MACHINE LEARNING

Unit-3

Important Q & A

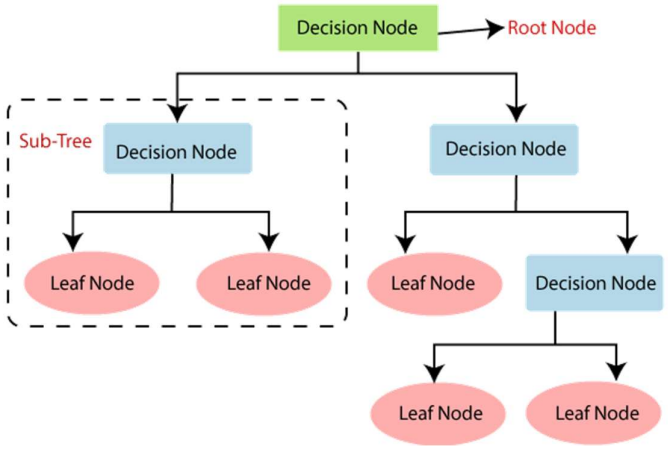
1) What is Decision Tree Classification algorithm? Why we use decision trees?

Ans) Decision Tree Classification Algorithm

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- The decisions or the test are performed on the basis of features of the given dataset.
- ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

Below diagram explains the general structure of a decision tree:

Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.



Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.

Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.

Branch/Sub Tree: A tree formed by splitting the tree.

Pruning: Pruning is the process of removing the unwanted branches from the tree.

Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

2) Explain Decision Tree Construction and its work with example? What are the advantages & disadvantages?

Ans) How does the Decision Tree algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

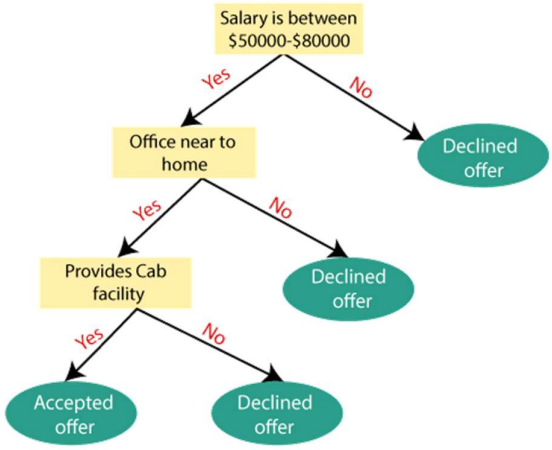
- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.

- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

The process of creating a decision tree involves:

1. **Selecting the Best Attribute**: Using a metric like Gini impurity, entropy, or information gain, the best attribute to split the data is selected.
2. **Splitting the Dataset**: The dataset is split into subsets based on the selected attribute.
3. **Repeating the Process**: The process is repeated recursively for each subset, creating a new internal node or leaf node until a stopping criterion is met (e.g., all instances in a node belong to the same class or a predefined depth is reached).

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



Attribute Selection Measures

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as **Attribute selection measure or ASM.** By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- **Information Gain**
- **Gini Index**

1. Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

1. Information Gain= Entropy(S)- [(Weighted Avg) *Entropy (each feature)

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

- **S= Total number of samples**
- **P(yes)= probability of yes**
- **P(no)= probability of no**

2. Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART (Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

Gini Index= 1- $\sum_j P_j^2$

Pruning: Getting an Optimal Decision tree

Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.

A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset. Therefore, a technique that decreases the size of the learning tree without reducing accuracy is known as Pruning. There are mainly two types of tree **pruning** technology used:

- **Cost Complexity Pruning**
- **Reduced Error Pruning.**

Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the **Random Forest algorithm.**
- For more class labels, the computational complexity of the decision tree may increase.

Pruning

To overcome **overfitting, pruning** techniques are used. Pruning reduces the size of the tree by removing nodes that provide little power in classifying instances. There are two main types of pruning:

- **Pre-pruning (Early Stopping)**: Stops the tree from growing once it meets certain criteria (e.g., maximum depth, minimum number of samples per leaf).
- **Post-pruning**: Removes branches from a fully grown tree that do not provide significant power.

Applications of Decision Trees

- **Business Decision Making**: Used in strategic planning and resource allocation.
- **Healthcare**: Assists in diagnosing diseases and suggesting treatment plans.
- **Finance**: Helps in credit scoring and risk assessment.
- **Marketing**: Used to segment customers and predict customer behavior.

3) Explain about CART (Classification and Regression Tree) with algorithm?

Ans) CART (Classification and Regression Trees) is a variation of the decision tree algorithm. It can handle both classification and regression tasks. Scikit-Learn uses the Classification and Regression Tree (CART) algorithm to train Decision Trees (also called “growing” trees). CART was first produced by Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone in 1984.

CART (Classification and Regression Tree) for Decision Tree

CART is a predictive algorithm used in Machine learning and it explains how the target variable's values can be predicted based on other matters. It is a decision tree where each fork is split into a predictor variable and each node has a prediction for the target variable at the end.

The term CART serves as a generic term for the following categories of decision trees:

- **Classification Trees:** The tree is used to determine which "class" the target variable is most likely to fall into when it is continuous.
- **Regression trees:** These are used to predict a continuous variable's value.

In the decision tree, nodes are split into sub-nodes based on a threshold value of an attribute. The root node is taken as the training set and is split into two by considering the best attribute and threshold value. Further, the subsets are also split using the same logic. This continues till the last pure sub-set is found in the tree or the maximum number of leaves possible in that growing tree.

CART Algorithm

Classification and Regression Trees (CART) is a decision tree algorithm that is used for both classification and regression tasks. It is a supervised learning algorithm that learns from labelled data to predict unseen data.

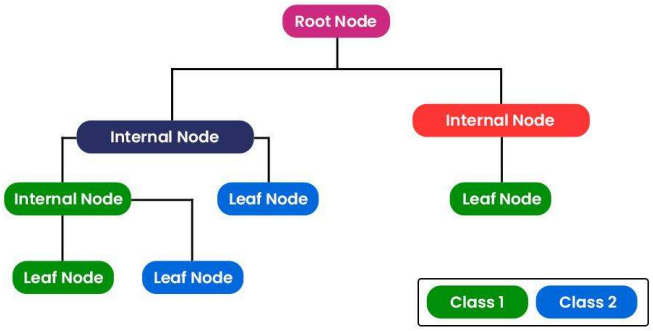
- **Tree structure:** CART builds a tree-like structure consisting of nodes and branches. The nodes represent different decision points, and the branches represent the possible outcomes of those decisions. The leaf nodes in the tree contain a predicted class label or value for the target variable.
- **Splitting criteria:** CART uses a greedy approach to split the data at each node. It evaluates all possible splits and selects the one that best reduces the impurity of the resulting subsets. For classification tasks, CART uses Gini impurity as the splitting criterion. The lower the Gini impurity, the more pure the subset is. For regression tasks, CART uses residual reduction as the splitting criterion. The lower the residual reduction, the better the fit of the model to the data.
- **Pruning:** To prevent overfitting of the data, pruning is a technique used to remove the nodes that contribute little to the model accuracy. Cost complexity pruning and information gain pruning are two popular pruning techniques. Cost complexity pruning involves calculating the cost of each node and removing nodes that have a negative cost. Information gain pruning involves calculating the information gain of each node and removing nodes that have a low information gain.

How does CART algorithm works

The CART algorithm works via the following process:

- The best-split point of each input is obtained.

- Based on the best-split points of each input in Step 1, the new “best” split point is identified.
- Split the chosen input according to the “best” split point.
- Continue splitting until a stopping rule is satisfied or no further desirable splitting is available.



CART algorithm uses Gini Impurity to split the dataset into a decision tree .It does that by searching for the best homogeneity for the sub nodes, with the help of the Gini index criterion.

How Does CART for Classification Work

CART for classification works by recursively splitting the training data into smaller and smaller subsets based on certain criteria. The goal is to split the data in a way that minimizes the impurity within each subset. Impurity is a measure of how mixed up the data is in a particular subset. For classification tasks, CART uses Gini impurity

- **Gini Impurity-** Gini impurity measures the probability of misclassifying a random instance from a subset labeled according to the majority class. Lower Gini impurity means more purity of the subset.
- **Splitting Criteria-** The CART algorithm evaluates all potential splits at every node and chooses the one that best decreases the Gini impurity of the resultant subsets. This process continues until a stopping criterion is reached, like a maximum tree depth or a minimum number of instances in a leaf node.

How Does CART works for Regression

Regression CART works by splitting the training data recursively into smaller subsets based on specific criteria. The objective is to split the data in a way that minimizes the residual reduction in each subset.

- **Residual Reduction-** Residual reduction is a measure of how much the average squared difference between the predicted values and the actual values for the target variable is reduced by splitting the subset. The lower the residual reduction, the better the model fits the data.

- **Splitting Criteria-** CART evaluates every possible split at each node and selects the one that results in the greatest reduction of residual error in the resulting subsets. This process is repeated until a stopping criterion is met, such as reaching the maximum tree depth or having too few instances in a leaf node.

4) What is Ensemble Learning explain with techniques? Explain Bagging & Boosting algorithms?

Ans) Ensemble means **'a collection of things'** and in Machine Learning terminology, Ensemble learning refers to the approach of combining multiple ML models to produce a more accurate and robust prediction compared to any individual model. It implements an ensemble of fast algorithms (classifiers) such as decision trees for learning and allows them to vote.

Ensemble Learning with examples

- Ensemble learning is a machine learning technique that combines the predictions from multiple individual models to obtain a better predictive performance than any single model. The basic idea behind ensemble learning is to leverage the wisdom of the crowd by aggregating the predictions of multiple models, each of which may have its own strengths and weaknesses. This can lead to improved performance and generalization.
- Several individual base models (experts) are fitted to learn from the same data and produce an aggregation of output based on which a final decision is taken. These base models can be machine learning algorithms such as decision trees (mostly used), linear models, support vector machines (SVM), neural networks, or any other model that is capable of making predictions.
- Most commonly used ensembles include techniques such as Bagging- used to generate Random Forest algorithms and Boosting- to generate algorithms such as Adaboost, Xgboost etc.

Ensemble Learning Techniques

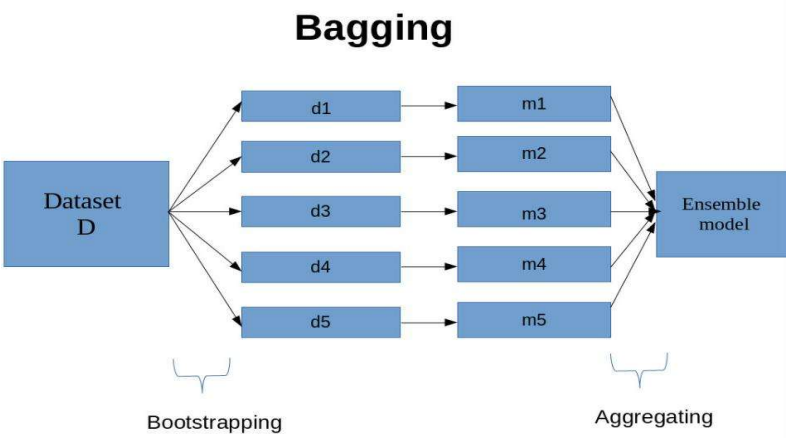
- **Gradient Boosting Machines (GBM):** Gradient Boosting is a popular ensemble learning technique that sequentially builds a group of decision trees and corrects the residual errors made by previous trees, enhancing its predictive accuracy. It trains each new weak learner to fit the residuals of the previous ensemble's predictions thus making it less sensitive to individual data points or outliers in the data.
- **Extreme Gradient Boosting (XGBoost)**: XGBoost features tree pruning, regularization, and parallel processing, which makes it a preferred choice for data scientists seeking robust and accurate predictive models.
- **CatBoost:** It is designed to handle features categorically that eliminates the need for extensive pre-processing.CatBoost is known for its high predictive accuracy, fast training, and automatic handling of overfitting.

- **Stacking:** It combines the output of multiple base models by training a combiner (an algorithm that takes predictions of base models) and generate more accurate prediction. Stacking allows for more flexibility in combining diverse models, and the combiner can be any machine learning algorithm.
- **Random Subspace Method (Random Subspace Ensembles):** It is an ensemble learning approach that improves the predictive accuracy by training base models on random subsets of input features. It mitigates overfitting and improves the generalization by introducing diversity in the model space.
- **Random Forest Variants:** They introduce variations in tree construction, feature selection, or model optimization to enhance performance.

Algorithm based on Bagging and Boosting

Bagging Algorithm

Bagging is a supervised learning technique that can be used for both regression and classification tasks. Here is an overview of the steps including Bagging classifier algorithm:

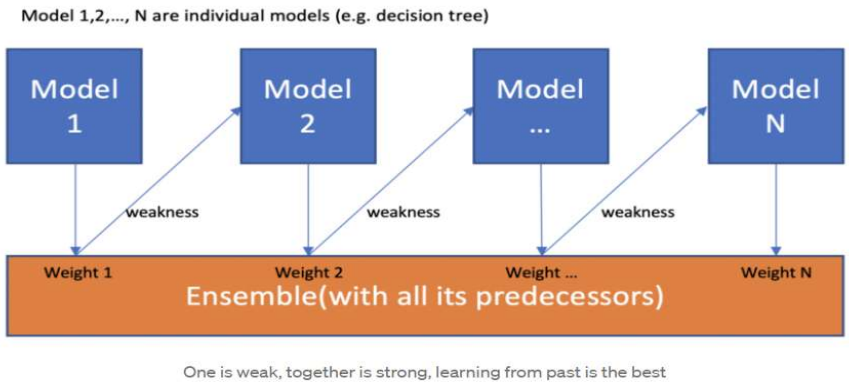


- **Bootstrap Sampling:** Divides the original training data into 'N' subsets and randomly selects a subset with replacement in some rows from other subsets. This step ensures that the base models are trained on diverse subsets of the data and there is no class imbalance.
- **Base Model Training**: For each bootstrapped sample, train a base model independently on that subset of data. These weak models are trained in parallel to increase computational efficiency and reduce time consumption.
- **Prediction Aggregation:** To make a prediction on testing data combine the predictions of all base models. For classification tasks, it can include majority voting or weighted majority while for regression, it involves averaging the predictions.

- **Out-of-Bag (OOB) Evaluation**: Some samples are excluded from the training subset of particular base models during the bootstrapping method. These “out-of-bag” samples can be used to estimate the model’s performance without the need for cross-validation.
- **Final Prediction:** After aggregating the predictions from all the base models, Bagging produces a final prediction for each instance.

Boosting Algorithm

Boosting is an ensemble technique that combines multiple weak learners to create a strong learner. The ensemble of weak models are trained in series such that each model that comes next, tries to correct errors of the previous model until the entire training dataset is predicted correctly. One of the most well-known boosting algorithms is AdaBoost (Adaptive Boosting).



One is weak, together is strong, learning from past is the best

Here are few popular boosting algorithm frameworks:

- **AdaBoost (Adaptive Boosting):** AdaBoost assigns different weights to data points, focusing on challenging examples in each iteration. It combines weighted weak classifiers to make predictions.
- **Gradient Boosting:** Gradient Boosting, including algorithms like Gradient Boosting Machines (GBM), XGBoost, and LightGBM, optimizes a loss function by training a sequence of weak learners to minimize the residuals between predictions and actual values, producing strong predictive models.

Uses of Ensemble Learning

Ensemble learning is a versatile approach that can be applied to a wide range of machine learning problems such as:-

- **Classification and Regression:** Ensemble techniques make problems like classification and regression versatile in various domains, including finance, healthcare, marketing, and more.
- **Anomaly Detection:** Ensembles can be used to detect anomalies in datasets by combining multiple anomaly detection algorithms, thus making it more robust.

- **Portfolio Optimization:** Ensembles can be employed to optimize investment portfolios by collecting predictions from various models to make better investment decisions.
- **Customer Churn Prediction:** In business and marketing analytics, by combining the results of various models capturing different aspects of customer behaviour, ensembles can be used to predict customer churn.
- **Medical Diagnostics:** In healthcare, ensembles can be used to make more accurate predictions of diseases based on various medical data sources and diagnostic models.
- **Credit Scoring:** Ensembles can be used to improve the accuracy of credit scoring models by combining the outputs of various credit risk assessment models.
- **Climate Prediction:** Ensembles of climate models help in making more accurate and reliable predictions for weather forecasting, climate change projections, and related environmental studies.
- **Time Series Forecasting**: Ensemble learning combines multiple time series forecasting models to enhance accuracy and reliability, adapting to changing temporal patterns.

5) Explain K-Means Clustering Algorithm with neat diagrams?

Ans) 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.

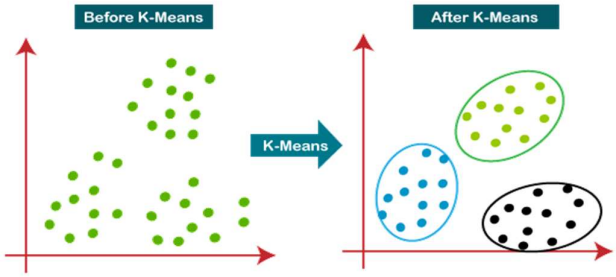
The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

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.

Advertisement

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.

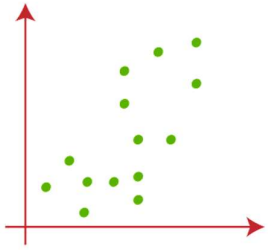
Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

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

Step-7: The model is ready.

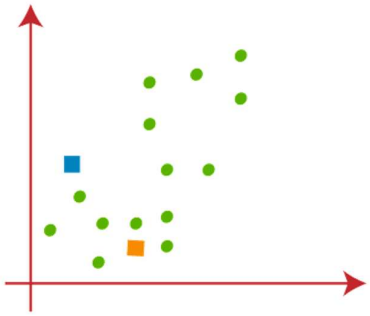
Let's understand the above steps by considering the visual plots:

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

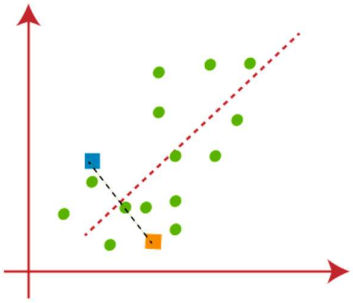


- Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.

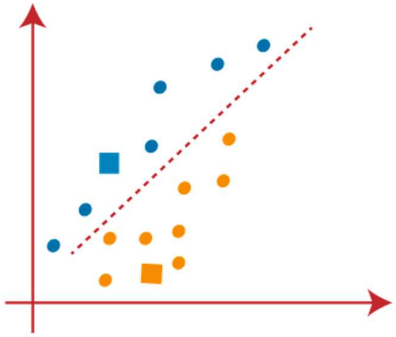
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the below image:



- 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:

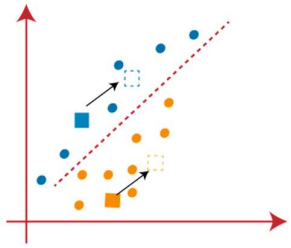


From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.

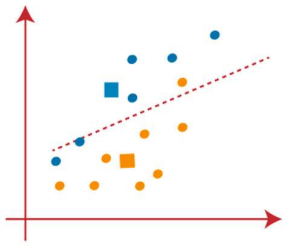


- As we need to find the closest cluster, so we will repeat the process by choosing **a new centroid**. To choose the new centroids, we will compute the center of gravity of

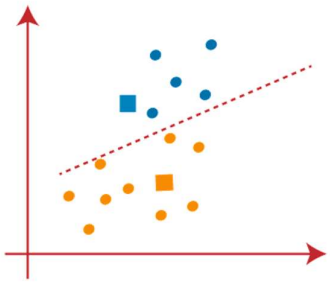
these centroids, and will find new centroids as below:



- 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:

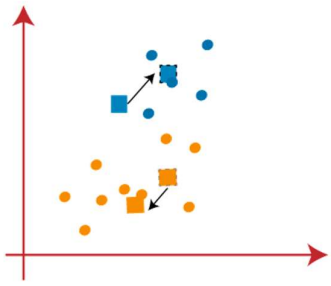


From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.

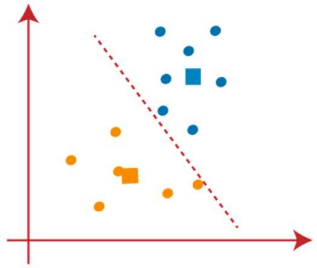


As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

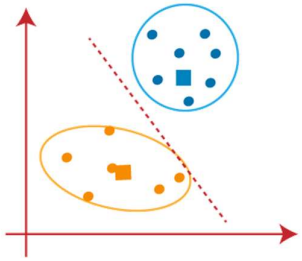
- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:



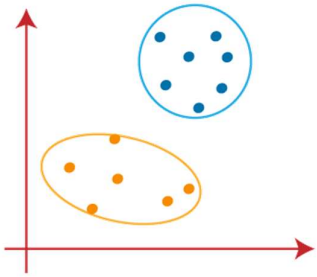
- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:



- 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:

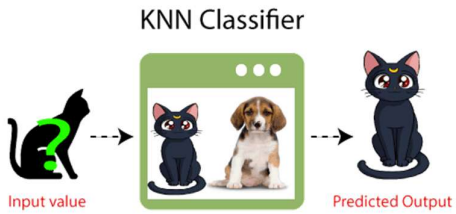


6) Explain K-Nearest Neighbor (KNN) Algorithm with neat diagrams?

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

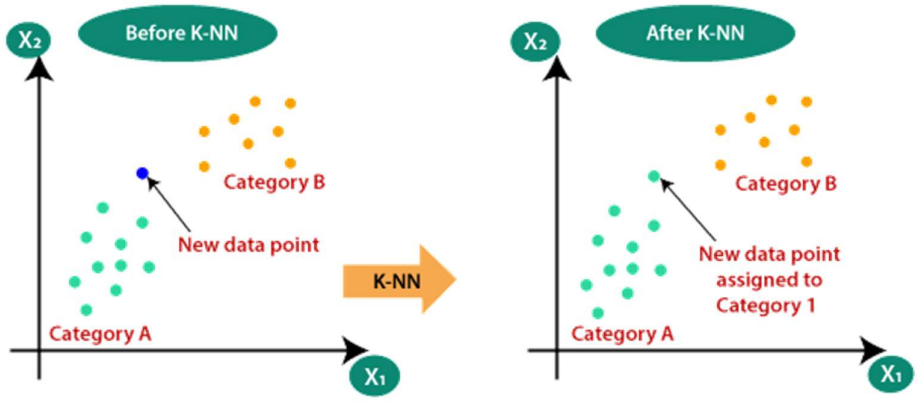
- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- K-NN algorithm 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:

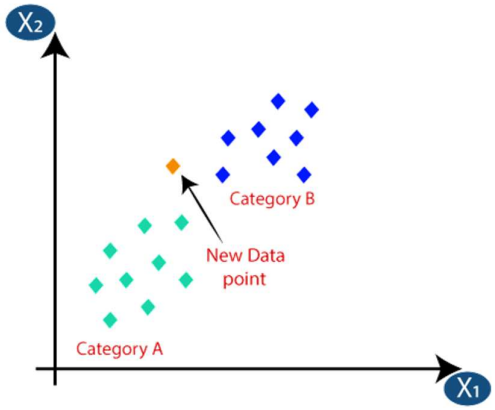


How does K-NN work?

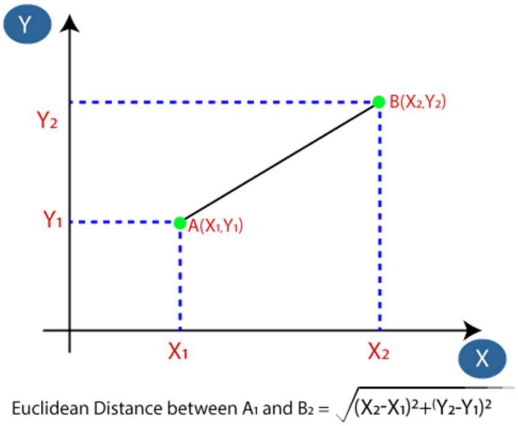
The K-NN working can be explained on the basis of the below algorithm:

- **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.

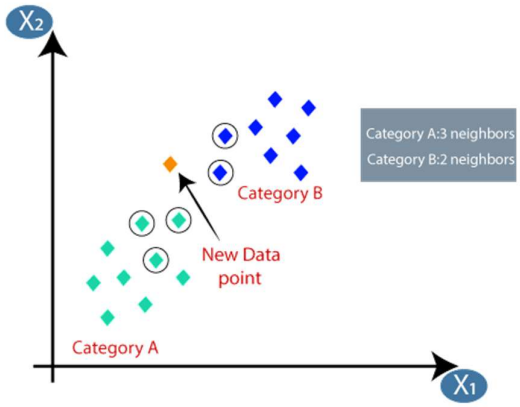
Suppose we have a new data point and we need to put it in the required category. Consider the below image:



- 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.

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.

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.