UNIT-2

Supervised Learning (Regression/Classification): Basic Methods: Distance based Methods, Nearest Neighbours, Decision Trees, Naive Bayes, **Linear Models:** Linear Regression, Logistic Regression, Generalized Linear Models, Support Vector Machines, **Binary Classification:** Multiclass/Structured outputs, MNIST, Ranking.

Concepts-1 Basic Methods: Distance based Methods

- Distance based Methods are a key part of several machine learning algorithms.
- These distance metrics are used to calculate the similarity between data points in both supervised and unsupervised learning. **Generally to calculate the similarity between data points.**
- In machine learning, there are four different types of distance based Methods.
  1. Euclidean Distance
  2. Manhattan Distance
  3. Minkowski Distance
  4. Hamming Distance

Euclidean Distance

- The shortest distance between two points is known as the **Euclidean Distance.** It is used to calculate the distance between two points in a two-dimensional space.
- The following algorithms used Euclidean Distance:
  1. K-Nearest Neighbors (KNN)
  2. K-Means Clustering

P1 = (x_1, y_1)

P2 = (x_2, y_2)

Then, the formula for Euclidean distance can be expressed as -

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

We can generalize this for n-dimensional space as -

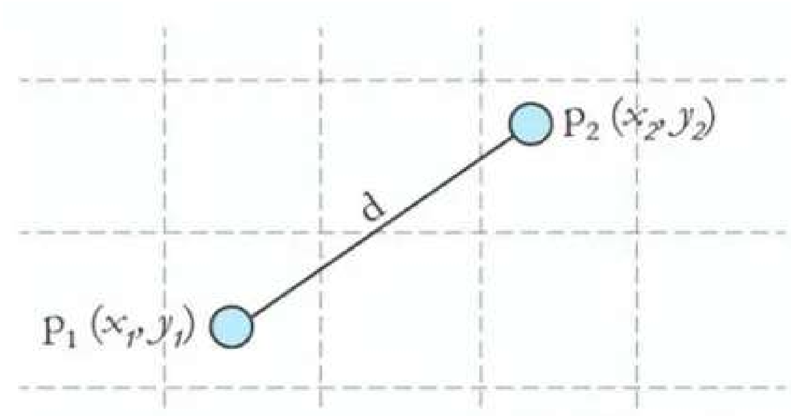
$$D_e = \left(\sum_{i=1}^{n} (x_i - y_i)^2\right)^{\frac{1}{2}}$$

Where,

n = number of dimensions

x_i, y_i = data points

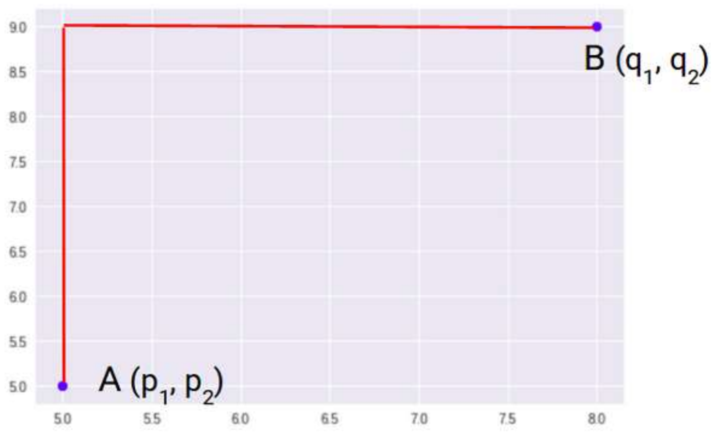
- Euclidean distance can be visualized graphically as



$$\text{Euclidean distance (d)} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Manhattan Distance

- **Manhattan Distance is the sum of absolute differences between points across all the dimensions**.
- The distance between two points measured **along axes at right angles.**



The Manhattan distance in a 2-dimensional space is given $d = |p_1 - q_1| + |p_2 - q_2|$

And the generalized formula for an n-dimensional space is given as:

$$D_m = \sum_{i=1}^{n} |p_i - q_i|$$

Where,

- n = number of dimensions
- pi, qi = data points

Minkowski Distance

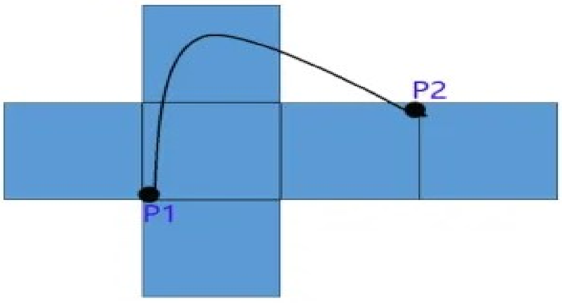
- Minkowski Distance is the generalized form of Euclidean and Manhattan Distance.
- The way distance are measured by the Minkowski metric of different orders between two objects with three variables **(In this images is displayed in a coordinates system with X, Y, Z axes).**

Formula for Minkowski Distance

Here, p is a parameter that defines the type of distance, n is the number of dimensions.

$|p_i - q_i|$ **is the absolute difference between the coordinates of A and B in each dimension.**

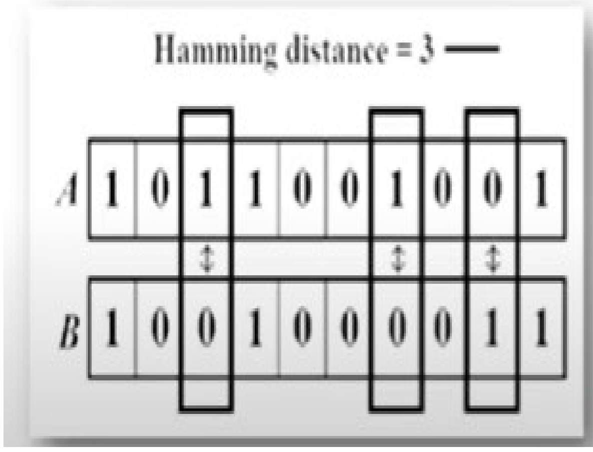
$$D = \left(\sum_{i=1}^{n} |p_i - q_i|^p \right)^{1/p}$$

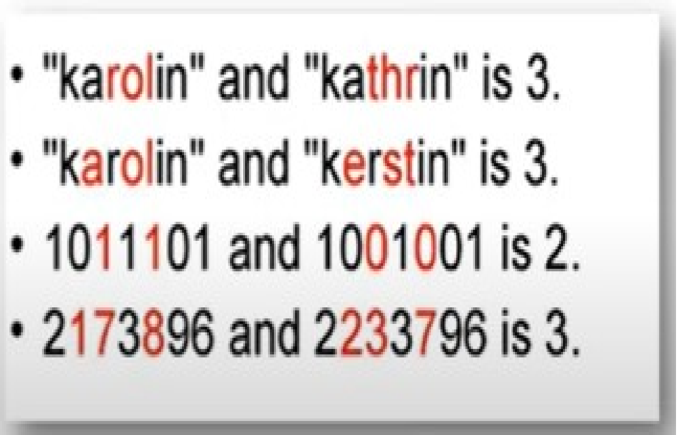


- When p is set to 1, the calculation is the same as the Manhattan distance. When p is set to 2, it is the same as the Euclidean distance. **p = 1 Manhattan distance. p = 2 Euclidean distance**.

Hamming distance

- Hamming distance measure the similarity between **two strings of the same length.**
- The Hamming distance between two strings of the same length is the number of positions at which the corresponding characters are different.





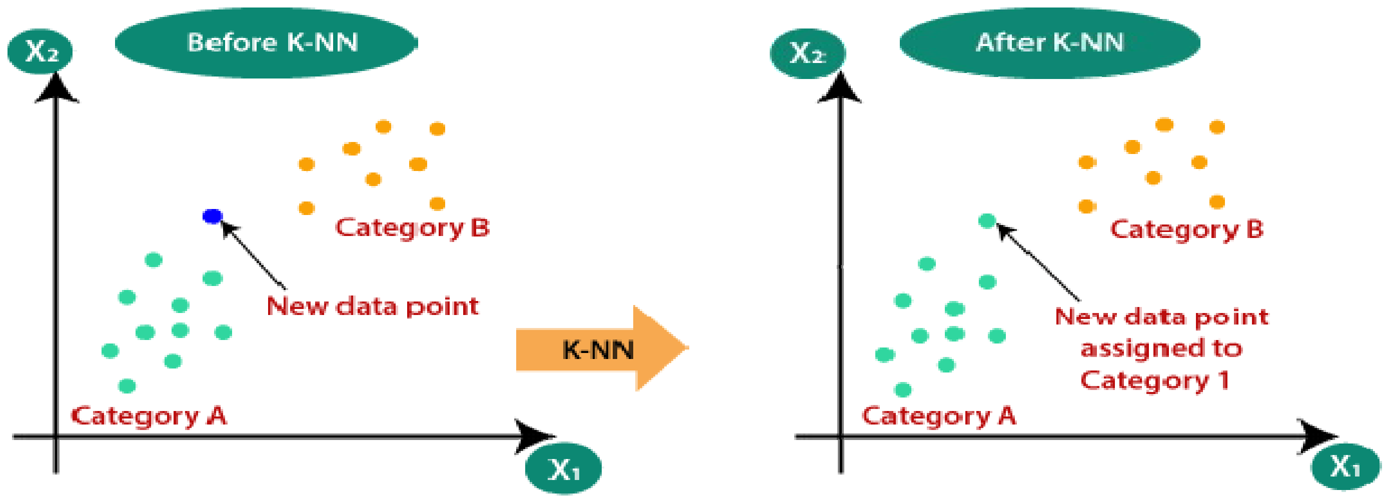
- "karolin" and "kathrin" is 3.
- "karolin" and "kerstin" is 3.
- 1011101 and 1001001 is 2.
- 2173896 and 2233796 is 3.

Concepts-2 K-Nearest Neighbours

- 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 data and available data and put the new data 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.
- 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.**
- **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.
- K-NN 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**.

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.

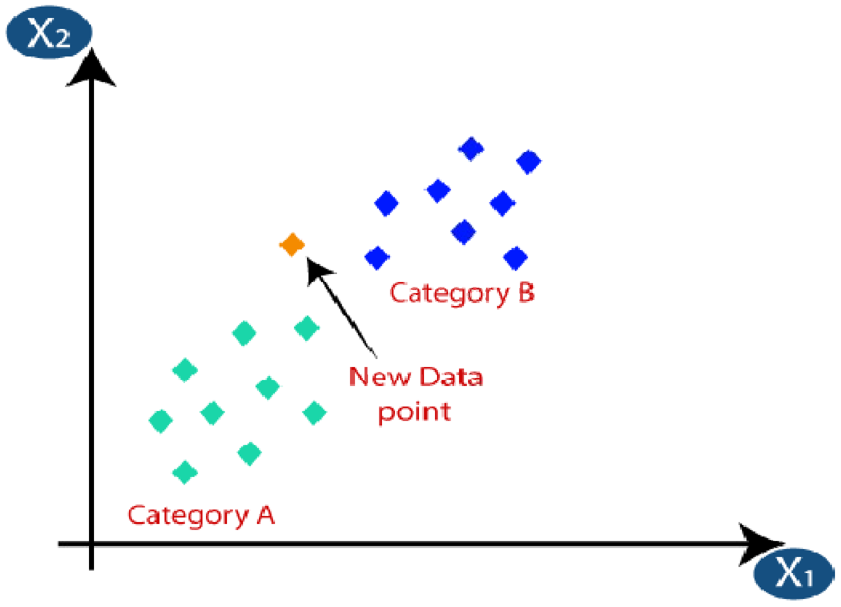


How does K-NN work?

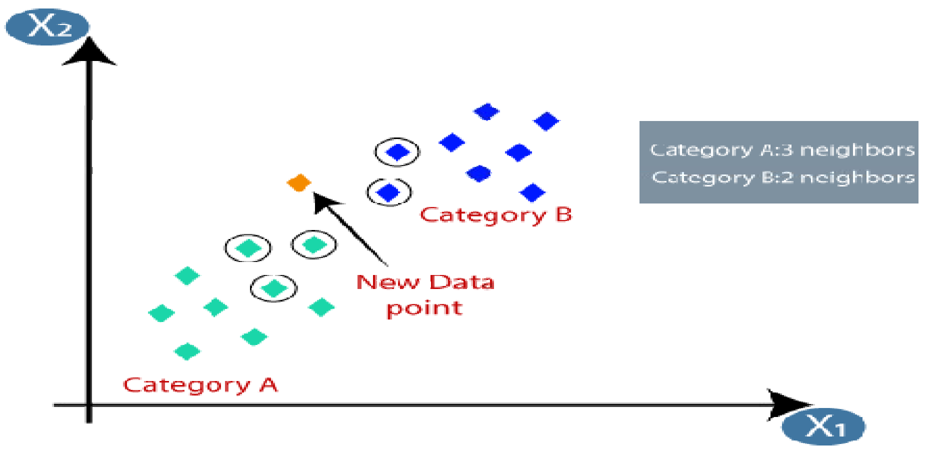
The K-NN working can be explained on the basis of the below algorithm:

1. Select the number K of the neighbours
2. Calculate the Euclidean distance of **K number of neighbours.**
3. Take the K nearest neighbours as per the calculated Euclidean distance.
4. Among these K neighbours, count the number of the data points in each category.
5. Assign the new data points to that category for which the number of the neighbour is maximum.
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 neighbours, so we will choose the **k=5.**
- Next, we will calculate the Euclidean distance between the data points.
- By calculating the Euclidean distance we got the nearest neighbours, as three nearest neighbours in category A and two nearest neighbours in category B.



- As we can see the 3 nearest neighbours 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?

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

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.

Example

- The Example for KNN algorithm Let consider the following dataset.
- Use the new data point (172, 68) and manually calculate the Euclidean distances between this new point and all other points in the dataset. Then we will use K=3(3 nearest neighbours) to make a prediction.

Dataset

Person	Height (cm)	Weight (kg)	Gender
P1	170	70	Male
P2	160	60	Female
P3	180	80	Male
P4	158	54	Female
P5	175	75	Male
P6	162	58	Female

New point for prediction: Height = 172 cm, Weight = 68 kg

Step 1: Calculating the Euclidean Distance

The Euclidean distance formula between two points (x_1, y_1) and (x_2, y_2) is:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- **Calculate the distance between the new point (172,68) and each person in the dataset:**

1. Distance from P1 $(170, 70)$:

$$d = \sqrt{(172 - 170)^2 + (68 - 70)^2} = \sqrt{2^2 + (-2)^2} = \sqrt{4 + 4} = \sqrt{8} \approx 2.83$$

2. Distance from P2 $(160, 60)$:

$$d = \sqrt{(172 - 160)^2 + (68 - 60)^2} = \sqrt{12^2 + 8^2} = \sqrt{144 + 64} = \sqrt{208} \approx 14.42$$

3. Distance from P3 $(180, 80)$:

$$d = \sqrt{(172 - 180)^2 + (68 - 80)^2} = \sqrt{(-8)^2 + (-12)^2} = \sqrt{64 + 144} = \sqrt{208} \approx 14.42$$

4. Distance from P4 $(158, 54)$:

$$d = \sqrt{(172 - 158)^2 + (68 - 54)^2} = \sqrt{14^2 + 14^2} = \sqrt{196 + 196} = \sqrt{392} \approx 19.80$$

5. Distance from P5 $(175, 75)$:

$$d = \sqrt{(172 - 175)^2 + (68 - 75)^2} = \sqrt{(-3)^2 + (-7)^2} = \sqrt{9 + 49} = \sqrt{58} \approx 7.62$$

6. Distance from P6 $(162, 58)$:

$$d = \sqrt{(172 - 162)^2 + (68 - 58)^2} = \sqrt{10^2 + 10^2} = \sqrt{100 + 100} = \sqrt{200} \approx 14.14$$

Step 2: Sorting the Distances

Person	Distance	Gender
P1	2.83	Male
P5	7.62	Male
P6	14.14	Female
P2	14.42	Female
P3	14.42	Male
P4	19.80	Female

Step 3: Selecting the K NearestNeighbours (K=3)

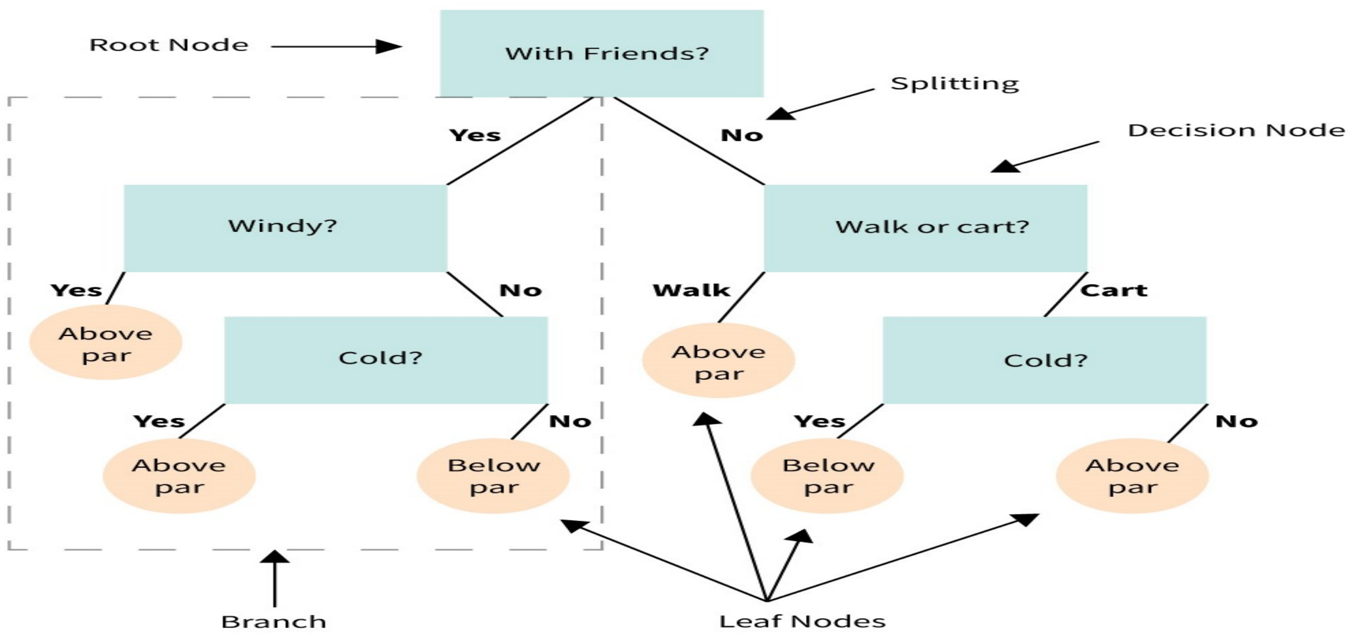
- The 3 closest neighbours based on the distances:
  - **a)** P1 (Male, distance = 2.83)
  - **b)** P5 (Male, distance = 7.62)
  - **c)** P6 (Female, distance = 14.14)

Step 4: Predicting the Gender

- Among the 3 nearest neighbors:
  a) **2 are Male** (P1, P5)
  b) **1 is Female** (P6)
- **Since the majority class is "Male" the predicted gender for the new person with height = 172 cm and weight = 68 kg is Male.**
- **The KNN algorithm predicts that the new person is Male based on the closest 3 neighbors in the dataset.**

Concepts-3 Decision Trees

- 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 or target.**
- 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.



- 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.

Why use Decision Trees?

- 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.

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

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.

Advantages of the Decision Tree

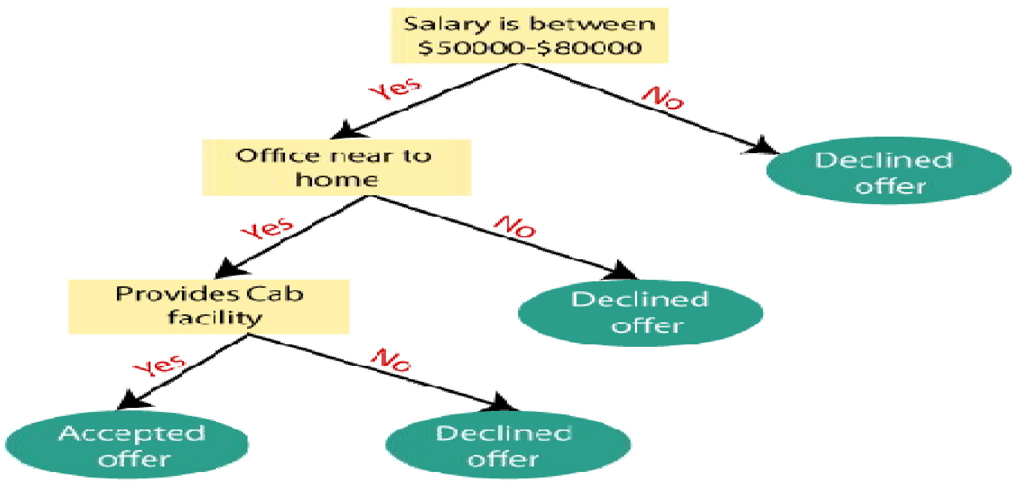
- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared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

Example

- 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 **Attribute Selection Measure**).
- 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).



Attribute Selection Measures

- 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 using ASM, we can easily select the best attribute for the nodes of the tree.
- There are two popular techniques for ASM, which are: **1. Information Gain** **2. Gini Index**

Information Gain

- Information gain is used to find the changes in Entropy after that splitting the dataset based on attributes.
- It calculates how much information a feature provides us about a class.
- According to the value of information again, 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 having the highest information gain is split first.

Information Gain can be calculated with the help of the following formula:

```
Information Gain (IG) = Entropy (D) - [(Weighted average)*Entropy (each Feature)
```

Entropy

- The concept is used to find the disorder present in the given dataset.
- In simple terms, we can say that Entropy tells the impurity present in the dataset.
- It determines the best splits for partitioning dataset.
- The negative sign is used to ensure the Entropy is a positive value.
- If the Entropy is high, the dataset is more heterogeneous for class labels. If the Entropy is low, it means the dataset is homogenous.

Following is the mathematical formula for Entropy:

$$Entropy\ (D) = -\sum_{i=1}^{n}(p_i).log_2(p_i)$$

In the above formula,

- Entropy (D) is the Entropy of the dataset "D".
- "n" is the dataset's number of distinct classes or categories.
- "p_i" is the proportion of data points in class i to the total data points in "D".

Gini Index

- It is also called the **Gini coefficient, Gini impurity**, or **Gini ratio**.
- Gini Index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and regression algorithm.
- The value of the Gini index lies between 0 and 1.
- An attribute with the low index should be preferred as compared to the high gini index.

- **If the Gini index value is 0, it denotes that the dataset is pure or that all the data points belong to the same class.**
- **If the Gini index value is 1, the dataset is impure, or the data points belong to various classes.**
- **If the Gini index value is 0.5, it denotes that the data points in the dataset are evenly distributed among various classes and the dataset is impure.**

The mathematical formula for calculating the Gini Index is given below:

$$Gini\ (D) = 1 - \sum_{i=1}^{n} (p_i)^2$$

In the above formula,

- Gini (D) is the Gini Index of the dataset "D".
- P_i is the probability of data points in the dataset that belong to the i-th class.

Note

- To build a decision tree either using the **ID3 algorithms or CART algorithms**
- By using **CART** Algorithms to build the decision tree with help of **GINI INDEX formula.**
- By using **ID3** Algorithms to build the decision tree with help of **INFORMATION AGAIN AND ENTROPY formula.**

CART Decision Tree Example

- **CART (Classification and Regression Trees)** is a variation of the decision tree algorithm. It can handle both classification and regression tasks.
- Classification And Regression Tree (CART) algorithm to train Decision Trees (also called "growing" trees).
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

Dataset

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No

7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Outlook

- **OUTLOOK** is a nominal feature. **It can be sunny, overcast or rain.** I will summarize the final decisions for outlook feature.

Outlook	Yes	No	Number of instances
Sunny	2	3	5
Overcast	4	0	4
Rain	3	2	5

Gini(Outlook=Sunny) = $1 - (2/5)^2 - (3/5)^2 = 1 - 0.16 - 0.36 = 0.48$

Gini(Outlook=Overcast) = $1 - (4/4)^2 - (0/4)^2 = 0$

Gini(Outlook=Rain) = $1 - (3/5)^2 - (2/5)^2 = 1 - 0.36 - 0.16 = 0.48$

Then, we will calculate weighted sum of gini indexes for outlook feature.

Gini(Outlook) = $(5/14) \times 0.48 + (4/14) \times 0 + (5/14) \times 0.48 = 0.171 + 0 + 0.171 = 0.342$

Temperature

- Similarly, **TEMPERATURE** is a nominal feature and **it could have 3 different values: Cool, Hot and Mild**. Summarize decisions for temperature feature.

Temperature	Yes	No	Number of instances
Hot	2	2	4
Cool	3	1	4
Mild	4	2	6

Gini(Temp=Hot) = $1 - (2/4)^2 - (2/4)^2 = 0.5$

Gini(Temp=Cool) = $1 - (3/4)^2 - (1/4)^2 = 1 - 0.5625 - 0.0625 = 0.375$

Gini(Temp=Mild) = $1 - (4/6)^2 - (2/6)^2 = 1 - 0.444 - 0.111 = 0.445$

We'll calculate weighted sum of gini index for temperature feature

Gini(Temp) = $(4/14) \times 0.5 + (4/14) \times 0.375 + (6/14) \times 0.445 = 0.142 + 0.107 + 0.190 = 0.439$

Humidity

- **HUMIDITY** is a binary class feature. **It can be high or normal.**

Humidity	Yes	No	Number of instances
High	3	4	7
Normal	6	1	7

Gini(Humidity=High) = $1 - (3/7)^2 - (4/7)^2 = 1 - 0.183 - 0.326 = 0.489$

Gini(Humidity=Normal) = $1 - (6/7)^2 - (1/7)^2 = 1 - 0.734 - 0.02 = 0.244$

Weighted sum for humidity feature will be calculated next

Gini(Humidity) = $(7/14) \times 0.489 + (7/14) \times 0.244 = 0.367$

Wind

WIND is a binary class similar to humidity. **It can be weak and strong.**

Wind	Yes	No	Number of instances
Weak	6	2	8
Strong	3	3	6

Gini(Wind=Weak) = $1 - (6/8)^2 - (2/8)^2 = 1 - 0.5625 - 0.062 = 0.375$

Gini(Wind=Strong) = $1 - (3/6)^2 - (3/6)^2 = 1 - 0.25 - 0.25 = 0.5$

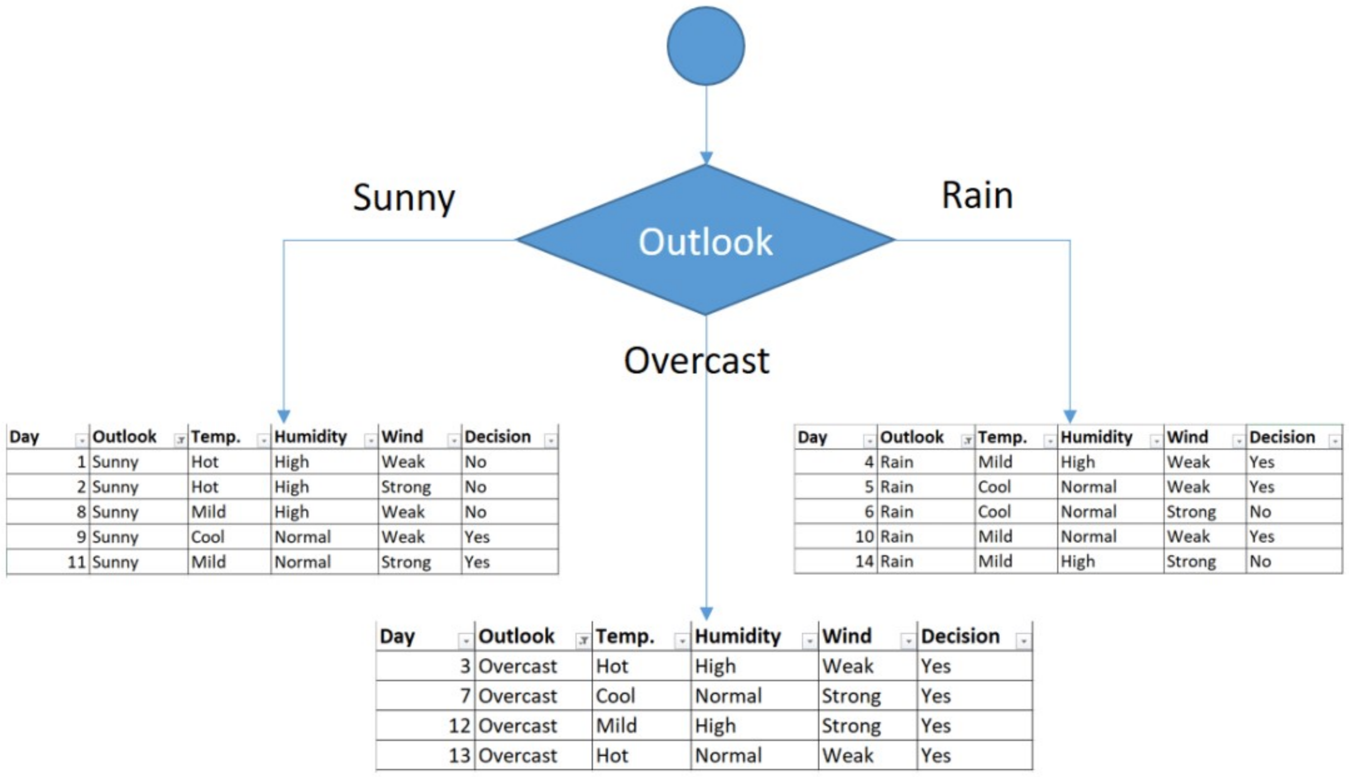
Gini(Wind) = $(8/14) \times 0.375 + (6/14) \times 0.5 = 0.428$

Time to decide

- We've calculated gini index values for each feature. **The winner will be outlook feature because its cost is the lowest.**

Feature	Gini index
Outlook	0.342
Temperature	0.439
Humidity	0.367
Wind	0.428

Decision Tree for Outlook

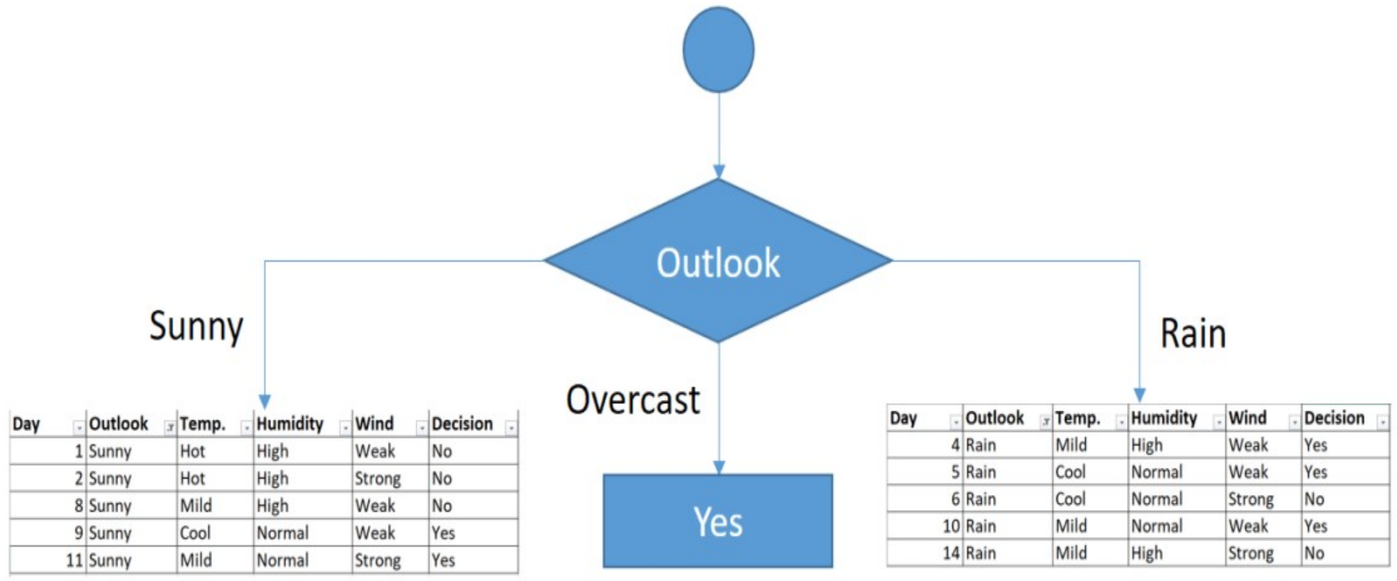


Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
10	Rain	Mild	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Day	Outlook	Temp.	Humidity	Wind	Decision
3	Overcast	Hot	High	Weak	Yes
7	Overcast	Cool	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes

➢ **Sub dataset in the overcast leaf has only yes decisions. This means that overcast leaf is over.**



Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
10	Rain	Mild	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

➢ Focus on the **sub dataset for sunny outlook. We need to find the gini index scores for temperature, humidity and wind features respectively.**

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

Gini of temperature for sunny outlook

Temperature	Yes	No	Number of instances
Hot	0	2	2
Cool	1	0	1
Mild	1	1	2

Gini(Outlook=Sunny and Temp.=Hot) = $1 - (0/2)^2 - (2/2)^2 = 0$

Gini(Outlook=Sunny and Temp.=Cool) = $1 - (1/1)^2 - (0/1)^2 = 0$

Gini(Outlook=Sunny and Temp.=Mild) = $1 - (1/2)^2 - (1/2)^2 = 1 - 0.25 - 0.25 = 0.5$

Gini(Outlook=Sunny and Temp.) = $(2/5)\text{x}0 + (1/5)\text{x}0 + (2/5)\text{x}0.5 = 0.2$

Gini of humidity for sunny outlook

Humidity	Yes	No	Number of instances
High	0	3	3
Normal	2	0	2

Gini(Outlook=Sunny and Humidity=High) = $1 – (0/3)^2 – (3/3)^2 = 0$

Gini(Outlook=Sunny and Humidity=Normal) = $1 – (2/2)^2 – (0/2)^2 = 0$

Gini(Outlook=Sunny and Humidity) = $(3/5)x0 + (2/5)x0 = 0$

Gini of wind for sunny outlook

Wind	Yes	No	Number of instances
Weak	1	2	3
Strong	1	1	2

Gini(Outlook=Sunny and Wind=Weak) = $1 – (1/3)^2 – (2/3)^2 = 0.266$

Gini(Outlook=Sunny and Wind=Strong) = $1- (1/2)^2 – (1/2)^2 = 0.2$

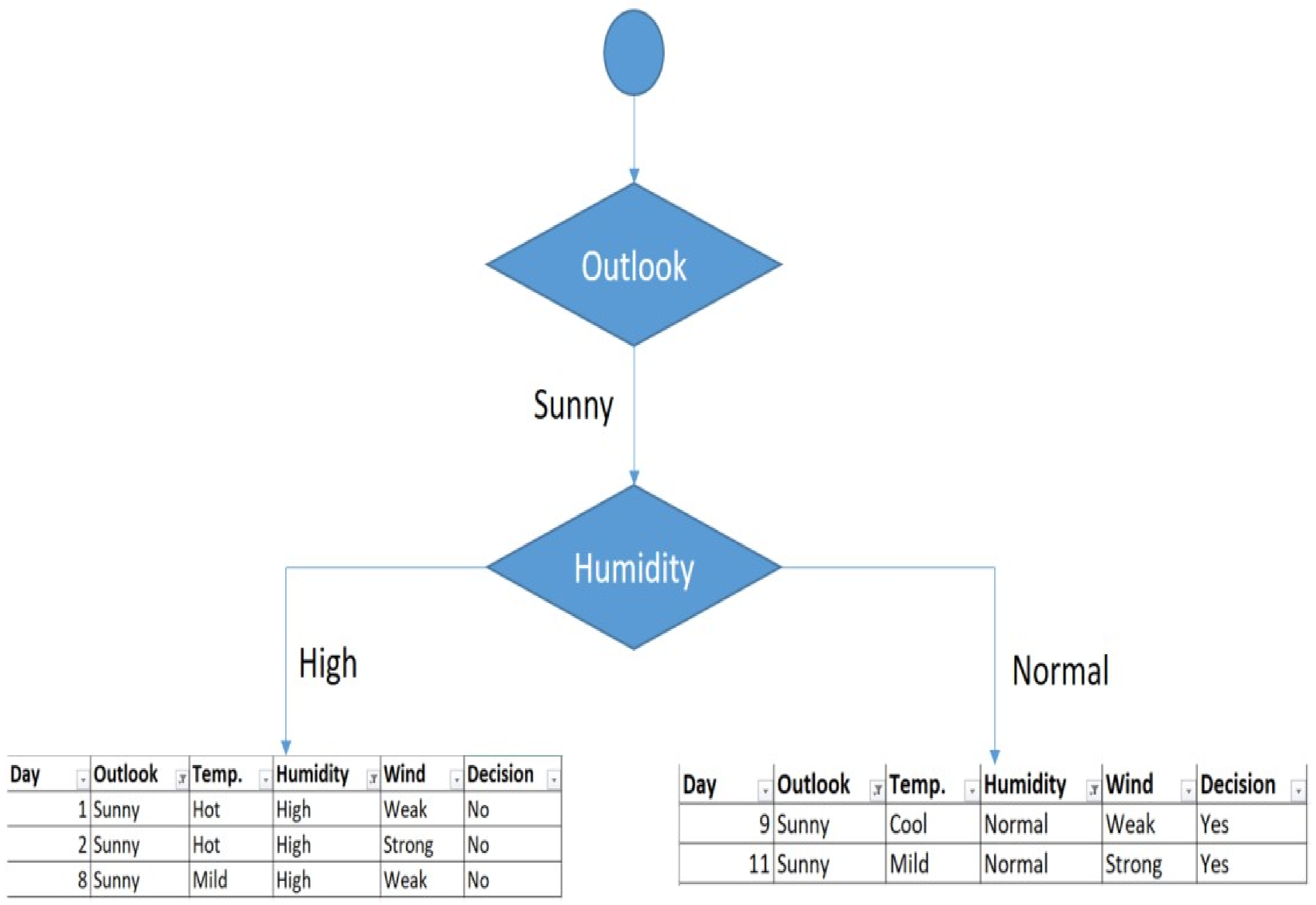
Gini(Outlook=Sunny and Wind) = $(3/5)x0.266 + (2/5)x0.2 = 0.466$

Decision for sunny outlook

- Calculated gini index scores for feature **when outlook is sunny**. **The winner is humidity because it has the lowest value.**

Feature	Gini index
Temperature	0.2
Humidity	0
Wind	0.466

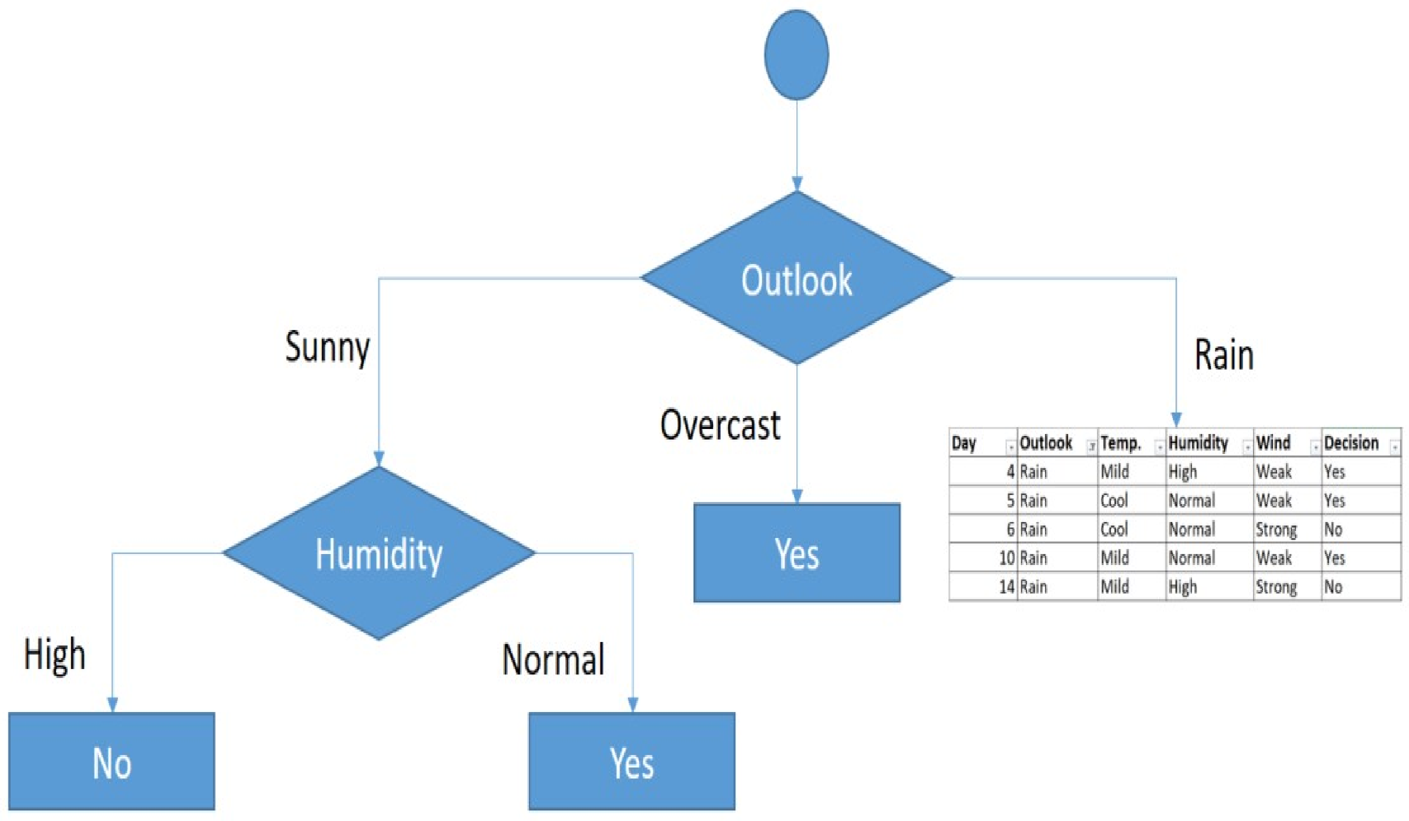
Decision Tree for Outlook (Sunny)



Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Mild	High	Weak	No

Day	Outlook	Temp.	Humidity	Wind	Decision
9	Sunny	Cool	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes

- Decision is always **no for high humidity and sunny outlook.** On the other hand, **decision will always be yes for normal humidity and sunny outlook.** This branch is over.



Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
10	Rain	Mild	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

- Now, we need to focus on rain outlook.

<u>Rain outlook</u>

Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
10	Rain	Mild	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

- Calculate **gini index scores for temperature, humidity and wind features when outlook is rain.**

Gini of temprature for rain outlook

Temperature	Yes	No	Number of instances
Cool	1	1	2
Mild	2	1	3

Gini(Outlook=Rain and Temp.=Cool) = $1 - (1/2)^2 - (1/2)^2 = 0.5$

Gini(Outlook=Rain and Temp.=Mild) = $1 - (2/3)^2 - (1/3)^2 = 0.444$

Gini(Outlook=Rain and Temp.) = $(2/5)\text{x}0.5 + (3/5)\text{x}0.444 = 0.466$

Gini of humidity for rain outlook

Humidity	Yes	No	Number of instances
High	1	1	2
Normal	2	1	3

Gini(Outlook=Rain and Humidity=High) = $1 - (1/2)^2 - (1/2)^2 = 0.5$

Gini(Outlook=Rain and Humidity=Normal) = $1 - (2/3)^2 - (1/3)^2 = 0.444$

Gini(Outlook=Rain and Humidity) = $(2/5)\text{x}0.5 + (3/5)\text{x}0.444 = 0.466$

Gini of wind for rain outlook

Wind	Yes	No	Number of instances
Weak	3	0	3
Strong	0	2	2

Gini(Outlook=Rain and Wind=Weak) = $1 – (3/3)^2 – (0/3)^2 = 0$

Gini(Outlook=Rain and Wind=Strong) = $1 – (0/2)^2 – (2/2)^2 = 0$

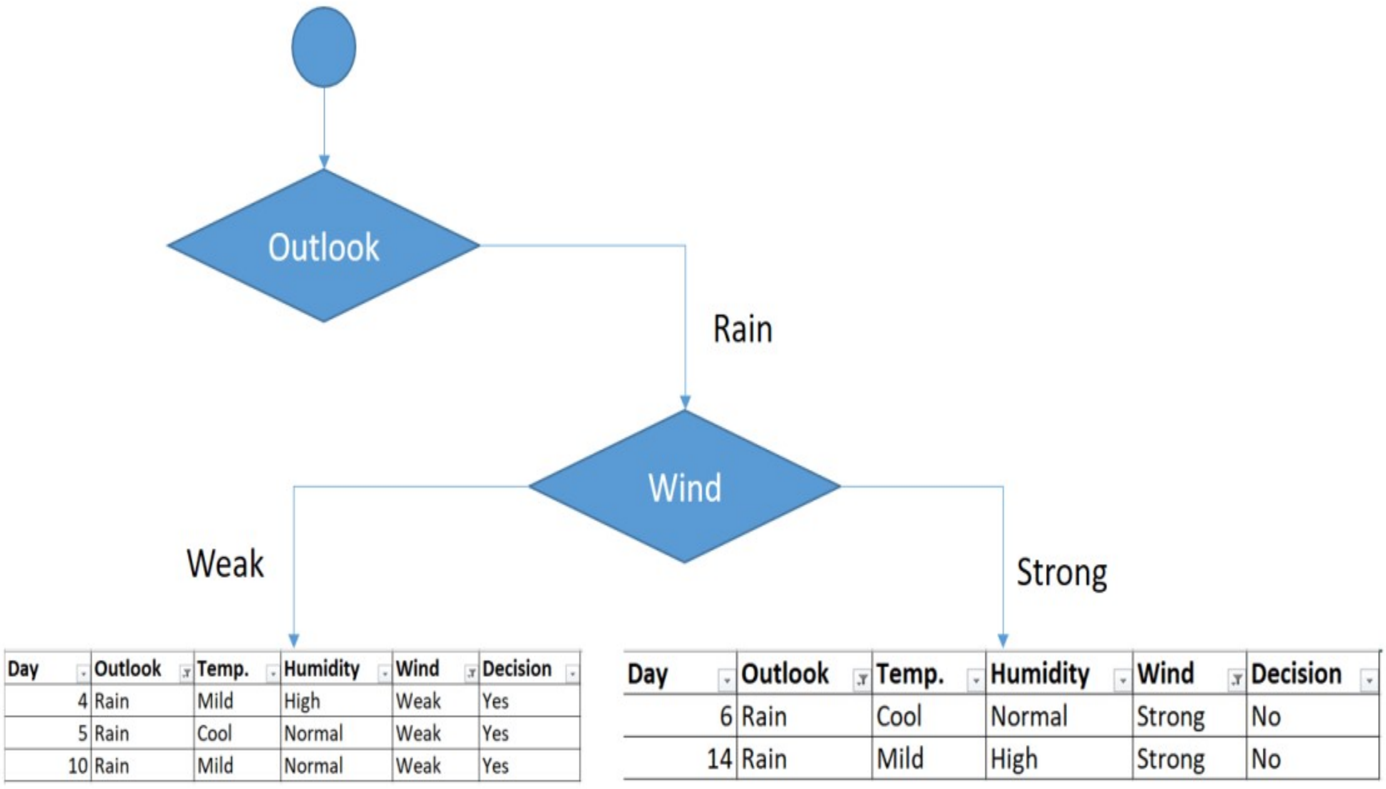
Gini(Outlook=Rain and Wind) = $(3/5)\text{x}0 + (2/5)\text{x}0 = 0$

Decision for rain outlook

- **The winner is wind feature for rain outlook because it has the minimum gini index score in features.**

Feature	Gini index
Temperature	0.466
Humidity	0.466
Wind	0

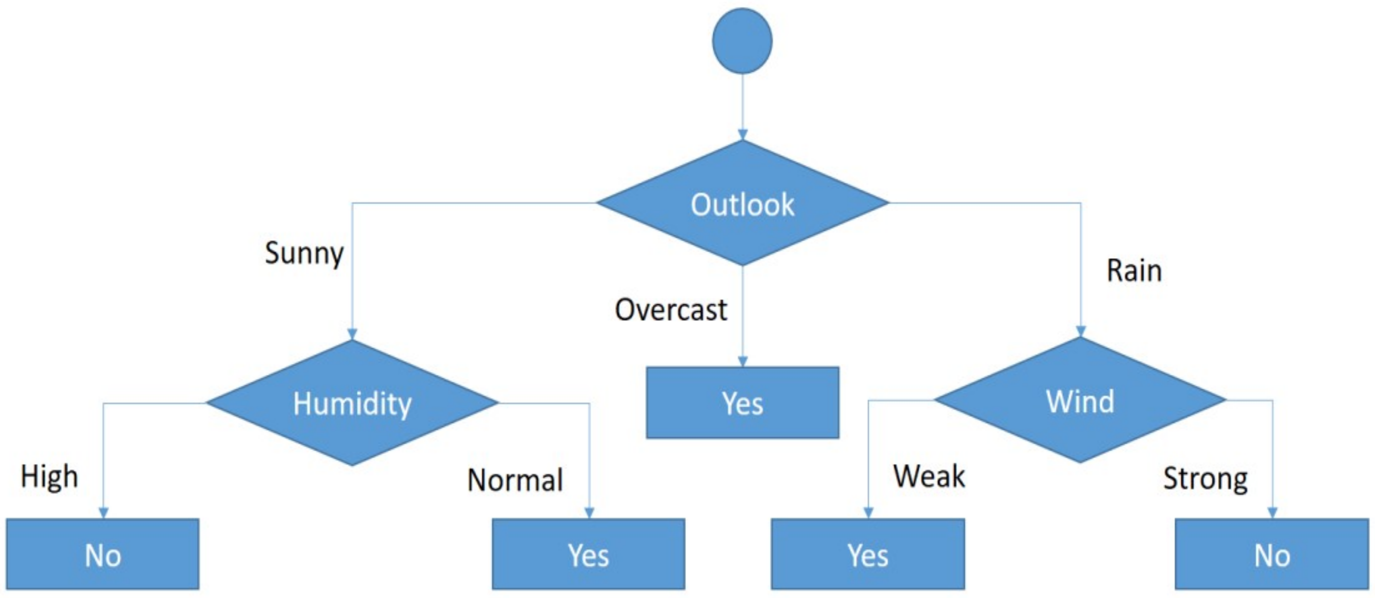
- **Put the wind feature for rain outlook branch and monitor the new sub data sets.**



Day	Outlook	Temp.	Humidity	Wind	Decision
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes

Day	Outlook	Temp.	Humidity	Wind	Decision
6	Rain	Cool	Normal	Strong	No
14	Rain	Mild	High	Strong	No

- Decision is always **yes when wind is weak.** On the other hand, decision is always **no if wind is strong.** This means that this branch is over.



<u>**Concept-4 Naive Bayes**</u>

- Naive Bayes algorithm is a <u>**supervised learning algorithm**</u>, which is based on <u>**Bayes theorem**</u> and used for solving <u>**classification problems.**</u>
- Naïve Bayes **that can be used for both classification and regression tasks.**
- It is mainly used in **text classification that includes a high-dimensional training dataset.**
- Naïve Bayes Classifier is one of the **simple and most effective Classification algorithms which help in building the fast machine learning models that can make quick predictions.**
- It is a **probabilistic classifier**, which means **it predicts on the basis of the probability of an object**.
- Some popular examples of **Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.**

<u>**Why is it called Naïve Bayes?**</u>

- The Naïve Bayes algorithm is comprised of two words Naïve and Bayes

<u>**Naive**</u>

- It is called Naive 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 on the bases of color, shape, and taste**, then **red, spherical, and sweet fruit is recognized as an apple.**
- Hence each feature individually contributes to identify that it is an apple without depending on each other.

<u>**Bayes**</u>

- It is called **Bayes** because it depends on the principle of **Bayes' Theorem.**

<u>**Bayes' Theorem**</u>

- Bayes' theorem to calculate the probability of **each class given the input features.**
- Bayes' theorem is a mathematical formula that is used to calculate the probability of an event given the probability of another event.

The formula for Bayes' theorem is given as

$$P(\text{class}|\text{features}) = \frac{P(\text{features}|\text{class}) \cdot P(\text{class})}{P(\text{features})}$$

Example

$$\overset{\text{Posterior Probability}}{P(Walks|X)} = \frac{\overset{\text{Likelihood}}{P(X|Walks)} * \overset{\text{Prior Probability}}{P(Walks)}}{\underset{\text{Marginal Likelihood}}{P(X)}}$$

Where

- **P (class | features) is Posterior probability:** It finds the probability of the class given the features.
- **P (features | class) is Likelihood probability**: It finds the probability of the features given the class.
- **P (class) is Prior Probability:** It finds the probability of the class.
- **P (features) is Marginal Probability:** It finds the probability of the features.

Working of Naive Bayes' Classifier

- Consider a dataset of **weather condition**s and corresponding target variable **"Play".**
- So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions.
- So, to solve this, we need to follow the below steps
  1. Convert the given dataset into **frequency tables**
  2. Generate **Likelihood table** by finding the probabilities of given features
  3. Now, use **Bayes theorem to calculate the posterior probability**

Problem: If the weather is sunny, then the Player should play or not?

Solution: To solve this, first consider the below dataset:

S.No	Outlook	Play
0	Rainy	Yes
1	Sunny	Yes
2	Overcast	Yes
3	Overcast	Yes
4	Sunny	No
5	Rainy	Yes
6	Sunny	Yes
7	Overcast	Yes
8	Rainy	No
9	Sunny	No
10	Sunny	Yes
11	Rainy	No
12	Overcast	Yes
13	Overcast	Yes

Frequency table for the Weather Conditions

Weather	Yes	No
Overcast	5	0
Rainy	2	2
Sunny	3	2
Total	10	4

Likelihood table weather condition

Weather	No	Yes	
Overcast	0	5	5/14= 0.35
Rainy	2	2	4/14=0.29
Sunny	2	3	5/14=0.35
All	4/14=0.29	10/14=0.71	

Applying Bayes' theorem

P (Yes | Sunny) = P (Sunny | Yes) * P (Yes) / P (Sunny)

P (Sunny | Yes) = 3/10 = **0.3**

P (Sunny) = **0.35**

P (Yes) = **0.71**

So P (Yes | Sunny) = **0.3 * 0.71 / 0.35 = 0.60**

P (No | Sunny) = P (Sunny | No) * P (No) / P (Sunny)

P (Sunny | NO) = 2/4 = **0.5**

P (No) = **0.29**

P (Sunny) = **0.35**

So P (No | Sunny) = **0.5*0.29/0.35 = 0.41**

- So as we can see from the above calculation that **P (Yes | Sunny) > P (No | Sunny).** Hence on a Sunny day, **Player can play the game.**

Advantages

- Naive Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for text classification problems.
- Low computational cost.

Disadvantages

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Applications of Naive Bayes Classifier

- It is used for Credit Scoring.
- It is used in medical data classification.
- It can be used in real-time predictions because Naïve Bayes Classifier
- It is used in Text classification such as Spam filtering and Sentiment analysis.

Concept-5 Binary Classification

What is classification?

- The Classification algorithms are a **supervised learning technique** that is used to identify the category of new observations on this basis of training data.
- The main goal of the classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data or Classes.
- **Classes can be called as targets/labels or categories. Such as, Yes or No, 0 or 1, Spam or Not Spam, Cat or dog, etc.**
- **Examples include spam email detection (spam vs. not spam), medical diagnosis (disease vs. no disease).**
- Evaluation Metrics for Classification model is **precision, recall, F1 score, accuracy.**

Types of Classifications

- The algorithm which implements the classification on a dataset is known as a classifier.
- There are two types of Classifier

Type 1: Binary Classifier

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

  Examples: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.
- Popular algorithms that can be used for binary classification include:

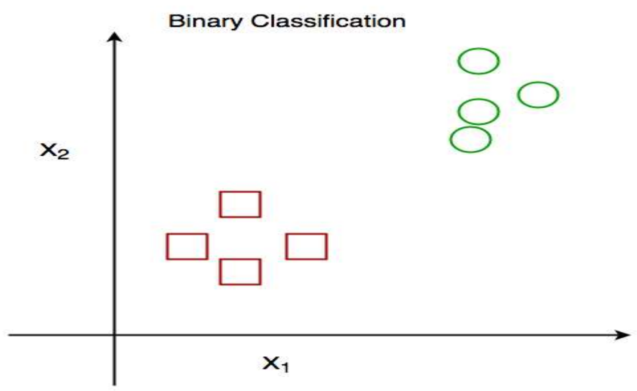
  1. Logistic Regression 2. K-Nearest Neighbours 3. Decision Trees 4. Naive Bayes

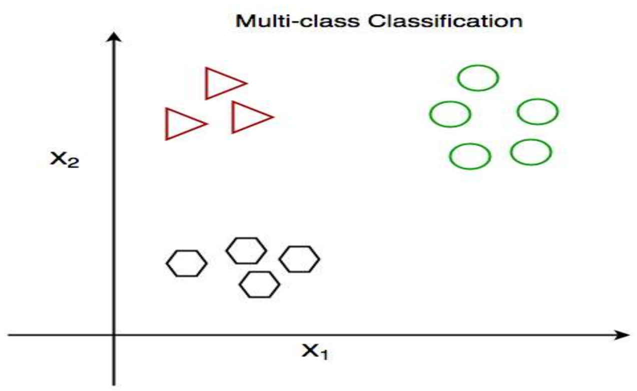
Type 2: Multi class Classifier

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

  Examples: Sentiment Analysis: Classify text into **Positive, Neutral, Negative** sentiment.

  Iris Flower Classification: Classify flowers into **Setosa, Versicolor, Virginica.**





Concept-6 Multi class/Structured Output/Classification

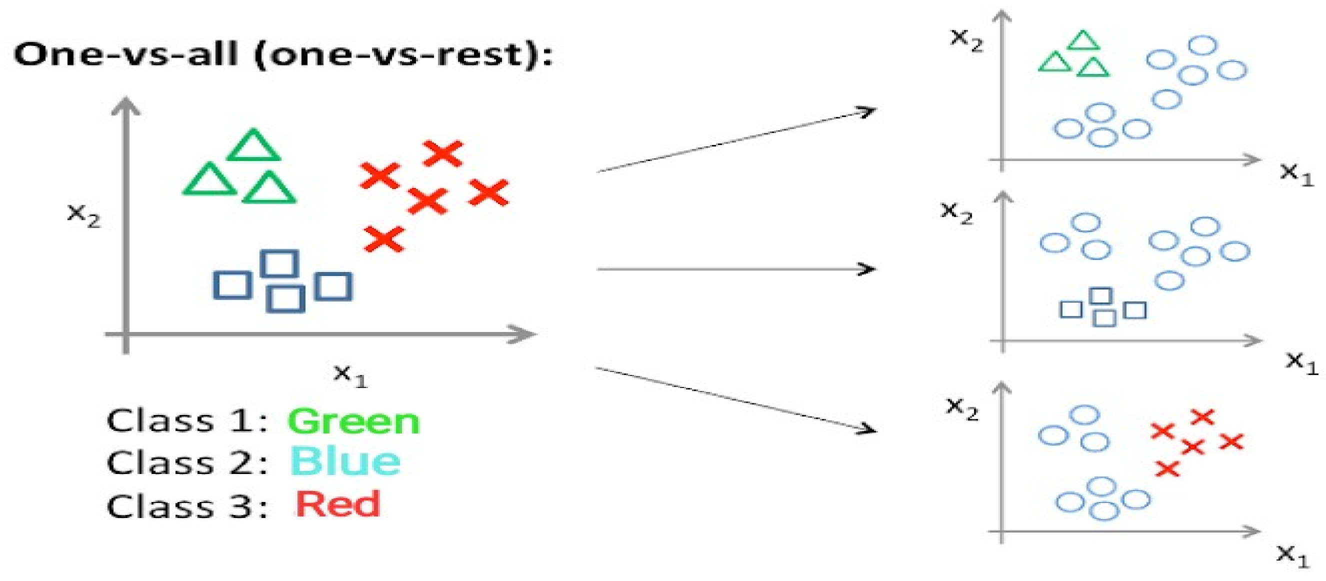
- Multiclass classification is a machine learning challenge focused on categorizing data into more than two classes.
- Multi class classification is the task of classifying elements into different classes.
- **Classes can be called as targets/labels or categories.**
- Examples of multi-class classification are
  1. **Classification of news in different categories,**
  2. **Classifying books according to the subject,**
  3. **Classifying students according to their streams etc.**
- The algorithm used to predict the class. Common algorithms include:
  a. Logistic Regression
  b. Decision Trees
  c. Support Vector Machines
  d. Neural Networks
  e. Naive Bayes
- Methods that can be used for multi-class classification include:
  1. **One vs All (One vs Rest) (N-class instance then N binary classifier models)**
  2. **One vs One (N-class instances then N*(N-1)/2 binary classifier models)**

One Vs rest:

- The One vs All classification, for the N-class instance dataset, we must generate the N-binary classifier models.
- The number of class lables present in the dataset and the number of generated binary classifiers must be the same.
- Consider we have three classes, for example, **Green, Blue, and Red.** Now, we create **three classifiers** here for three respective classes.

Classifier 1:- [Green] vs [Red, Blue] **Classifier 2:- [Blue] vs [Green, Red]**

Classifier 3:- [Red] vs [Blue, Green]



Example

- You can see that there are **three class labels Green, Blue, and Red present in the dataset.** Now we must create a training dataset for each class.

Features			Classes
x1	x2	x3	G
x4	x5	x6	B
x7	x8	x9	R
x10	x11	x12	G
x13	x14	x15	B
x16	x17	x18	R

Class 1 :- Green

Class 2 :- Blue

Class 3 :- Red

- We created the training datasets by putting +1 in the class column for that feature value, which is aligned to that particular class only.
- For the costs of the remaining features, we put -1 in the class column.

Main Dataset

Features			Classes
x1	x2	x3	G
x4	x5	x6	B
x7	x8	x9	R
x10	x11	x12	G
x13	x14	x15	B
x16	x17	x18	R

Class 1 :- Green Class 2 :- Blue Class 3 :- Red

Training Dataset 1
Class :- Green

Features			Green
x1	x2	x3	**+1**
x4	x5	x6	-1
x7	x8	x9	-1
x10	x11	x12	**+1**
x13	x14	x15	-1
x16	x17	x18	-1

Training Dataset 2
Class :- Blue

Features			Blue
x1	x2	x3	-1
x4	x5	x6	**+1**
x7	x8	x9	-1
x10	x11	x12	-1
x13	x14	x15	**+1**
x16	x17	x18	-1

Training Dataset 3
Class :- Red

Features			Red
x1	x2	x3	-1
x4	x5	x6	-1
x7	x8	x9	**+1**
x10	x11	x12	-1
x13	x14	x15	-1
x16	x17	x18	**+1**

- Let's understand with one example by taking **three test features values as y1, y2, and y3,** respectively. We passed test data to the classifier models.
- Let's say, we got the outcome as,

  Green class classifier -> Positive with a probability score of (0.9)

  Blue class classifier -> Positive with a probability score of (0.4)

  Red class classifier -> Negative with a probability score of (0.5)
- Hence, based on the **positive responses** and decisive probability score, we can say that our test input belongs to the **Green class.**

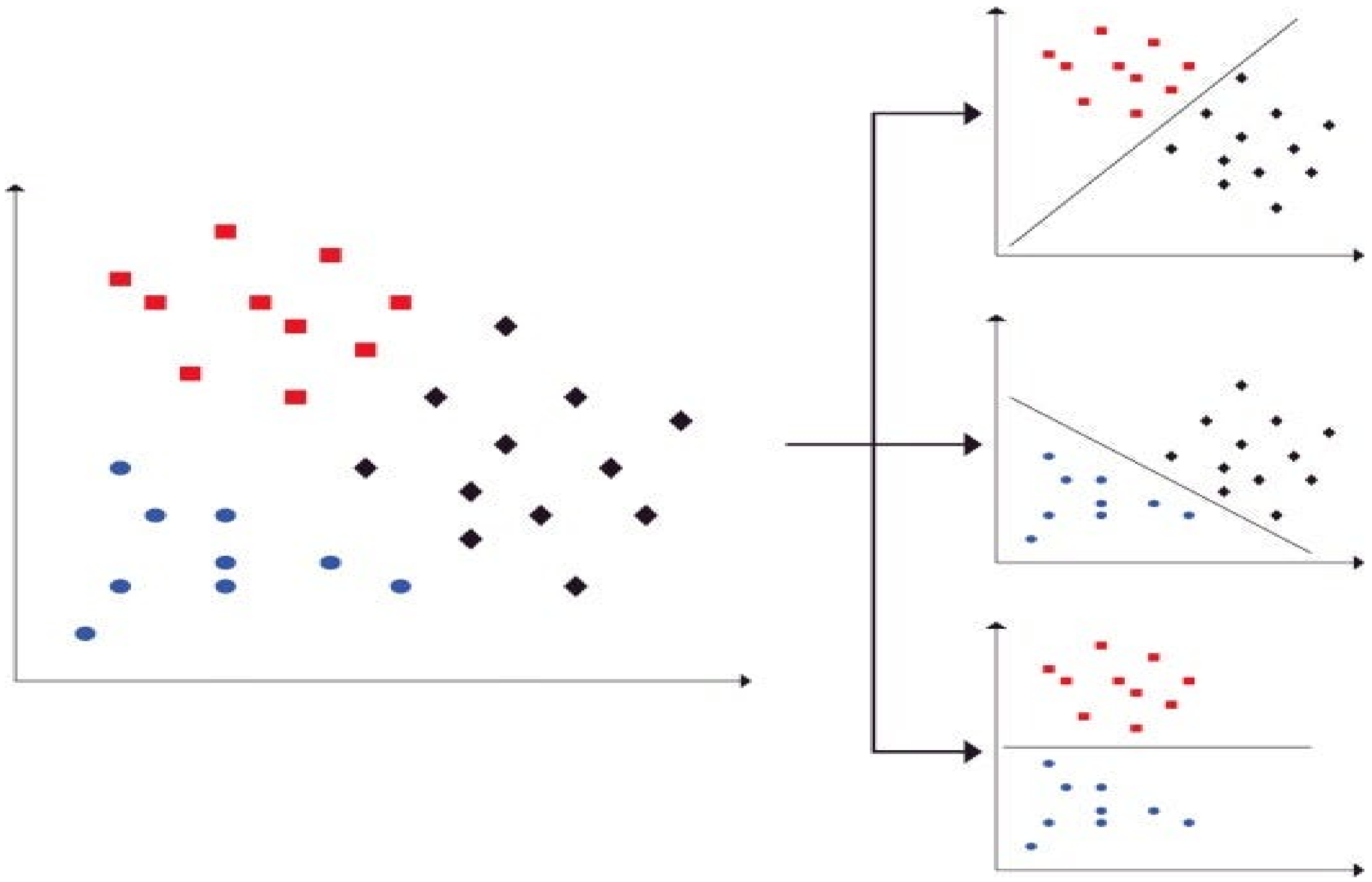
<u>**One Vs One**</u>

- In One-vs-One classification, for the **N-class instances dataset**, we must generate the **N* (N-1)/2 binary classifier models.**
- We split the primary dataset into one dataset for each class opposite to every other class.
- Taking the above example, we have a classification problem having three types: **Green, Blue, and Red (N=3)**
- We divide this problem into N* (N-1)/2 = 3 binary classifier problems:

  Classifier 1: Green vs. Blue

  Classifier 2: Green vs. Red

  Classifier 3: Blue vs. Red
- Each binary classifier predicts one class label.
- When we input the test data to the classifier, then the model with the majority counts is concluded as a result.



Binary vs Multiclass Classification

Binary Classification

- Only two class instances are present in the dataset.
- It requires only one classifier model.
- Confusion Matrix is easy to derive and understand.

  Example: Check email is spam or not, predicting gender based on height and weight.
- Popular algorithms that can be used for binary classification include:

  1. Logistic Regression 2. K-Nearest Neighbours 3. Decision Trees 4. Naive Bayes

Multi-class Classification

- Multiple class labels are present in the dataset.
- The Confusion matrix is easy to derive but complex to understand
- **The algorithm used to predict the Multi class classification. Common algorithms include:**

  1. Logistic Regression 2. Decision Trees 3. Support Vector Machines

  4. Neural Networks 5. Naive Bayes
- The number of classifier models depends on the classification technique we are applying to.

  One vs. All: N-class instances then N binary classifier models

  One vs. One: N-class instances then N*(N-1)/2 binary classifier models

  Example: Check whether the fruit is apple, banana, and orange.

Concept-7 Linear Regression

- **Regression are used if there is a relationship between the input variable and the output variable**
- **These are used to predict continuous output variables.**
- Linear Regression is one of the most **simple machine learning algorithms** that come under supervised learning technique and used for solving **regression problems.**
- **The output for linear regression should only be the continuous vales such as sales, salary, age, product price, etc.**
- **Linear regression algorithm shows a linear relationship between a dependent variable (x) and one or more independent variables (y) hence called as linear regression.**
- **The goal of the linear regression is to find the best fit line that can accurately predict the output.** And the relationship should be of **linear nature**.
- **Independent variables, also called inputs or predictors, do not depend on other features of datasets.**

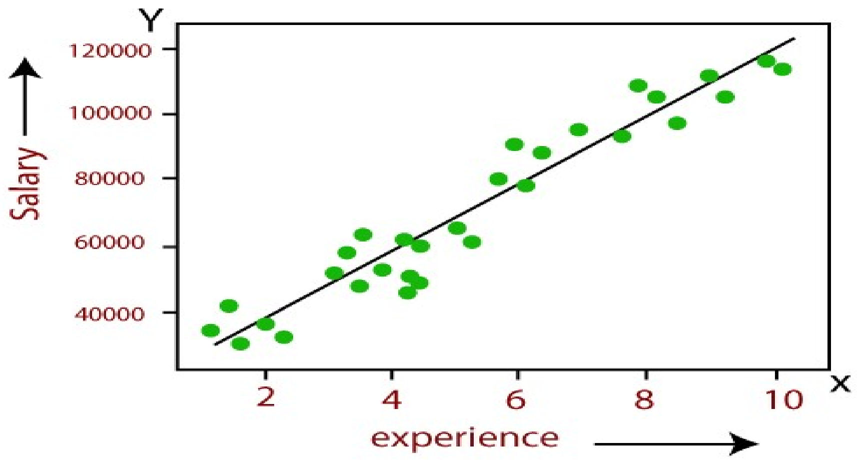
  Example: Experience.

  Dependent variables, also called output or responses, depend on the independent variables.

  Example: Salary.
- The relationship between variables in the linear regression model can be explained using the below image. **Here we are predicting the salary of an employee on the basis of the year of experience.**

The Regression line can be written as: $y=a0+a1x+\varepsilon$

Where y is dependent variable, x is independent; a0 and a1 are the coefficients and ε is the random error



- **Dependent Variable is on Y-axis (Salary) and independent variable is on X-axis (Experience).**
- Some popular applications of linear regression are: **Analyzing trends and sales estimates, Salary forecasting, Real estate prediction.**
- The linear regression model provides a sloped straight line representing the relationship between the variables.

Types of Linear Regression

- Linear regression can be further divided into two types of the algorithm:

Simple Linear Regression:

- If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called **Simple Linear Regression**.

Multiple Linear Regression:

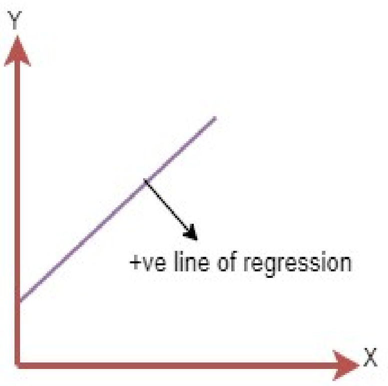
- If more than one independent variables is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called **Multiple Linear Regression.**

Linear Regression Line

- A linear line showing the relationship between the dependent and independent variables is called a regression line.
- A regression line can show two types of relationship:

Positive Linear Relationship

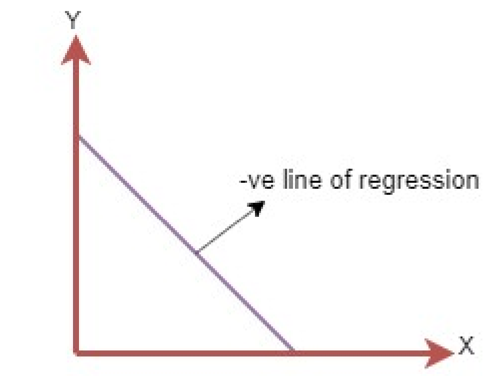
- If the dependent variable **increases** on the Y-axis and independent variable **increases** on X-axis, then such a relationship is termed as a **Positive linear relationship.**



The line equation will be: $Y = a_0 + a_1x$

Negative Linear Relationship

- If the dependent variable **decreases** on the Y-axis and independent variable **increases** on the X-axis, then such a relationship is called a **negative linear relationship.**



The line of equation will be: **$Y= -a_0+a_1x$**

Finding the best fit line

- Our main goal is to **find the best fit line** that **means the error between predicted values and actual values should be minimized. The best fit line will have the least error.**

Cost function

- For Linear Regression, we use **the Mean Squared Error (MSE) cost function**, which is the average of squared error occurred between the predicted values and actual values.
- For the above linear equation, MSE can be calculated as:

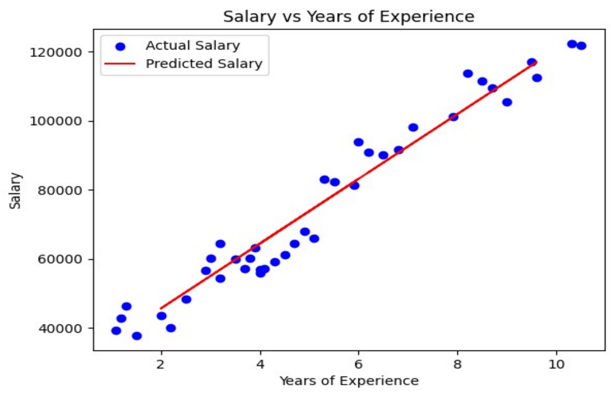
$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Where, N=Total number of observation, yi and yi are actual values and predicted values.

Program

```
# Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

# Load the dataset (assuming CSV file)
# You can replace 'salary_data.csv' with the actual path to your dataset
# Sample dataset contains two columns: 'YearsExperience' and 'Salary'
df = pd.read_csv('salary_data.csv')

# Features (Years of Experience) and Target (Salary)
X = df[['Experience Years']]  # Independent variable (Years of Experience)
y = df['Salary']  # Dependent variable (Salary)

# Split the data into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Create a Linear Regression model and train it
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Display the regression line on a plot
plt.scatter(X, y, color='blue', label='Actual Salary')
plt.plot(X_test, y_pred, color='red', label='Predicted Salary')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.title('Salary vs Years of Experience')
plt.legend()
plt.show()
```



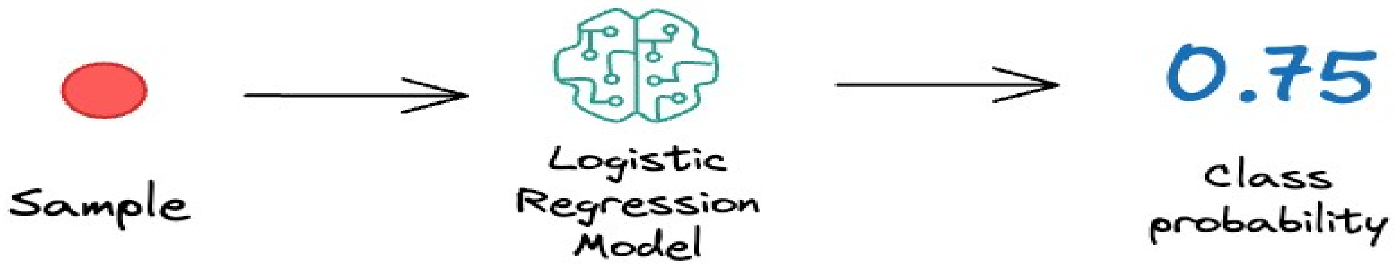
```
# Example: Predicting salary for a custom input
custom_years_experience = np.array([[10]])  # Input: 10 years of experience

predicted_salary = model.predict(custom_years_experience)

print(f"Predicted Salary for 10 years of experience: {predicted_salary[0]}")
```

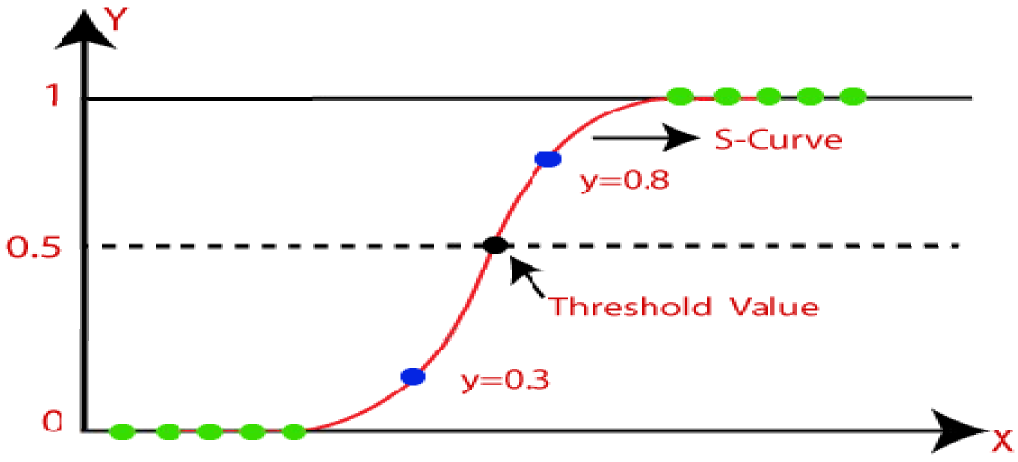
Output

```
Predicted Salary for 10 years of experience: 120796.56290121133
```

Concept-8 Logistic Regression

- Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique.
- Logistic regression predicts the output of categorical values. Therefore the outcome must be a categorical value.
- It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, **it gives the probabilistic values which lie between 0 and 1**.



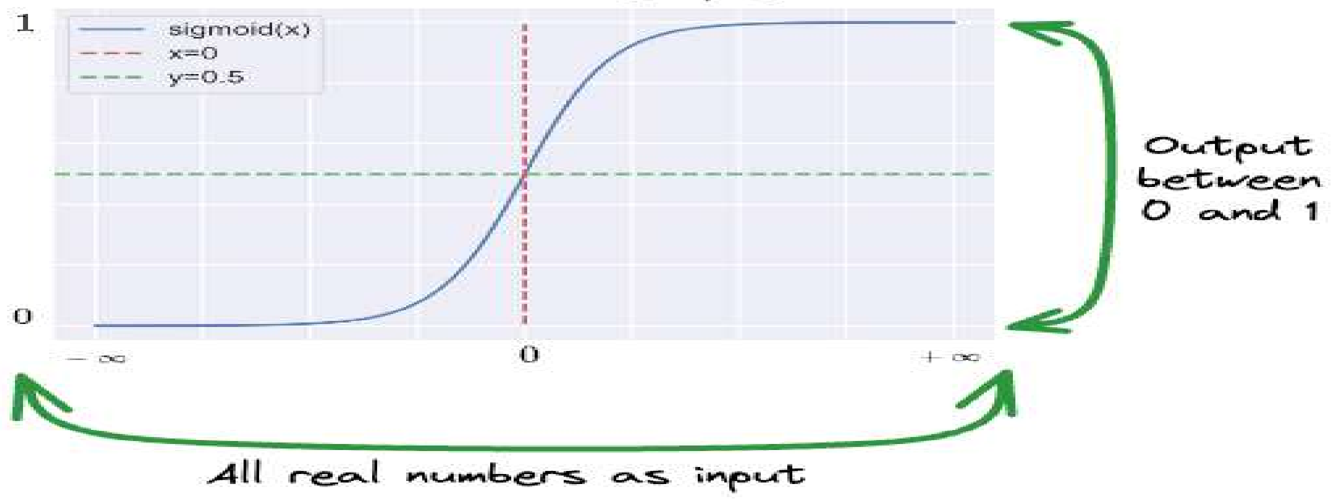


- Logistic Regression is much similar to the Linear Regression **except that how they are used.**
- Logistic regression is used for binary classification where we use sigmoid function, that takes input as independent variables and produces a probability value between 0 and 1.
- **For example, we have two classes Class 0 and Class 1 if the value of the logistic function for an input is greater than 0.5 (threshold value) then it belongs to Class 1 otherwise it belongs to Class 0.**
- Linear Regression is used for solving Regression problems, whereas **Logistic regression is used for solving the classification problems**.
- In Logistic regression, instead of **fitting a regression line**, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
- Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.

Logistic Function (Sigmoid Function)

- The sigmoid function is a mathematical function used to map the **predicted values to probabilities.**
- **Sigmoid function converts the continuous variable data into the probability i.e. between 0 and 1**.
- The value of the logistic regression must be **between 0 and 1,** which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the **Sigmoid function or the logistic function.**
- In this we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

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



Logistic Regression Equation

- The Logistic regression equation can be obtained from the Linear Regression equation.
- We know the equation of the straight line can be written as:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 + \cdots + b_n x_n$$

- The equation for logistic regression is:

$$\log\left[\frac{y}{1-y}\right] = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 + \cdots + b_n x_n$$

Type of Logistic Regression

- Logistic Regression can be classified into three types:

  Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.

  Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"

  Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Program

```
# Import necessary libraries
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

# Step 1: Generate or Load Data
# Example dataset: income, credit_score, loan_amount, and approval_status
data = {
   "income": [30000, 50000, 20000, 40000, 45000, 32000, 80000, 100000],
   "credit_score": [650, 720, 600, 710, 680, 640, 750, 800],
   "loan_amount": [10000, 20000, 15000, 25000, 22000, 18000, 40000, 50000],
   "approval_status": [1, 1, 0, 1, 1, 0, 1, 1],  # 1 = Approved, 0 = Rejected
}

# Convert to a Pandas DataFrame
df = pd.DataFrame(data)

# Step 2: Prepare Features and Labels
X = df[["income", "credit_score", "loan_amount"]]  # Features
y = df["approval_status"]  # Target variable

# Step 3: Split Data into Training and Testing Sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Step 4: Build the Logistic Regression Model
model = LogisticRegression()
model.fit(X_train, y_train)
```

```
# Step 5: Predict for New Inputs
def predict_loan_approval(income, credit_score, loan_amount):
   prediction = model.predict([[income, credit_score, loan_amount]])
    return "Approved" if prediction[0] == 1 else "Rejected"

# Example Prediction
print("\nExample Prediction:")
print(predict_loan_approval(40000, 700, 20000))  # Test with custom inputs
```

Output

```
Example Prediction:     Approved
```

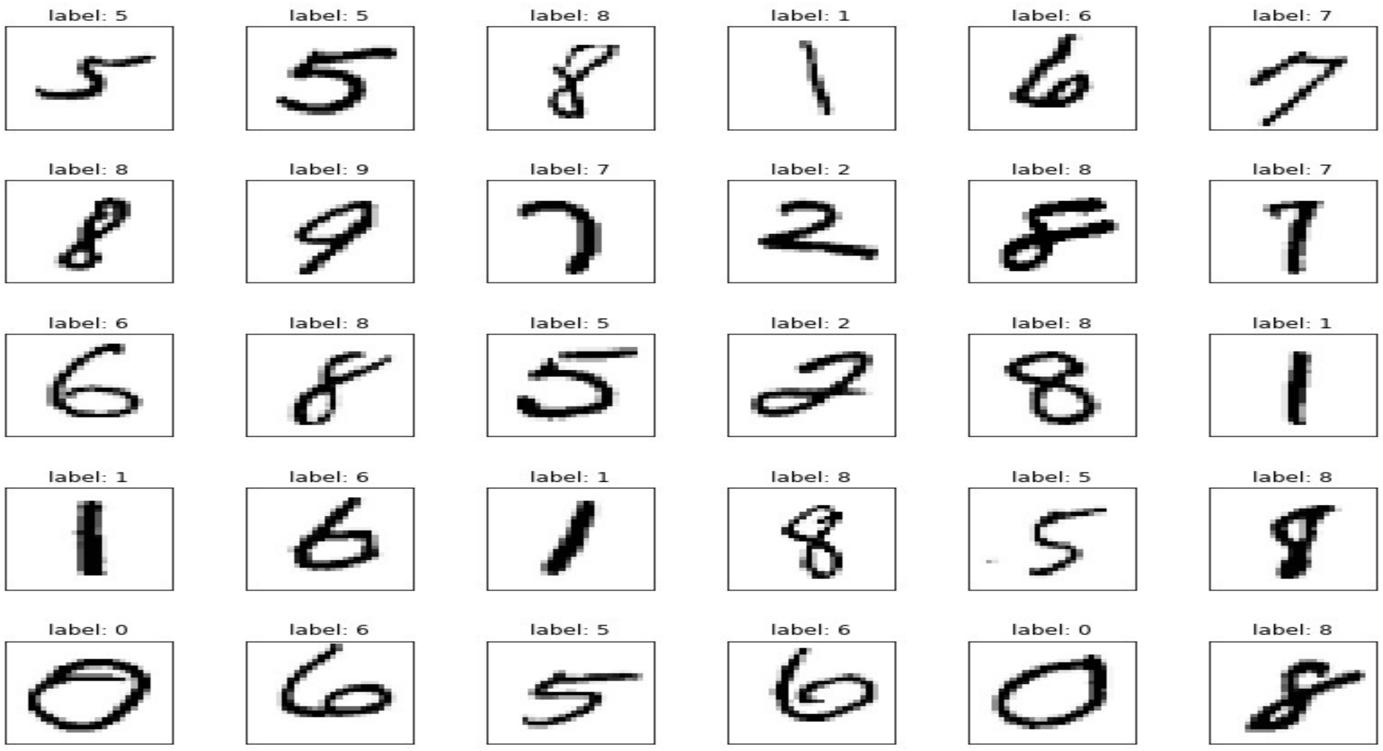
Linear Regression (vs) Logistic Regression

Linear Regression	Logistic Regression
Linear regression is used to predict the continuous dependent variable using a given set of independent variables.	Logistic Regression is used to predict the categorical dependent variable using a given set of independent variables.
Linear Regression is used for solving **Regression problem.**	Logistic regression is used for solving **Classification problems.**
In Linear regression, we predict the value of **continuous variables.**	In logistic Regression, we predict the values of **categorical variables.**
In linear regression, we find the best fit line, by which we can easily predict the output.	In Logistic Regression, we find the S-curve by which we can classify the samples.
Least square estimation method is used for estimation of accuracy.	Maximum likelihood estimation method is used for estimation of accuracy.
The output for Linear Regression must be a **continuous value, such as price, age, etc.**	The output of Logistic Regression must be a **Categorical value such as 0 or 1, Yes or No, etc.**
In Linear regression, it is required that relationship between dependent variable and independent variable must be linear.	In Logistic regression, it is not required to have the linear relationship between the dependent and independent variable.
In linear regression, there may be collinearity between the independent variables.	In logistic regression, there should not be collinearity between the independent variable.

Concept-9 MNIST

- The MNIST **(Modified National Institute of Standards and Technology database)**
- MNIST Dataset which is set of **70,000 small images** of digits handwritten by **high school students and employees of the US Census Bureau.**
- The MNIST dataset is a popular dataset used for training and testing in the field of machine learning for **handwritten digit recognition.**
- Each image is **labelled** with the digit it represents
- **MNIST is a collection of gray-scale images of handwritten digits.**
- This collection is made up of 60,000 images for training and 10,000 images for testing model performance. Each image is 28 X 28 pixels in gray-scale.

Example of MNIST Datasets



- The MNIST digits dataset is often used by data scientists who want to try machine learning techniques and pattern recognition methods on real-world data while spending minimal effort on pre-processing and formatting.

Structure of MNIST dataset

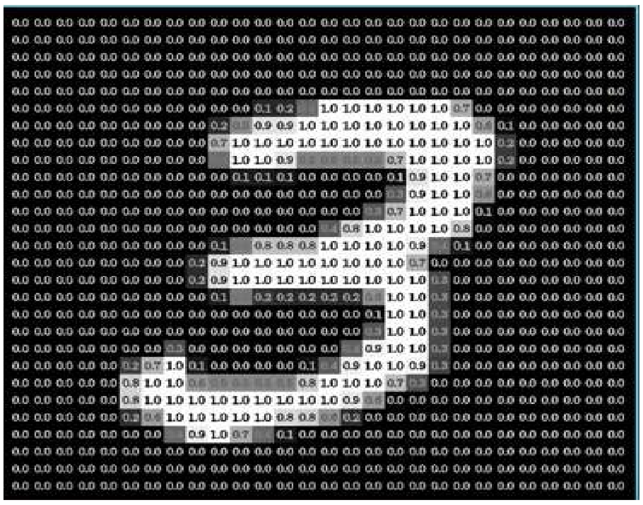
- The MNIST dataset is a collection of 70,000 handwritten digits (0-9), with each image being 28x28 pixels. Here is the dataset information in the specified format:
- **Number of Instances:** 70,000 images
- **Number of Attributes:** 784 (28x28 pixels)
- **Target:** Column represents the digit (0-9) corresponding to the handwritten image
- **Pixel 1-784:** Each pixel value (0-255) represents the grayscale intensity of the corresponding pixel in the image.
- **The dataset is divided into two main subsets:**
- **Training Set:** Consists of 60,000 images along with their labels, commonly used for training machine learning models.
- **Test Set:** Contains 10,000 images with their corresponding labels, used for evaluating the performance of trained models.
- **Scikit**-Learn provides many helper functions to download popular datasets called as MNIST Datasets
- **The following code fetches the MNIST Datasets**

```
from sklearn.datasets import fetch_openml
MNIST=fetch_openml("MNIST_784",version=1)
X, y = MNIST ["data"], MINST ["target"]
X.shape    (70000,784)
y.shape    (70000,)
```

- The MNIST dataset is actually already split into a training set (the first 60,000 images) a test set (the last 10,000 images).

  X_train,X_test,y_train,y_test=X[:60000],X[60000:],y[:60000],y[60000:]

<u>28x28 pixels images of digit number "3" in this here 0 (black) or 255 (white)</u>



<u>Concept-10 Ranking</u>

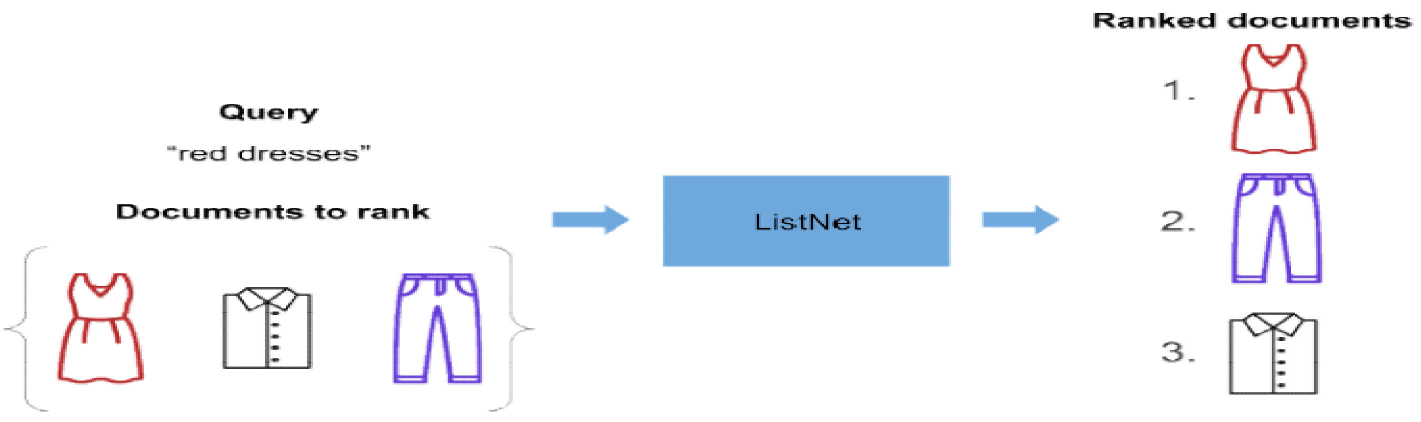
- Ranking is a type of supervised machine learning (ML) that uses labeled datasets to train its data and models to classify future data to predict outcomes. Quite simply, the goal of a ranking model is to sort data in an optimal and relevant order.
- A ranking algorithm is a procedure that ranks items in a dataset according to some criterion.
- Ranking was first largely deployed **within search engines.**
- Ranking algorithms are used in many different applications**, such as web search, recommender systems**.
- Ranking algorithms are used in search engines to rank webpages according to their relevance to a user's search query.

<u>Key Components of Ranking</u>

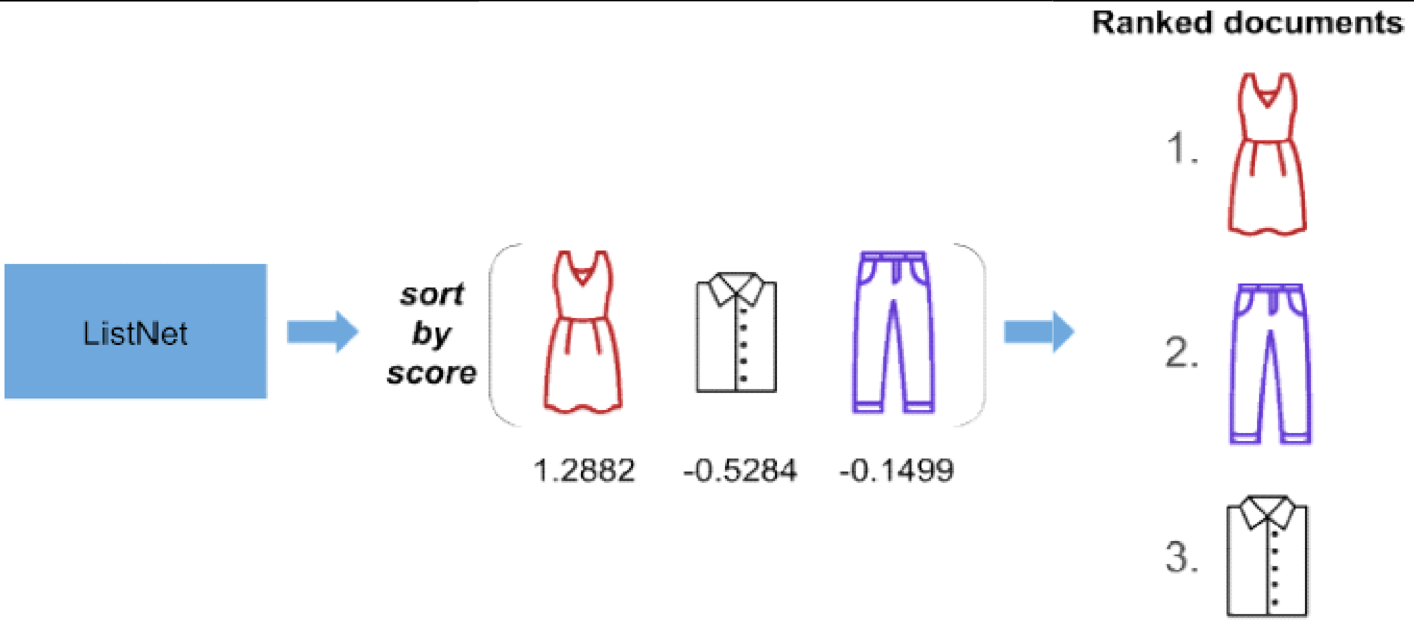
Query: The input to the ranking system (e.g., a search query or user preferences).

Items: The entities that are ranked (e.g., web pages, products, or documents).

Relevance: A measure of how well each item satisfies the query (e.g., how relevant a webpage is to a search query).



- **ListNet** outputs a bunch **of real numbers.** Each real number is a **score** assigned to the document we want to rank. We simply sort the documents in **descending order of score**, and this tells us how the original list of documents should be ranked.



Types of Ranking Problems

Pointwise Ranking

- Each item is evaluated independently, and a score or rank is assigned to it. The model does not consider the order of items at this stage.
- The task is to predict a score for each item that reflects its relevance to the query.
- This approach is often addressed **with regression or classification algorithms.**

Example: In a movie recommendation system, you could rate each movie independently (e.g., 1-5 stars).

Movies	Predicted Rating (1-5)
Movie A	4.8
Movie B	3.9
Movie C	2.1

Pairwise Ranking

- The model learns by comparing pairs of items. It looks at two items at a time and determines which one is better or more relevant to the query.
- The task is to determine the relative preference between items (e.g., which is more relevant).
- A ranking model using pairwise approaches can use techniques like logistic regression or neural networks for binary classification between pairs of items.

Example: For a search query, the model compares two documents and predicts which one is more relevant. The idea is that for any two items A and B, the model should be able to say whether A is better than B, or vice versa.

Pair (Movie A, Movie B): **A is preferred**

Pair (Movie A, Movie C): **A is preferred**

Pair (Movie B, Movie C): **B is preferred**

Listwise Ranking

- In listwise ranking, the model considers the entire list of items together. It optimizes the order of the whole list rather than considering items in isolation (pointwise) or comparing pairs.

- The goal is to optimize the ranking of all items in the list at once to achieve the best overall order.
- **Algorithms like LambdaMART and RankNet are popular for listwise ranking problems.** These models aim to optimize the entire list by considering **ranking metrics like mean reciprocal rank (MRR), normalized discounted cumulative gain (NDCG), or mean average precision (MAP).**
- **Example**: For a search engine result, the model doesn't just rank individual documents. It looks at the whole list of documents and ensures that the most relevant documents are at the top.

Rank	Movies	Predicted Rating (1-5)
1	Movie A	4.8
2	Movie B	3.9
3	Movie C	2.1

Applications of Ranking in Machine Learning

- **Search Engines**: Ranking web pages based on relevance to a user's query.
- **Recommendation Systems**: Ranking movies, products, or music based on user preferences.
- **Advertising**: Ranking ads to show the most relevant ones to users.
- **Document Retrieval**: Ranking research papers or articles based on relevance to a query.

Concept-10 Generalized Linear Models

- Generalized linear models (GLMs) explain how linear regression and Logistic regression is a member of a much broader class of models.
- GLMs can be used to construct the models for regression and classification problems by using the type of distribution which best describes the data or labels given for training the model.
    a. Binary classification data – Bernoulli distribution
    b. Real valued data – Gaussian distribution
    c. Count-data – Poisson distribution

Bernoulli distribution

- A Bernoulli distribution is a discrete probability distribution that models a random variable with only two possible outcomes.
- These outcomes are typically labeled as "success" and "failure," or else they are represented numerically as 1 and 0.

Bernoulli trials

- A Bernoulli trial is a random experiment with exactly two possible outcomes.
    a. Flipping a coin (heads or tails)
    b. Answering a true/false question
    c. Determining if a customer will make a purchase (buy or not buy)
- Each Bernoulli trial is independent, meaning the outcome of one trial does not affect the probability of success in subsequent trials.
- A Bernoulli distribution describes the probability of success in a single Bernoulli trial.
- It is characterized by a single parameter, p, which represents the probability of success.
- The probability of failure is consequently 1 - p.

- Mathematically, we can express a Bernoulli distribution as follows, where X is the random variable representing the outcome of the Bernoulli trial.

$$P(X = 1) = p, \quad P(X = 0) = 1 - p$$

Gaussian distribution

- The Gaussian distribution, also known as the normal distribution, is a continuous probability distribution that is widely used in statistical modeling and Machine Learning.
- It is a bell-shaped curve that is symmetrical around its mean and is characterized by its mean and standard deviation.
- The Gaussian distribution is important because it is a common and useful model for data, and because it allows for efficient calculations in statistical analysis
- Gaussian Distribution formula is

f(x) = (1 / sqrt(2 * pi * sigma^2)) * exp(-((x – mu)^2) / (2 * sigma^2))

Where,

- **X** is a real number representing a possible value of a continuous random variable.
- **mu** is the mean of the distribution, and sigma is the standard deviation.
- **(1 / sqrt(2 * pi * sigma ^ 2))** – is the normalization factor that ensures that the area under the curve of the distribution is equal to 1; and **exp (-((x – mu) ^ 2) / (2 * sigma^2))** – is a bell-shaped curve that is centered at the **mean, mu**, and has a standard deviation of sigma.

Poisson distribution

- Poisson distribution is a probability distribution that is used to show how many times an event occurs over a specific period.
- It is the discrete probability distribution of the number of events occurring in a given time period, given the average number of times the event occurs over that time period.
- It is the distribution related to probabilities of events that are extremely rare but have a large number of independent opportunities of occurrence.

$$P(x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

Important Question

1. Compare linear regression with logistic regression
2. Describe the importance of K-Values in nearest neighbour algorithms in detail.
3. Explain the step to be followed in distance based classification models
4. Write a note on linear regression. Implement the linear regression to predict the stock market price prediction
5. How can decision tree be used to classify the attributes? Explain the algorithm steps
6. What is the sigmoid function? Where it can be used? Explain
7. What is bayes theorem? Explain Naïve bayes with an example.
8. What is ranking in binary classification in machine learning? What is the best algorithm for raking?
9. What is the purpose of sigmoid function in logistic regression? Explain.
10. Discuss about multi class classification technique.
11. What is the decision tree? How to choose attribute selection in decision tree?
12. Explain about decision tree classifier with an example.
13. Explain about MNIST dataset. Describe the procedure to apply classification technique.
14. What are general linear models? Give their parametric equations
15. What is the role of distance measure ML algorithms? Illustrate.
16. Explain KNN algorithm with an example.
17. Explain the working principal of logistic regression. How is it different from linear regression? Given an example.
18. What is multi-class classification? With MNIST data sets, explain the algorithm.
19. Differentiate classification and regression problems and their solutions with examples.