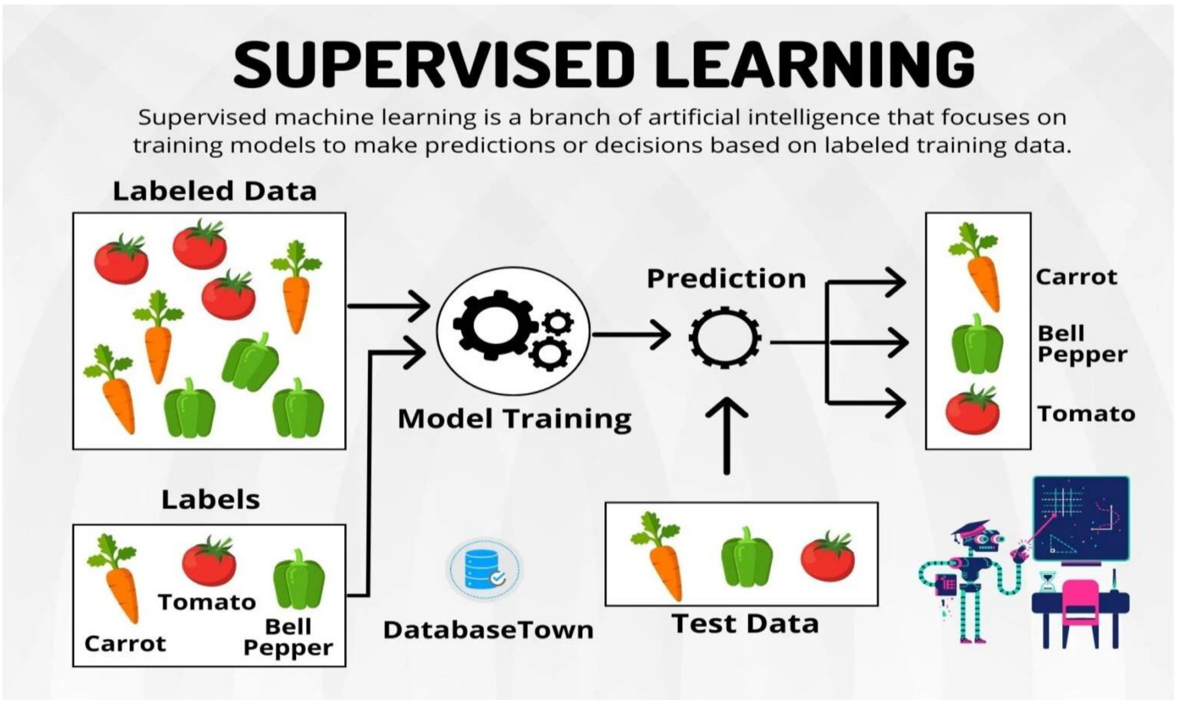
UNIT-4

Classification & Regression Algorithms

4.1 Supervised learning

- Supervised learning is the type of machine learning in which machines are trained using well "labeled" training data, and based on that data, machines predict the output. The labeled data means some input data is already tagged with the correct output.
- Examples of supervised learning are as follows:

  How does supervised learning work?


SUPERVISED LEARNING
Supervised machine learning is a branch of artificial intelligence that focuses on training models to make predictions or decisions based on labeled training data.
Labeled Data
Labels
Carrot
Tomato
Bell Pepper
Model Training
Prediction
DatabaseTown
Test Data
Carrot
Bell Pepper
Tomato

4.2 Types of Supervised Machine Learning Algorithms

- There are two types of supervised learning algorithms:
  1. Classification
  2. Regression

4.2.1 Classification

- Classification algorithms are used when the output variable is categorical, which means there are two classes Yes-No, Male-Female, True-false, etc.

- For example:

  A bank may have a customer dataset containing credit history, loans, investment details, etc. and they may want to know if any customer will default. In the historical data, we will have Features and Targets.

  - Features will be attributes of a customer such as credit history, loans, investments, etc.
  - Target will represent whether a particular customer has defaulted in the past (normally represented by 1 or 0 / True or False / Yes or No.)

- Classification algorithms are used for predicting discrete outcomes, if the outcome can take two possible values such as True or False, Default or No, or Yes or No, it is known as Binary Classification.
- When the outcome contains more than two possible values, it is known as Multiclass Classification.

  Many machine learning algorithms can be used for classification tasks. Some of them are:

  - Logistic Regression
  - Decision Tree Classifier
  - K Nearest Neighbour Classifier
  - Random Forest Classifier
  - Neural Networks

- **Binary Classifier:** If the classification problem has only two possible outcomes, it is called a Binary Classifier.

  Example: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.

- **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called a Multi-class Classifier.

  Example: Classifications of types of crops, Classification of types of music.

- Classification Algorithms can be further divided into the Mainly two main categories:
    - **Linear Models**
        - Logistic Regression
        - Support Vector Machines
    - **Non-linear Models**
        - K-Nearest Neighbours
        - Naïve Bayes
        - Decision Tree Classification
        - Random Forest Classification

4.2.2 Learner in Classification

- **Lazy Learners:** Lazy Learner first stores the training dataset and waits until it receives the test dataset.

  In the Lazy learner case, classification is done based on the most related data stored in the training dataset.
  It takes less time in training but more time for predictions.
  Example: K-NN algorithm, Case-based reasoning
- **Eager Learners:** Eager Learners develop a classification model based on a training dataset before receiving a test dataset.

  Opposite to Lazy learners, Eager Learner takes more time in learning and less time in prediction.
  Example: Decision Trees, Naïve Bayes, ANN.

4.2.3 Regression

- Regression algorithms are used if there is a relationship between the input variable and the output variable.
- It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc.

Types of Regression in Supervised Learning.

- **Linear Regression:** Here, we have only one independent variable to predict the output, i.e., the dependent variable.
- **Multiple Regression:** Here, we have more than one independent variable to predict the output, i.e., the dependent variable.
- **Polynomial Regression:** The graph between the dependent and independent variables follows a polynomial function.

Regression	**Classification**
Regression is the task of predicting a continuous quantity.	Classification is the task of predicting a discrete class label.
Regression Means to predict the output value using training data.	Classification means to group the output into a class.
A regression problem requires the prediction of a quantity.	In a classification problem data is labelled into one of two or more class.
If it is a real number or continuous then it is regression problem.	If it is discrete or categorical variable, then it is classification problem.
A regression problem with multiple input variables is called a multivariable regression problem.	A classification problem with two classes is called binary, more than 2 classes is called as multi-classification problem.
Example : Predict the house prices.	**Example** : Is that E-mail is Spam or not a Spam.

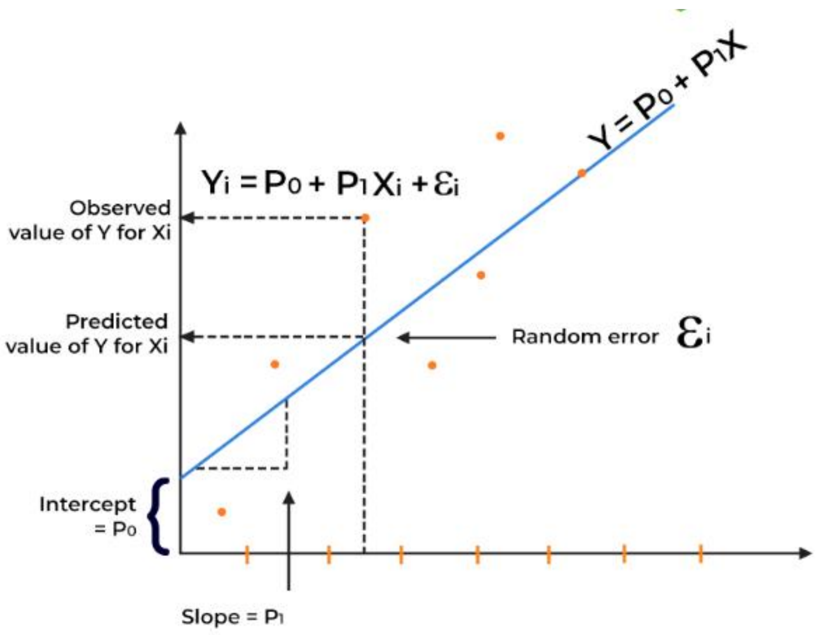
4.3. Supervised Machine Learning Algorithms:

- Linear Regression
- Support Vector Machines
- Decision Trees
- Random Forest
- Logistic Regression
- K-NN
- Naïve Bayes.

4.3.1 Linear Regression Algorithm

- Linear regression is a statistical method that predicts the relationship between two variables and is used for predictive analysis
- Linear regression makes predictions for continuous/real or numeric variables such as **sales, salary, age, product price,** etc.

  Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called linear regression.


$Y = P_0 + P_1X$
$Y_i = P_0 + P_1X_i + \varepsilon_i$
Observed value of Y for Xi
Predicted value of Y for Xi
Random error ε_i
Intercept $= P_0$
Slope $= P_1$

- Mathematical equation for linear regression:

- $y = p0 + p1x + \varepsilon$ is the formula used for simple linear regression.

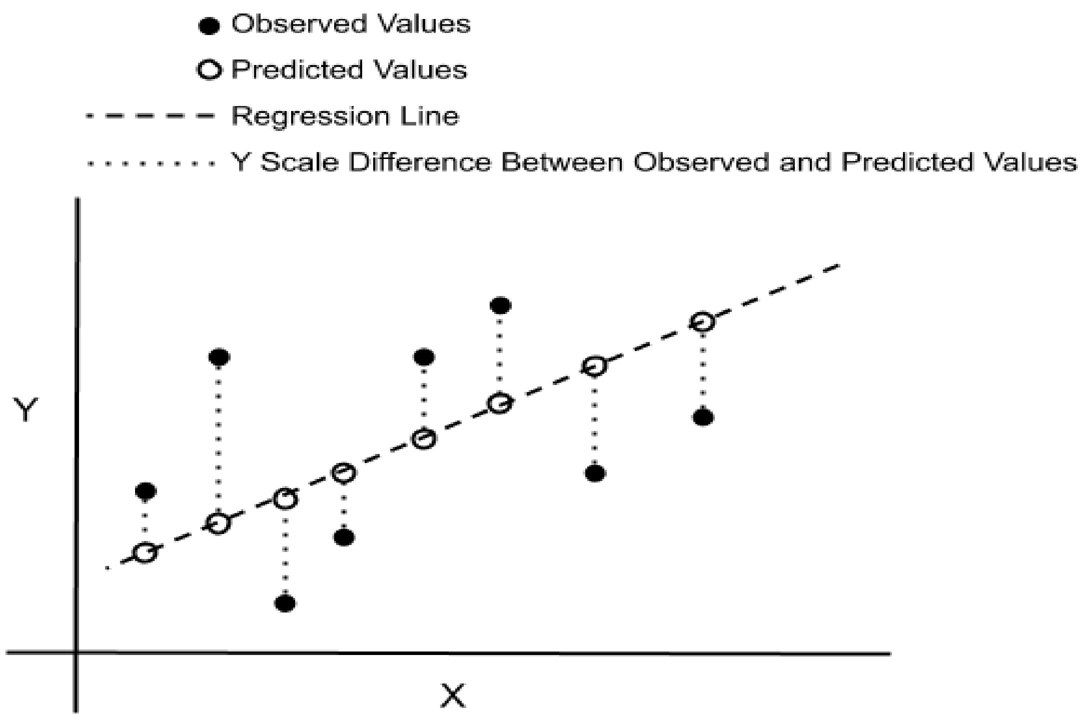
  y is the predicted value of the dependent variable (y) for any given value of the independent variable (x).

  p0 is the intercept, the predicted value of y when the x is 0.

  p1 is the regression coefficient – how much we expect y to change as x increases.

  x is the independent variable (the variable we expect is influencing y).

  ε (Random error) is the error of the estimate, or how much variation there is in our regression coefficient estimate.



● Observed Values
O Predicted Values
- - - - Regression Line
. . . . Y Scale Difference Between Observed and Predicted Values
Y
X

- **Assumption for Linear Regression Model**

  Linear regression is a powerful tool for understanding and predicting the behavior of a variable, however, it needs to meet a few conditions to be accurate and dependable solutions.

- **Linearity**: The independent and dependent variables have a linear relationship with one another. This implies that changes in the dependent variable follow those in the independent variable(s) linearly.
- **Independence**: The observations in the dataset are independent of each other. This means that the value of the dependent variable for one observation does not depend on the value of the dependent variable for another observation.
- **Homoscedasticity**: Across all levels of the independent variable(s), the variance of the errors is constant. This indicates that the amount of the independent variable(s) has no impact on the variance of the errors.
- **Normality**: The errors in the model are normally distributed.
- **No multicollinearity**: There is no high correlation between the independent variables. This indicates that there is little or no correlation between the independent variables.

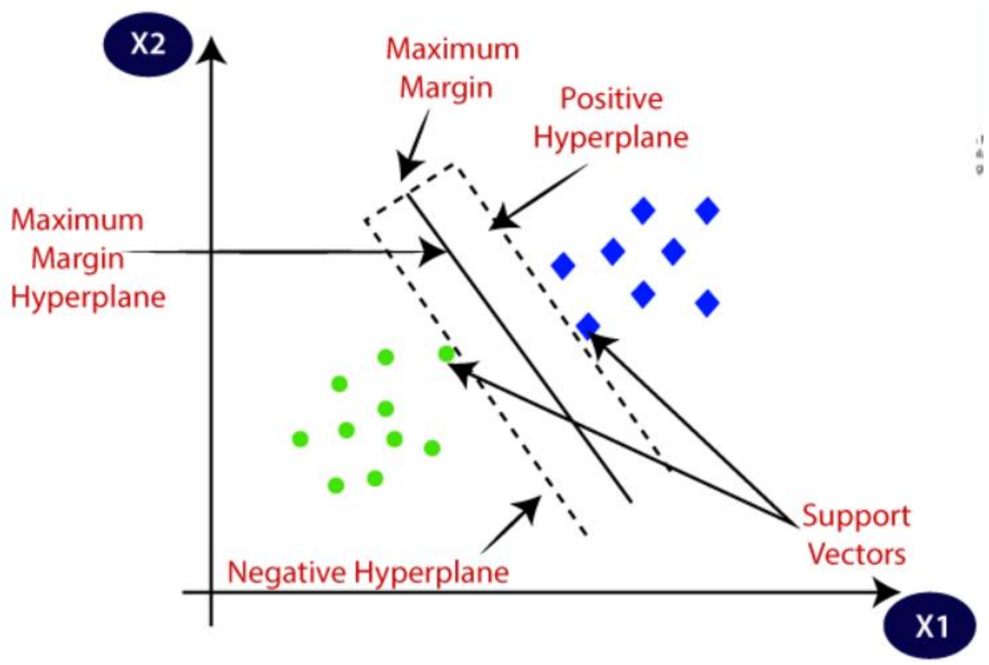
4.3.2 Support Vector Machines Algorithm

- Support Vector Machine (SVM) is a supervised machine learning algorithm that can be used for both classification and regression challenges. it is mostly used in classification problems.
- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

Support Vector Machine Terminology

Hyperplane: Hyperplane is the decision boundary that is used to separate the data points of different classes in a feature space. In the case of linear classifications, it will be a linear equation i.e. $wx+b = 0$.

The **hyperplane** with the maximum margin is called the **optimal hyperplane**.

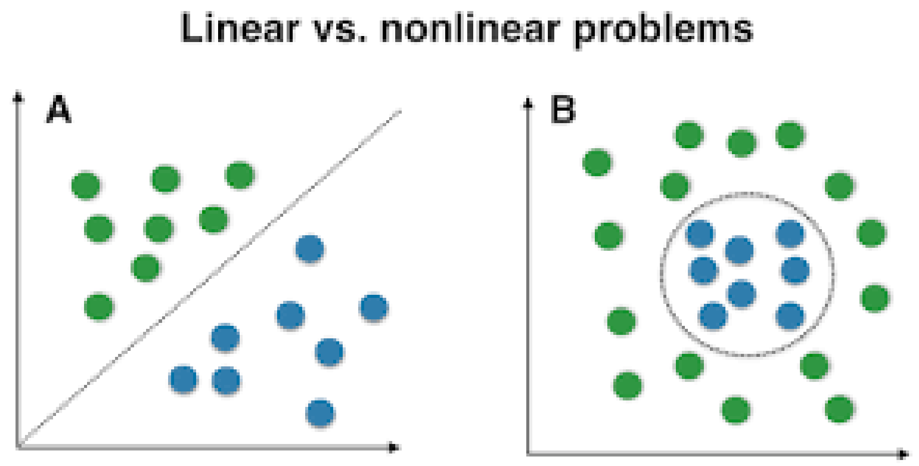


- **Support Vectors:** Support vectors are the closest data points to the hyperplane, which plays a critical role in deciding the hyperplane and margin.
- **Margin**: Margin is the distance between the support vector and hyperplane. The main objective of the support vector machine algorithm is to maximize the margin. The wider margin indicates better classification performance.
- **Kernel**: Kernel is the mathematical function, which is used in SVM to map the original input data points into high-dimensional feature spaces, so, that the hyperplane can be easily found out even if the data points are not linearly separable in the original input space. Some of the common kernel functions are linear, polynomial, radial basis function (RBF), and sigmoid.
- **Hard Margin:** The maximum-margin hyperplane or the hard margin hyperplane is a hyperplane that properly separates the data points of different categories without any misclassifications.
- **Soft Margin:** When the data is not perfectly separable or contains outliers, SVM permits a soft margin technique. Each data point has a slack variable introduced by the soft-margin SVM formulation, which softens the strict margin requirement and permits certain misclassifications or violations. It

discovers a compromise between increasing the margin and reducing violations.

SVM can be of two types:

- **Linear SVM**: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called Linear SVM classifier.
- **Non-linear SVM**: Non-linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data, and the classifier used is called a Non-linear SVM classifier



- **Strengths of SVM**
  - SVM can be used for both classification and regression.
  - It is robust, i.e. not much impacted by data with noise or outliers.
  - The prediction results using this model are very promising.
- **Weaknesses of SVM**
  - SVM is applicable only for binary classification, i.e. when there are only two classes in the problem domain.

- The SVM model is very complex – almost like a black box when it deals with a high-dimensional data set. Hence, it is very difficult and close to impossible to understand the model in such cases.
- It is slow for a large dataset, i.e. a data set with either a large number of features or a large number of instances.

4.3.3 K-Nearest Neighbours Algorithm

K-nearest neighbors (K-NN) algorithm is a type of supervised ML algorithm that can be used for both classification as well as regression.

How does K-NN work?

The K-NN working can be explained based on 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 neighbors is maximum.
- **Step 6:** Our model is ready.

Why the K-NN algorithm is called a lazy learner?

Eager learners follow the general steps of machine learning, i.e. perform an abstraction of the information obtained from the input data and then follow it through by a generalization step. However, as we have seen in the case of the K-NN algorithm, these steps are completely skipped. It stores the training data and directly applies the philosophy of nearest neighborhood finding to arrive at the classification. So, for K-NN, there is no learning happening in the real sense. Therefore, K-NN falls under the category of lazy learner.

- **Strengths of the k-NN algorithm**
  - Extremely simple algorithm – easy to understand

    - Very effective in certain situations, eg. for recommender system design
    - Very fast or almost no time is required for the training phase

- **Weaknesses of the k-NN algorithm**

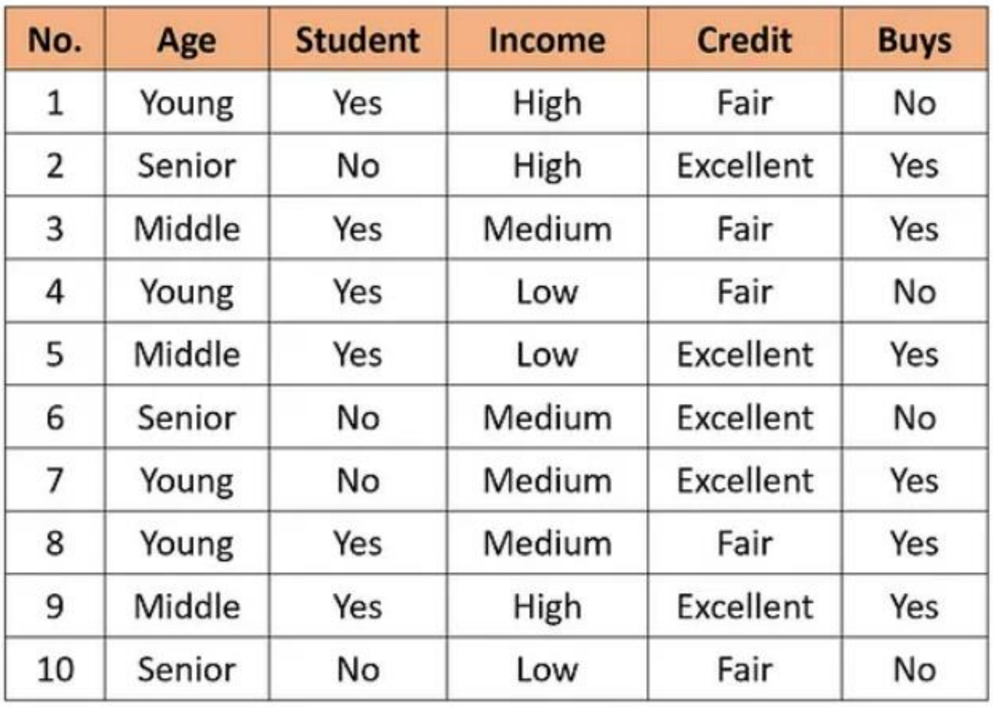
  Classification is done completely based on the training data. So, it has a heavy reliance on the training data. If the training data does not represent the problem domain comprehensively, the algorithm fails to make an effective classification. The classification process is very slow. Also, a large amount of computational space is required to load the training data for classification.

4.3.4 Naïve Bayes Algorithm

- The Naïve Bayes algorithm is a supervised learning algorithm, which is based on the **Bayes theorem** and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- **It is a probabilistic classifier, which means it predicts based on the probability of an object**.
- Some popular examples of Naïve Bayes Algorithm are **spam filtration, Sentiment analysis, and classifying articles**.
- **Naïve**: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features. Such as if the fruit is identified based on color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple. Hence each feature individually contributes to identifying that it is an apple without depending on each other.
- **Bayes**: It is called Bayes because it depends on the principle of Bayes' Theorem.

Example: Customer Purchase Prediction

Table with data on the past purchases of customers in a store:



No.	Age	Student	Income	Credit	Buys
1	Young	Yes	High	Fair	No
2	Senior	No	High	Excellent	Yes
3	Middle	Yes	Medium	Fair	Yes
4	Young	Yes	Low	Fair	No
5	Middle	Yes	Low	Excellent	Yes
6	Senior	No	Medium	Excellent	No
7	Young	No	Medium	Excellent	Yes
8	Young	Yes	Medium	Fair	Yes
9	Middle	Yes	High	Excellent	Yes
10	Senior	No	Low	Fair	No

- The training set

  Each row in the table contains the age of the customer, whether they are a student or not, their level of income, their credit rating, and whether or not they have purchased the product.

  A new customer with the following properties arrives at the store: <Age = Young, Student = Yes, Income = Low, Credit = Excellent>

  You need to predict whether this customer will buy the product or not.

  We first compute the class prior probabilities by counting the number of rows that have Buys = Yes (6 out of 10) and the number of rows that have Buys = No (4 out of 10):

$$P(\text{Buys} = \text{Yes}) = 6/10 = 0.6$$
$$P(\text{Buys} = \text{No}) = 4/10 = 0.4$$

$$P(\text{Age} = \text{Young}|\text{Buys} = \text{Yes}) = 2/6 = 0.333$$
$$P(\text{Age} = \text{Young}|\text{Buys} = \text{No}) = 2/4 = 0.5$$

$$P(\text{Student} = \text{Yes}|\text{Buys} = \text{Yes}) = 4/6 = 0.667$$
$$P(\text{Student} = \text{Yes}|\text{Buys} = \text{No}) = 2/4 = 0.5$$

$$P(\text{Income} = \text{Low}|\text{Buys} = \text{Yes}) = 1/6 = 0.167$$
$$P(\text{Income} = \text{Low}|\text{Buys} = \text{No}) = 2/4 = 0.5$$

$$P(\text{Credit} = \text{Excellent}|\text{Buys} = \text{Yes}) = 4/6 = 0.667$$
$$P(\text{Credit} = \text{Excellent}|\text{Buys} = \text{No}) = 1/4 = 0.25$$

Then, we compute the likelihood of the features in each class:

Therefore, the class posterior probabilities are:

α is the normalization factor (*α* = 1 / *P*(**x**)).

Since *P*(Buys = Yes|**x**) > *P*(Buys = No|**x**), we predict that the customer will buy the product.

If we want to get the actual probability that the customer will buy the product, we can first find the normalization factor using the fact that the two posterior probabilities must sum to 1:

$$0.0148\alpha + 0.0125\alpha = 1 \Rightarrow \alpha = \frac{1}{0.0148 + 0.0125} = 36.63$$

Then, we can plug it in the posterior probability for Buy = Yes:

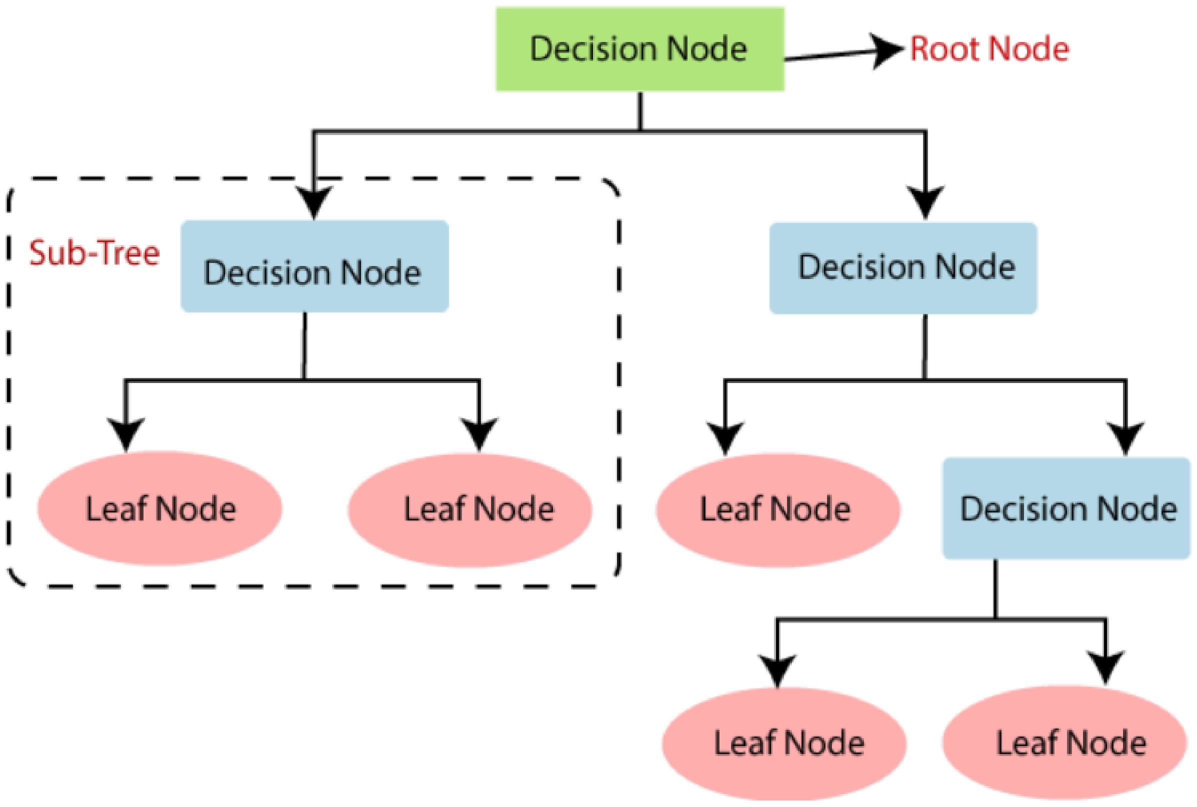
$$\begin{aligned} P(\text{Buys} = \text{Yes}|\mathbf{x}) &= P(\text{Yes}) \cdot P(\text{Young}|\text{Yes}) \cdot P(\text{Student}|\text{Yes}) \cdot P(\text{Low}|\text{Yes}) \cdot P(\text{Excellent}|\text{Yes}) \\ &= 0.6 \cdot 0.333 \cdot 0.667 \cdot 0.167 \cdot 0.667\alpha = 0.0148\alpha \end{aligned}$$

$$\begin{aligned} P(\text{Buys} = \text{No}|\mathbf{x}) &= P(\text{No}) \cdot P(\text{Young}|\text{No}) \cdot P(\text{Student}|\text{No}) \cdot P(\text{Low}|\text{No}) \cdot P(\text{Excellent}|\text{No}) \\ &= 0.4 \cdot 0.5 \cdot 0.5 \cdot 0.5 \cdot 0.25\alpha = 0.0125\alpha \end{aligned}$$

The probability that the customer will buy the product is 54.21%.

4.3.5 Decision Trees 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 tests are performed based on 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.
- 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 splits the tree into subtrees.



Decision Tree Terminologies

- **Root Node:** The 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 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.

- **Attribute Selection Measures**

  While implementing a Decision tree, the main issue arises as to how to select the best attribute for the root node and sub-nodes. So, to solve such problems there is a technique which is called an **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**
  - **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 built 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:
    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

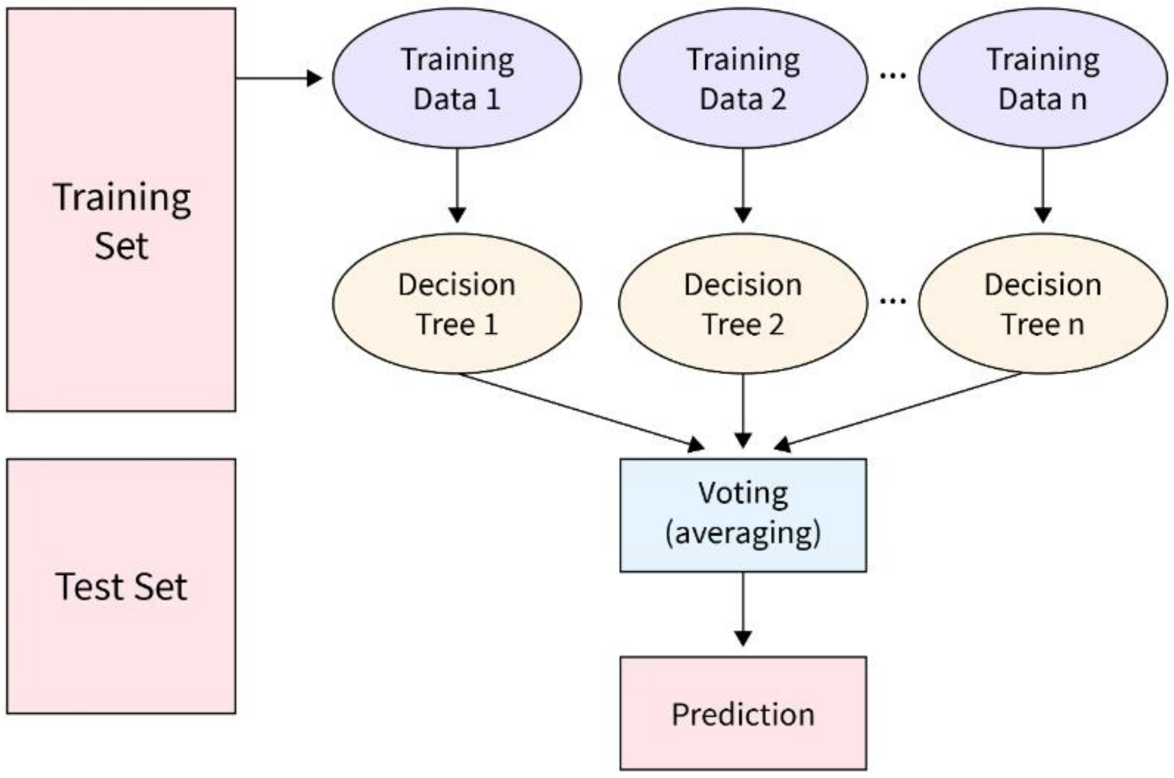
- **Gini Index:**
  - The Gini index is a measure of impurity or purity used while creating a decision tree in the CART (Classification and Regression Tree) algorithm.
  - An attribute with the low Gini index should be preferred as compared to the high Gini index.
  - It only creates binary splits, and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

$$\text{Gini Index} = 1 - \sum_j P_j^2$$

- **Strengths of decision tree**
  - It produces very simple understandable rules. For smaller trees, not much Mathematical and computational knowledge is required to understand this Model.
  - Works well for most of the problems.
  - It can handle both numerical and categorical variables.
  - It can work well both with small and large training data sets.
  - Decision trees provide a definite clue of which features are more useful for classification.
- **Weaknesses of decision tree**
  - Decision tree models are often biased towards features having more number of possible values, i.e. levels.
  - This model gets over-fitted or under-fitted quite easily.
  - Decision trees are prone to errors in classification problems with many classes and a relatively small number of training examples.
  - A decision tree can be computationally expensive to train.
  - Large trees are complex to understand.

4.3.6 Random Forest Algorithm

- Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting



- **Use of Random Forest**

  Below are some points that explain why we should use the Random Forest algorithm:
  - It takes less training time as compared to other algorithms.
  - It predicts output with high accuracy, even for a large dataset it runs efficiently.
  - It can also maintain accuracy when a large proportion of data is missing

- **Working of Random Forest Algorithm**

  Random Forest works in two phases. first is to create the random forest by combining N decision trees, and the second is to make predictions for each tree created in the first phase.

  The Working process can be explained in the below steps and diagram:

  Step 1: Select random K data points from the training set.

  Step 2: Build the decision trees associated with the selected data points (Subsets).

  Step 3: Choose the number N for decision trees that you want to build.

  Step 4: Repeat Steps 1 & 2.

  Step 5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

- **Strengths of random forest**
  - It runs efficiently on large and expansive data sets.
  - It has a robust method for estimating missing data and maintains precision when a large proportion of the data is absent.
  - It has powerful techniques for balancing errors in a class population of unbalanced data sets.
  - It gives estimates (or assessments) about which features are the most important ones in the overall classification.
  - It generates an internal unbiased estimate (gauge) of the generalization error as the forest generation progresses.
  - Generated forests can be saved for future use on other data.
  - Lastly, the random forest algorithm can be used to solve both classification and regression problems.

- **Weaknesses of random forest**
  - This model, because it combines several decision tree models, is not as easy to understand as a decision tree model.
  - It is computationally much more expensive than a simple model like a decision tree.

- **Advantages of Supervised learning:**
  - With the help of supervised learning, the model can predict the output based on prior experiences.
  - It performs classification and regression tasks.
  - In supervised learning, we can have an exact idea about the classes of objects.
  - The supervised learning model helps us to solve various real-world problems such as fraud detection, spam filtering, etc.
- **Disadvantages of supervised learning:**
  - Supervised learning models are not suitable for handling complex tasks.
  - Supervised learning cannot predict the correct output if the test data is different from the training dataset.
  - Training requires lots of computation time.
  - In supervised learning, we need enough knowledge about the classes of objects.