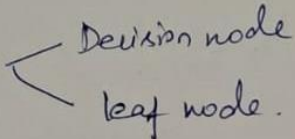


## UNIT-2

### DECISION TREES

Decision Tree based Algorithm:

- Supervised learning technique
- It is a tree structured classifier, where internal nodes represent - features of dataset  
branches - decision rules  
leaf node - outcome.

In decision tree    
Decision node  
leaf node.

- To build a tree, we use CART algorithm which is class & regression alg.
- Simply ask questions Y/N and further split into sub trees.



eg graph.

⇒ The y axis is the posterior probability  $P(w_i|x)$  and the x axis is our feature 'x'.

⇒ The axis where posterior probability of both the class are equal is called decision boundary

$$\Rightarrow P(w_1|x) = P(w_2|x)$$

The  $w_1$  and  $w_2$  magnitude are extended to other classes which is called risk or probability error.

Calculation of probability error:  
Mathematically,

$$w_1 \text{ is } P(w_2|x)$$

and

$$w_2 \text{ is } P(w_1|x)$$

$$P(E|x) = \text{minimum}(P(w_1|x), P(w_2|x)).$$

Our probability of total error is the minimum of the posterior probability of both the class.



## Decision Tree Algorithm

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too.

The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data).

In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

## Types of Decision Trees

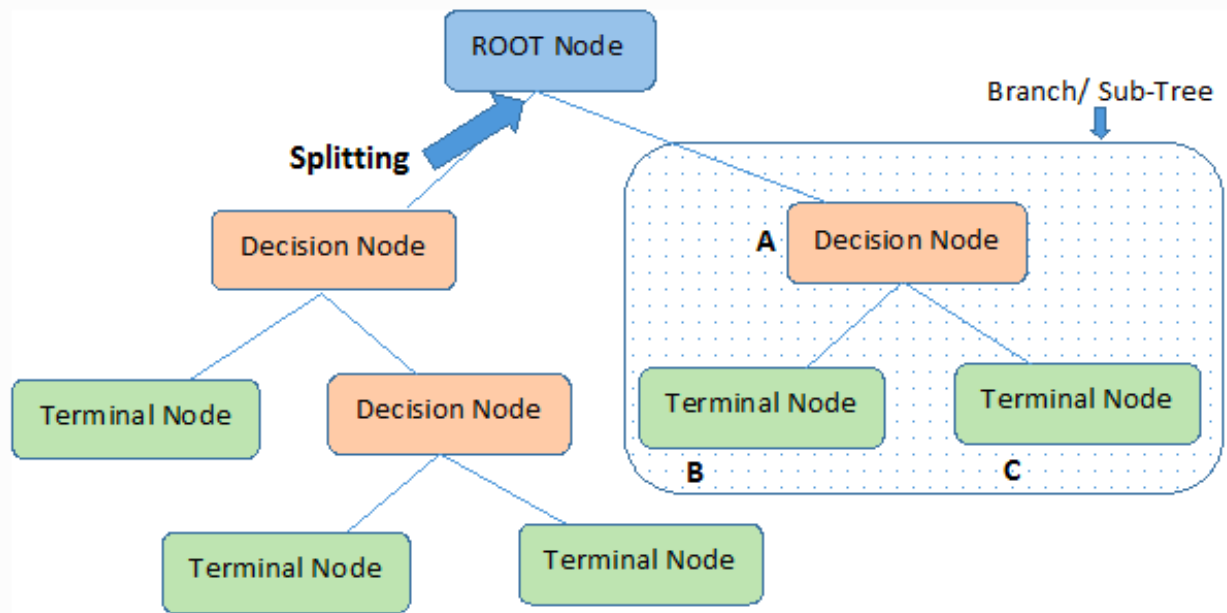
Types of decision trees are based on the type of target variable we have. It can be of two types:

1. **Categorical Variable Decision Tree:** Decision Tree which has a categorical target variable then it called a **Categorical variable decision tree**.
2. **Continuous Variable Decision Tree:** Decision Tree has a continuous target variable then it is called **Continuous Variable Decision Tree**.

**Example:-** Let's say we have a problem to predict whether a customer will pay his renewal premium with an insurance company (yes/ no). Here we know that the income of customers is a significant variable but the insurance company does not have income details for all customers. Now, as we know this is an important variable, then we can build a decision tree to predict customer income based on occupation, product, and various other variables. In this case, we are predicting values for the continuous variables.

## Terminology related to Decision Trees

1. **Root Node:** It represents the entire population or sample and this further gets divided into two or more homogeneous sets.
2. **Splitting:** It is a process of dividing a node into two or more sub-nodes.
3. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called the decision node.
4. **Leaf / Terminal Node:** Nodes do not split is called Leaf or Terminal node.
5. **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
6. **Branch / Sub-Tree:** A subsection of the entire tree is called branch or sub-tree.
7. **Parent and Child Node:** A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.



**Note:-** A is parent node of B and C.

Decision trees classify the examples by sorting them down the tree from the root to some leaf/terminal node, with the leaf/terminal node providing the classification of the example.

Each node in the tree acts as a test case for some attribute, and each edge descending from the node corresponds to the possible answers to the test case. This process is recursive in nature and is repeated for every subtree rooted at the new node.

### Assumptions while creating Decision Tree

Below are some of the assumptions we make while using Decision tree:

- In the **beginning**, the whole training set is considered as the **root**.
- Feature values are preferred to be **categorical**. If the **values are continuous** then they are discretized prior to building the model.
- Records are **distributed recursively** on the basis of attribute values.
- Order to placing attributes as **root or internal node** of the tree is done by **using some statistical approach**.

Decision Trees follow **Sum of Product (SOP)** representation. The Sum of product (SOP) is also known as **Disjunctive Normal Form**. For a class, every branch from the root of the tree to a leaf node having the same class is conjunction (product) of values, different branches ending in that class form a disjunction (sum).

The primary challenge in the decision tree implementation is to identify which attributes do we need to consider as the root node and each level. Handling this is to know as the attributes selection. We have different attributes selection measures to identify the attribute which can be considered as the root note at each level.

## How do Decision Trees work?

The decision of making strategic splits heavily affects a tree's accuracy. The decision criteria are different for classification and regression trees.

Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that the purity of the node increases with respect to the target variable. The decision tree splits the nodes on all available variables and then selects the split which results in most homogeneous sub-nodes.

The algorithm selection is also based on the type of target variables. Let us look at some algorithms used in Decision Trees:

<b>ID3</b> →	(extension	of	D3)
<b>C4.5</b> →	(successor	of	ID3)
<b>CART</b> →	(Classification	And	Regression Tree)
<b>CHAID</b> →	(Chi-square automatic interaction detection	Performs multi-level splits when	
	computing	classification	trees)
<b>MARS</b> →	(multivariate adaptive regression splines)		

The ID3 algorithm builds decision trees using a top-down greedy search approach through the space of possible branches with no backtracking. A greedy algorithm, as the name suggests, always makes the choice that seems to be the best at that moment.

### Steps in ID3 algorithm:

1. It begins with the original set  $S$  as the root node.
2. On each iteration of the algorithm, it iterates through the very unused attribute of the set  $S$  and calculates Entropy( $H$ ) and Information gain( $IG$ ) of this attribute.
3. It then selects the attribute which has the smallest Entropy or Largest Information gain.
4. The set  $S$  is then split by the selected attribute to produce a subset of the data.
5. The algorithm continues to recur on each subset, considering only attributes never selected before.

## Attribute Selection Measures

If the dataset consists of  $N$  attributes then deciding which attribute to place at the root or at different levels of the tree as internal nodes is a complicated step. By just randomly selecting any node to be the root can't solve the issue. If we follow a random approach, it may give us bad results with low accuracy.

For solving this attribute selection problem, researchers worked and devised some solutions. They suggested using some criteria like :

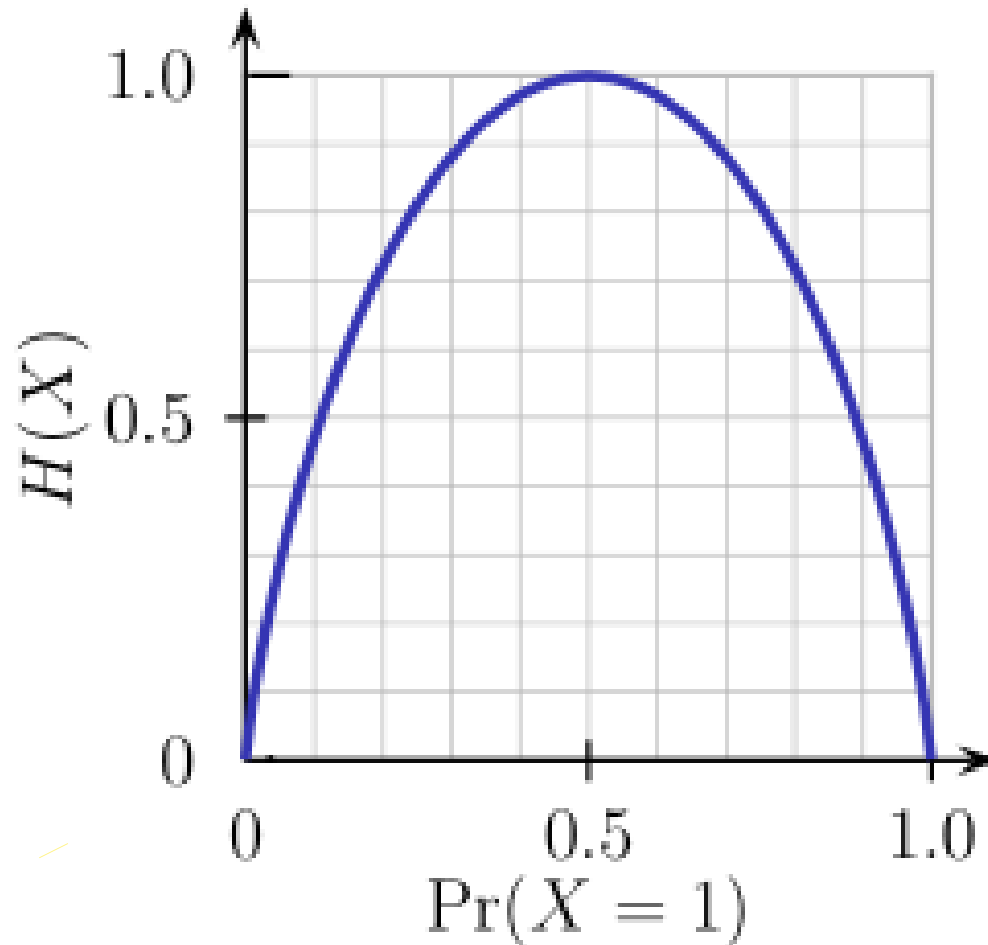
<b>Entropy,</b>	
<b>Information</b>	<b>gain,</b>
<b>Gini</b>	<b>index,</b>
<b>Gain</b>	<b>Ratio,</b>
<b>Reduction</b>	<b>in</b>
<b>Chi-Square</b>	<b>Variance</b>

These criteria will calculate values for every attribute. The values are sorted, and attributes are placed in the tree by following the order i.e, the attribute with a high value(in case of information

gain) is placed at the root. While using Information Gain as a criterion, we assume attributes to be categorical, and for the Gini index, attributes are assumed to be continuous.

## Entropy

Entropy is a measure of the randomness in the information being processed. The higher the entropy, the harder it is to draw any conclusions from that information. Flipping a coin is an example of an action that provides information that is random.



From the above graph, it is quite evident that the entropy  $H(X)$  is zero when the probability is either 0 or 1. The Entropy is maximum when the probability is 0.5 because it projects perfect randomness in the data and there is no chance if perfectly determining the outcome.

**ID3 follows the rule — A branch with an entropy of zero is a leaf node and A branch with entropy more than zero needs further splitting.**

Mathematically Entropy for 1 attribute is represented as:

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

Play Golf	
Yes	No
9	5



$$\begin{aligned}
 \text{Entropy(PlayGolf)} &= \text{Entropy}(5,9) \\
 &= \text{Entropy}(0.36, 0.64) \\
 &= -(0.36 \log_2 0.36) - (0.64 \log_2 0.64) \\
 &= 0.94
 \end{aligned}$$

Where  $S \rightarrow$  Current state, and  $P_i \rightarrow$  Probability of an event  $i$  of state  $S$  or Percentage of class  $i$  in a node of state  $S$ .

Mathematically Entropy for multiple attributes is represented as:

$$E(T, X) = \sum_{c \in X} P(c)E(c)$$

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



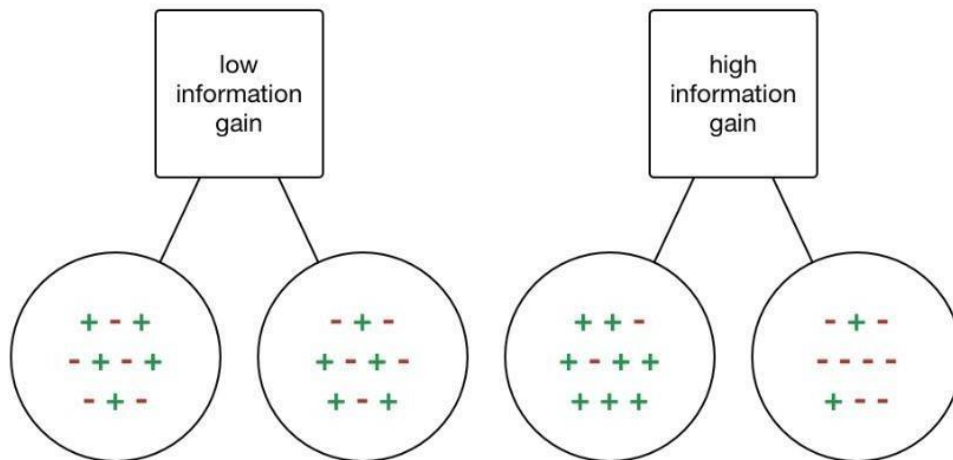
$$\begin{aligned}
 E(\text{PlayGolf}, \text{Outlook}) &= P(\text{Sunny}) * E(3,2) + P(\text{Overcast}) * E(4,0) + P(\text{Rainy}) * E(2,3) \\
 &= (5/14) * 0.971 + (4/14) * 0.0 + (5/14) * 0.971 \\
 &= 0.693
 \end{aligned}$$



where  $T \rightarrow$  Current state and  $X \rightarrow$  Selected attribute

### Information Gain

Information gain or **IG** is a statistical property that measures how well a given attribute separates the training examples according to their target classification. Constructing a decision tree is all about finding an attribute that returns the highest information gain and the smallest entropy.



### Information Gain

Information gain is a decrease in entropy. It computes the difference between entropy before split and average entropy after split of the dataset based on given attribute values. ID3 (Iterative Dichotomiser) decision tree algorithm uses information gain.

Mathematically, IG is represented as:

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

$$\begin{aligned} \text{IG}(\text{PlayGolf}, \text{Outlook}) &= E(\text{PlayGolf}) - E(\text{PlayGolf}, \text{Outlook}) \\ &= 0.940 - 0.693 \\ &= 0.247 \end{aligned}$$

In a much simpler way, we can conclude that:

$$\text{Information Gain} = \text{Entropy}(\text{before}) - \sum_{j=1}^K \text{Entropy}(j, \text{after})$$

### Information Gain

Where —before‖ is the dataset before the split, K is the number of subsets generated by the split, and (j, after) is subset j after the split.

### Gini Index

You can understand the Gini index as a cost function used to evaluate splits in the dataset. It is calculated by subtracting the sum of the squared probabilities of each class from one. It favors larger partitions and easy to implement whereas information gain favors smaller partitions with distinct values.

$$\text{Gini} = 1 - \sum_{i=1}^C (p_i)^2$$

Gini Index

Gini Index works with the categorical target variable —Success‖ or —Failure‖. It performs only Binary splits.

Higher value of Gini index implies higher inequality, higher heterogeneity.

### Steps to Calculate Gini index for a split

1. Calculate Gini for sub-nodes, using the above formula for success(p) and failure(q) ( $p^2+q^2$ ).
  2. Calculate the Gini index for split using the weighted Gini score of each node of that split.
- CART (Classification and Regression Tree) uses the Gini index method to create split points.

### Gain ratio

Information gain is biased towards choosing attributes with a large number of values as root nodes. It means it prefers the attribute with a large number of distinct values.

C4.5, an improvement of ID3, uses Gain ratio which is a modification of Information gain that reduces its bias and is usually the best option. Gain ratio overcomes the problem with information gain by taking into account the number of branches that would result before making the split. It corrects information gain by taking the intrinsic information of a split into account.

Let us consider if we have a dataset that has users and their movie genre preferences based on variables like gender, group of age, rating, blah, blah. With the help of information gain, you split at \_Gender\_ (assuming it has the highest information gain) and now the variables \_Group of Age\_ and \_Rating\_ could be equally important and with the help of gain ratio, it will penalize a variable with more distinct values which will help us decide the split at the next level.

$$\text{Gain Ratio} = \frac{\text{Information Gain}}{\text{SplitInfo}} = \frac{\text{Entropy (before)} - \sum_{j=1}^K \text{Entropy}(j, \text{after})}{\sum_{j=1}^K w_j \log_2 w_j}$$

Gain Ratio

Where —before is the dataset before the split, K is the number of subsets generated by the split, and (j, after) is subset j after the split.

### Reduction in Variance

**Reduction in variance** is an algorithm used for continuous target variables (regression problems). This algorithm uses the standard formula of variance to choose the best split. The split with lower variance is selected as the criteria to split the population:

$$\text{Variance} = \frac{\sum (X - \bar{X})^2}{n}$$

Above X-bar is the mean of the values, X is actual and n is the number of values.

#### Steps to calculate Variance:

1. Calculate variance for each node.
2. Calculate variance for each split as the weighted average of each node variance.

### Chi-Square

The acronym CHAID stands for *Chi*-squared Automatic Interaction Detector. It is one of the oldest tree classification methods. It finds out the statistical significance between the differences between sub-nodes and parent node. We measure it by the sum of squares of standardized differences between observed and expected frequencies of the target variable.

It works with the categorical target variable —Success or —Failure. It can perform two or more splits. Higher the value of Chi-Square higher the statistical significance of differences between sub-node and Parent node.

It generates a tree called CHAID (Chi-square Automatic Interaction Detector).

Mathematically, Chi-squared is represented as:

$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

Where:

$\chi^2$  = Chi Square obtained

$\sum$  = the sum of

$O$  = observed score

$E$  = expected score

#### Steps to Calculate Chi-square for a split:

1. Calculate Chi-square for an individual node by calculating the deviation for Success and Failure both
2. Calculated Chi-square of Split using Sum of all Chi-square of success and Failure of each node of the split

#### How to avoid/counter Overfitting in Decision Trees?

The common problem with Decision trees, especially having a table full of columns, they fit a lot. Sometimes it looks like the tree memorized the training data set. If there is no limit set on a decision tree, it will give you 100% accuracy on the training data set because in the worse case it will end up making 1 leaf for each observation. Thus this affects the accuracy when predicting samples that are not part of the training set.

Here are two ways to remove overfitting:

1. Pruning Decision Trees.
2. Random Forest

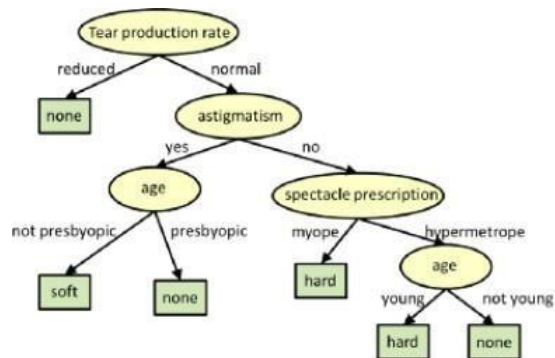
#### Pruning Decision Trees

The splitting process results in fully grown trees until the stopping criteria are reached. But, the fully grown tree is likely to overfit the data, leading to poor accuracy on unseen data.

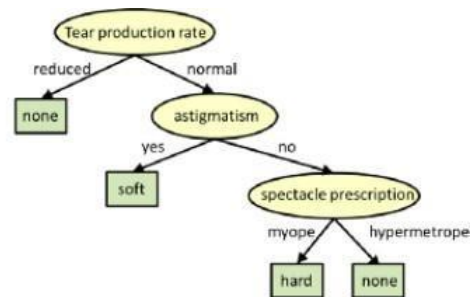


Pruning in action

In **pruning**, you trim off the branches of the tree, i.e., remove the decision nodes starting from the leaf node such that the overall accuracy is not disturbed. This is done by segregating the actual training set into two sets: training data set,  $D$  and validation data set,  $V$ . Prepare the decision tree using the segregated training data set,  $D$ . Then continue trimming the tree accordingly to optimize the accuracy of the validation data set,  $V$ .



Original Tree



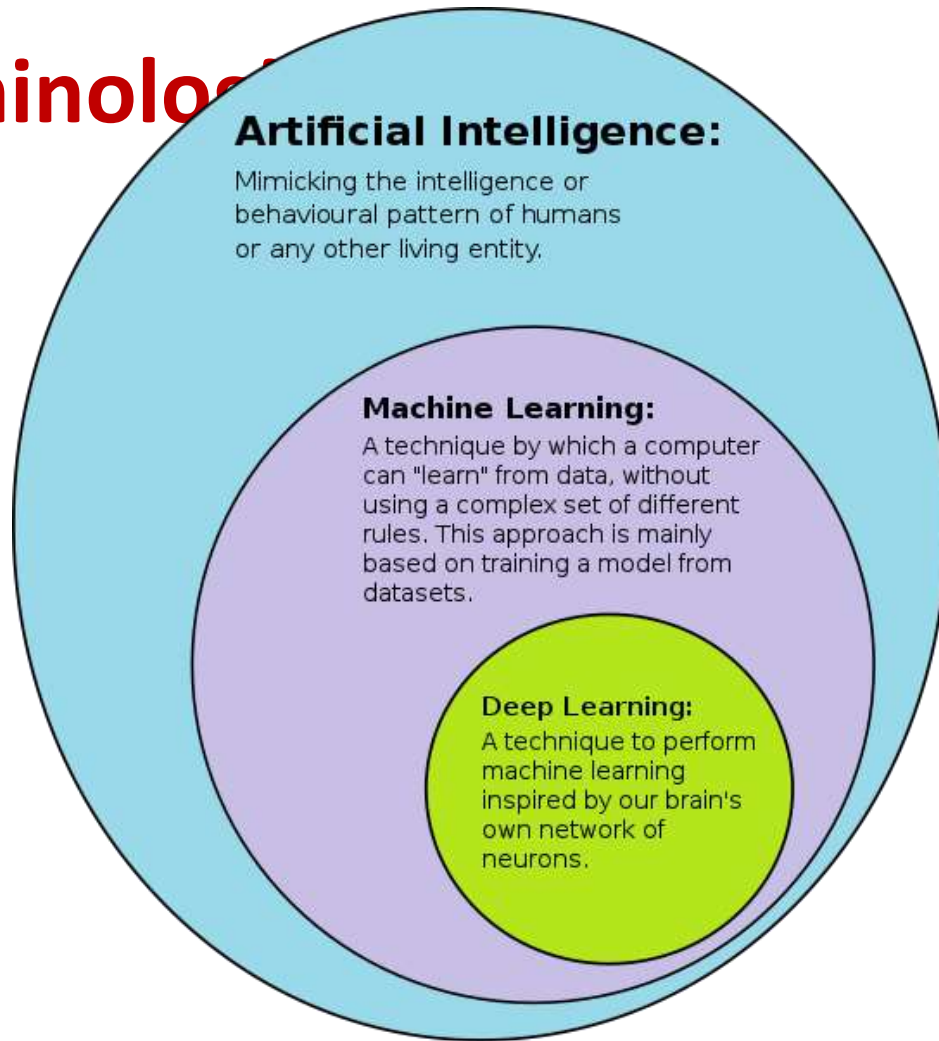
Pruned Tree



# Machine Learning

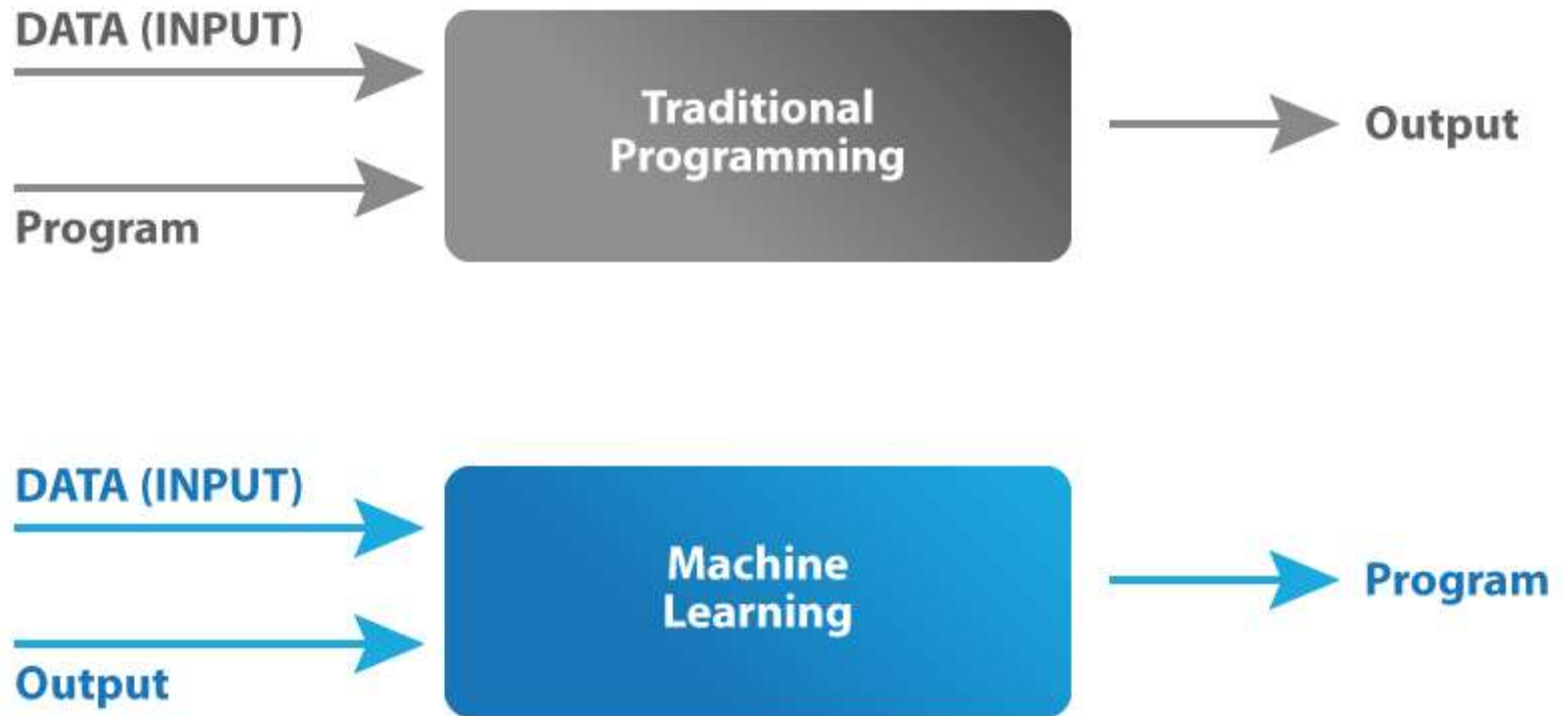
# Some Common Terminology

- **Artificial Intelligence**
  - ability to imitate human behaviors
- **Machine Learning**
  - ability to automatically learn and improve from experiences without being explicitly programmed
- **Deep Learning**





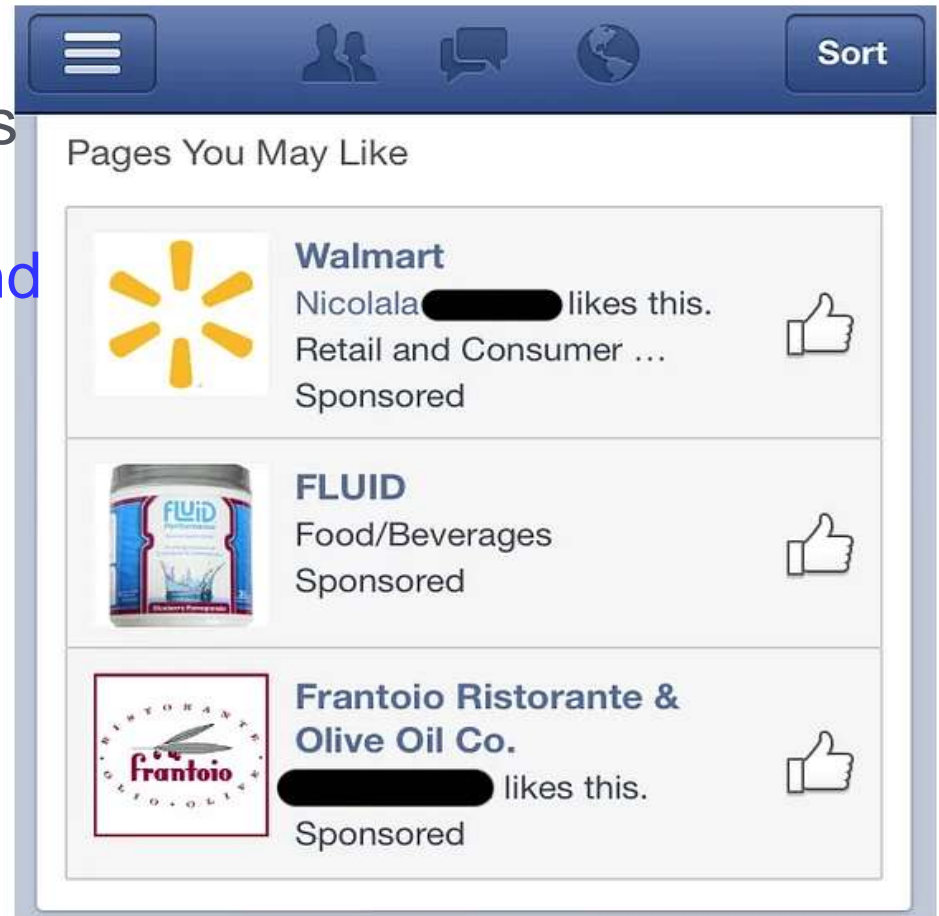
# What is Machine Learning??



# Applications of Machine Learning

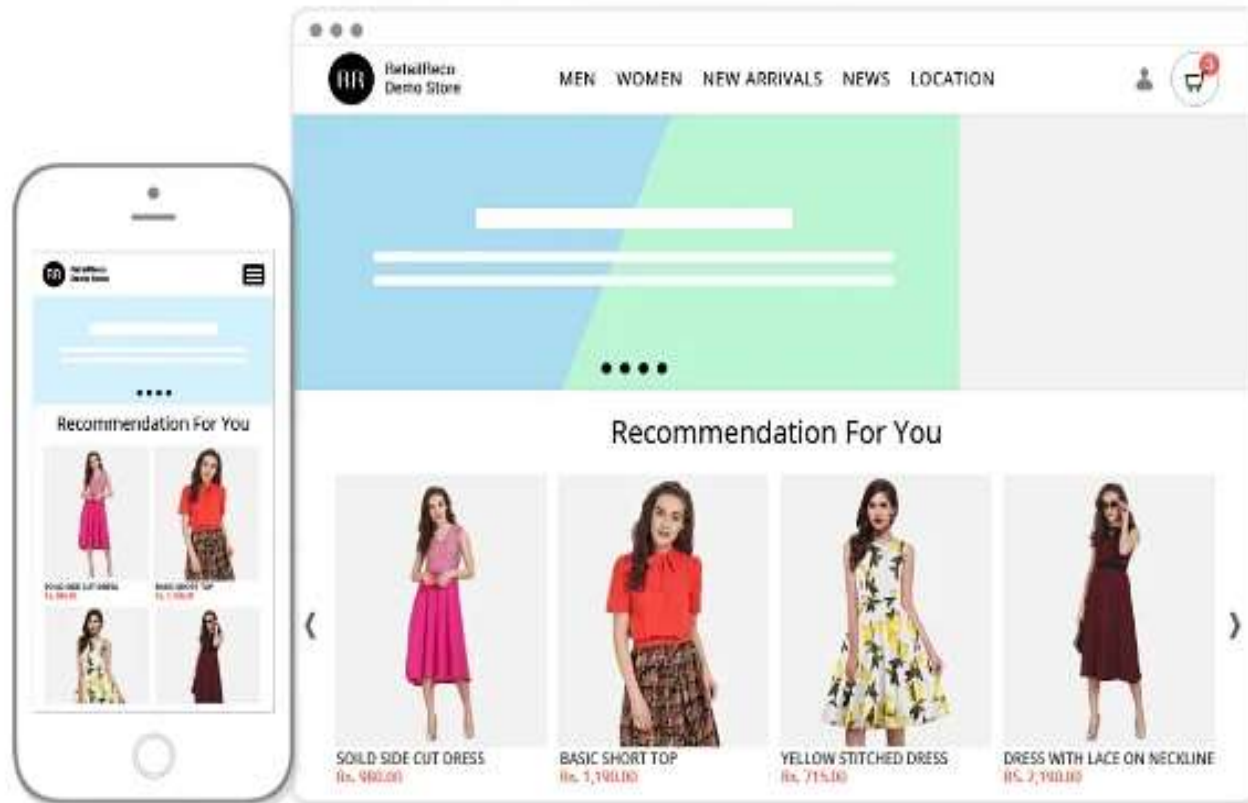
## 1. Social Media Features

For instance, Facebook notices and records your activities, chats, likes, and comments, and the time you spend on specific kinds of posts.



## 2. Product Recommendations

websites track your behavior based on your previous purchases, searching patterns, and cart history, and then make product recommendations.



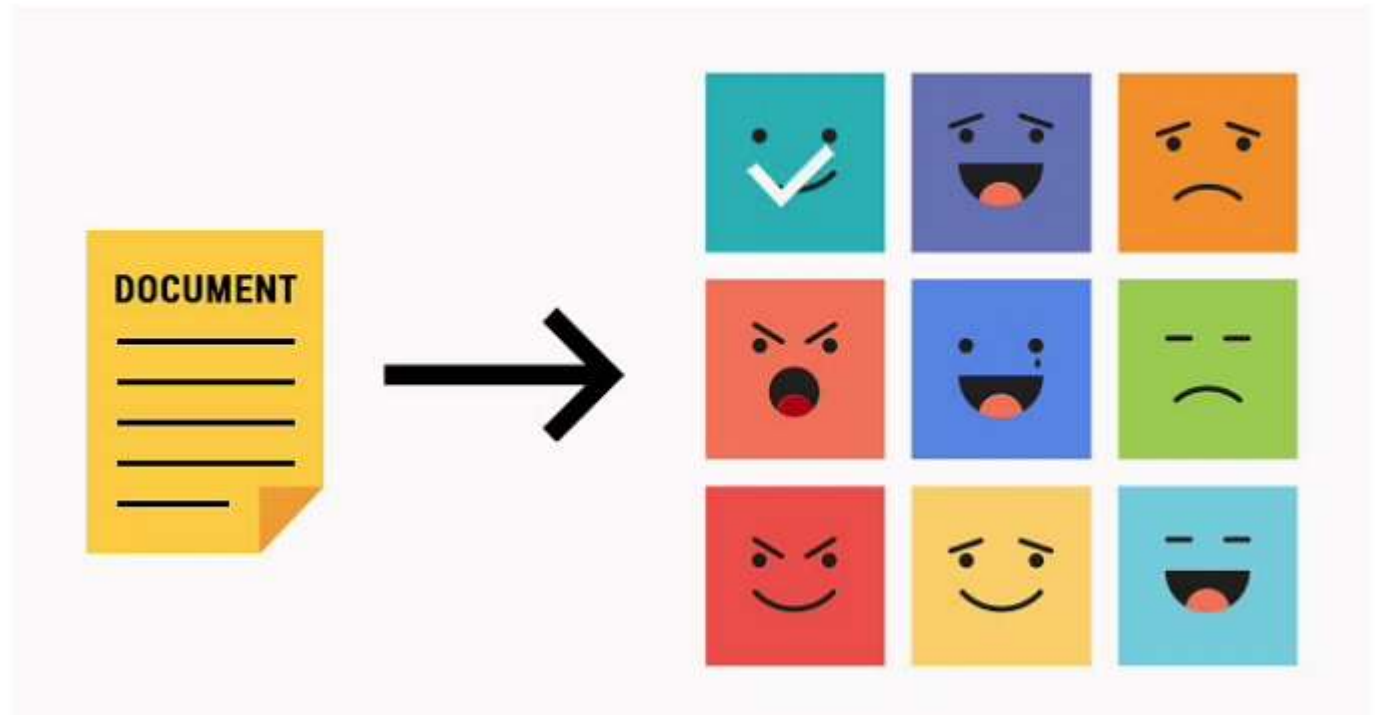
# 3. Image Recognition

cataloging and detecting a feature or an object in the digital image, is one of the most significant and notable machine learning and AI techniques.

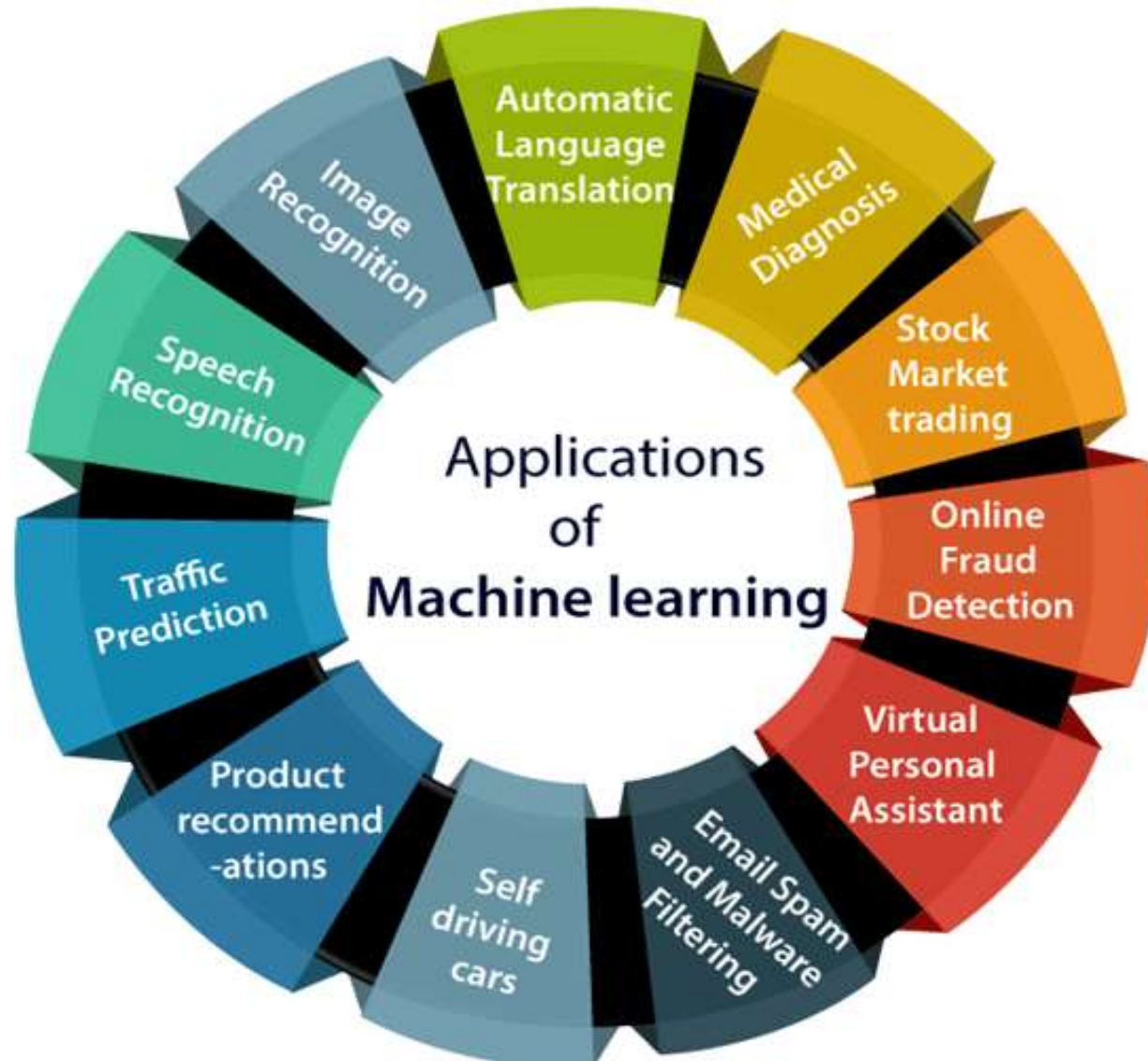


# 4. Sentiment Analysis

determines the emotion or opinion of the speaker or the writer



# Applications of Machine Learning

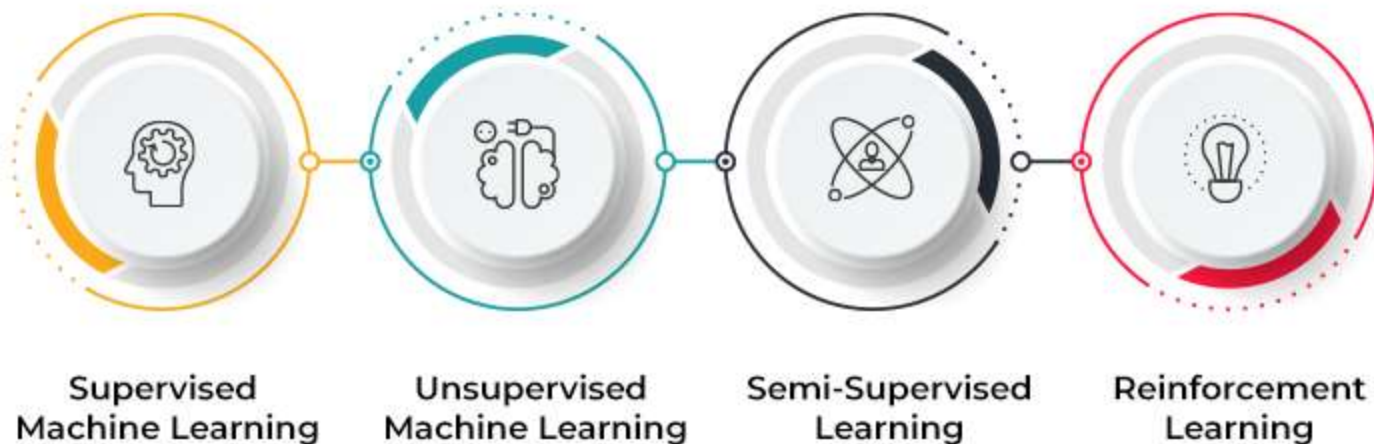




# Types of Machine Learning



## TYPES OF MACHINE LEARNING



# Ensemble Learning

When you want to purchase a new car, how will you decide which car is to be bought??

will you walk up to the first car shop and purchase one based on the advice of the dealer?

OR



**web portals** where people have posted their **reviews** and compare different car models,

checking for their **features and prices**.

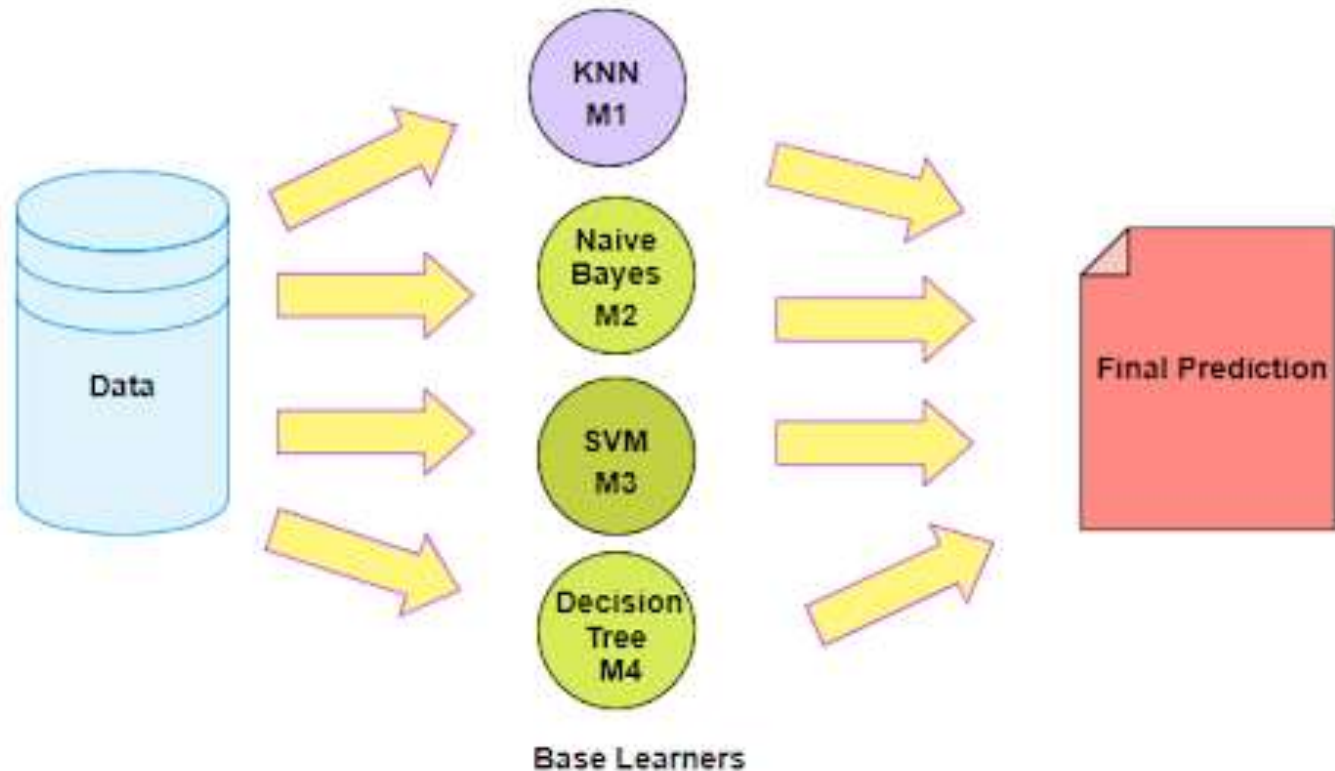
**make a decision considering the opinions of other people as well.**

ask your **friends and colleagues** for their opinion



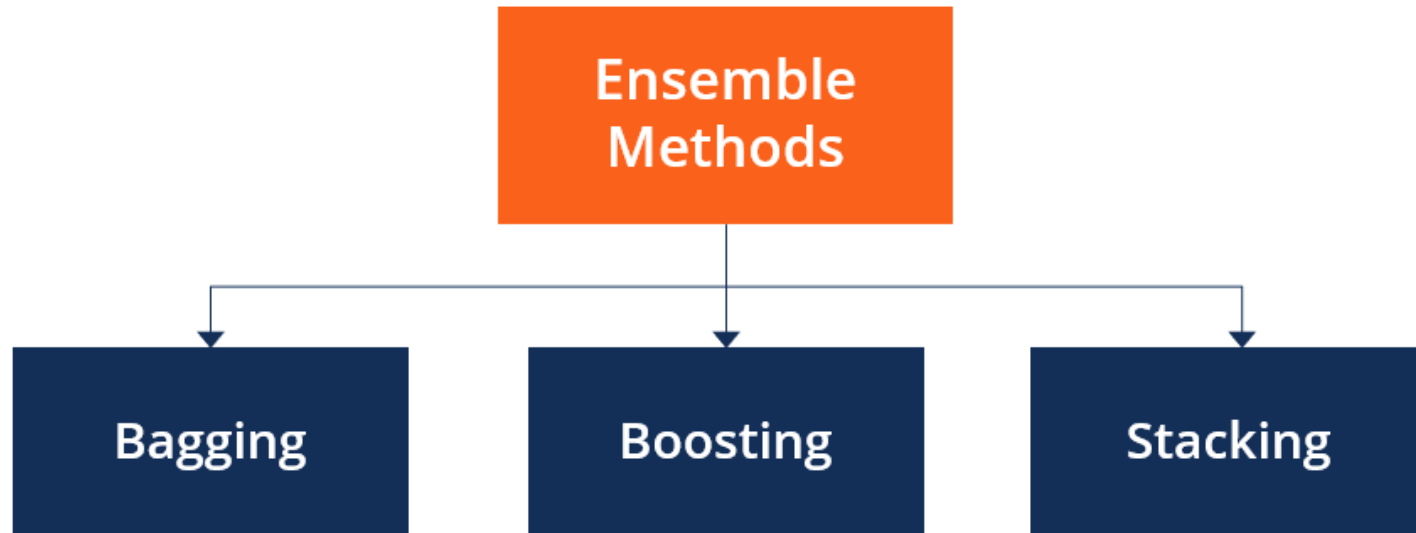
# Ensemble Learning

Ensemble learning **combine the decisions from multiple models** to improve the overall performance.



Ensemble learning is a widely-used and preferred machine learning technique in which multiple individual models, often called base models, are combined to produce an effective optimal prediction model.

# Ensemble Learning

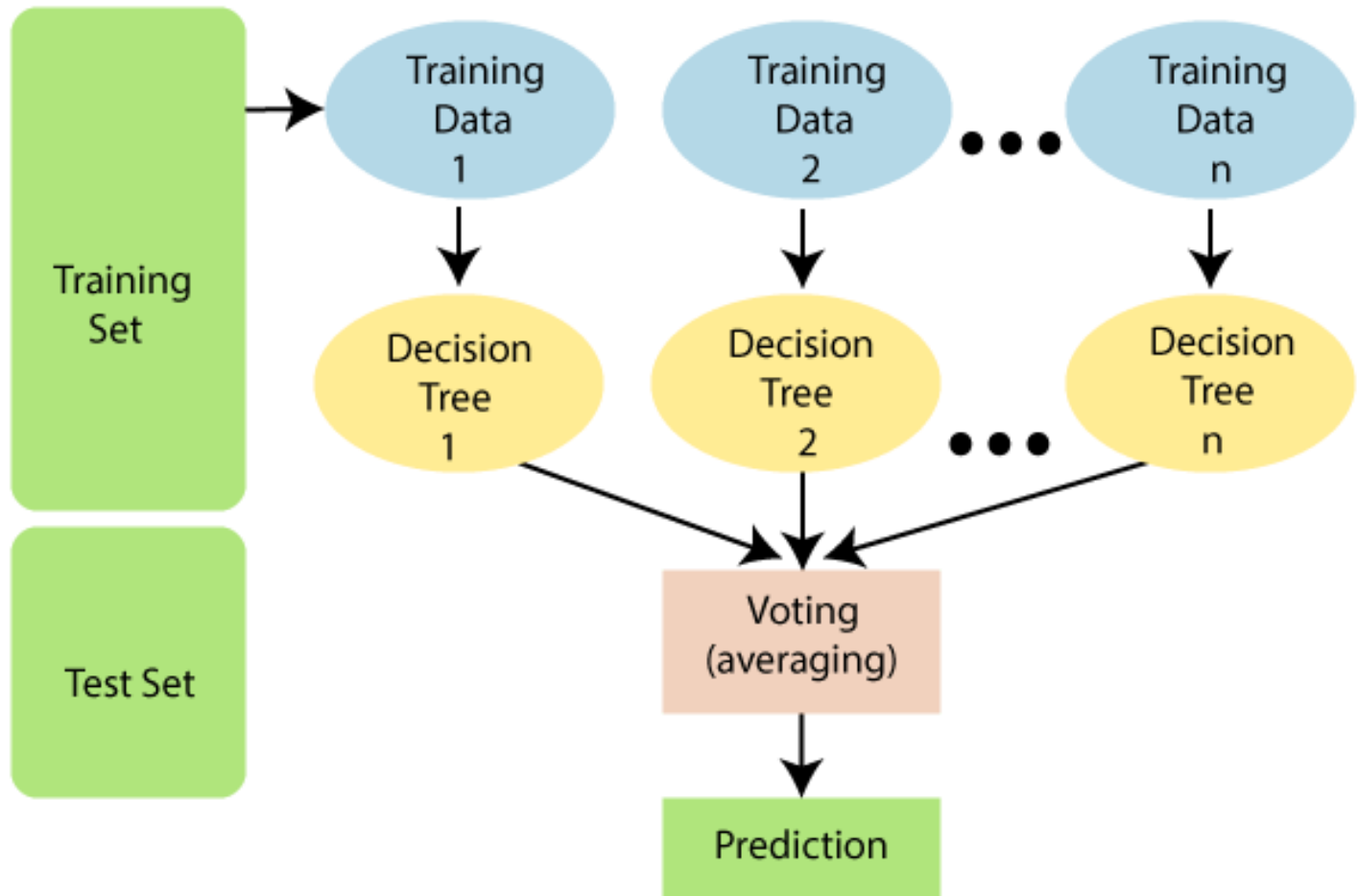


# Random Forest

- supervised learning technique.
- It can be used for both **Classification and Regression** problems in ML.
- It is based on the concept of **ensemble learning**, which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

***Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."***

# Random Forest



# Random Forest algorithm

Phase 1 :To create the random forest by combining N decision tree,

Phase 2: To make predictions for each tree created in the first phase.

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

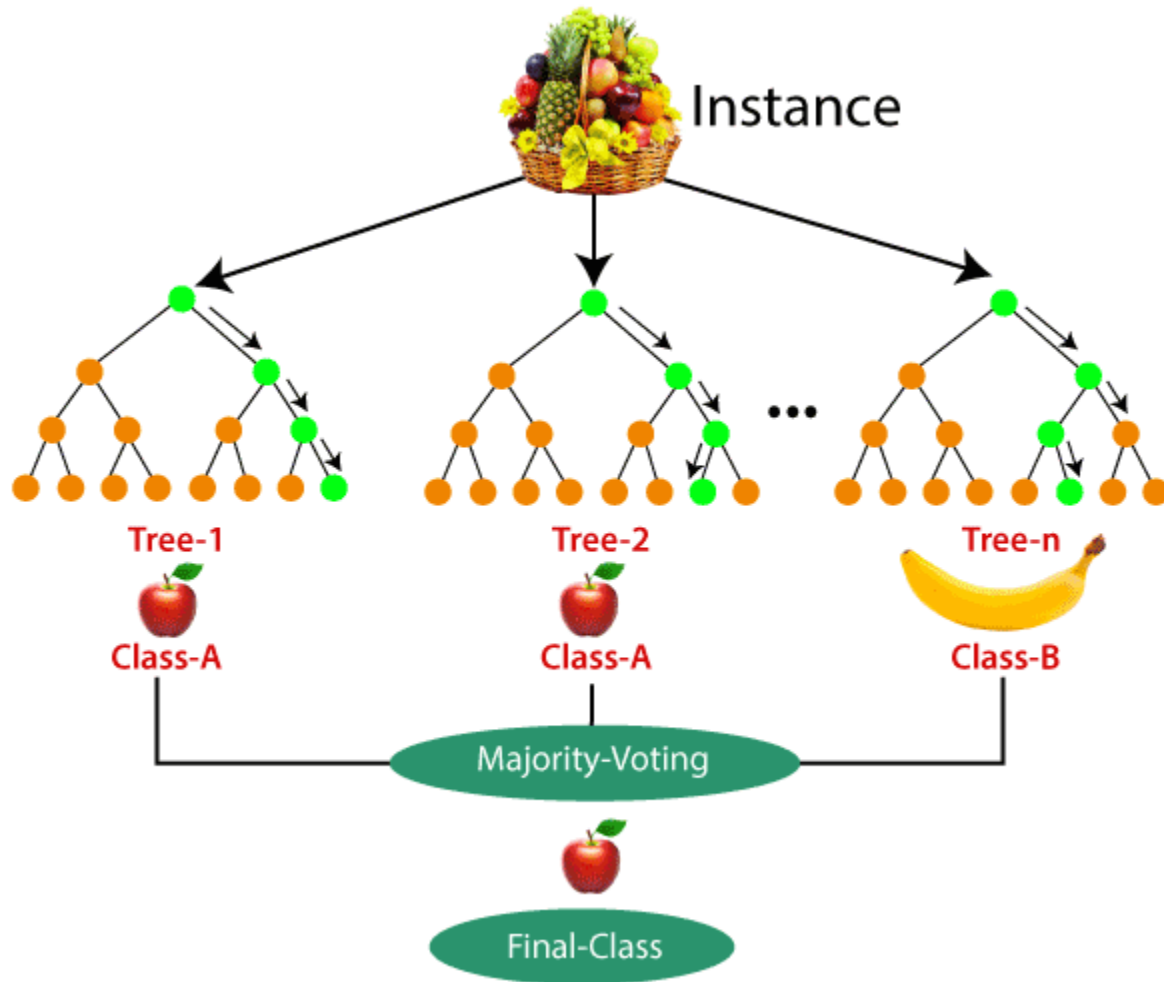
**Step-2:** Build the decision trees associated with the selected data points (Subsets).

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

**Step-4:** Repeat Step 1 & 2.

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

# Random Forest Example



# Random Forest

## Advantages of Random Forest

- Random Forest is capable of performing both **Classification and Regression** tasks.
- It is capable of handling **large datasets with high dimensionality**.
- It enhances the **accuracy** of the model and prevents the **overfitting issue**.

## Disadvantages of Random Forest

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

# Applications of Random Forest

- 1.Banking:** Banking sector mostly uses this algorithm for the identification of loan risk.
- 2.Medicine:** With the help of this algorithm, disease trends and risks of the disease can be identified.
- 3.Land Use:** We can identify the areas of similar land use by this algorithm.
- 4.Marketing:** Marketing trends can be identified using this algorithm.



## Decision Trees Example Problem

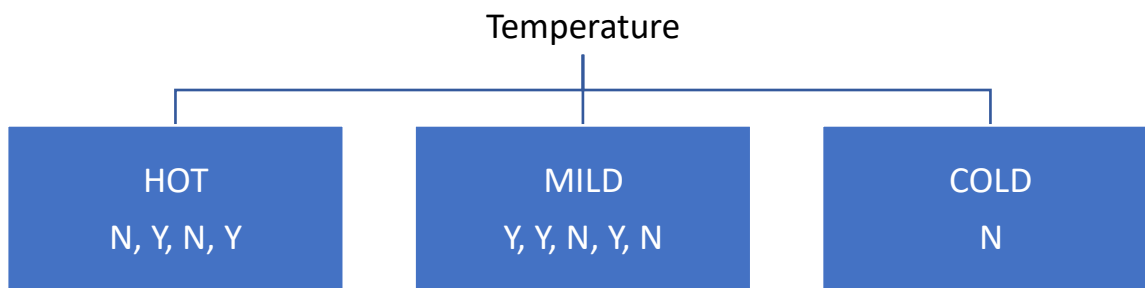
Consider the following data, where the Y label is whether or not the child goes out to play.

Day	Weather	Temperature	Humidity	Wind	Play?
1	Sunny	Hot	High	Weak	No
2	Cloudy	Hot	High	Weak	Yes
3	Sunny	Mild	Normal	Strong	Yes
4	Cloudy	Mild	High	Strong	Yes
5	Rainy	Mild	High	Strong	No
6	Rainy	Cool	Normal	Strong	No
7	Rainy	Mild	High	Weak	Yes
8	Sunny	Hot	High	Strong	No
9	Cloudy	Hot	Normal	Weak	Yes
10	Rainy	Mild	High	Strong	No

Step 1: Calculate the IG (information gain) for each attribute (feature)

$$\begin{aligned}
 \text{Initial entropy} = H(Y) &= -\sum_y P(Y = y) \log_2 P(Y = y) \\
 &= -P(Y = \text{yes}) \log_2 P(Y = \text{yes}) - P(Y = \text{no}) \log_2 P(Y = \text{no}) \\
 &= -(0.5) \log_2(0.5) - (0.5) \log_2(0.5) \\
 &= 1
 \end{aligned}$$

**Temperature:**

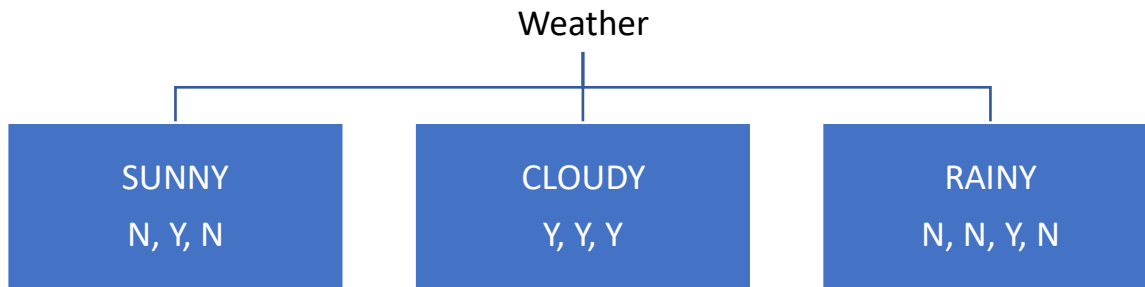


Total entropy of this division is:

$$\begin{aligned}
 H(Y | temp) &= -\sum_x P(temp = x) \sum_y P(Y = y | temp = x) \log_2 P(Y = y | temp = x) \\
 &= -(P(temp = H) \sum_y P(Y = y | temp = H) \log_2 P(Y = y | temp = H) + \\
 &\quad P(temp = M) \sum_y P(Y = y | temp = M) \log_2 P(Y = y | temp = M) + \\
 &\quad P(temp = C) \sum_y P(Y = y | temp = C) \log_2 P(Y = y | temp = C)) \\
 &= -((0.4)((\frac{1}{2}) \log_2 (\frac{1}{2}) + (\frac{1}{2}) \log_2 (\frac{1}{2})) + (0.5)((\frac{3}{5}) \log_2 (\frac{3}{5}) + (\frac{2}{5}) \log_2 (\frac{2}{5})) + \\
 &\quad (0.1)((1) \log_2 (1) + (0) \log_2 (0))) \\
 &= 0.7884
 \end{aligned}$$

$$IG(Y, temp) = 1 - 0.7884 = 0.2116$$

**Weather:**

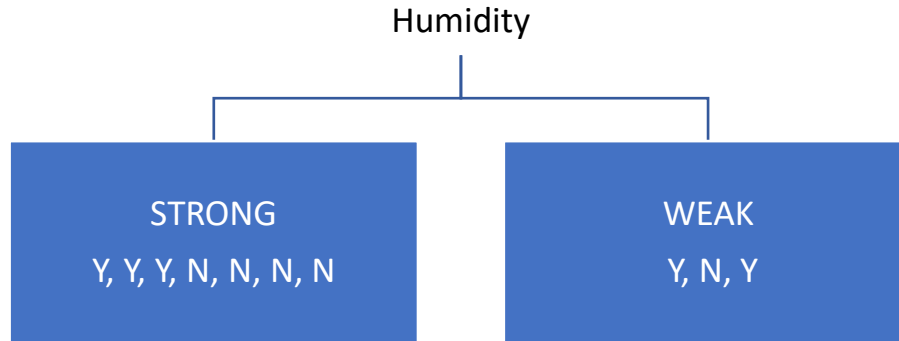


Total entropy of this division is:

$$\begin{aligned}
 H(Y \mid \text{weather}) &= - \sum_x P(\text{weather} = x) \sum_y P(Y = y \mid \text{weather} = x) \log_2 P(Y = y \mid \text{weather} = x) \\
 &= -(P(\text{weather} = S) \sum_y P(Y = y \mid \text{weather} = S) \log_2 P(Y = y \mid \text{weather} = S) + \\
 &\quad P(\text{weather} = C) \sum_y P(Y = y \mid \text{weather} = C) \log_2 P(Y = y \mid \text{weather} = C) + \\
 &\quad P(\text{weather} = R) \sum_y P(Y = y \mid \text{weather} = R) \log_2 P(Y = y \mid \text{weather} = R)) \\
 &= -((0.3)((\frac{1}{3}) \log_2 (\frac{1}{3}) + (\frac{2}{3}) \log_2 (\frac{2}{3})) + (0.3)((1) \log_2 (1) + (0) \log_2 (0)) + \\
 &\quad (0.4)((\frac{1}{4}) \log_2 (\frac{1}{4}) + (\frac{3}{4}) \log_2 (\frac{3}{4}))) \\
 &= 0.6
 \end{aligned}$$

$$IG(Y, \text{weather}) = 1 - 0.6 = 0.4$$

Humidity:

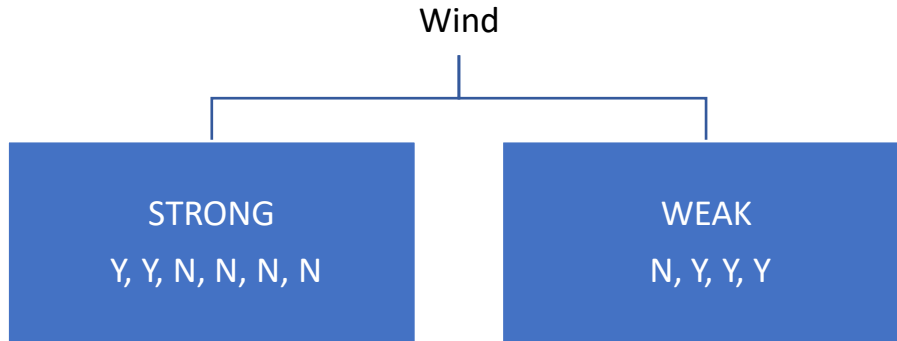


Total entropy of this division is:

$$\begin{aligned}
 H(Y | \text{hum}) &= - \sum_x P(\text{hum} = x) \sum_y P(Y = y | \text{hum} = x) \log_2 P(Y = y | \text{hum} = x) \\
 &= -(P(\text{hum} = H) \sum_y P(Y = y | \text{hum} = H) \log_2 P(Y = y | \text{hum} = H) + \\
 &\quad P(\text{hum} = N) \sum_y P(Y = y | \text{hum} = N) \log_2 P(Y = y | \text{hum} = N)) \\
 &= -((0.7)((\frac{3}{7}) \log_2 (\frac{3}{7}) + (\frac{4}{7}) \log_2 (\frac{4}{7})) + (0.3)((\frac{2}{3}) \log_2 (\frac{2}{3}) + (\frac{1}{3}) \log_2 (\frac{1}{3}))) \\
 &= 0.8651
 \end{aligned}$$

$$IG(Y, \text{hum}) = 1 - 0.8651 = 0.1349$$

Wind:



Total entropy of this division is:

$$\begin{aligned} H(Y | \text{wind}) &= - \sum_x P(\text{wind} = x) \sum_y P(Y = y | \text{wind} = x) \log_2 P(Y = y | \text{wind} = x) \\ &= -(P(\text{wind} = S) \sum_y P(Y = y | \text{wind} = S) \log_2 P(Y = y | \text{wind} = S) + \\ &\quad P(\text{wind} = W) \sum_y P(Y = y | \text{wind} = W) \log_2 P(Y = y | \text{wind} = W)) \\ &= -((0.6)((\frac{2}{6}) \log_2 (\frac{2}{6}) + (\frac{4}{6}) \log_2 (\frac{4}{6})) + (0.4)((\frac{1}{4}) \log_2 (\frac{1}{4}) + (\frac{3}{4}) \log_2 (\frac{3}{4}))) \\ &= 0.8755 \end{aligned}$$

$$IG(Y, \text{wind}) = 1 - 0.8755 = 0.1245$$

Step 2: Choose which feature to split with!

$$IG(Y, \text{wind}) = 0.1245$$

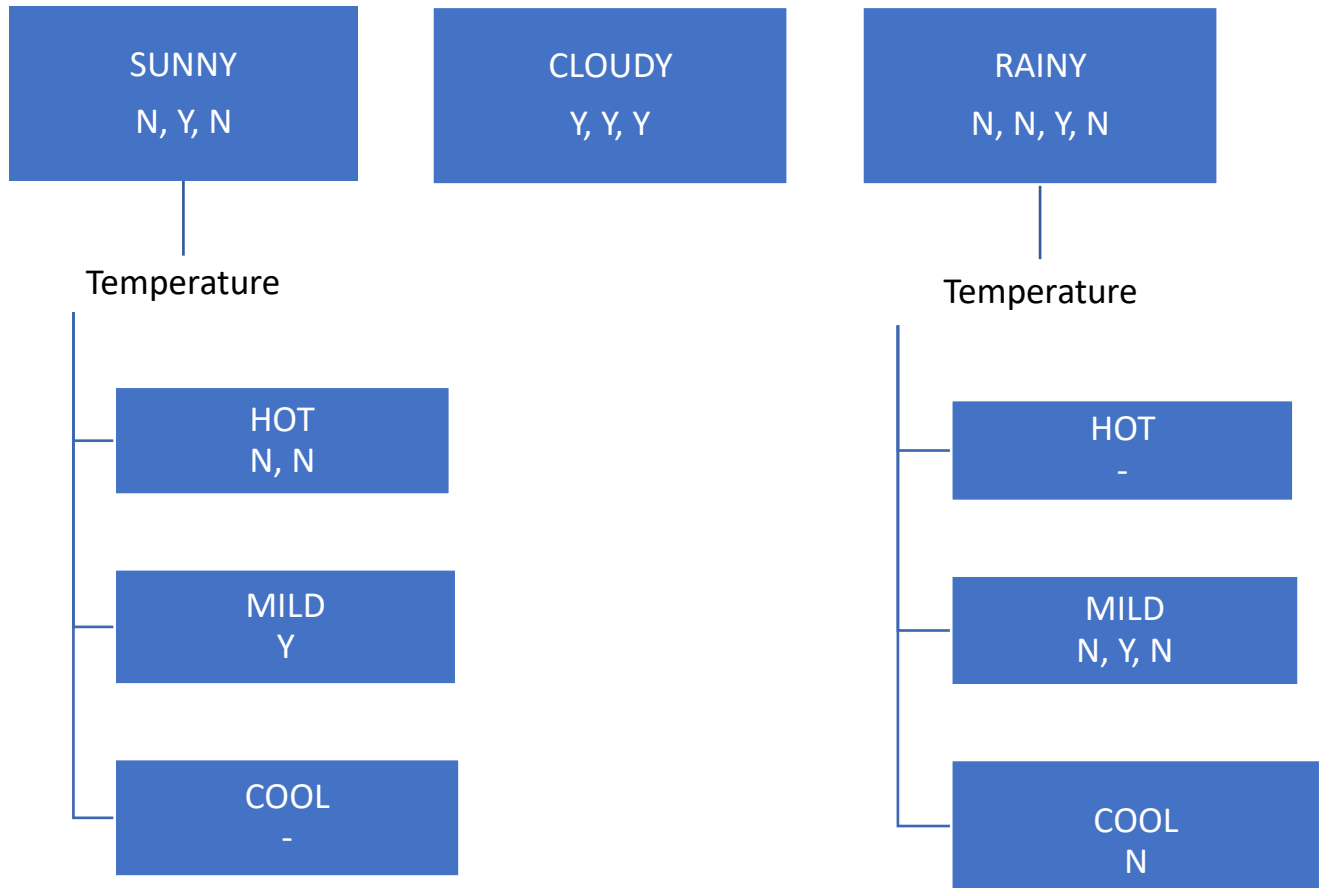
$$IG(Y, \text{hum}) = 0.1349$$

$$IG(Y, \text{weather}) = 0.4$$

$$IG(Y, \text{temp}) = 0.2116$$

Step 3: Repeat for each level (sad, I know)

### Temperature



Entropy of "Sunny" node =  $-\left(\left(\frac{1}{3}\right) \log_2 \left(\frac{1}{3}\right) + \left(\frac{2}{3}\right) \log_2 \left(\frac{2}{3}\right)\right) = 0.9183$

Entropy of its children = 0

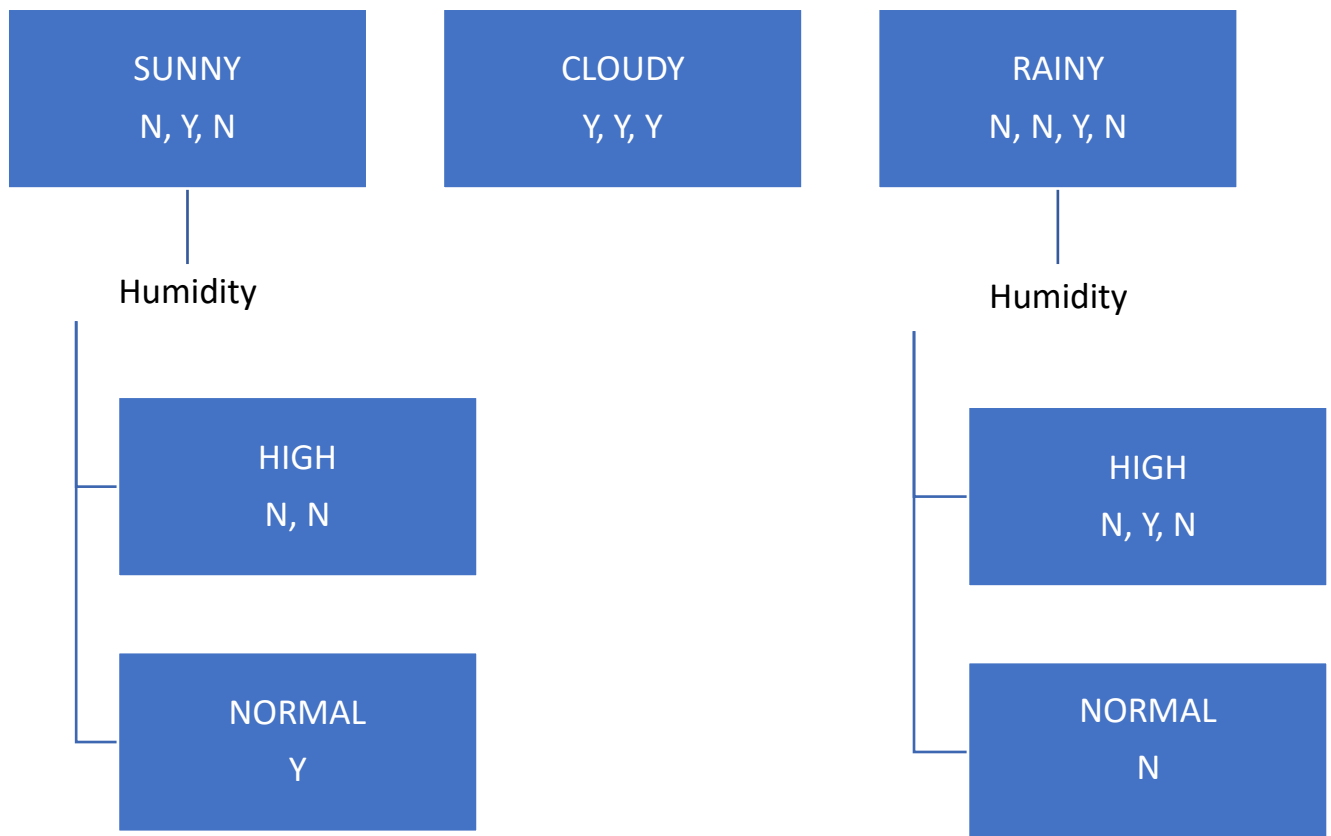
IG = 0.9183

Entropy of "Rainy" node =  $-\left(\left(\frac{1}{4}\right) \log_2 \left(\frac{1}{4}\right) + \left(\frac{3}{4}\right) \log_2 \left(\frac{3}{4}\right)\right) = 0.8113$

Entropy of children =  $-\left(\left(\frac{3}{4}\right) \left(\left(\frac{1}{3}\right) \log_2 \left(\frac{1}{3}\right) + \left(\frac{2}{3}\right) \log_2 \left(\frac{2}{3}\right)\right) + 0\right) = 0.6887$

IG = 0.1226

## Humidity



Entropy of "Sunny" node =  $-\left(\left(\frac{1}{3}\right) \log_2 \left(\frac{1}{3}\right) + \left(\frac{2}{3}\right) \log_2 \left(\frac{2}{3}\right)\right) = 0.9183$

Entropy of its children = 0

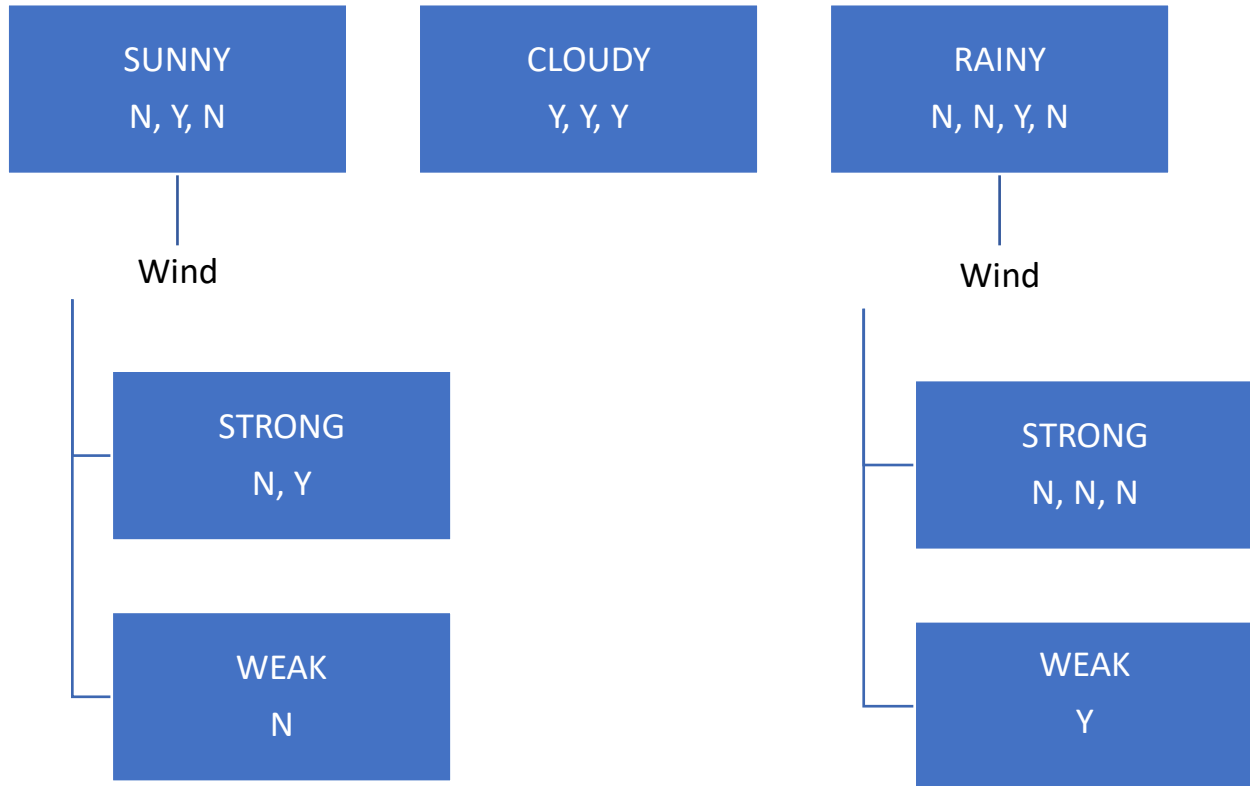
IG = 0.9183

Entropy of "Rainy" node =  $-\left(\left(\frac{1}{4}\right) \log_2 \left(\frac{1}{4}\right) + \left(\frac{3}{4}\right) \log_2 \left(\frac{3}{4}\right)\right) = 0.8113$

Entropy of children =  $-\left(\left(\frac{3}{4}\right) \left(\left(\frac{1}{3}\right) \log_2 \left(\frac{1}{3}\right) + \left(\frac{2}{3}\right) \log_2 \left(\frac{2}{3}\right)\right) + 0\right) = 0.6887$

IG = 0.1226

## Wind



Entropy of "Sunny" node =  $-\left(\left(\frac{1}{3}\right) \log_2 \left(\frac{1}{3}\right) + \left(\frac{2}{3}\right) \log_2 \left(\frac{2}{3}\right)\right) = 0.9183$

Entropy of its children =  $-\left(\frac{2}{3}\right)\left(\left(\frac{1}{2}\right) \log_2 \left(\frac{1}{2}\right) + \left(\frac{1}{2}\right) \log_2 \left(\frac{1}{2}\right)\right) + 0 = 0.6667$

IG = 0.2516

Entropy of "Rainy" node =  $-\left(\left(\frac{1}{4}\right) \log_2 \left(\frac{1}{4}\right) + \left(\frac{3}{4}\right) \log_2 \left(\frac{3}{4}\right)\right) = 0.8113$

Entropy of children = 0

IG = 0.8113



Step 4: Choose feature for each node to split on!

**“Sunny node”:**

$$IG(Y, \text{weather}) = IG(\text{humidity}) = 0.9183$$

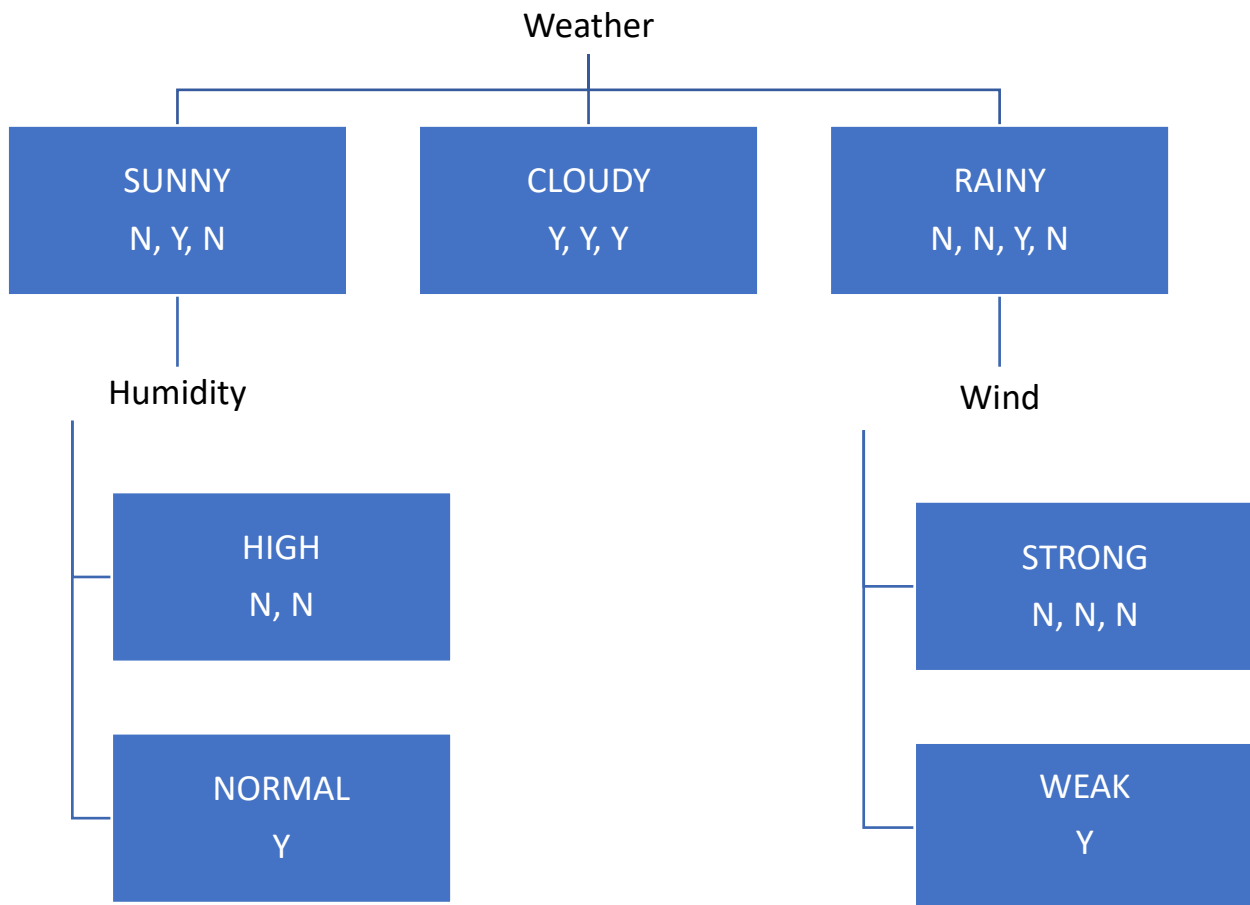
$$IG(Y, \text{wind}) = 0.2516$$

**“Rainy node”:**

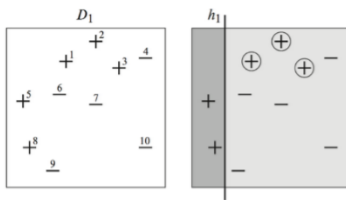
$$IG(Y, \text{weather}) = IG(Y, \text{humidity}) = 0.1226$$

$$IG(Y, \text{wind}) = 0.8113$$

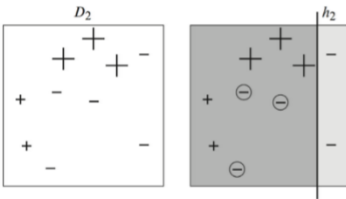
Final Tree!



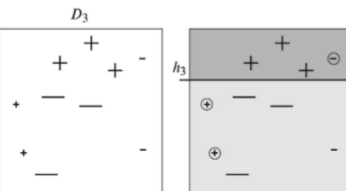
# Boosting



	1	2	3	4	5	6	7	8	9	10	
$D_1(i)$	<u>0.10</u>	<u>0.10</u>	<u>0.10</u>	0.10	0.10	0.10	0.10	0.10	0.10	0.10	$\epsilon_1 = 0.30, \alpha_1 \approx 0.42$
$e^{-\alpha_1 y_i h_1(x_i)}$	1.53	1.53	1.53	0.65	0.65	0.65	0.65	0.65	0.65	0.65	
$D_1(i) e^{-\alpha_1 y_i h_1(x_i)}$	0.15	0.15	0.15	0.07	0.07	0.07	0.07	0.07	0.07	0.07	$Z_1 \approx 0.92$



$D_2(i)$	0.17	0.17	0.17	0.07	0.07	<u>0.07</u>	<u>0.07</u>	0.07	<u>0.07</u>	0.07	$\epsilon_2 \approx 0.21, \alpha_2 \approx 0.65$
$e^{-\alpha_2 y_i h_2(x_i)}$	0.52	0.52	0.52	0.52	0.52	1.91	1.91	0.52	1.91	0.52	
$D_2(i) e^{-\alpha_2 y_i h_2(x_i)}$	0.09	0.09	0.09	0.04	0.04	0.14	0.14	0.04	0.14	0.04	$Z_2 \approx 0.82$



$D_3(i)$	0.11	0.11	0.11	<u>0.05</u>	<u>0.05</u>	0.17	0.17	<u>0.05</u>	0.17	0.05	$\epsilon_3 \approx 0.14, \alpha_3 \approx 0.92$
$e^{-\alpha_3 y_i h_3(x_i)}$	0.40	0.40	0.40	2.52	2.52	0.40	0.40	2.52	0.40	0.40	
$D_3(i) e^{-\alpha_3 y_i h_3(x_i)}$	0.04	0.04	0.04	0.11	0.11	0.07	0.07	0.11	0.07	0.02	$Z_3 \approx 0.69$

$$H = \text{sign} \left( 0.42 \begin{array}{|c|} \hline \text{white} \\ \hline \end{array} + 0.65 \begin{array}{|c|} \hline \text{gray} \\ \hline \end{array} + 0.92 \begin{array}{|c|} \hline \text{white} \\ \hline \end{array} \right) = \begin{array}{|c|} \hline \text{gray} \\ \hline \end{array}$$

([https://www.ccs.neu.edu/home/vip/teach/MLcourse/4\\_boosting/slides/boosting.pdf](https://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/boosting.pdf))