Knn – algorithms

# K-Nearest Neighbor(KNN) Algorithm for Machine Learning

Some facts:

* K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
* K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
* K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
* K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
* K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.
* It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
* KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.
* Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.



## Why do we need a K-NN Algorithm?

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:



## How does K-NN work?

The K-NN working can be explained on the basis of the below algorithm:

* Step-1: Select the number K of the neighbors
* Step-2: Calculate the Euclidean distance of K number of neighbors
* Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
* Step-4: Among these k neighbors, count the number of the data points in each category.
* Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
* Step-6: Our model is ready.

Suppose we have a new data point and we need to put it in the required category. Consider the below image:



* Firstly, we will choose the number of neighbors, so we will choose the k=5.
* Next, we will calculate the Euclidean distance between the data points. The Euclidean distance is the distance between two points, which we have already studied in geometry. It can be calculated as:



* By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B. Consider the below image:



* As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

## Required Data Preparation:

1. Data Scaling: To locate the data point in multidimensional feature space, it would be helpful if all features are on the same scale. Hence normalization or standardization of data will help.

2. Dimensionality Reduction: KNN may not work well if there are too many features. Hence dimensionality reduction techniques like feature selection, principal component analysis can be implemented.

3. Missing value treatment: If out of M features one feature data is missing for a particular example in the training set then we cannot locate or calculate distance from that point. Therefore deleting that row or imputation is required.

## How to select the value of K in the K-NN Algorithm?

Below are some points to remember while selecting the value of K in the K-NN algorithm:

* There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
* A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.
* Large values for K are good, but it may find some difficulties.

## Advantages of KNN Algorithm:

* It is simple to implement.
* It is robust to the noisy training data
* It can be more effective if the training data is large.

## Disadvantages of KNN Algorithm:

* Always needs to determine the value of K which may be complex some time.
* The computation cost is high because of calculating the distance between the data points for all the training samples.

**Features of KNN:**

**1. kNN Is a Supervised Machine Learning Algorithm**

In supervised models, you have two types of variables at the same time:

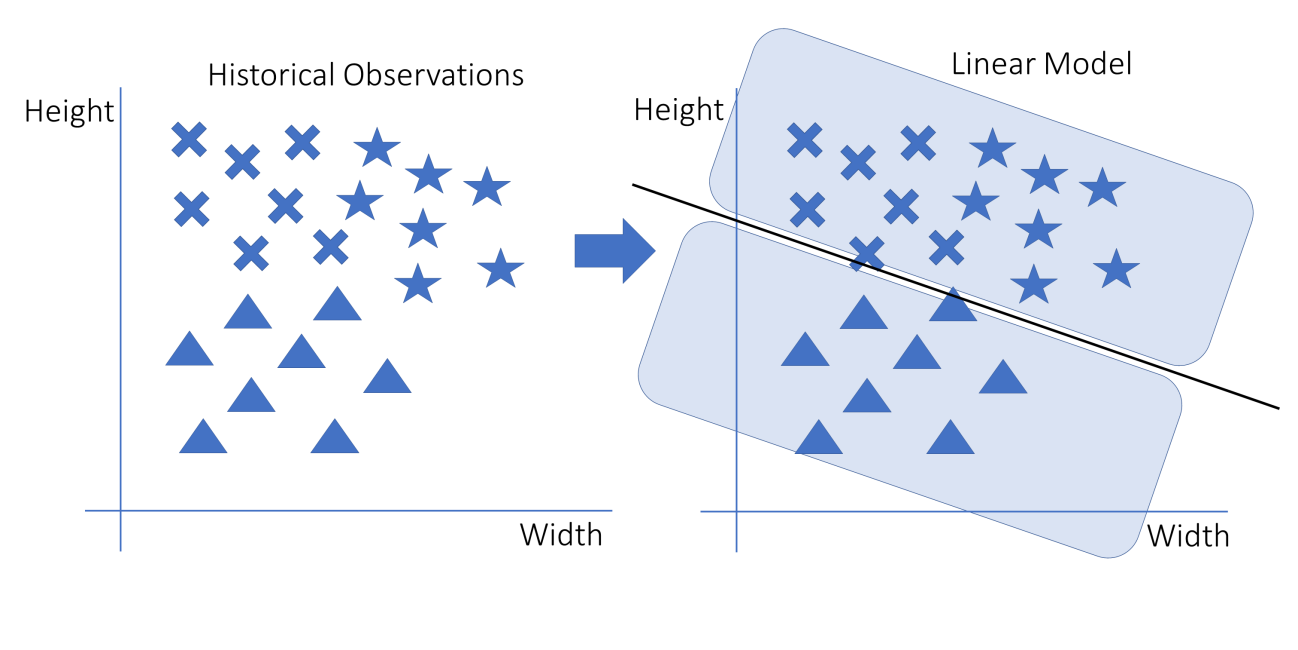
1. A target variable, which is also called the dependent variable or the y variable.
2. Independent variables, which are also known as x variables or explanatory variables.

The target variable is the variable that you want to predict. It depends on the independent variables and it isn’t something that you know ahead of time. The independent variables are variables that you do know ahead of time. You can plug them into an equation to predict the target variable. In this way, it’s relatively similar to the y = ax + b case.

### 2. kNN Is a Nonlinear Learning Algorithm

A second property that makes a big difference in machine learning algorithms is whether or not the models can estimate nonlinear relationships.

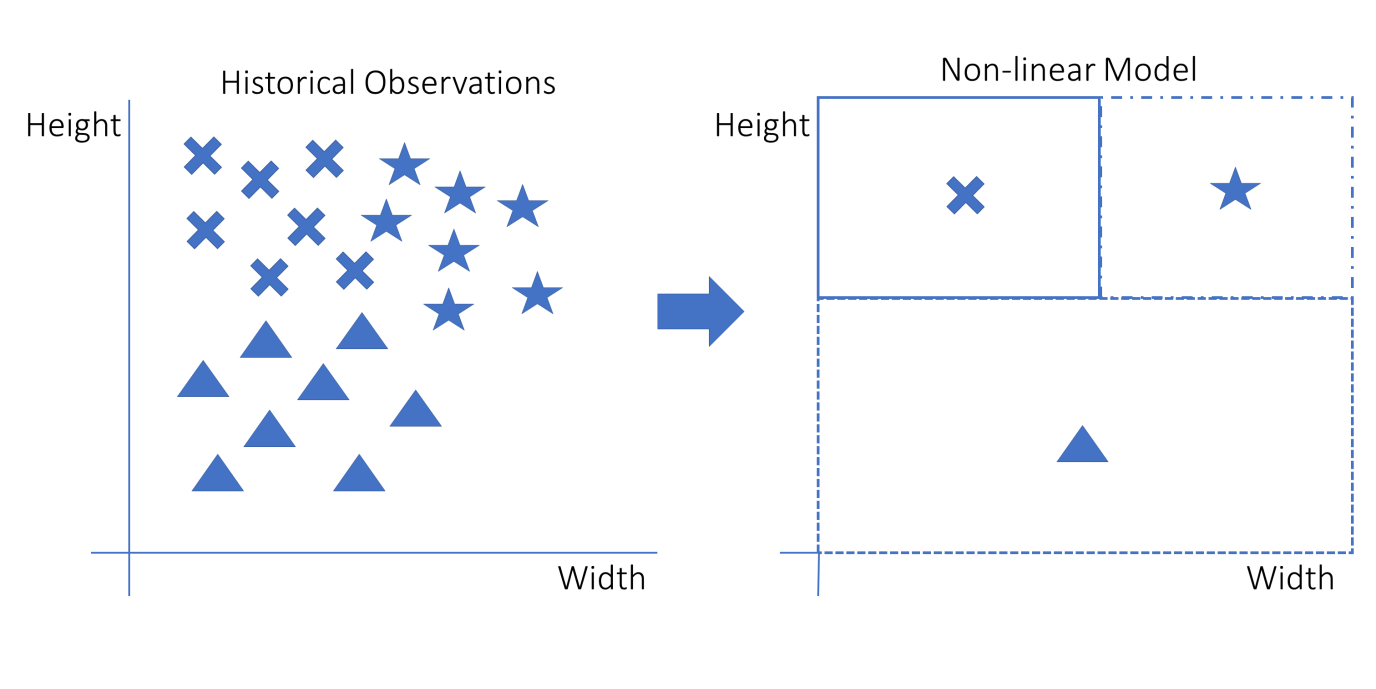
Linear models are models that predict using lines or hyperplanes. In the image, the model is depicted as a line drawn between the points. The model y = ax + b is the classical example of a linear model. You can see how a linear model could fit the example data in the following schematic drawing:

[](https://files.realpython.com/media/knn_04_MLlinear_wide.05d1160927c7.png)

In this picture, the data points are depicted on the left with stars, triangles, and crosses. On the right is a linear model that can separate triangles from non-triangles. The decision is a line. Every point above the line is a non-triangle, and everything below the line is a triangle.

If you wanted to add another independent variable to the previous graph, you would need to draw it as an additional dimension, thereby creating a cube with the shapes inside it. Yet a line wouldn’t be able to cut a cube into two parts. The multidimensional counterpart of the line is the hyperplane. A linear model is therefore represented by a hyperplane, which in the case of two-dimensional space happens to be a line.

Nonlinear models are models that use any approach other than a line to separate their cases. A well-known example is the decision tree, which is basically a long list of if … else statements. In the nonlinear graph, if … else statements would allow you to draw squares or any other form that you wanted to draw. The following graph depicts a nonlinear model applied to the example data:

[](https://files.realpython.com/media/knn_05_MLnonlinear_wide.5f554e71c368.png)

This graph shows how a decision can be nonlinear. The decision rule is made up of three squares. The box in which a new data point falls will define its predicted shape. Note that it’s not possible to fit this at once using a line: Two lines are needed. This model could be re-created with if … else statements as follows:

* If the data point’s height is low, then it’s a triangle.
* Else, if the data point’s width is low, then it’s a cross.
* Else, if none of the above is true, then it’s a star.

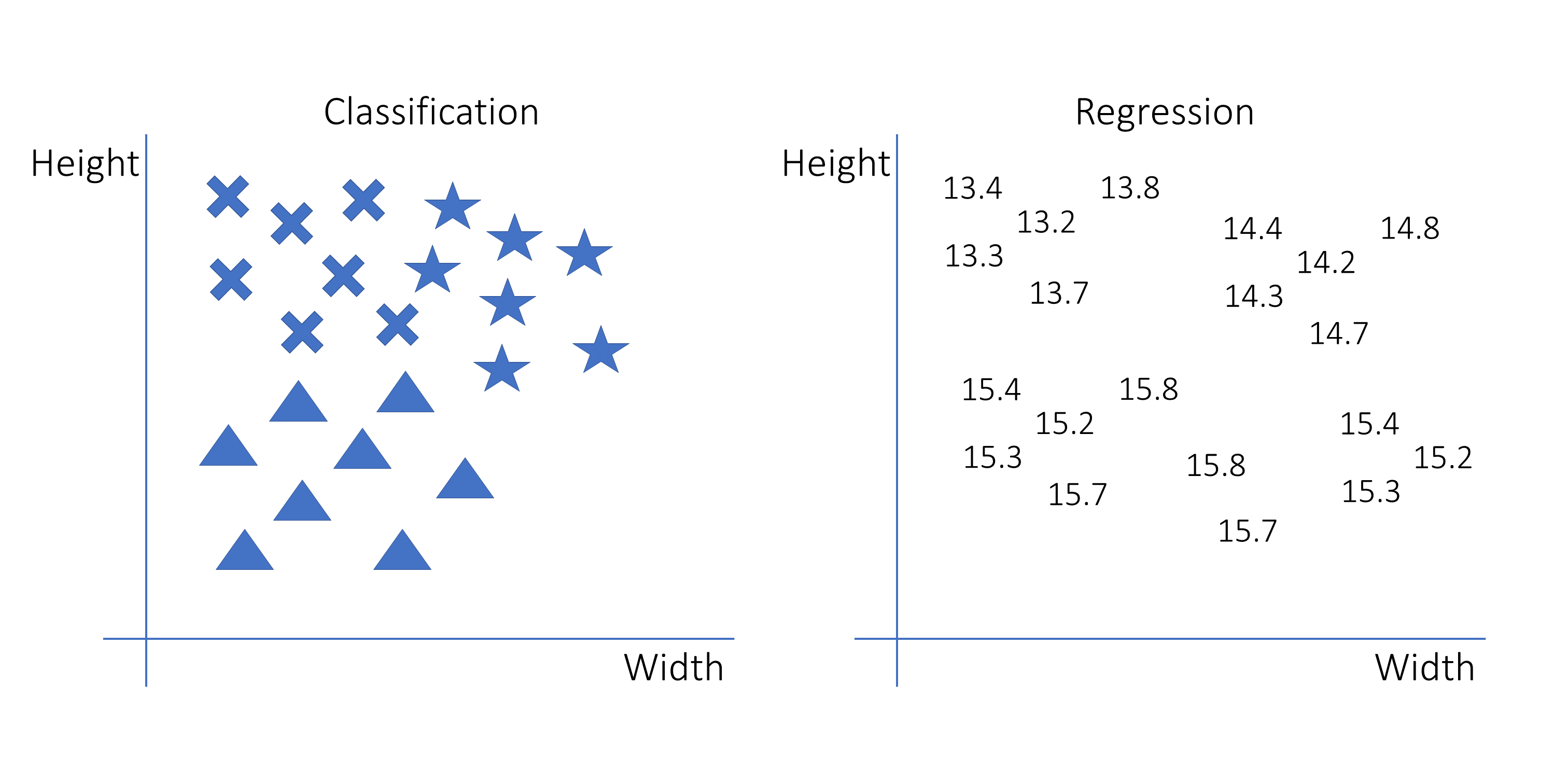
kNN is an example of a nonlinear model.

**3. kNN Is a Supervised Learner for Both Classification and Regression**

Supervised machine learning algorithms can be split into two groups based on the type of target variable that they can predict:

1. Classification is a prediction task with a categorical target variable. Classification models learn how to classify any new observation. This assigned class can be either right or wrong, not in between. A classic example of classification is the iris dataset, in which you use physical measurements of plants to predict their species. A famous algorithm that can be used for classification is logistic regression.
2. Regression is a prediction task in which the target variable is numeric. A famous example of regression is the Housing Prices Challenge on Kaggle. In this machine learning contest, participants try to predict the sales prices of houses based on numerous independent variables.

In the following graphic, you can see what a regression and a classification would look like using the previous example:

[](https://files.realpython.com/media/knn_06_MLclassificationregression.6029d11323aa.png)

The left part of this image is a classification. The target variable is the shape of the observation, which is a categorical variable. The right part is a regression. The target variable is numeric. The decision rules could be exactly the same for the two examples, but their interpretations are different.

For a single prediction, classifications are either right or wrong, while regressions have an error on a continuous scale. Having a numeric error measure is more practical, so many classification models predict not only the class but also the probability of being in either of the classes.

Some models can only do regression, some can only do classification, and some can do both. The kNN algorithm seamlessly adapts to both classification and regression.

### kNN Is Fast and Interpretable

As a final criterion to characterize machine learning models, you need to take into account model complexity. Machine learning, and especially artificial intelligence, is currently blooming and is being used in many complicated tasks, such as understanding text, images, and speech, or for self-driving cars.

More advanced and complex models like neural networks can probably learn anything that a k-Nearest Neighbors model can. After all, those advanced models are very strong learners. However, be aware that this complexity also has its price. In order to make the models fit to your prediction, you’ll generally spend much more time on development.

You’ll also need much more data to fit a more complex model, and data is not always available. Last but not least, more complex models are more difficult for us humans to interpret, and sometimes this interpretation can be very valuable.

This is where the force of the kNN model lies. It allows its users to understand and interpret what’s happening inside the model, and it’s very fast to develop. This makes kNN a great model for many machine learning use cases that don’t require highly complex techniques.

**K-means clustering**

Clustering is the process of dividing the entire data into groups (also known as clusters) based on the patterns in the data.

## **How is Clustering an Unsupervised Learning Problem?**

when we have a target variable to predict based on a given set of predictors or independent variables, such problems are called supervised learning problems. There might be situations where we do not have any target variable to predict. Such problems, without any fixed target variable, are known as **unsupervised learning problems.** In these problems, we only have the independent variables and no target/dependent variable.In clustering, we do not have a target to predict. We look at the data and then try to club similar observations and form different groups. Hence it is an unsupervised learning problem.

## **Properties of Clusters**

1. All the data points in a cluster should be similar to each other
2. The data points from different clusters should be as different as possible.

**K-means**

* 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).

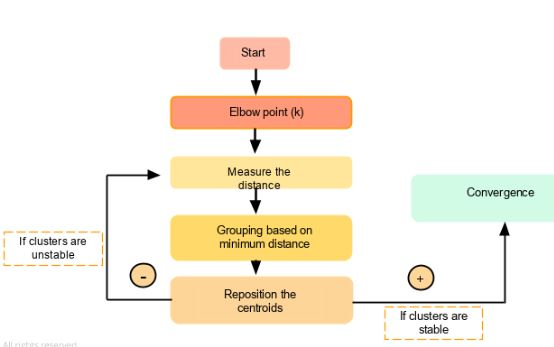
Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

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.

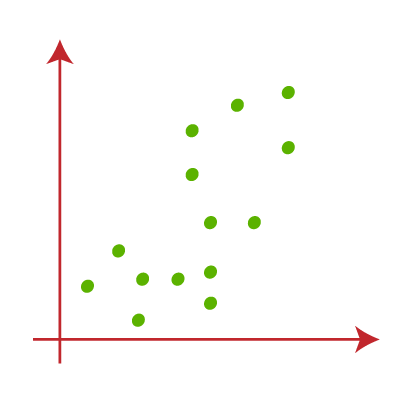
Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

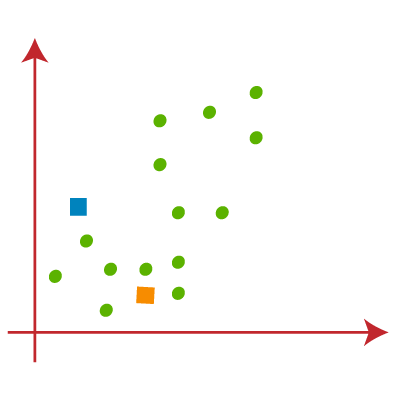
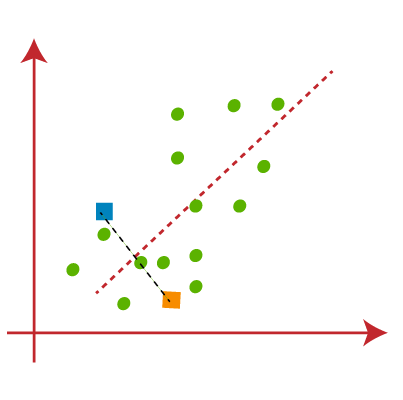
Step-7: The model is ready.



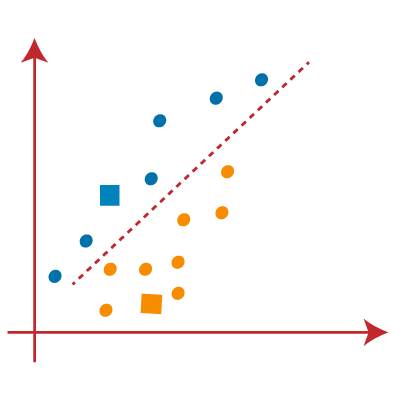
Let's understand the above steps by considering the visual plots:

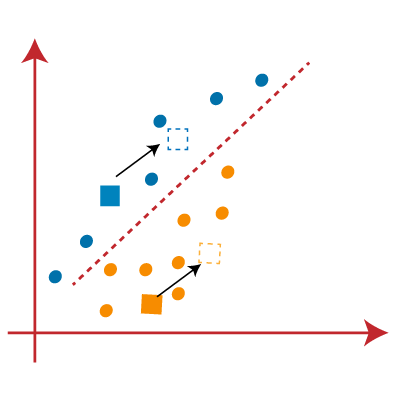
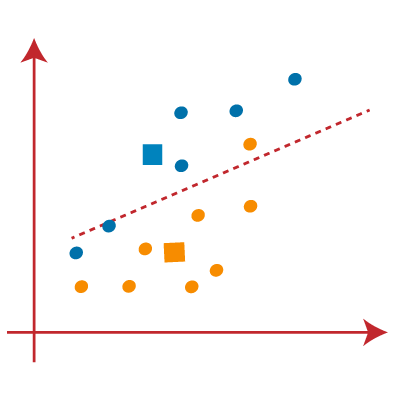
Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:



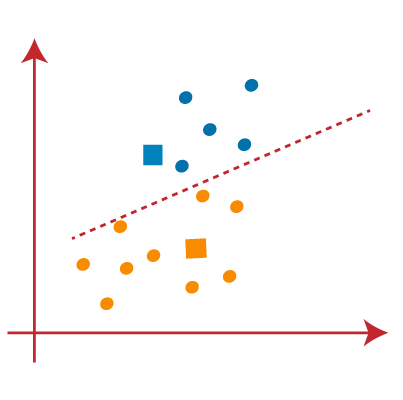
* Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
* We need to choose some random k points or centroid to form the cluster. These points can be either the points 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.

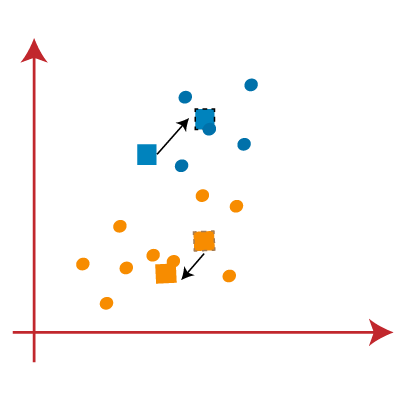
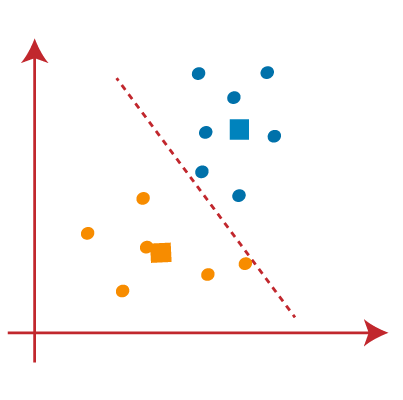
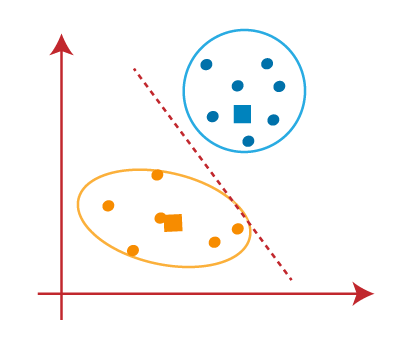


* As we need to find the closest cluster, so we will repeat the process by choosing a new centroid. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:  
  
* Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:  
  

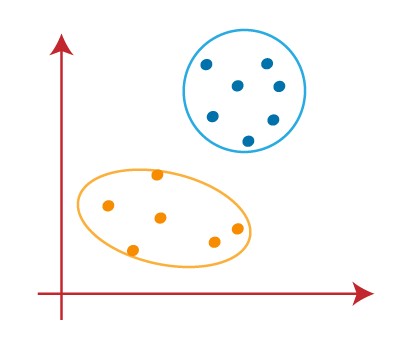
From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.



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:



## How to choose the value of "K number of clusters" in K-means Clustering?

The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

### Elbow Method

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. WCSS stands for Within Cluster Sum of Squares, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

WCSS= ∑Pi in Cluster1 distance(Pi C1)2 +∑Pi in Cluster2distance(Pi C2)2+∑Pi in CLuster3 distance(Pi C3)2

In the above formula of WCSS,

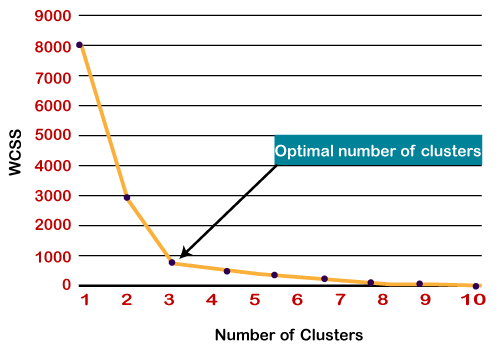
∑Pi in Cluster1 distance(Pi C1)2: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms.

To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:

* **It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).**
* **For each value of K, calculates the WCSS value.**
* **Plots a curve between calculated WCSS values and the number of clusters K.**
* **The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.**

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



## **Applications of K-Means Clustering**

K-Means clustering is used in a variety of examples or business cases in real life, like:

* Academic performance
* Diagnostic systems
* Search engines
* Wireless sensor networks

### **Academic Performance**

Based on the scores, students are categorized into grades like A, B, or C.

Diagnostic systems

The medical profession uses k-means in creating smarter medical decision support systems, especially in the treatment of liver ailments.

Search engines

Clustering forms a backbone of search engines. When a search is performed, the search results need to be grouped, and the search engines very often use clustering to do this.

Wireless sensor networks

The clustering algorithm plays the role of finding the cluster heads, which collect all the data in its respective cluster.

# Drawbacks

1. Kmeans algorithm is good in capturing structure of the data if clusters have a spherical-like shape. It always try to construct a nice spherical shape around the centroid. That means, the minute the clusters have a complicated geometric shapes, kmeans does a poor job in clustering the data.
2. kmeans algorithm doesn’t let data points that are far-away from each other share the same cluster even though they obviously belong to the same cluster.
3. Scale/standardize the data when applying kmeans algorithm.
4. Elbow method in selecting number of clusters doesn’t usually work because the error function is monotonically decreasing for all ks.
5. Kmeans gives more weight to the bigger clusters.
6. Kmeans assumes spherical shapes of clusters (with radius equal to the distance between the centroid and the furthest data point) and doesn’t work well when clusters are in different shapes such as elliptical clusters.
7. If there is overlapping between clusters, kmeans doesn’t have an intrinsic measure for uncertainty for the examples belong to the overlapping region in order to determine for which cluster to assign each data point.
8. Kmeans may still cluster the data even if it can’t be clustered such as data that comes from uniform distributions.

## Advantages of k-means

1. Simple: It is easy to implement k-means and identify unknown groups of data from complex data sets. The results are presented in an easy and simple manner.
2. Flexible: K-means algorithm can easily adjust to the changes. If there are any problems, adjusting the cluster segment will allow changes to easily occur on the algorithm.
3. Suitable in a large dataset: K-means is suitable for a large number of datasets and it’s computed much faster than the smaller dataset. It can also produce higher clusters.
4. Efficient: The algorithm used is good at segmenting the large data set. Its efficiency depends on the shape of the clusters. K-means works well in hyper-spherical clusters.
5. Time complexity: K-means segmentation is linear in the number of data objects thus increasing execution time. It doesn’t take more time in classifying similar characteristics in data like hierarchical algorithms.
6. Tight clusters: Compared to hierarchical algorithms, k-means produce tighter clusters especially with globular clusters.
7. Easy to interpret: The results are easy to interpret. It generates cluster descriptions in a form minimized to ease understanding of the data.
8. Computation cost: Compared to using other clustering methods, a k-means clustering technique is fast and efficient in terms of its computational cost O(K\*n\*d).
9. Accuracy: K-means analysis improves clustering accuracy and ensures information about a particular problem domain is available. Modification of the k-means algorithm based on this information improves the accuracy of the clusters.
10. Spherical clusters: This mode of clustering works great when dealing with spherical clusters. It operates with an assumption of joint distributions of features since each cluster is spherical. All the clusters features or characters have equal variance and each is independent of the other.