

Classification in Supervised Learning Techniques

Madushanka Wadumulla
PGC in IA
Beng, IIESL, AMEI



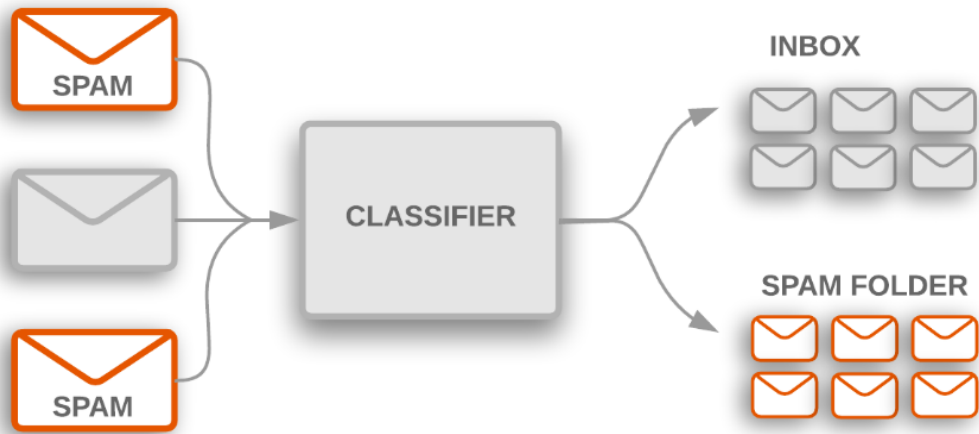
Outline

- Qualitative Responses
- Classification
- Logistic Regression
- K-Nearest Neighbors
- DT
- SVM

Qualitative Responses

- What about the response variable is a categorical (Qualitative) variable?
- Ex:-
- Yes/ No
- True/ False
- Good/ Bad
- Pass/ Fail
- High/ Medium/ Low
- Class 01/ Class 02/ Class 03/ Class 04
- Red/ Green/ Blue etc.
- In these situations, our goal is to classify observations to these groups. This is called as the Classification.

Classification

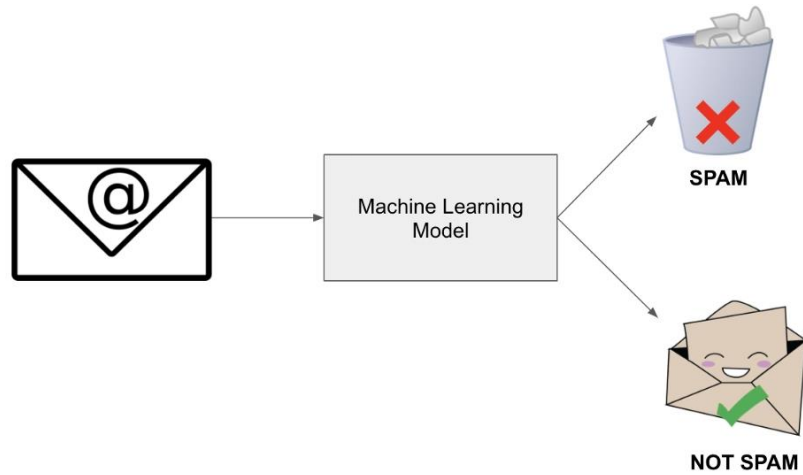


- Classification is a process of categorizing data or objects into predefined classes or categories based on their features or attributes.
- Machine Learning classification is a type of supervised learning technique where an algorithm is trained on a labeled dataset to predict the class or category of new, unseen data.
- The main objective of classification machine learning is to build a model that can accurately assign a label or category to a new observation based on its features.

Classification Types

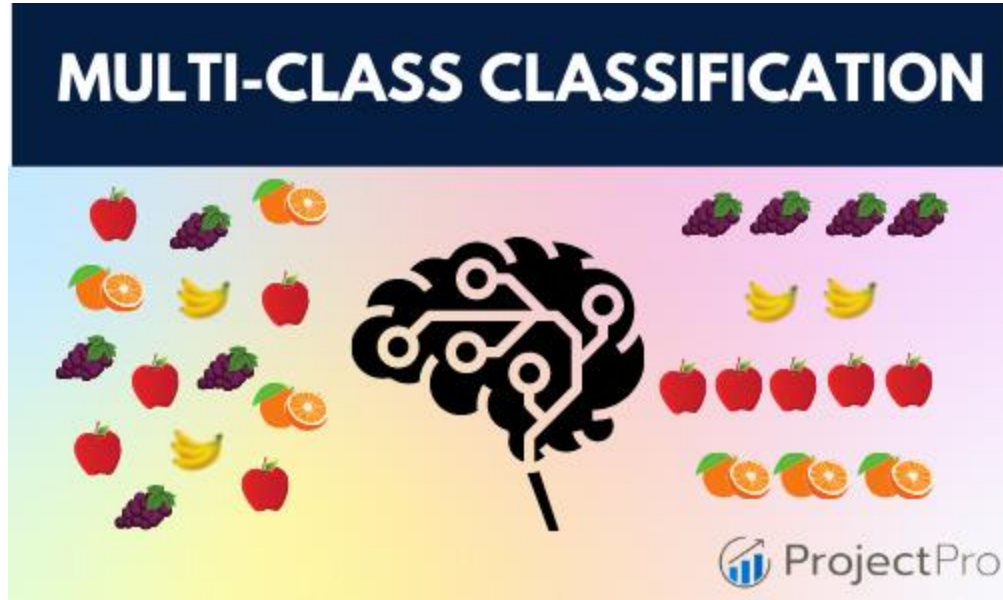
- Binary classification
- Multi-Label Classification
- Multi-Class Classification
- Imbalanced Classification

Binary classification



- A binary classification refers to those tasks which can give either of any two class labels as the output.
- Generally, one is considered as the normal state and the other is considered to be the abnormal state.
- Buy or not.
- Spam or not.
- Pass or Fail

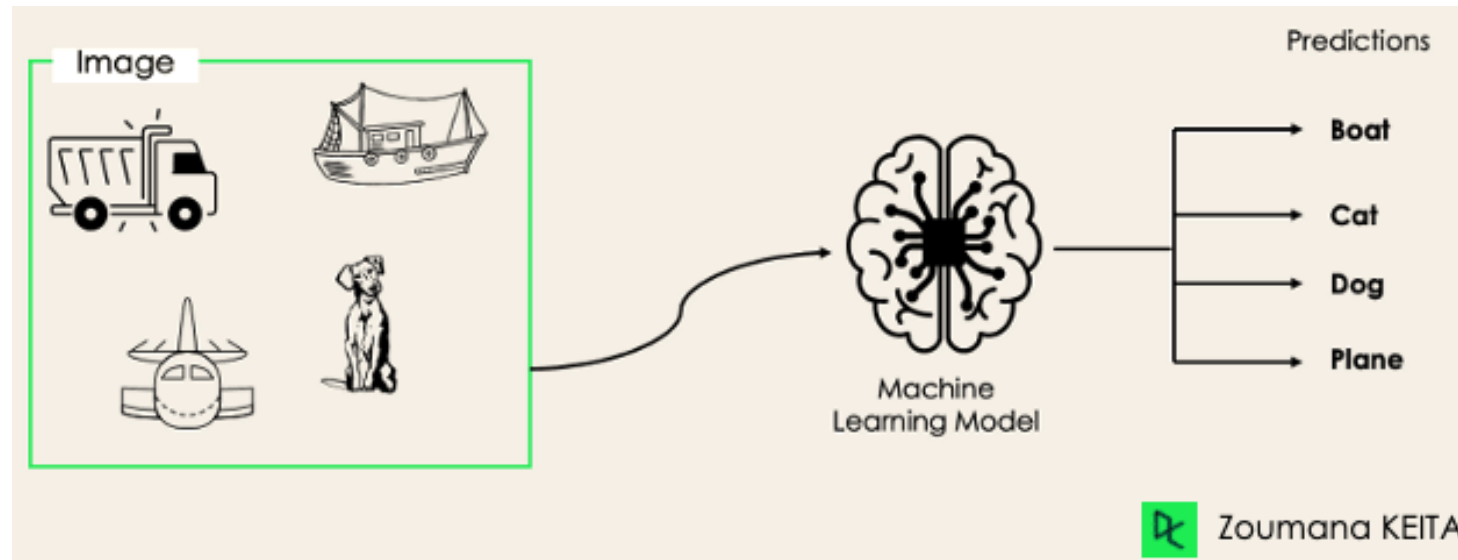
Multi-Class Classification



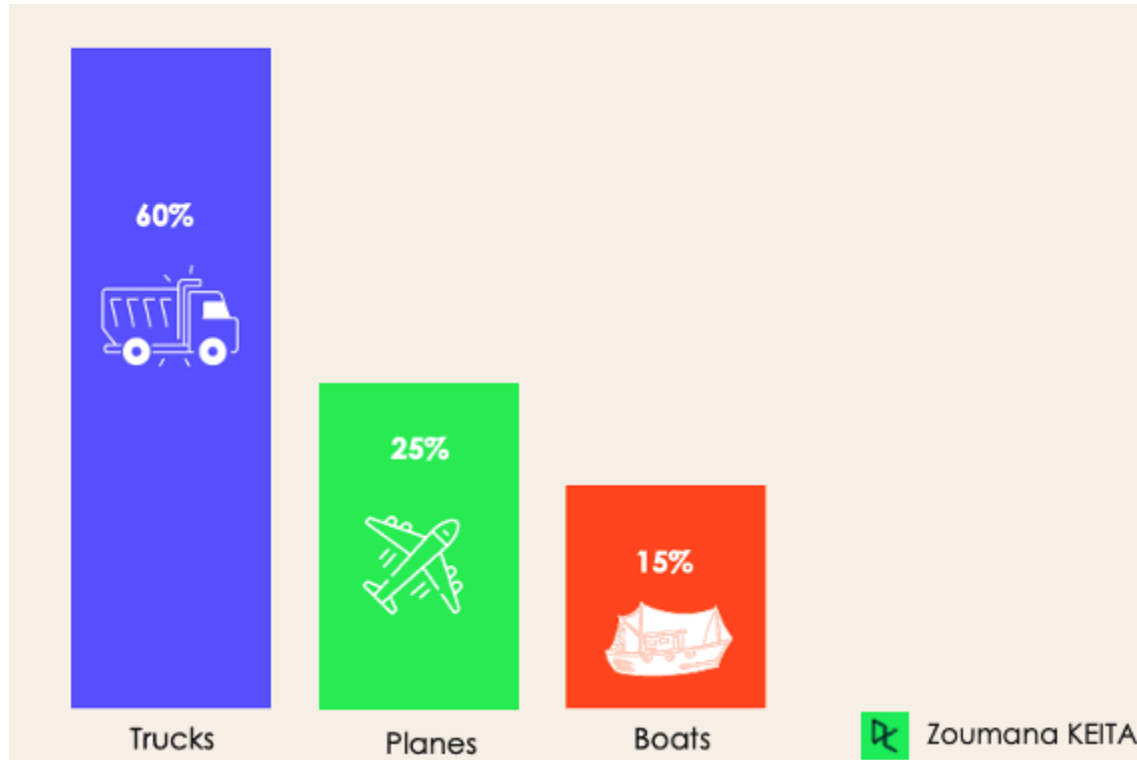
- These types of classification problems have no fixed two labels but can have any number of labels.
- Some popular examples of multi-class classification are :
 - Plant Species Classification
 - Face Classification

Multi-Label Classification

- In multi-label classification tasks, we try to predict 0 or more classes for each input example.
- In this case, there is no mutual exclusion because the input example can have more than one label.
- Such a scenario can be observed in different domains, such as auto-tagging in Natural Language Processing, where a given text can contain multiple topics.
- Similarly to computer vision, an image can contain multiple objects, as illustrated below: the model predicted that the image contains: a plane, a boat, a truck, and a dog.



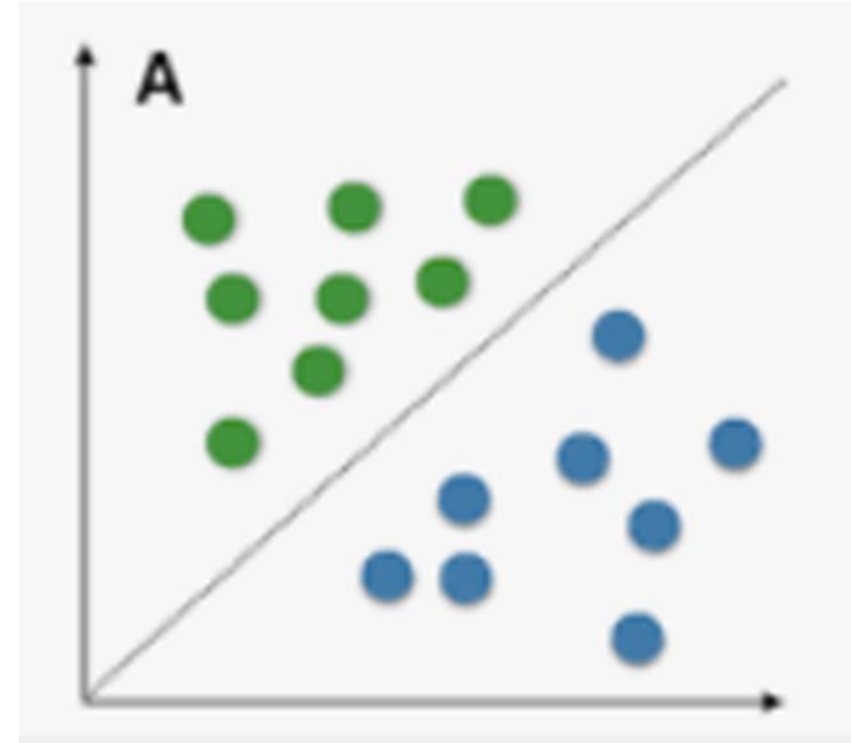
Imbalanced Classification



- For the imbalanced classification, the number of examples is unevenly distributed in each class, meaning that we can have more of one class than the others in the training data.
- Let's consider the following 3-class classification scenario where the training data contains: 60% of trucks, 25% of planes, and 15% of boats.
- The imbalanced classification problem could occur in the following scenario:
 - Fraudulent transaction detections in financial industries
 - Rare disease diagnosis

Classification Algorithms Categories

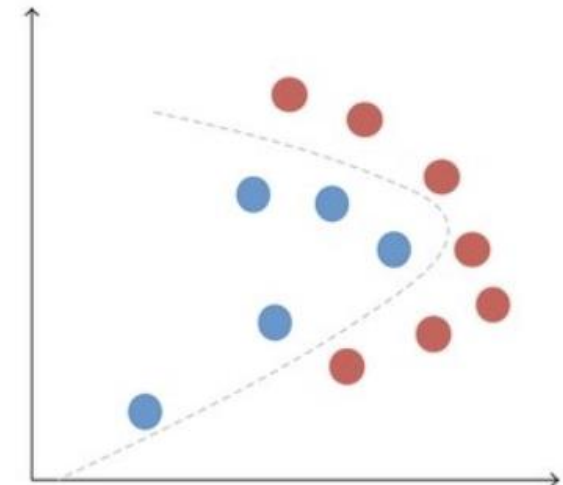
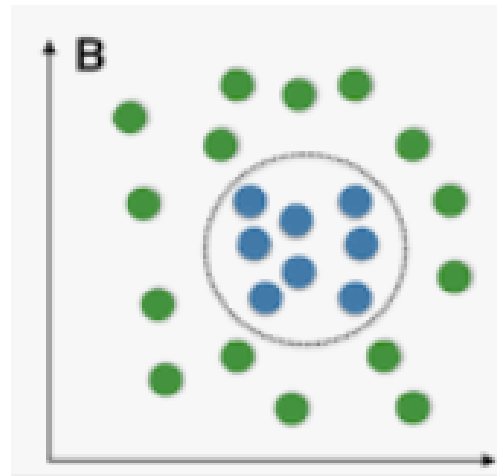
- **Linear Classifiers**
- Linear models create a linear decision boundary between classes.
- They are simple and computationally efficient.
- Some of the linear classification models are as follows:
 - **Logistic Regression**
 - **Support Vector Machines having kernel = 'linear'**
 - Single-layer Perceptron
 - Stochastic Gradient Descent (SGD) Classifier



Classification Algorithms Categories

Non Linear Classifiers

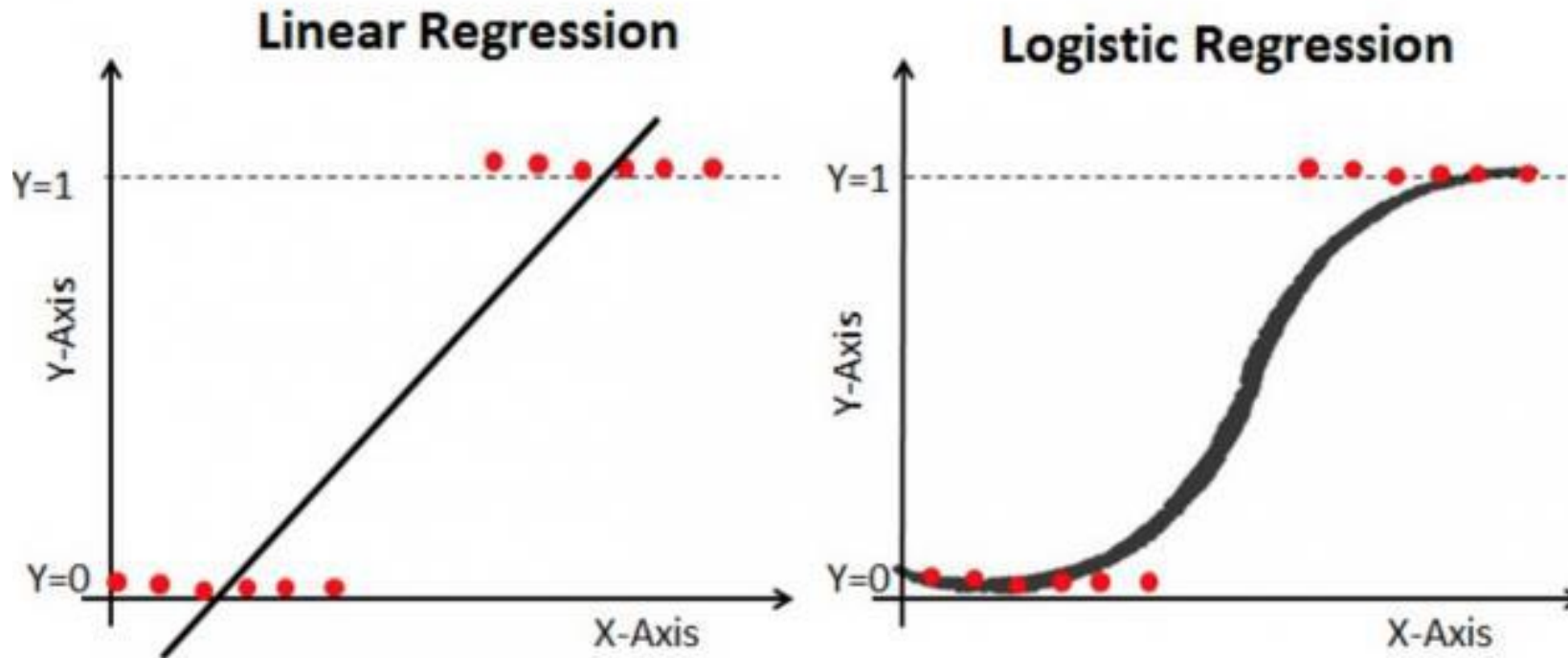
- Non-linear models create a non-linear decision boundary between classes.
- They can capture more complex relationships between the input features and the target variable.
- Some of the non-linear classification models are as follows:
 - **K-Nearest Neighbours**
 - **Kernel SVM**
 - Naive Bayes
 - **Decision Tree Classification**
 - Ensemble learning classifiers:
 - **Random Forests,**
 - AdaBoost,
 - Bagging Classifier,
 - Voting Classifier,
 - ExtraTrees Classifier
 - Multi-layer Artificial Neural Networks



Logistic Regression

Logistic Regression

- Logistic regression is used when we have a dichotomous variable (Two levels categorical variable Ex- Yes/ No) as the response variable. In that case we cannot use linear regression for the predictions.



Here what we can calculate as the output of the model is the probability of belonging to the category.

Logistic Regression

- Then we cannot work with this model, $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \beta_p x_p$. We have to convert this model such that the output is positive as well as the output should be lying between 0 and 1. Then we have to deal with the Sigmoid function.

$$S(x) = \frac{1}{1 + e^{-x}}$$

So, we put above model inside to the sigmoid function and satisfy above discussed criteria.

$$P = S(y) = \frac{1}{1 + e^{-y}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \beta_p x_p)}}$$

$$P = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \dots + \beta_p x_p)}}$$

Then we can show that,

$$\log \left(\frac{P}{1 - P} \right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdots + \beta_p x_p$$

Logistic Regression

- Here is $\log\left(\frac{P}{1-P}\right)$ called as Logit. Parameters of the model will be estimated using Maximum Likelihood Estimation.
- When predicting some variable using this model a score will be given through this model and the probability should be found and then with that probability according to some cutoff value, the prediction will be done.

Consider the example,

$P \geq \text{Some cutoff} \longrightarrow \text{Assign to class 01}$

$P < \text{Some cutoff} \longrightarrow \text{Assign to class 02}$

Generally this cutoff values is 0.5 in most cases. But this can be changed according to the situation.

Model Evaluation

- Logistic regression is used when we have a dichotomous variable (Two levels categorical variable Ex- Yes/ No) as the response variable. In that case we cannot use linear regression for the predictions.

The mainly discussed two techniques for evaluating a model in regression can be used for evaluating in the classification too.

- Validation Set Approach
- Cross Validation

Using these approaches we can measure some metrics for evaluating a model. The most popular metrics among these metrics is,

Miss Classificatio Error (MCE)

Accuracy=1-MCE

F1 Score

True positive (TP)

- You predicted that a woman is pregnant, and she actually is
- Predicted positive and it's true



True negative (TN)

- You predicted that a woman is not Pregnant, and she actually is not
- Predicted negative and it's true



False positive (FP)

- You predicted a man is pregnant, But he actually is not
- Predicted positive and it's false



FALSE POSITIVE

False negative (FN)

- You predicted a woman is not pregnant, but she actually is
- Predicted negative and it's false



False negative (FN)

- You predicted a woman is not pregnant, but she actually is
- Predicted negative and it's false

Confusion Matrix

- For evaluating the above discussed metrics, a special tool is used which is called as Confusion Matrix.
- Consider the following example where we have to predict a categorical variable with two levels; Yes & No. Consider that we have predicted 100 testing observations using a trained model. Then the confusion matrix can be obtained as,

	Actual Output		
Predicted Output		Yes	No
	Yes	40	10
	NO	20	30

		Predicted	
		Negative (N) -	Positive (P) +
Actual	Negative -	True Negative (TN)	False Positive (FP) Type I Error
	Positive +	False Negative (FN) Type II Error	True Positive (TP)

Performance Metrics for Classification

- In a classification problem, the category or classes of data is identified based on training data.
- The model learns from the given dataset and then classifies the new data into classes or groups based on the training.
- It predicts class labels as the output, such as Yes or No, 0 or 1, Spam or Not Spam, etc. To evaluate the performance of a classification model, different metrics are used, and some of them are as follows:
 - Accuracy
 - Precision
 - Recall
 - F-Score
 - AUC(Area Under the Curve)-ROC

Accuracy of the machine learning model

- Represents the number of correctly classified data instances over the total number of data instances.

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN}$$

Precision of the machine learning model

- The precision determines the proportion of positive prediction that was actually correct.
- It can be calculated as the True Positive or predictions that are actually true to the total positive predictions (True Positive and False Positive).

$$Precision = \frac{TP}{TP + FP}$$

Recall of the machine learning model

- It aims to calculate the proportion of actual positive that was identified incorrectly.
- It can be calculated as true positive or predictions that are actually true to the total number of positives, either correctly predicted as positive or incorrectly predicted as negative (true positive and false negative).

$$Recall = \frac{TP}{TP + FN}$$

F1-score of the machine learning model

- F-score or F1 Score is a metric to evaluate a binary classification model on the basis of predictions that are made for the positive class. Usually, it should be greater than 0.8 for a good model.

$$F1 \text{ Score} = 2 * \frac{Precision * Recall}{Precision + Recall}$$

True Positive Rate

- It represents the proportion of observations that are predicted to be positive when they are indeed positive.
- It is calculated as:

$$TPR = \frac{TP}{Actual\ Positive} = \frac{TP}{TP + FN}$$

False Positive Rate

- This represents the proportion of observations that are predicted to be positive when they are actually negative.
- It is calculated as:

$$\textit{False Positive Rate} = \frac{FP}{\textit{Actual Negative}} = \frac{FP}{TN+FP}$$

False Positive Rate

- This represents the proportion of observations that are predicted to be positive when they are actually negative.
- It is calculated as:

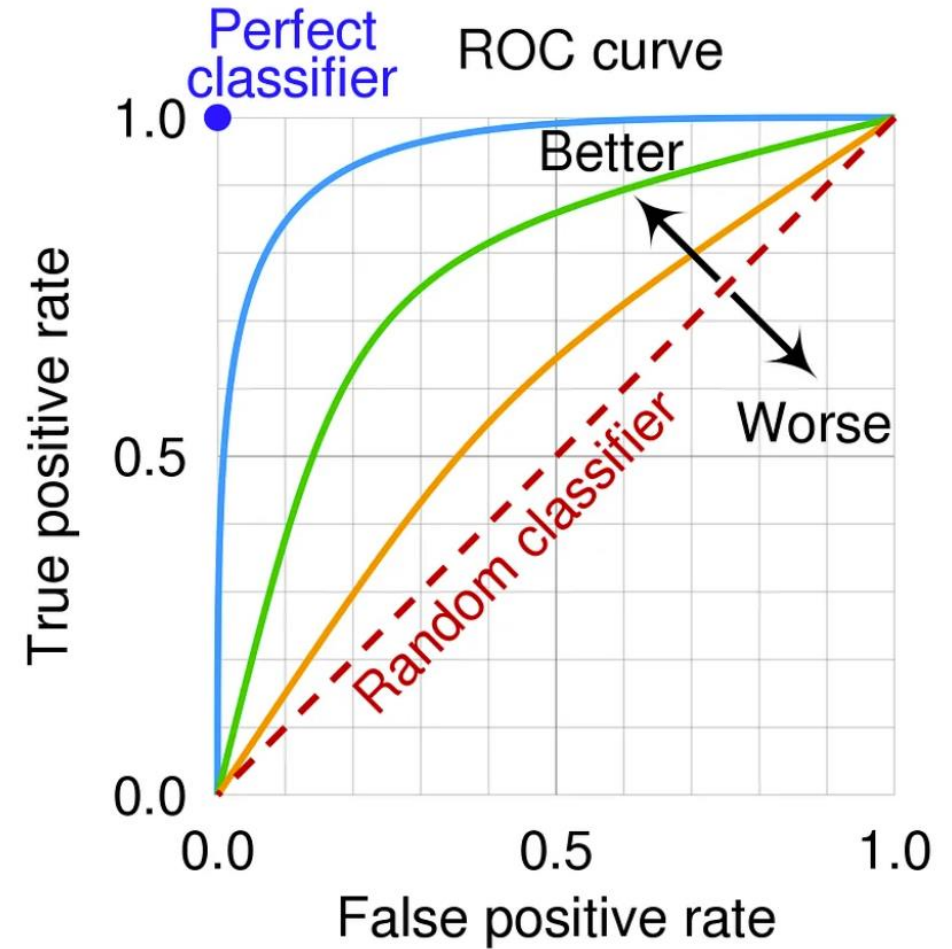
$$\textit{False Positive Rate} = \frac{FP}{\textit{Actual Negative}} = \frac{FP}{TN+FP}$$

AUC-ROC (Receiver Operating Characteristics)

- Sometimes we need to visualize the performance of the classification model on charts; then, we can use the AUC-ROC curve.
- It is one of the popular and important metrics for evaluating the performance of the classification model.
- Firstly, let's understand ROC (Receiver Operating Characteristic curve) curve.
- ROC represents a graph to show the performance of a classification model at different threshold levels.
- The curve is plotted between two parameters, which are:
 - **True Positive Rate**
 - **False Positive Rate**

AUC-ROC

- AUC is equal to 1, the classifier is able to perfectly distinguish between all Positive and Negative class points.
- When AUC is equal to 0, the classifier would be predicting all Negatives as Positives and vice versa.
- When AUC is 0.5, the classifier is not able to distinguish between the Positive and Negative classes



Area Under Curve (AUC)

- With the area under the ROC curve, an idea can be got about the accuracy of a model.

AUC Range	Classification Accuracy
0.9-1.0	Excellent
0.8-0.9	Good
0.7-0.8	Fair
0.6-0.7	Bad
0.5-0.6	Very Bad

Actual	Predicted
1-Spam	1-Spam
1-Spam	0-Not Spam
1-Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	1-Spam
0-Not Spam	0-Not Spam
0-Not Spam	1-Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam
0-Not Spam	1-Spam
0-Not Spam	0-Not Spam
0-Not Spam	0-Not Spam

True Positives (TP):

The number of instances correctly predicted as “Spam” (actual = 1, predicted = 1).

True Negatives (TN):

The number of instances correctly predicted as “Not Spam” (actual = 0, predicted = 0).

False Positives (FP):

The number of instances incorrectly predicted as “Spam” when they are actually “Not Spam” (actual = 0, predicted = 1).

False Negatives (FN):

The number of instances incorrectly predicted as “Not

Actual	Predicted	Status
1-Spam	1-Spam	TP
1-Spam	0-Not Spam	FN
1-Spam	0-Not Spam	FN
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	1-Spam	FP
0-Not Spam	0-Not Spam	TN
0-Not Spam	1-Spam	FP
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN
0-Not Spam	1-Spam	FP
0-Not Spam	0-Not Spam	TN
0-Not Spam	0-Not Spam	TN

Actual	Predicted
Dog	Dog
Cat	Dog
Dog	Dog
Cat	Cat
Dog	Dog
Dog	Dog
Cat	Cat
Dog	Cat
Cat	Cat
Dog	Cat
Dog	Dog
Dog	Dog
Dog	Dog
Cat	Cat
Dog	Dog
Dog	Dog
Cat	Cat
Dog	Dog
Dog	Dog
Cat	Cat

Actual	Predicted	Status
Dog	Dog	TN
Cat	Dog	FN
Dog	Dog	TN
Cat	Cat	TP
Dog	Dog	TN
Dog	Dog	TN
Cat	Cat	TP
Dog	Cat	FP
Cat	Cat	TP
Dog	Cat	FP
Dog	Dog	TN
Dog	Dog	TN
Dog	Dog	TN
Cat	Cat	TP
Dog	Dog	TN
Dog	Dog	TN
Cat	Cat	TP
Dog	Dog	TN
Dog	Dog	TN
Cat	Cat	TP

- TP: Predicted as "Cat" and actually "Cat"
- TN: Predicted as "Dog" and actually "Dog"
- FP: Predicted as "Cat" but actually "Dog"
- FN: Predicted as "Dog" but actually "Cat"

Class Imbalance Problem in Classification

- Consider the following example where we have to predict Yes/ No using a model. Consider the response variable data in the training dataset.

Category	Number of Observations
Yes	1500
No	300

Here we can clearly see that the observations in both classes are not balanced. This problem is called the Class Imbalance Problem. Find solutions for that

Class Imbalance Problem in Classification

- For dealing with this problem, we have several options.
 - SMOTE: Synthetic Minority Oversampling Technique
 - ADASYN: Adaptive Synthetic Sampling Approach
 - Hybridization: SMOTE + Tomek Links
 - Hybridization: SMOTE + ENN

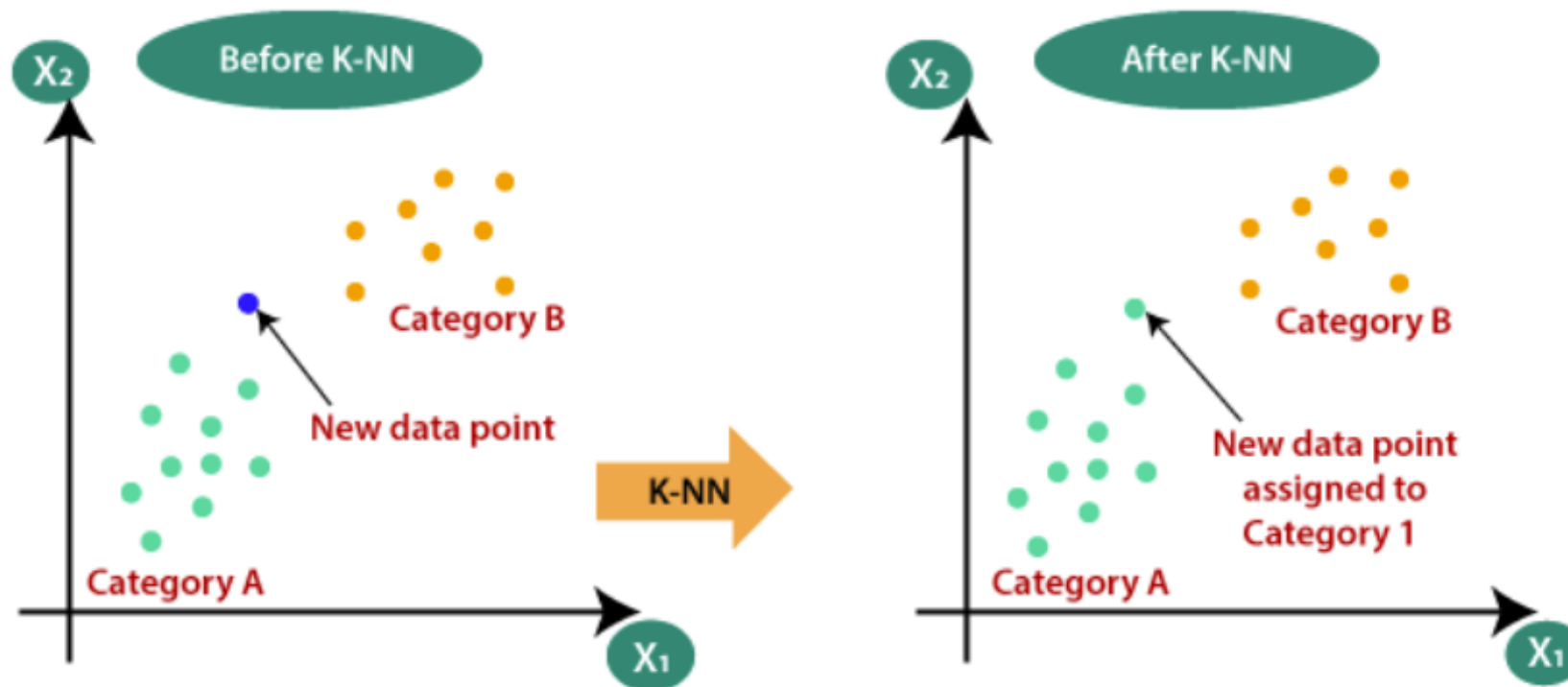
K-Nearest Neighbors

K-Nearest Neighbors

- The K-Nearest Neighbors (KNN) algorithm is a supervised machine learning method employed to tackle classification and regression problems.
- 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 is a non-parametric algorithm, which means it does not make any assumption on underlying data.

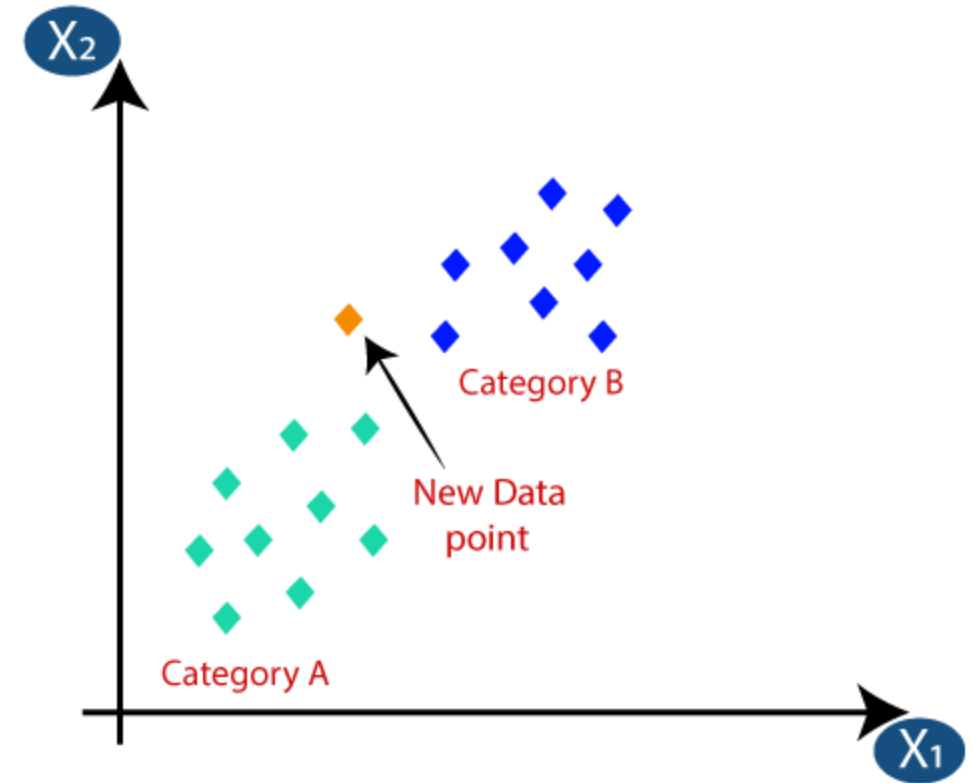
K-Nearest Neighbors

- Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x_1 , 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.



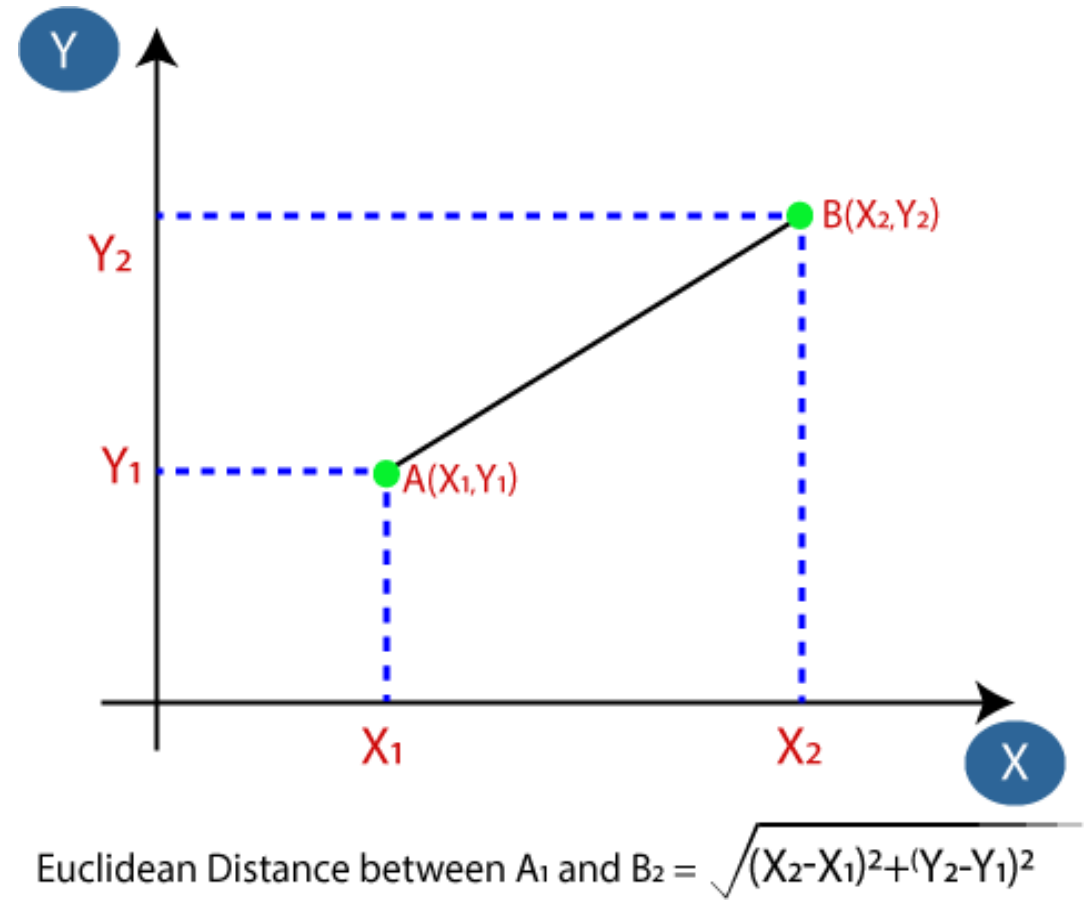
How K-NN Works? Step 1

- Suppose we have a new data point, and we need to put it in the required category to solve this type of problem, we need a K-NN algorithm.
- First, we will choose the number of neighbors, so we will choose the $k=5$.



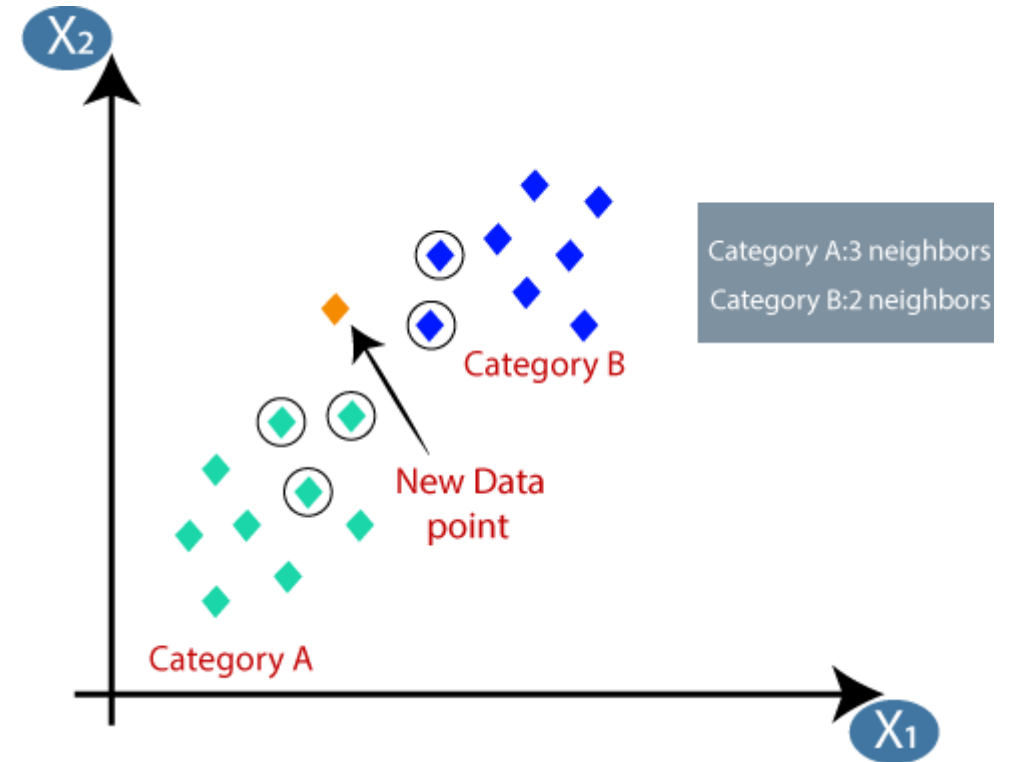
Step 2

- Next, we will calculate the Euclidean distance between the data points.
- The Euclidean distance is the distance between two points.



Step 3

- By calculating the Euclidean distance, we got the nearest neighbors.
- Suppose three nearest neighbors in category A and two nearest neighbors in category B.
- 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?

- There is no 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.

Pros & Cons of K-NN

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

Distance Metrics

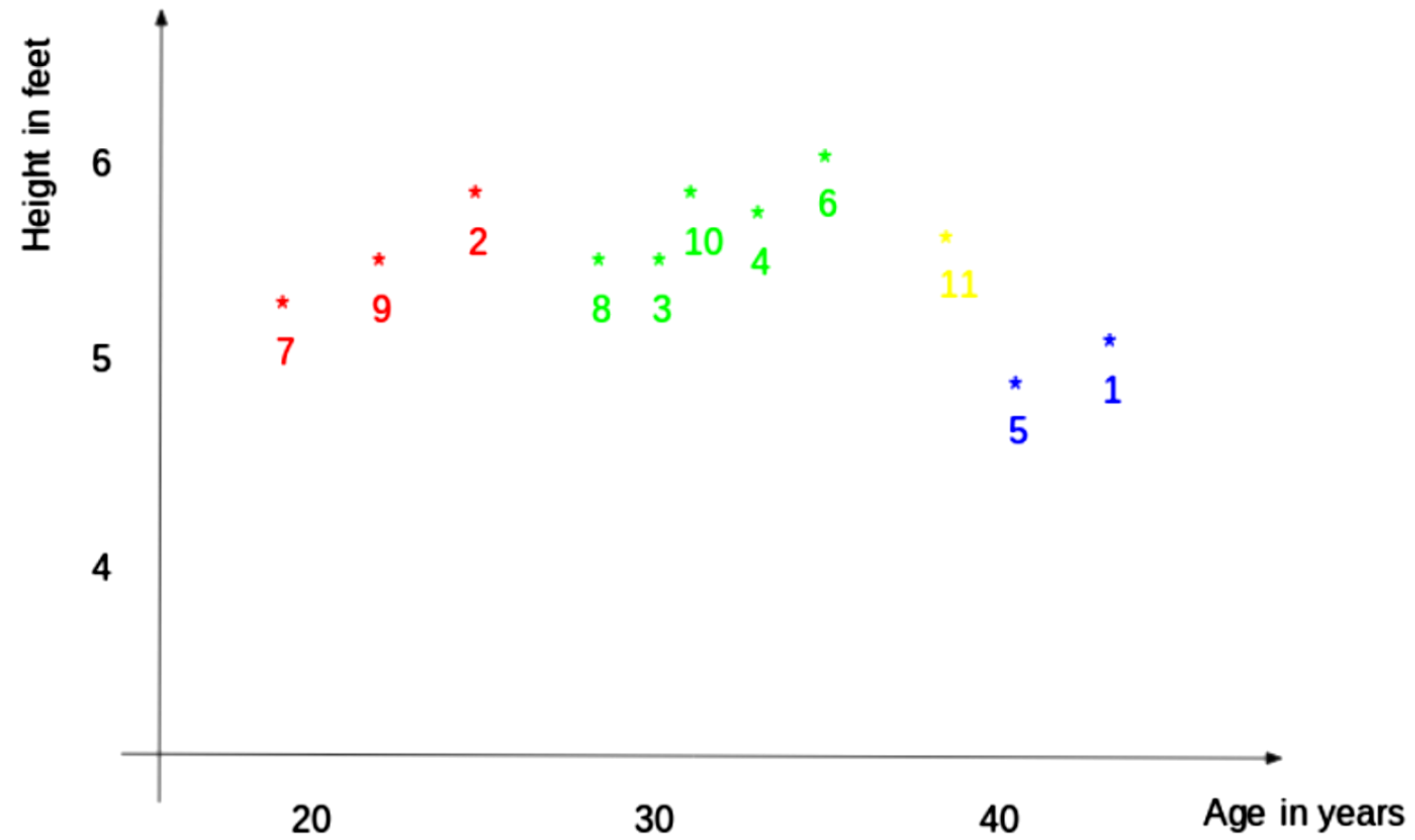
- There are several types of distance measures techniques but we only use some of them and they are listed below:
 - **1. Euclidean distance**
 - 2. Manhattan distance
 - 3. Minkowski distance
 - 4. Hamming distance

- Consider the following example.

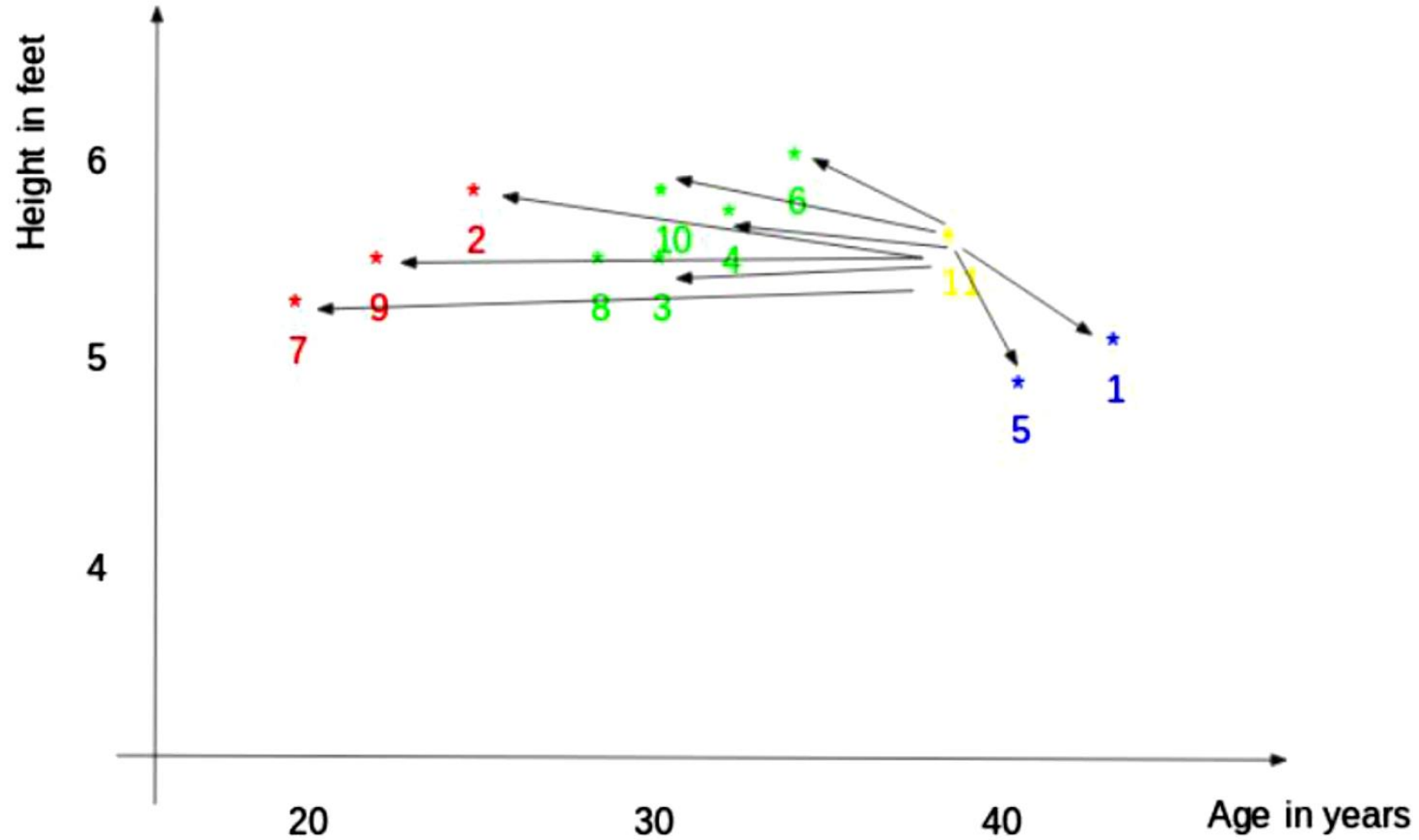
ID	Height	Age	Weight
1	5	45	77
2	5.11	26	47
3	5.6	30	55
4	5.9	34	59
5	4.8	40	72
6	5.8	36	60
7	5.3	19	40
8	5.8	28	60
9	5.5	23	45
10	5.6	32	58
11	5.5	38	?

Think that we need to find the Weight for observation 11 using the given data.

- If we plot this,

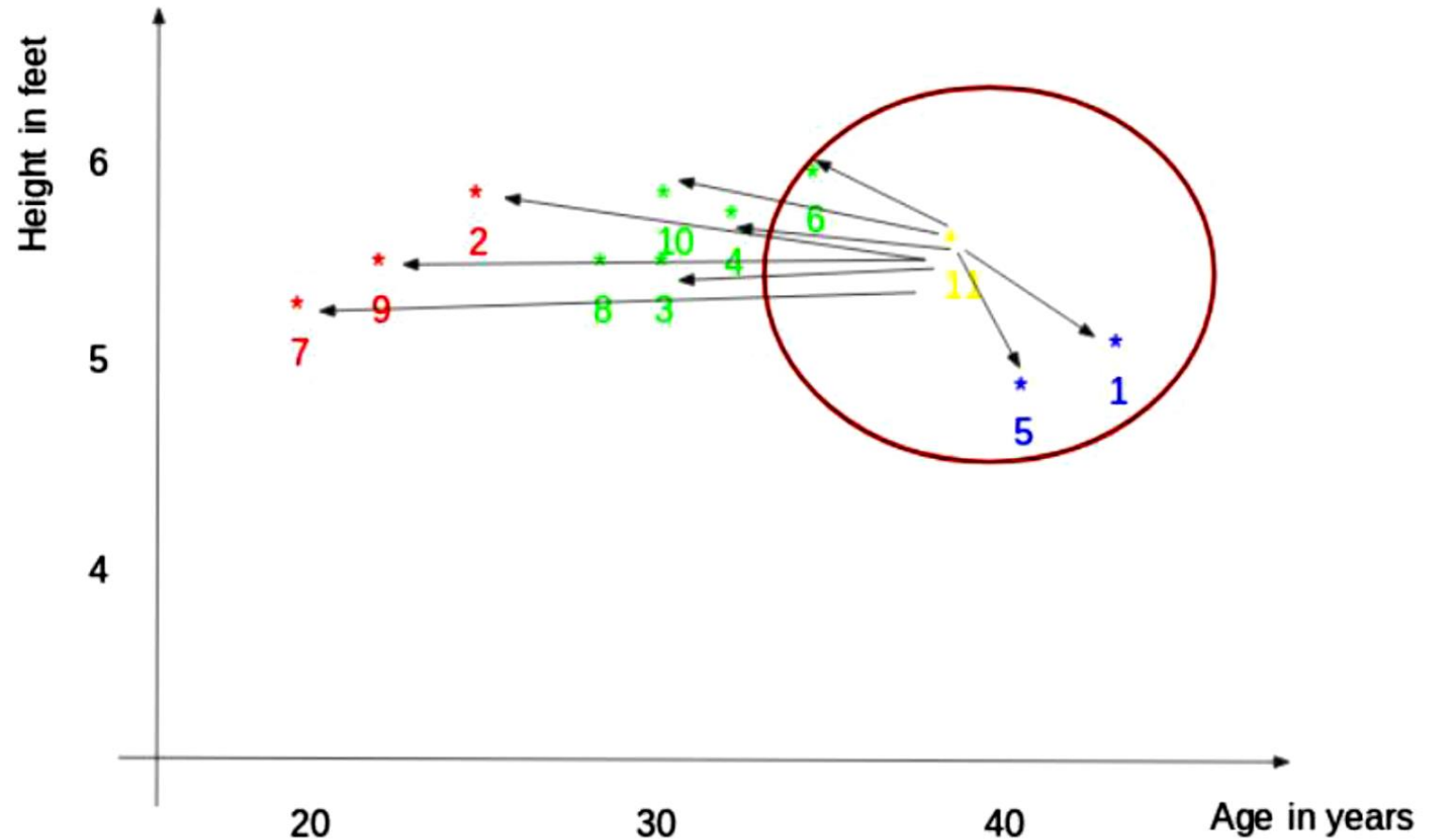


- First, the distance between the new point and each training point is calculated.



- The closest k data points are selected (based on the distance). Consider here k=3, so in this example, points 1, 5, 6 will be selected if the value of k is 3..

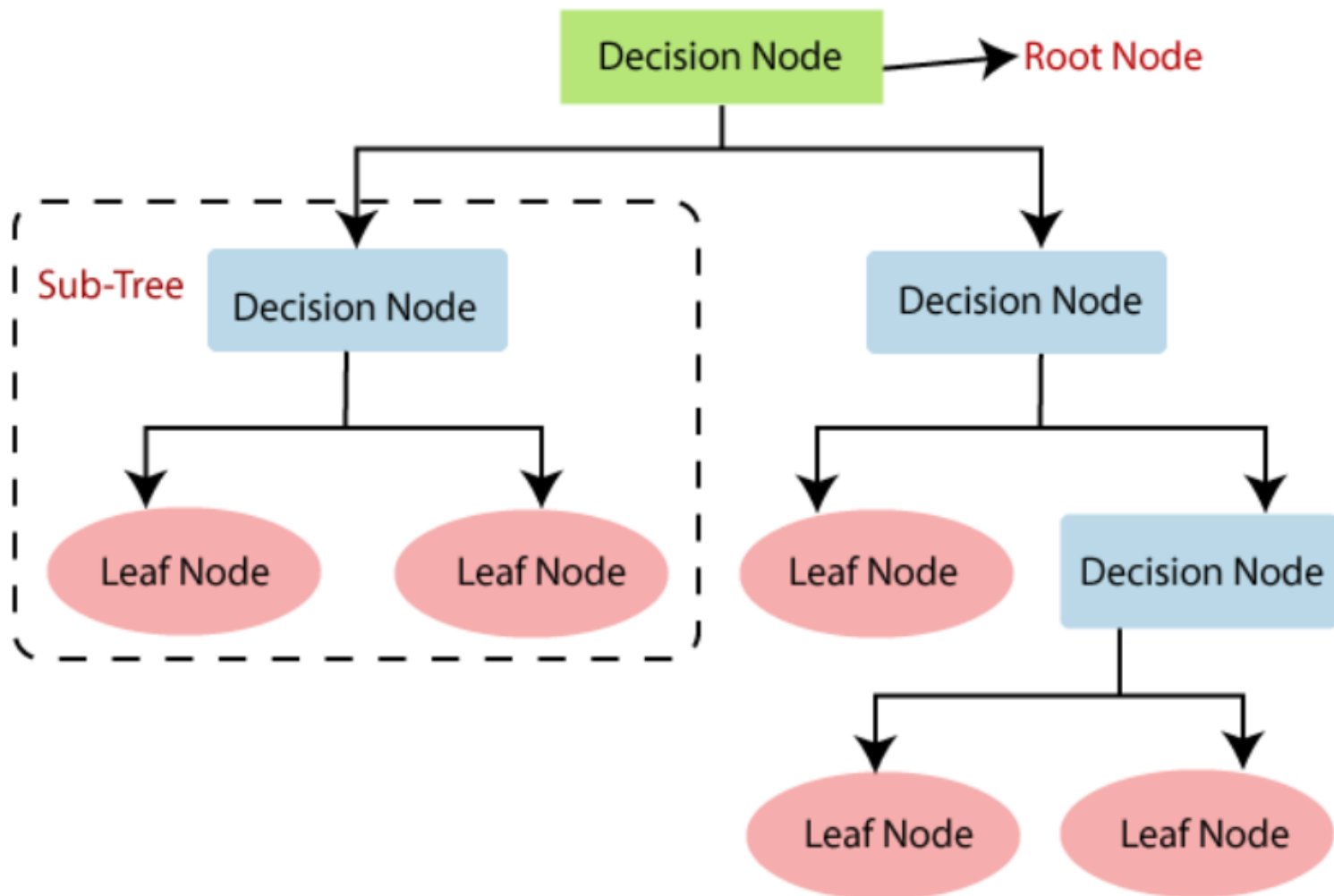
- Then the average of these data points is the final prediction for the new point.
- Here, we have weight of ID11 is $(77+72+60)/3 = 69.66$ kg.
- If we have a classification case, the most frequent category will be chosen to assign.



Decision Tree

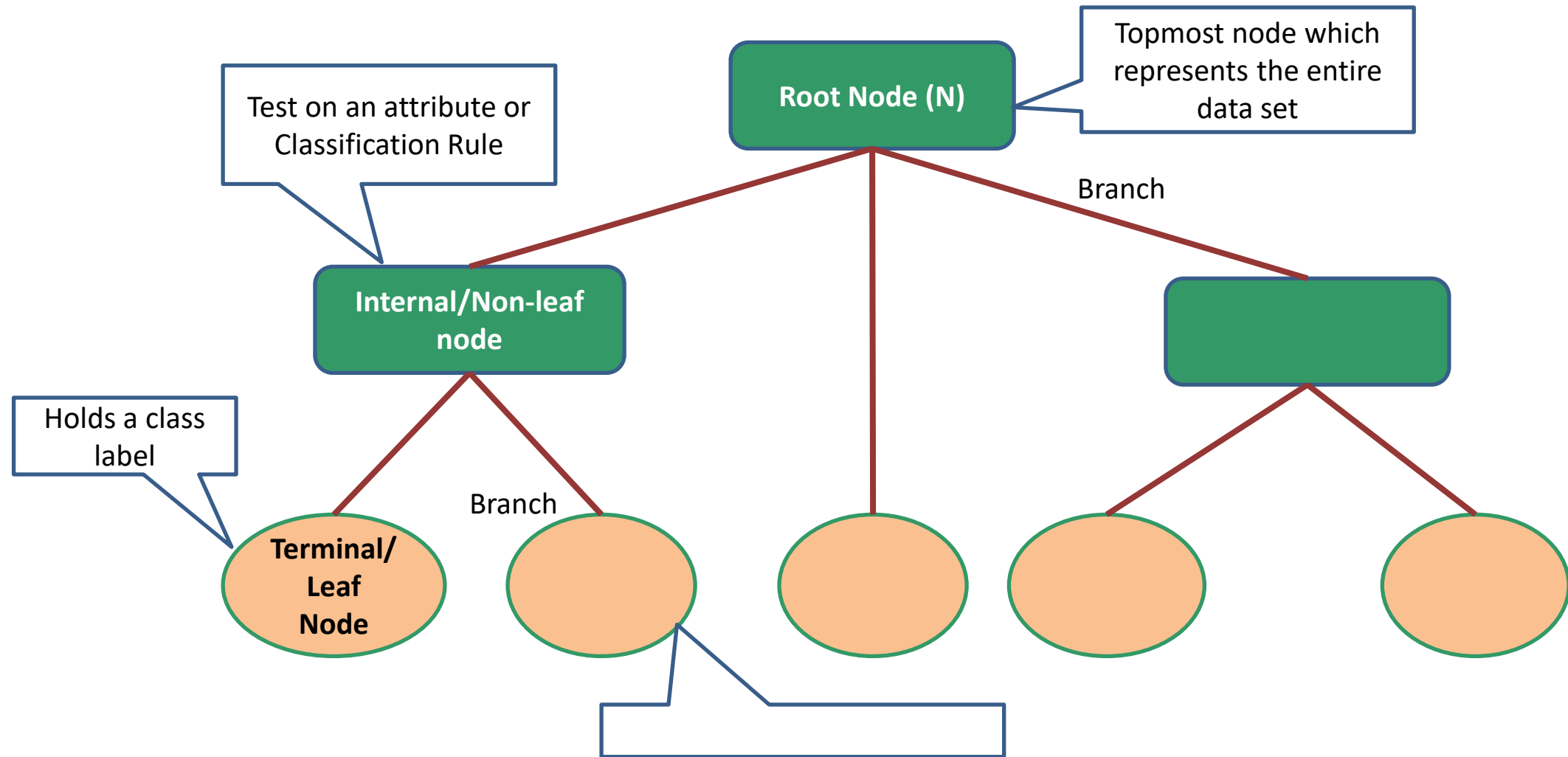
Decision Tree

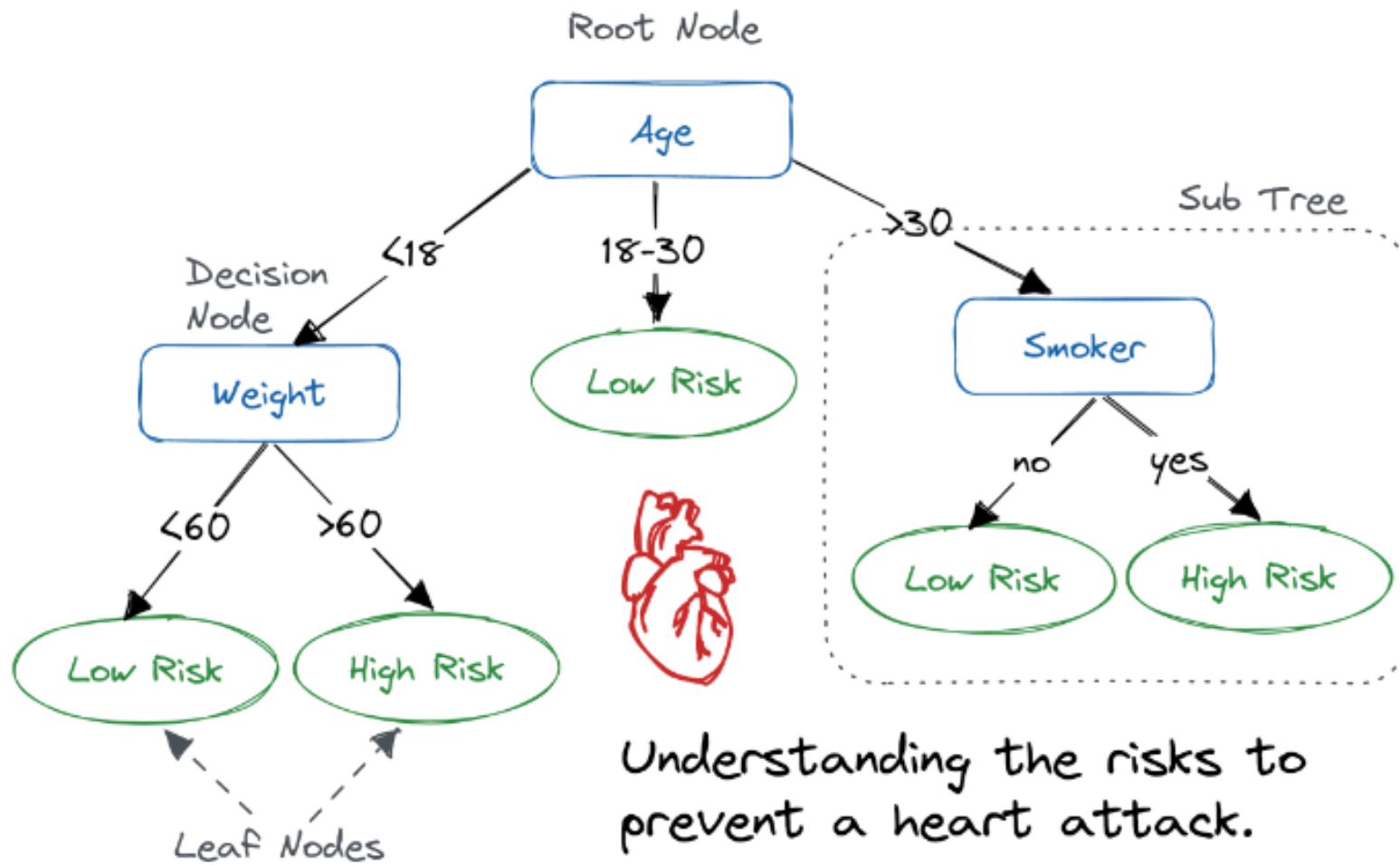
- A decision tree is a non-parametric supervised learning algorithm for classification and regression tasks.
- It has a hierarchical tree structure consisting of a root node, branches, internal nodes, and leaf nodes.
- It is a tree-structured classifier, where **internal nodes represent the features 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.



Decision Tree

Decision tree structure

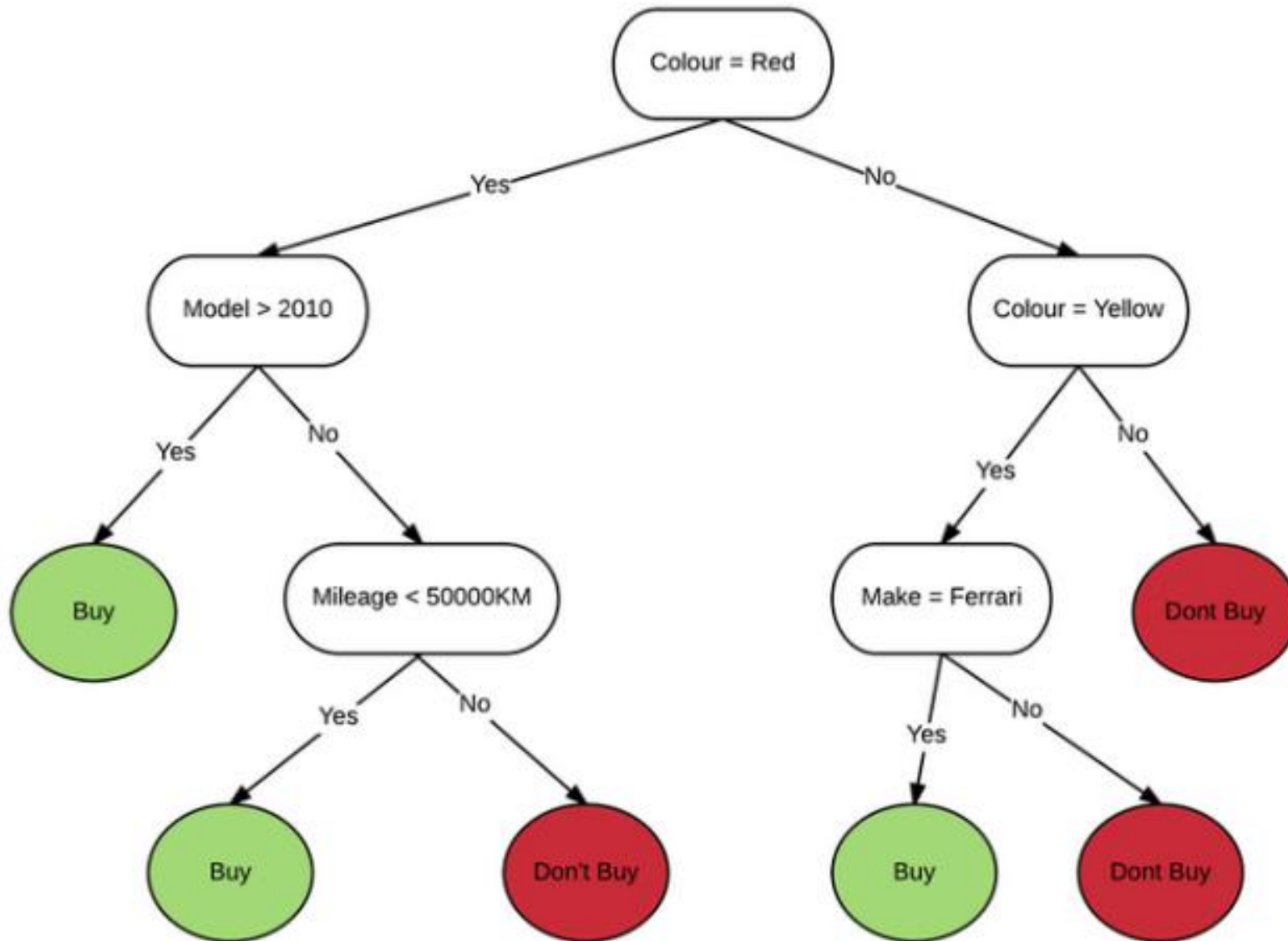




Example 1

BUYING A CAR

Example 2



Terminologies

Root Node: Main question

It represents the entire population or sample, and this further gets divided into two or more homogeneous sets.

Leaf/ Terminal Node: Answer

Nodes do not split is called Leaf or Terminal node.

Branch Node: Intermediate Process

When a sub-node splits into further sub-nodes, then it is called a decision node.

Branch / Sub-Tree:

A subsection of the entire tree is called a branch or sub-tree.

Splitting:

It is a process of dividing a node into two or more sub-nodes.

Pruning:

Pruning is when we selectively remove branches from a tree. The goal is to remove unwanted

How decision tree algorithms work?

- **Starting at the Root:**
 - The algorithm begins at the top, called the “root node,” representing the entire dataset.
- **Asking the Best Questions:**
 - It looks for the most important feature or question that splits the data into the most distinct groups. This is like asking a question at a fork in the tree.
- **Branching Out:**
 - Based on the answer to that question, it divides the data into smaller subsets, creating new branches.
 - Each branch represents a possible route through the tree.
- **Repeating the Process:**
 - The algorithm continues asking questions and splitting the data at each branch until it reaches the final “leaf nodes,” representing the predicted outcomes or classifications.

Attribute selection measures

The attribute selection measure supports a ranking for every attribute defining the given training tuples. The attribute having the best method for the measure is selected as the splitting attribute for the given tuples.

Methods available:

- Information Gain
- Gain Ratio
- Gini Index

Algorithms to build a decision tree

CART (Classification and Regression Trees)

This makes use of Gini impurity as the metric.

ID3 (Iterative Dichotomiser 3)

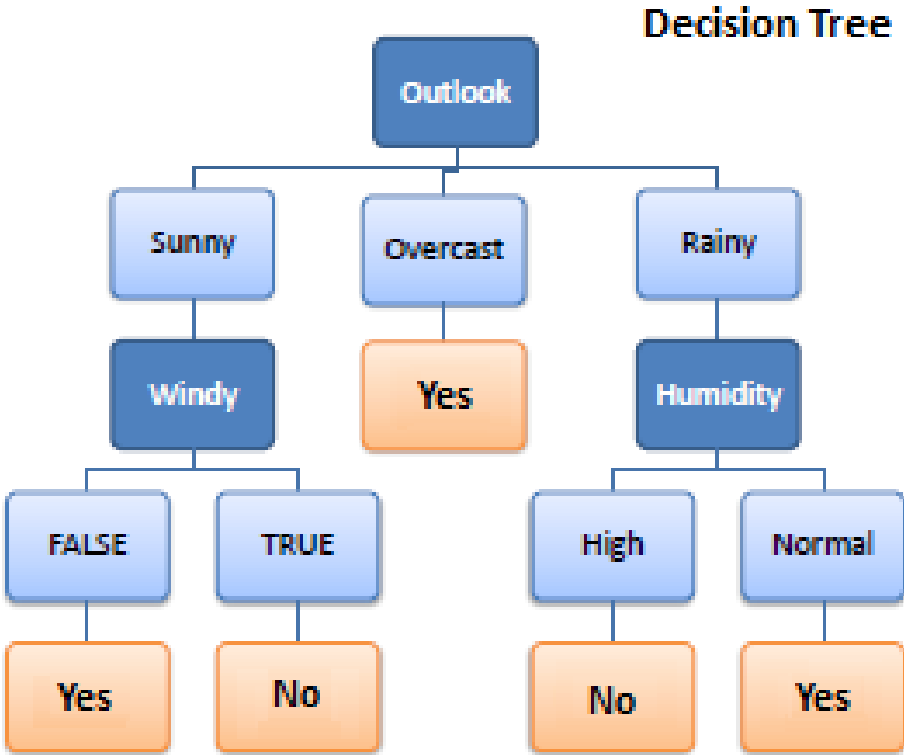
This uses entropy and information gain as metric.

C4.5

Improve version of ID3 (Use gain Ratio)

Example

Predictors				Target
Outlook	Temperature	Humidity	Windy	Play golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



Building a decision tree

The core algorithm: ID3 by J. R. Quinlan

- Adopts a top down, greedy search through the space of all possible branches
- Partitions data into several subsets
- Uses Entropy and Information Gain
 - Entropy: evaluates the homogeneity of the values in subsets (0 if completely homogeneous)
 - Information Gain: Entropy before partitioning – entropy after partitioning
 - Ideally node/attribute having the highest information gain is accept a partition

Entropy

- Entropy is the measure of the degree of randomness or uncertainty in the dataset.
 - In the case of classifications, It measures the randomness based on the distribution of class labels in the dataset.
 - The entropy can be defined as:
-
- p_i = Probability that an arbitrary sample in the data set belongs to class i . (ratios of elements of each label in the set)
 - Binary Entropy Function

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

$$E(S) = -P_{(+)} \log_2 P_{(+)} - P_{(-)} \log_2 P_{(-)}$$

Information Gain Calculation

STEP 1: Involves two calculations

Initial entropy

Outlook	Temperature	Humidity	Windy	Play golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

Play Golf	
Yes	No
9	5

$$E(S) = \sum_{i=1}^c -p_i \log_2 p_i$$

$$E(S) = -P_{(+)} \log_2 P_{(+)} - P_{(-)} \log_2 P_{(-)}$$

$$E(\text{PlayGolf}) = -\left(\frac{9}{14} \log_2 \frac{9}{14}\right) - \left(\frac{5}{14} \log_2 \frac{5}{14}\right)$$

$$E(s) = -(0.6429 (-0.6373)) - (0.3571 (-1.4856))$$

$$E(s) = -(-0.4097) - (-0.5305)$$

$$E(s) = (0.4097) + (0.5305)$$

$$E(s) = 0.9402$$

Information Gain Calculation

Entropy after split

Outlook	Temperature	Humidity	Windy	Play golf
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Sunny	Mild	High	True	No
Sunny	Cool	Normal	True	No

Outlook	Temperature	Humidity	Windy	Play golf
Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

Outlook	Temperature	Humidity	Windy	Play golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Rainy	Mild	Normal	True	Yes

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

Information Gain Calculation

Entropy after split

$$E(\text{PlayGolf}, \text{Outlook}) = P(\text{Sunny}) * E(3,2) + P(\text{Overcast}) * E(4,0) + P(\text{Rainy}) * E(2,3)$$

		Play Golf		
		Yes	No	
Outlook	Sunny	3	2	5
	Overcast	4	0	4
	Rainy	2	3	5
				14

$$\begin{aligned} E(\text{PlayGolf}, \text{Outlook}) &= \frac{5}{14} * \left(- \left(\frac{3}{5} \log_2 \frac{3}{5} \right) - \left(\frac{2}{5} \log_2 \frac{2}{5} \right) \right) \\ &+ \frac{4}{14} * \left(- \left(\frac{4}{4} \log_2 \frac{4}{4} \right) - \left(\frac{0}{4} \log_2 \frac{0}{4} \right) \right) \\ &+ \frac{5}{14} * \left(- \left(\frac{2}{5} \log_2 \frac{2}{5} \right) - \left(\frac{3}{5} \log_2 \frac{3}{5} \right) \right) \end{aligned}$$

$$E(s) = 0.694$$

Information Gain Calculation

Step 2:

Calculate the information gain for each split (along each attribute)

$$\underline{Gain(T, X)} = Entropy(T) - Entropy(T, X)$$

$$G(\text{PlayGolf}, \text{Outlook}) = E(\text{PlayGolf}) - E(\text{PlayGolf}, \text{Outlook})$$

$$= 0.940 - 0.693 = 0.247$$

Information Gain Calculation

Outlook	Temperature	Humidity	Windy	Play golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Overcast	Hot	Normal	False	Yes


Outlook	Temperature	Humidity	Windy	Play golf
Sunny	Mild	High	False	Yes
Rainy	Mild	High	False	No
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Sunny	Mild	High	True	No

Outlook	Temperature	Humidity	Windy	Play golf
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Cool	Normal	False	Yes

		Play Golf	
		Yes	No
Temp. ☹	Hot	2	2
	Mild	4	2
	Cool	3	1

Information Gain Calculation

Entropy after split

		Play Golf	
		Yes	No
Temp. 	Hot	2	2
	Mild	4	2
	Cool	3	1

$$\begin{aligned} E(\text{PlayGolf}, \text{Temperature}) &= \frac{4}{14} * \left(- \left(\frac{2}{4} \log_2 \frac{2}{4} \right) - \left(\frac{2}{4} \log_2 \frac{2}{4} \right) \right) \\ &+ \frac{6}{14} * \left(- \left(\frac{4}{6} \log_2 \frac{4}{6} \right) - \left(\frac{2}{6} \log_2 \frac{2}{6} \right) \right) \\ &+ \frac{4}{14} * \left(- \left(\frac{3}{4} \log_2 \frac{3}{4} \right) - \left(\frac{1}{4} \log_2 \frac{1}{4} \right) \right) \\ &= 0.2857 * (-(0.5*-1)-(0.5*-1)) + 0.4286 * (-(0.6667*-0.5849) - (0.3333 * -1.585)) \\ &\quad + 0.2857 * (-(0.75*-0.415) - (0.25*-2)) \\ &= (0.2857 * (0.5+0.5)) + (0.4286 * (0.3900 + 0.5283)) + (0.2857 * (0.3113 + 0.5)) \\ &= (0.2857 * 1) + (0.4286 * 0.9183) + (0.2857 * 0.8113) \\ &= (0.2857 + 0.3936 + 0.2317) \\ &= 0.911 \end{aligned}$$

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

$$\begin{aligned} G(\text{PlayGolf}, \text{Temperature}) &= E(\text{PlayGolf}) - E(\text{PlayGolf}, \text{Temperature}) \\ &= 0.940 - 0.911 \\ &= 0.029 \end{aligned}$$

Information Gain Calculation

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

		Play Golf	
		Yes	No
Temp.	Hot	2	2
	Mild	4	2
	Cool	3	1
Gain = 0.029			

		Play Golf	
		Yes	No
Humidity	High	3	4
	Normal	6	1
Gain = 0.152			

		Play Golf	
		Yes	No
Windy	False	6	2
	True	3	3
Gain = 0.048			


Information Gain Calculation

Step 3:

Select the split with highest gain

Step 4:

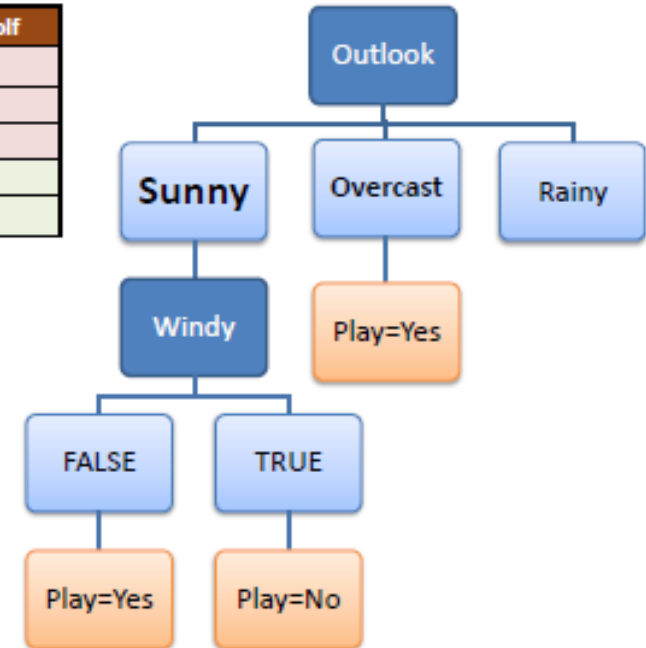
a branch with entropy 0 is a leaf

		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3
Gain = 0.247			

Information Gain Calculation

Branch with entropy more than 0 needs further splitting

Temp	Humidity	Windy	Play Golf
Mild	High	FALSE	Yes
Cool	Normal	FALSE	Yes
Mild	Normal	FALSE	Yes
Cool	Normal	TRUE	No
Mild	High	TRUE	No



ID3 runs recursively on each non-leaf branch until entropy becomes 0

Decision Tree to Decision Rules

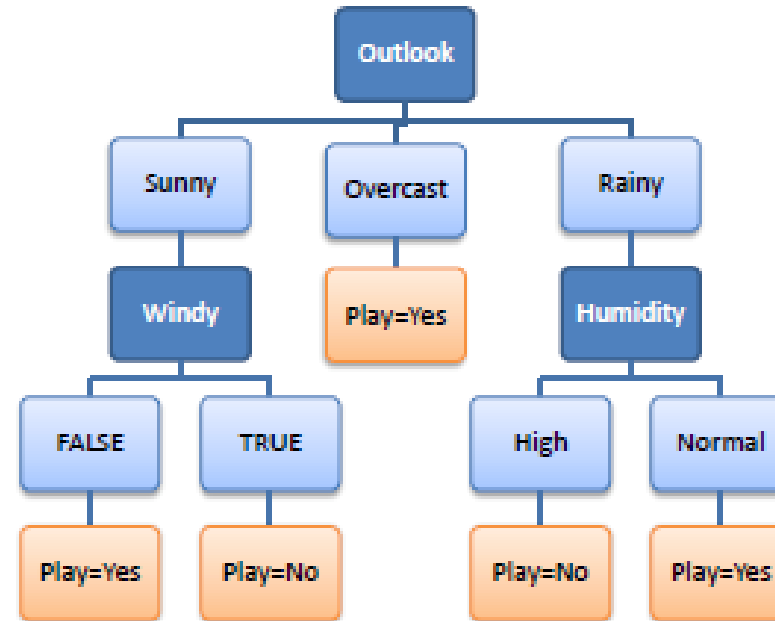
R_1 : IF (Outlook=Sunny) AND (Windy=FALSE) THEN Play=Yes

R_2 : IF (Outlook=Sunny) AND (Windy=TRUE) THEN Play=No

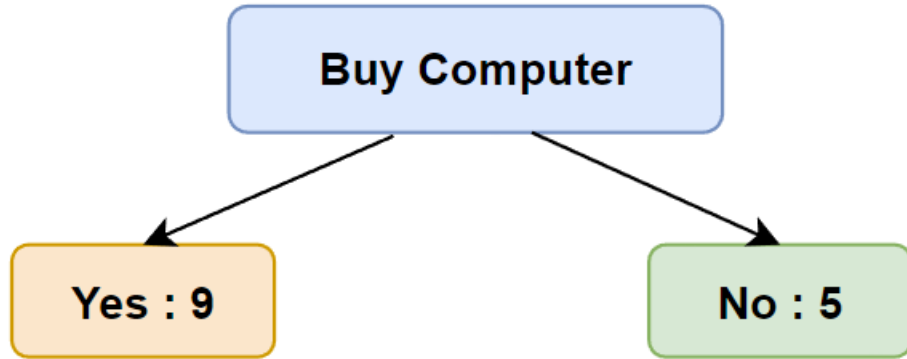
R_3 : IF (Outlook=Overcast) THEN Play=Yes

R_4 : IF (Outlook=Rainy) AND (Humidity=High) THEN Play=No

R_5 : IF (Outlook=Rainy) AND (Humidity=Normal) THEN Play=Yes



#	Age	Income	Student	Credit Rating	Buy Computer
1	Youth	High	No	Fair	No
2	Youth	High	No	Excellent	No
3	Middle-aged	High	No	Fair	Yes
4	Senior	Medium	No	Fair	Yes
5	Senior	Low	Yes	Fair	Yes
6	Senior	Low	Yes	Excellent	No
7	Middle-aged	Low	Yes	Excellent	Yes
8	Youth	Medium	No	Fair	No
9	Youth	Low	Yes	Fair	Yes
10	Senior	Medium	Yes	Fair	Yes
11	Youth	Medium	Yes	Excellent	Yes
12	Middle-aged	Medium	No	Excellent	Yes
13	Middle-aged	High	Yes	Fair	Yes
14	Senior	Medium	No	Excellent	No



$$Gini(D) = 1 - \sum_{i=1}^m P_i^2$$

$$Gini(D) = 1 - (P_+^2 + P_-^2)$$

$$Gini(D) = 1 - \left(\left(\frac{9}{14}\right)^2 + \left(\frac{5}{14}\right)^2 \right)$$

$$Gini(D) = 1 - ((0.6429)^2 + (0.3571)^2)$$

$$Gini(D) = 1 - (0.4133 + 0.1275)$$

$$Gini(D) = 1 - (0.5408)$$

$$Gini(D) = 0.4592 = 0.460$$

Gini Impurity or index

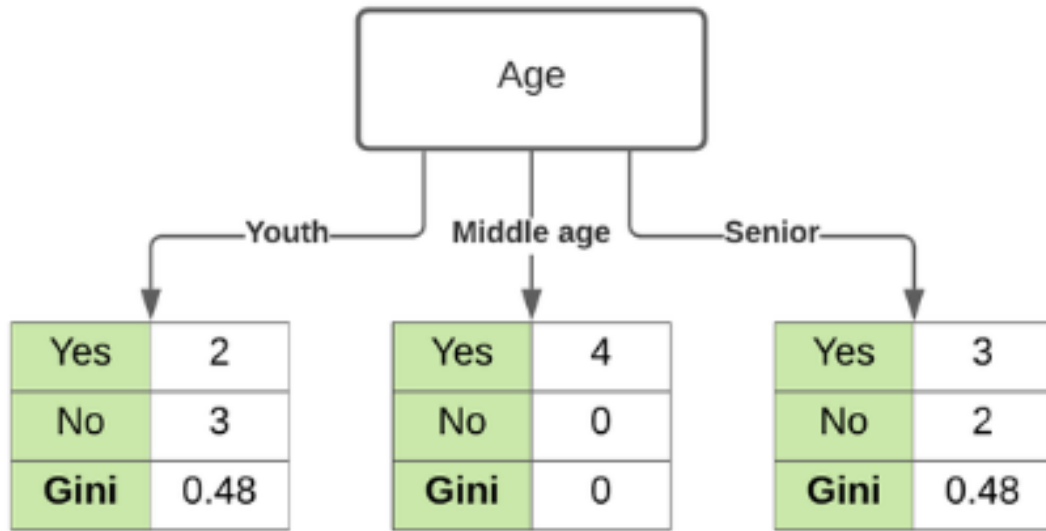
The Gini Index considers a binary split for each attribute.

You can compute a weighted sum of the impurity of each partition.

If a binary split on attribute A partitions data D into D1 and D2, the Gini index of D is:

When training a decision tree, the attribute that provides the smallest is chosen to split the node.

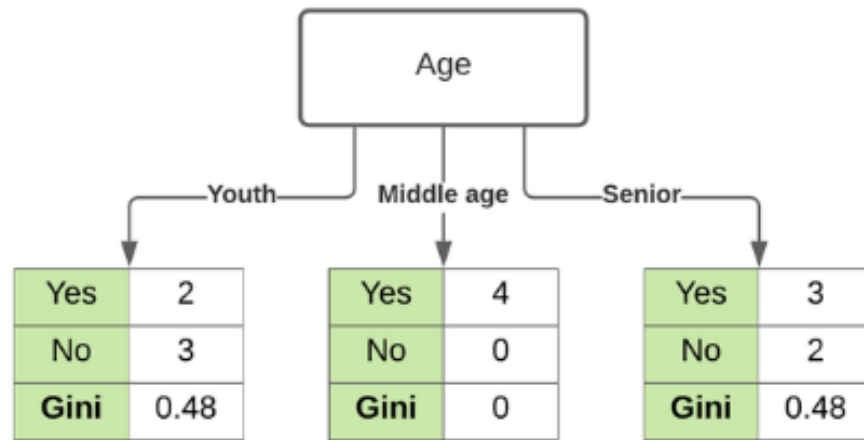
$$\text{Gini}_A(D) = \frac{|D_1|}{|D|} \text{Gini}(D_1) + \frac{|D_2|}{|D|} \text{Gini}(D_2)$$



Gini Impurity for Age is 0.343

$$Gini(D) = 1 - \sum_{i=1}^m P_i^2$$

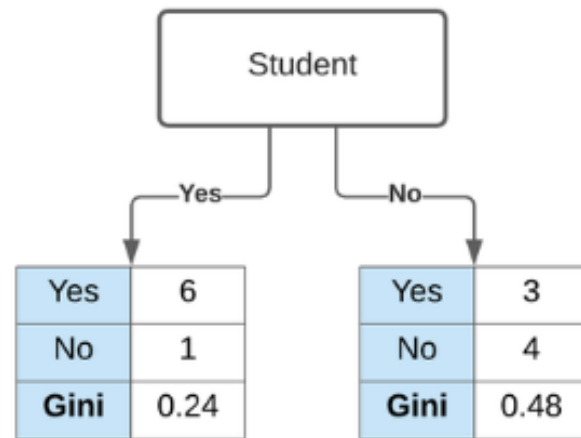
$$\begin{aligned}
 Gini(D) &= \frac{5}{14} * \left(1 - \left(\left(\frac{2}{5}\right)^2 + \left(\frac{3}{5}\right)^2 \right) \right) + \\
 &\quad \frac{4}{14} * \left(1 - \left(\left(\frac{4}{4}\right)^2 + \left(\frac{0}{4}\right)^2 \right) \right) + \\
 &\quad \frac{5}{14} * \left(1 - \left(\left(\frac{3}{5}\right)^2 + \left(\frac{2}{5}\right)^2 \right) \right) \\
 &= 0.3571 * (1 - (0.16 + 0.36)) + \\
 &\quad 0.2857 * (1 - (1 + 0)) + 0.3571 * (1 - (0.36 + 0.16)) \\
 &= 0.3571 * (0.48) + 0 + 0.3571 * (0.48) \\
 &= 0.1714 + 0.1714 \\
 &= 0.3428 \\
 &= 0.343
 \end{aligned}$$



Gini Impurity for Age is 0.343



Gini Impurity for Income is 0.440



Gini Impurity for Student is 0.367



Gini Impurity for Credit Rating is 0.429

Best

In order to obtain information gain for an attribute, the weighted impurities of the branches is subtracted from the original impurity.

The best split can also be chosen by maximizing the Gini gain. Gini gain is calculated as follows:

Gini gain

$$\Delta Gini(A) = Gini(D) - Gini_A(D)$$

$$Gini (Age) = Gini (D) - Gini_{Age} (D) = 0.460 - 0.343 = 0.117$$

$$Gini (Income) = Gini (D) - Gini_{Income} (D) = 0.460 - 0.440 = 0.02$$

$$Gini (Student) = Gini (D) - Gini_{Student} (D) = 0.460 - 0.367 = 0.093$$

$$Gini (Credit Rating) = Gini (D) - Gini_{Credit Rating} (D) = 0.460 - 0.429 \\ = 0.031$$

The best split can also be chosen by maximizing the Gini gain.

Advantages of Decision Trees

- 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 Decision Trees

- 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.

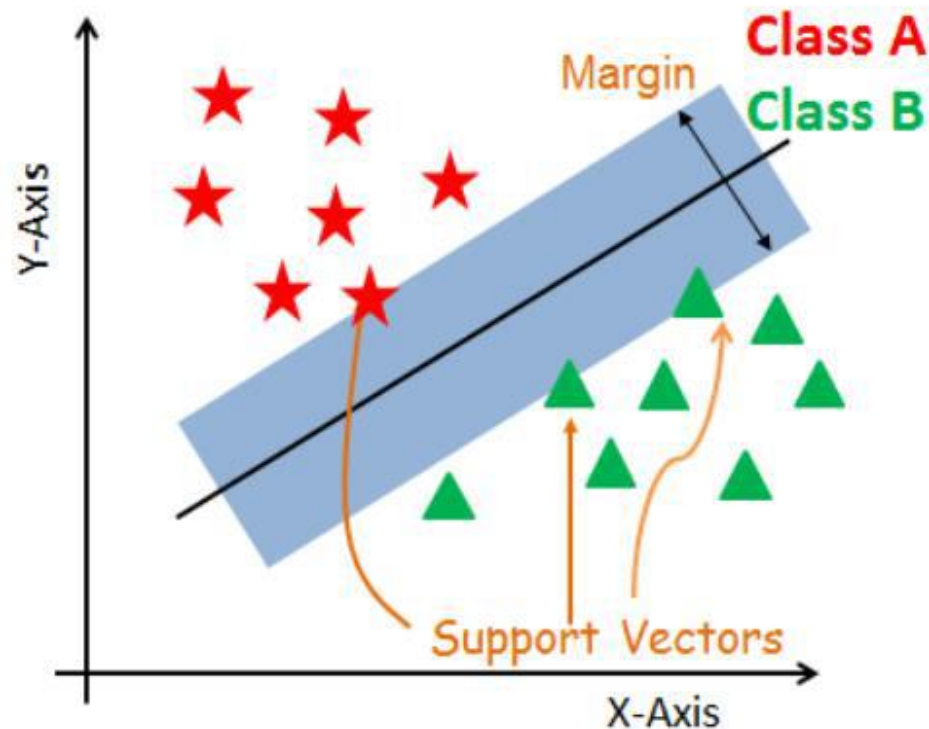
Applications

- **Credit scoring:** Decision trees can be used to predict whether a loan applicant is likely to default on their loan.
- **Medical diagnosis:** Decision trees can be used to diagnose diseases based on a patient's symptoms.
- **Customer segmentation:** Decision trees can be used to segment customers into different groups based on their characteristics.
- **Fraud detection:** Decision trees can be used to detect fraudulent transactions.

Support Vector Machines

SVM – In Short

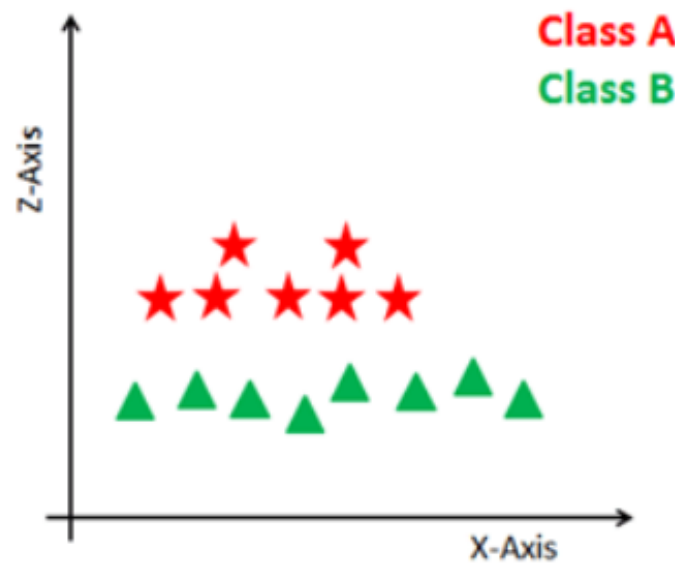
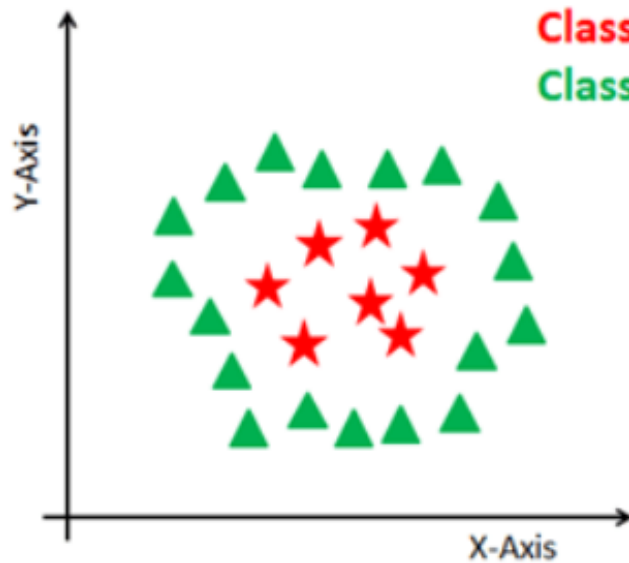
- SVM constructs a hyperplane in multidimensional space to separate different classes. SVM generates optimal hyperplane in an iterative manner, which is used to minimize an error. The core idea of SVM is to find a maximum marginal hyperplane(MMH) that best divides the dataset into classes.



This is generally used for classification. But for regression, this can be extended.

SVM – In Short

- Some problems can't be solved using linear hyperplane, as shown in the figure below. In such situation, SVM uses a kernel trick to transform the input space to a higher dimensional space as shown on the right.



Some popular kernels are,

- Linear Kernel
- Polynomial Kernel
- Radial Basis Function Kernel