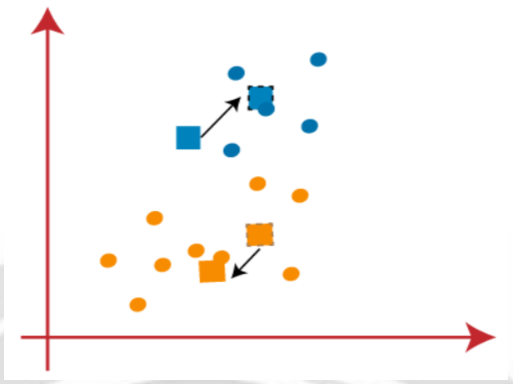


- From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.

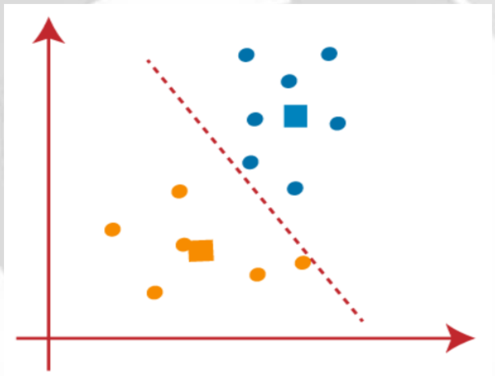


As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

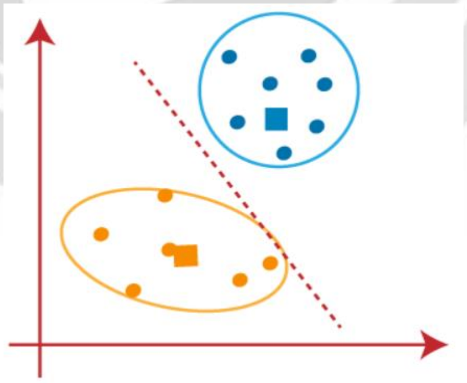
- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image



- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be



- We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image



- As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image