Decision Tree Technique

Decision Tree

- **Decision Tree**
- "A decision tree in machine learning is a flowchart structure in which each node represents a "test" on the attribute and each branch represents the outcome of the test."
- The end node (called leaf node) represents a class label.

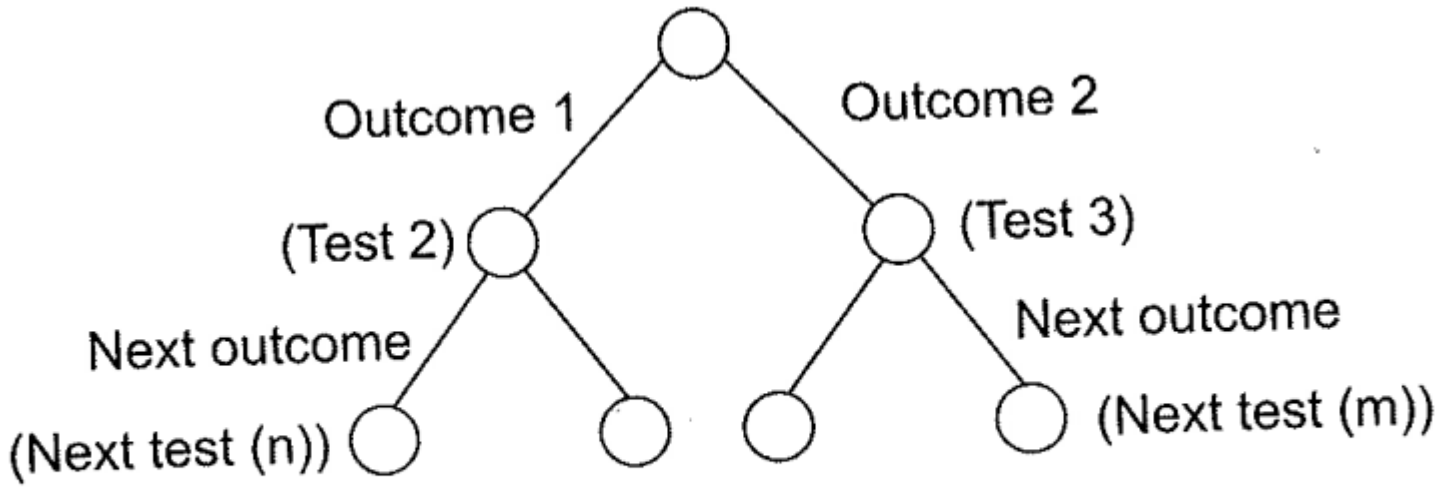
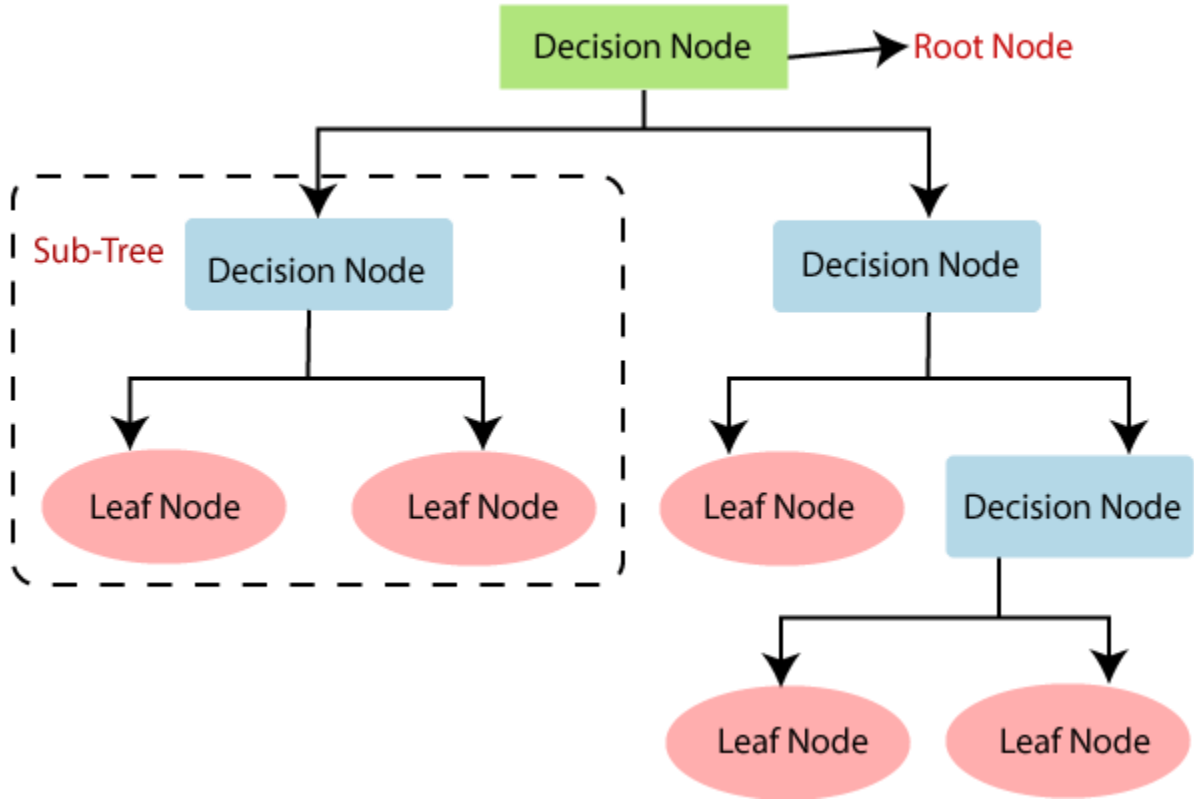


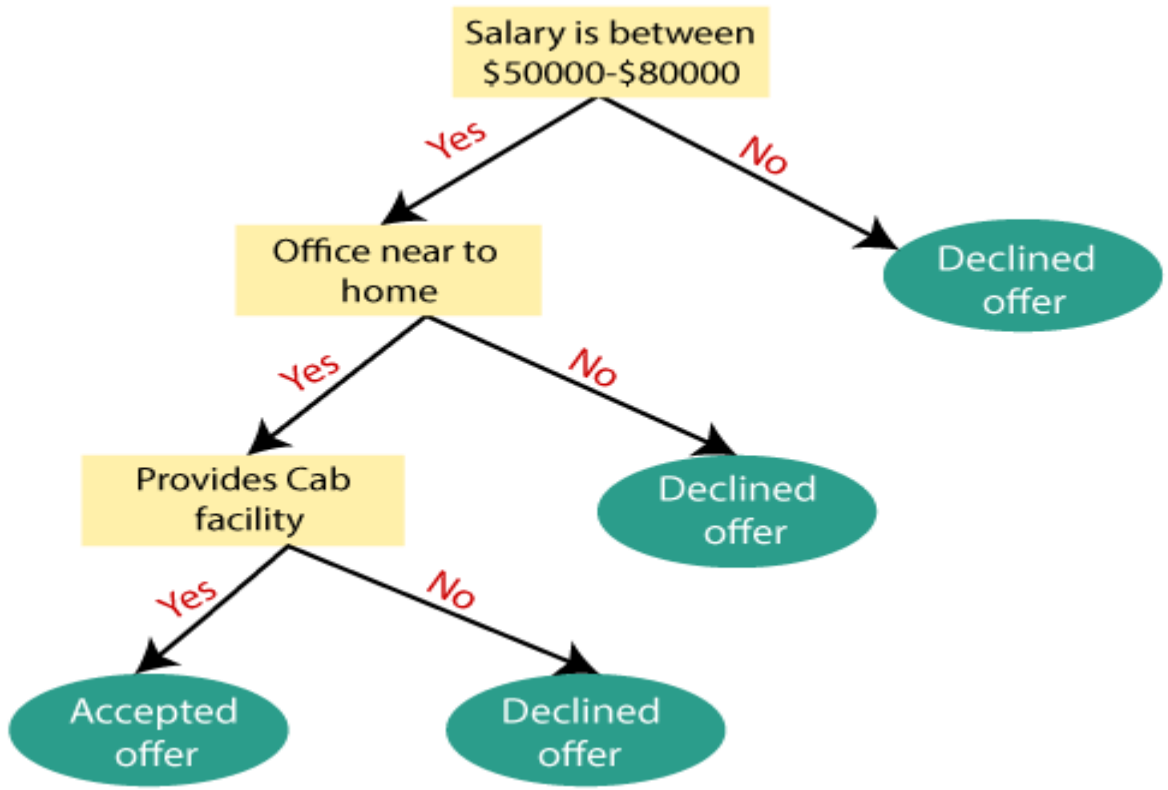
Fig. 5.1. Decision tree in ML

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.

- It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**

- In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.


Decision Node
Root Node
Sub-Tree
Decision Node
Decision Node
Leaf Node
Leaf Node
Leaf Node
Decision Node
Leaf Node
Leaf Node


Salary is between $50000-$80000
Yes
No
Office near to home
Declined offer
Yes
No
Provides Cab facility
Declined offer
Yes
No
Accepted offer
Declined offer

- To make a prediction using the decision tree, we start at the root node and follow the branches based on the feature values of the new instance.

- Eventually, we reach a leaf node and output the corresponding prediction.

Attribute Selection Measures

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

- The decision tree algorithm starts with the entire dataset and selects the best feature to split the data based on a criterion, such as information gain or Gini impurity.

- The selected feature becomes the root node of the tree, and the data is split into different branches based on the possible attribute values.

1. Information gain

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

Information Gain= Entropy(S)-
sum[(Weighted Avg) *Entropy(each feature)]

2. The Gini Index

- The Gini Index is calculated by subtracting the sum of the squared probabilities of each class from one.
- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index. Perfectly classified, Gini Index would be zero.
- Gini index can be calculated using the below formula:

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

Decision Tree Terminology

- **Entropy E(x):** The "average amount of information" contained by a random variable (x) is called Entropy. It is denoted by (E) or (H).
- In other words, entropy is the "measure of randomness of information" of a variable. Entropy (E) is the measure of impurity or uncertainty associated with a random variable (X).

$$\mathrm{E}(x) \text{ or } \mathrm{H}(x) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \,.\, \log_2 \mathrm{P}(x_i)$$

Example Calculate the entropy (E) of a single attribute "Playing Golf" problem when following data is given.

Given data

Play Golf	
Yes	No
9	5

Solution. $$\text{Entropy, } E(S) = \sum_{i=1}^{n} - p_i \cdot \log_2 p_i$$

where S = current state, p_i = Prob. of event (i) of state (S).

Entropy (Play Golf) = Entropy (5, 9)

$$\text{Prob. of Play Golf = Yes} \quad \Rightarrow \quad \frac{9}{14} = 0.64$$

$$\text{Prob. of Play Golf = No} \quad \Rightarrow \quad \frac{5}{14} = 0.36$$

$$\text{Entropy, } E(S) = - (0.36 \log_2 (0.36)) - (0.64 \log_2 (0.64))$$

$$E(S) = 0.94$$ **Ans.**

Example 5.2. Calculate the entropy of multiple attributes of "Playing Golf" problem with given data set.

Given data

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

Solution. The entropy of multiple attribute is given as:

$$E(T, X) = \sum_{c \in X} P(c) \cdot E(c) \qquad \ldots(5.3)$$

where T = current state, X = Selected attribute

E(Play Golf, Outlook) = P(Sunny), E(3, 2) + P(Overcast) . E(4, 0) + P(Rainy) E(2, 3)

E(Play Golf Outlook) = (5/14) . 0.971 + (4/14).0 + (5/14) . 0.971

$$E(\text{Play Golf, Outlook}) = 0.693$$ **Ans.**

INFORMATION GAIN (IG)

Definition: *In machine learning and decision trees, the information gain (IG) is defined as the reduction (decrease) in entropy.*

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

where, T = Current state (S)

X = Selected attribute

or

$$IG(S, A) = \text{Entropy } (S) - \sum \frac{|S_V|}{|S|} . \text{Entropy } (S_V)$$

Example Calculate the information gain (IG) for following data:

$$Entropy\ (T) = E(Play\ Golf) = 0.94$$

$$Entropy\ (T, X) = E(Play\ Golf, Outlook) = 0.693.$$

Solution.

$$IG = Entropy\ (T) - Entropy\ (T, X)$$

$$IG = 0.94 - 0.693 = 0.247 \quad \textbf{Ans.}$$

ID3 Algorithm

ID3 Steps

1. Calculate the Information Gain of each feature.
2. Considering that all rows don’t belong to the same class, split the dataset **S** into subsets using the feature for which the Information Gain is maximum.
3. Make a decision tree node using the feature with the maximum Information gain.
4. If all rows belong to the same class, make the current node as a leaf node with the class as its label.
5. Repeat for the remaining features until we run out of all features, or the decision tree has all leaf nodes.

Example Make a decision tree from the given training data of Table 1.

Table 1. Training examples (data) for target concept "Play Tennis"

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

Here, nine "Yes", Five "No" i.e. 9 (+ve) and 5(-ve) example in this Table 1

Solution. Let us consider a data set (S) as shown in Table 5.1. It consists of 14 training examples in which 9 (+ve) examples and 5 (–ve) examples denoted by (9 +, 5 –).

- **Entropy of Entire Data Set Collection**

$$\text{Entropy (S)} = \sum_{i=1}^{C} - p_i \cdot \log_2 p_i$$

p_i = Probability or proportion of +ve and –ve examples

$$\text{Entropy (9 +, 5 –)} = -\left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right) - \left(\frac{5}{14}\right).\log_2\left(\frac{5}{14}\right)$$

$$\text{Entropy (9 +, 5 –)} = 0.940$$

- **Root Node Selection**

4 Attributes = Outlook, Temp., Humidity, Wind

1 Target = Play Tennis

The attribute which gives highest information gain (IG) is selected as root node.

- **Attribute 1 = Outlook**

Values (outlook) = Sunny, Overcast, Rain

We will calculate the entropy (S) of each value *i.e.*, S_{sunny}, $S_{overcast}$ and S_{rain}

(*i*) Entropy of Sunny (S_{sunny}): Count number of training examples having "Sunny" in the Table 5.1. There are 2 +ve and 3 –ve training examples which consists "Sunny" in the column of "Outlook" attribute.

$\therefore$ $S_{sunny} = (2+, 3-)$

Entropy (S_{sunny}) = (Proportion of +ve Examples) $\log_2$ (Proportion of +ve Examples) – (Proportion of – ve Examples) . $\log_2$ (Proportion of – ve Examples) ...(5.7)

$$= \frac{2}{5}\log_2\frac{2}{5} - \frac{3}{5}\log\frac{3}{5}$$

$$\boxed{\text{Entropy } (S_{sunny}) = 0.971}$$

(*ii*) Entropy of Overcast: $S_{overcast} \leftarrow (4+, 0-)$

$$\text{Entropy } (S_{overcast}) = -\left(\frac{4}{4}\right)\log_2\left(\frac{4}{4}\right) - \left(\frac{0}{4}\right)\log_2\left(\frac{0}{4}\right) = 0$$

(*iii*) Entropy of Rain: $S_{rain} \leftarrow (3+, 2-)$

$$\text{Entropy } (S_{rain}) = -\left(\frac{3}{5}\right)\log_2\left(\frac{3}{5}\right) - \left(\frac{2}{5}\right)\log_2\left(\frac{2}{5}\right) = 0.971.$$

- **Information Gain of Outlook:** (w.r.t. entire dataset S)

$$\text{Gain}\left(\underset{\text{Entire Dataset}}{S}, \underset{\text{Attribute}}{\text{Outlook}}\right) = \text{Entropy of all data set (S)} - \sum_{v \in (\text{Sunny, Overcast, Rain})} \frac{|S_v|}{|S|} \text{Entropy } (S_v)$$

...(5.8)

S_v = Number of times any attribute value is appearing *e.g.*,
Sunny is appearing 5 times
Overcast is appearing 4 times
Rain is appearing 5 times.

$$\text{Gain (S, Outlook)} = \text{Entropy (S)} - \left(\frac{5}{14}\right) \text{Entropy } (S_{\text{sunny}}) - \left(\frac{4}{14}\right) \text{Entropy } (S_{\text{overcast}}) - \left(\frac{5}{14}\right) \text{Entropy } (S_{\text{rain}})$$

$$\text{Gain (S, Outlook)} = 0.94 - \left(\frac{5}{14}\right) 0.971 - \left(\frac{4}{14}\right) . 0 - \left(\frac{5}{14}\right) . 0.971$$

$$\boxed{\text{Gain (S, Outlook)} = 0.2464}$$

This is the information gain (IG) of outlook attribute *i.e.*, 0.2464.

Similarly, we calculate the information gain of remaining 3 attributes in Table 5.1 *i.e.*, Temp., Humidity and Wind.

- **Attribute 2 = Temperature**

$$\text{Value (Temp.)} = \text{Hot, Mild, Cool}$$

$$S = (9+\ ,\ 5-)$$

$$\text{Entropy (S)} = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.94$$

$$S_{\text{Hot}} = (2+,\ 2-)$$

$$\text{Entropy } (S_{\text{Hot}}) = -\frac{2}{4}\log_2\frac{2}{4} - \frac{2}{4}\log_2\frac{2}{4} = 1$$

$$S_{\text{Mild}} = (4+,\ 2-)$$

$$\text{Entropy } (S_{\text{Mild}}) = -\frac{4}{6}\log\frac{4}{6} - \frac{2}{6}\log_2\frac{2}{6} = 0.9183$$

$$S_{\text{Cool}} = (3+,\ 1-)$$

$$\text{Entropy } (S_{\text{Cool}}) = -\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4} = 0.8113$$

$$\text{Information Gain (S, Temp)} = \text{Entropy(S)} - \sum_{\substack{v \in \\ \text{(Hot, Cool, Mild)}}} \frac{|S_v|}{|S|} \text{Entropy } (S_v) \quad \ldots(5.9)$$

$$\text{Information Gain (S, Temp.)} = \text{Entropy (S)} - \frac{4}{14} \text{ Entropy } (S_{Hot})$$

$$- \frac{6}{14} \text{ Entropy } (S_{Mild}) - \frac{4}{14} \text{ Entropy } (S_{Cool})$$

$$\text{I.G.} = 0.94 - \left(\frac{4}{14}\right) \times 1 - \left(\frac{6}{14}\right) \times 0.9183 - \left(\frac{4}{14}\right) \times 0.8113$$

$$= 0.0289$$

Similarly, we obtain the Information Gain of remaining two attributes as:

I. Gain (S, Outlook) = 0.2464 ← Highest IG
I. Gain (S, Temp.) = 0.0289
I. Gain (S, Humidity) = 0.1516
I. Gain (S wind) = 0.0478

$\therefore$ Root node will be outlook.

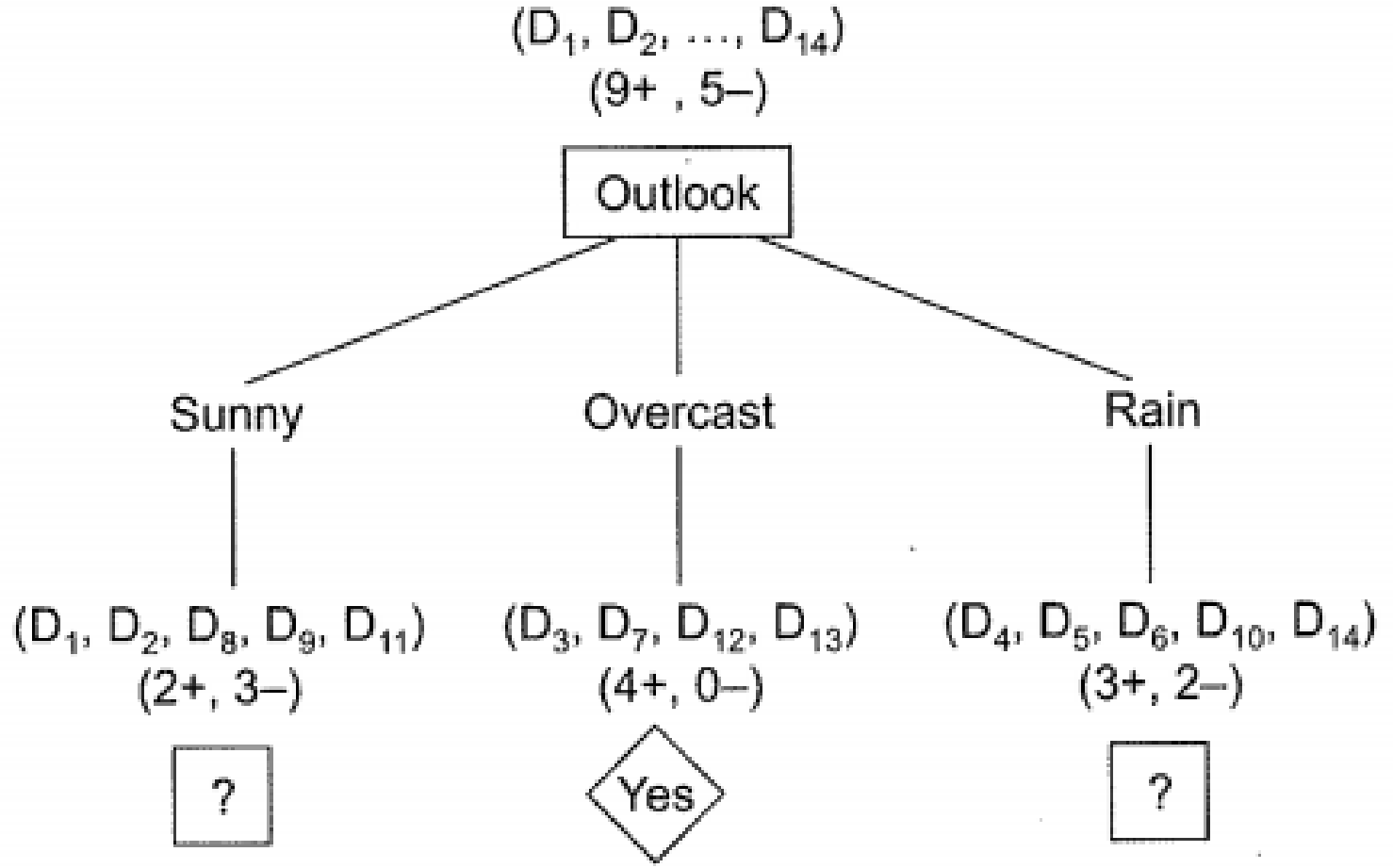


Fig. 5.6. Decision tree at middle stage

Now, we need to test dataset for custom subsets of outlook attribute.

Overcast outlook on decision

Basically, decision will always be yes if outlook were overcast.

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

Sunny outlook on decision

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

Here, there are 5 instances for sunny outlook. Decision would be probably 3/5 percent no, 2/5 percent yes.

1- Gain(Outlook=Sunny|Temperature) = 0.570

2- Gain(Outlook=Sunny|Humidity) = 0.970

3- Gain(Outlook=Sunny|Wind) = 0.019

Now, humidity is the decision because it produces the highest score if outlook were sunny.

At this point, decision will always be no if humidity were high.

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

On the other hand, decision will always be yes if humidity were normal

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

Finally, it means that we need to check the humidity and decide if outlook were sunny.

Rain outlook on decision

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

1- Gain(Outlook=Rain | Temperature) = 0.0199730940219749

2- Gain(Outlook=Rain | Humidity) = 0.0199730940219749

3- Gain(Outlook=Rain | Wind) = 0.970950594454668

Here, wind produces the highest score if outlook were rain. That's why, we need to check wind attribute in 2nd level if outlook were rain.

So, it is revealed that decision will always be yes if wind were weak and outlook were rain.

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

What's more, decision will be always no if wind were strong and outlook were rain.

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

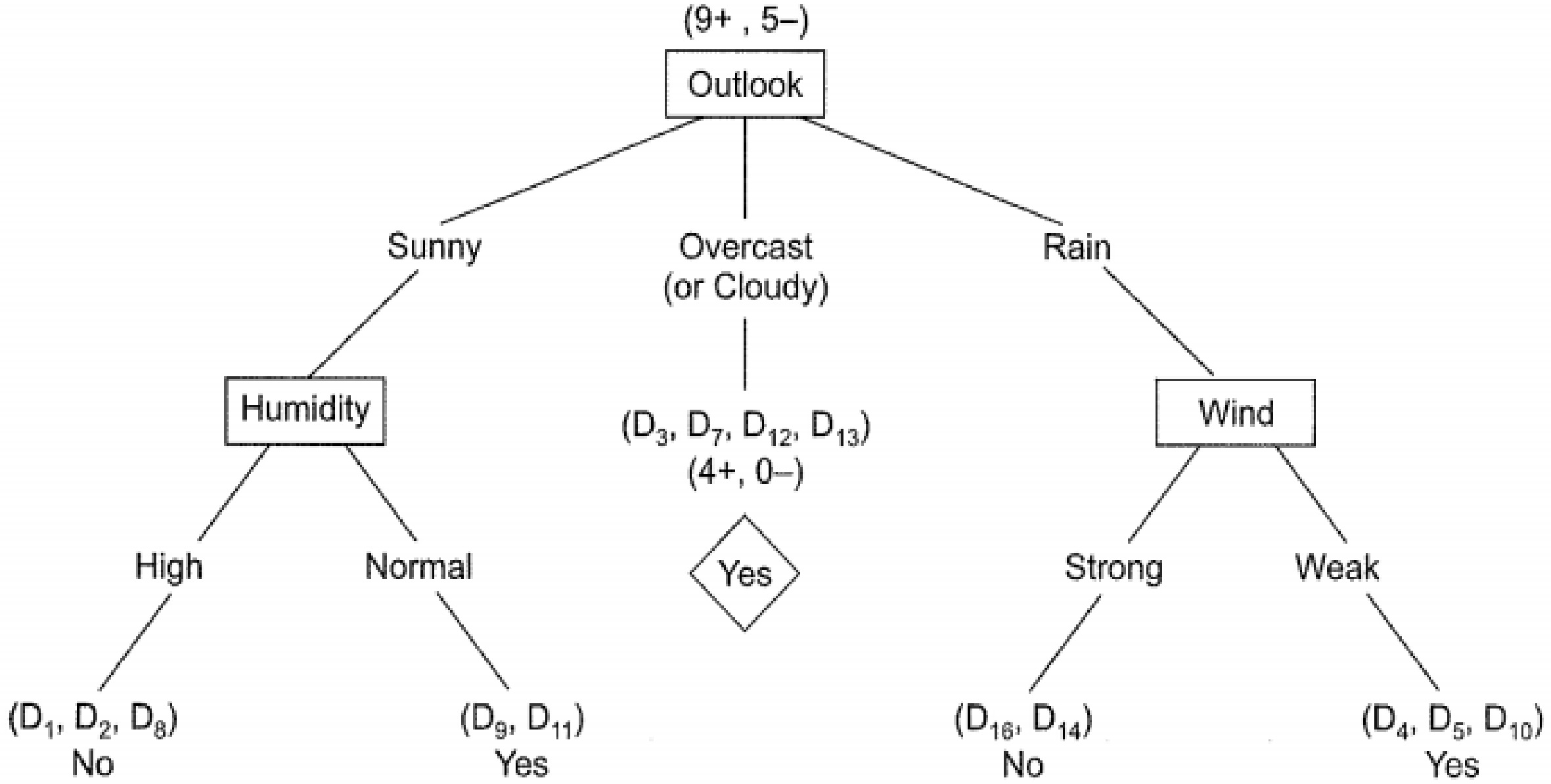


Fig. 5.7. Final decision tree using ID3 algorithm

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.

$$Logloss = \frac{1}{N}\sum_{i=1}^{N} logloss_i$$

$$Logloss = -\frac{1}{N}\sum_{i=1}^{N}[y_i \, ln \, p_i + (1 - y_i) \, ln(1 - p_i)]$$

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.

Overfitting issues in Decision Tree

- Overfitting refers to the condition when the model completely fits the training data but fails to generalize the testing unseen data.
- Overfit condition arises when the model memorizes the noise of the training data and fails to capture important patterns.
- A perfectly fit decision tree performs well for training data but performs poorly for unseen test data.

Random Forest Technique

- Random Forest is a popular machine learning algorithm that belongs to the 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.*

- A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.

- The fundamental concept behind random forest is a simple but powerful one — the wisdom of crowds.

- 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.“

- The name "random forest" comes from the fact that the algorithm creates an ensemble of decision trees and introduces randomness during the training process.

- Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

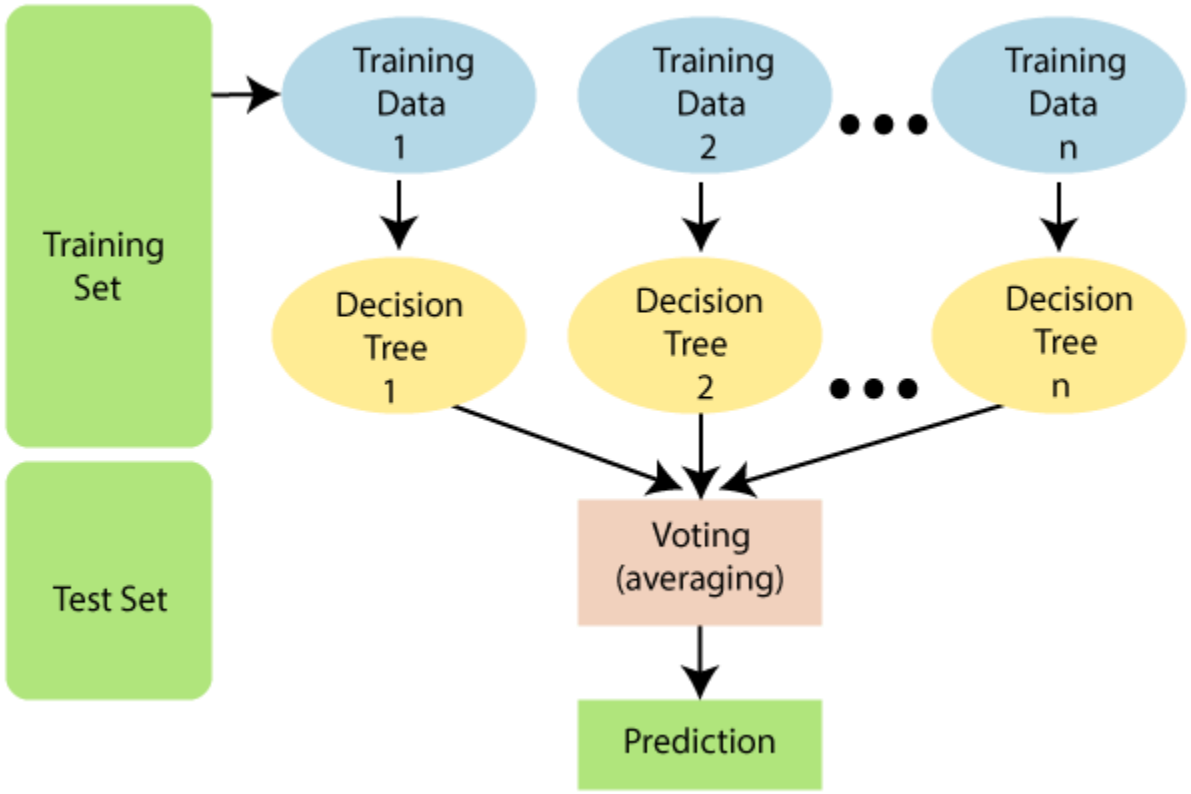
- The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

- Random Forest employs a technique called bootstrapping, where multiple random subsets (with replacement) are created from the original training dataset.

- Each subset, known as a bootstrap sample, is of the same size as the original dataset but contains some repeated instances and may exclude others.

- For each bootstrap sample, a decision tree is constructed using a subset of features randomly chosen from the available features.

Pictorial understanding of Random Forest Algorithm

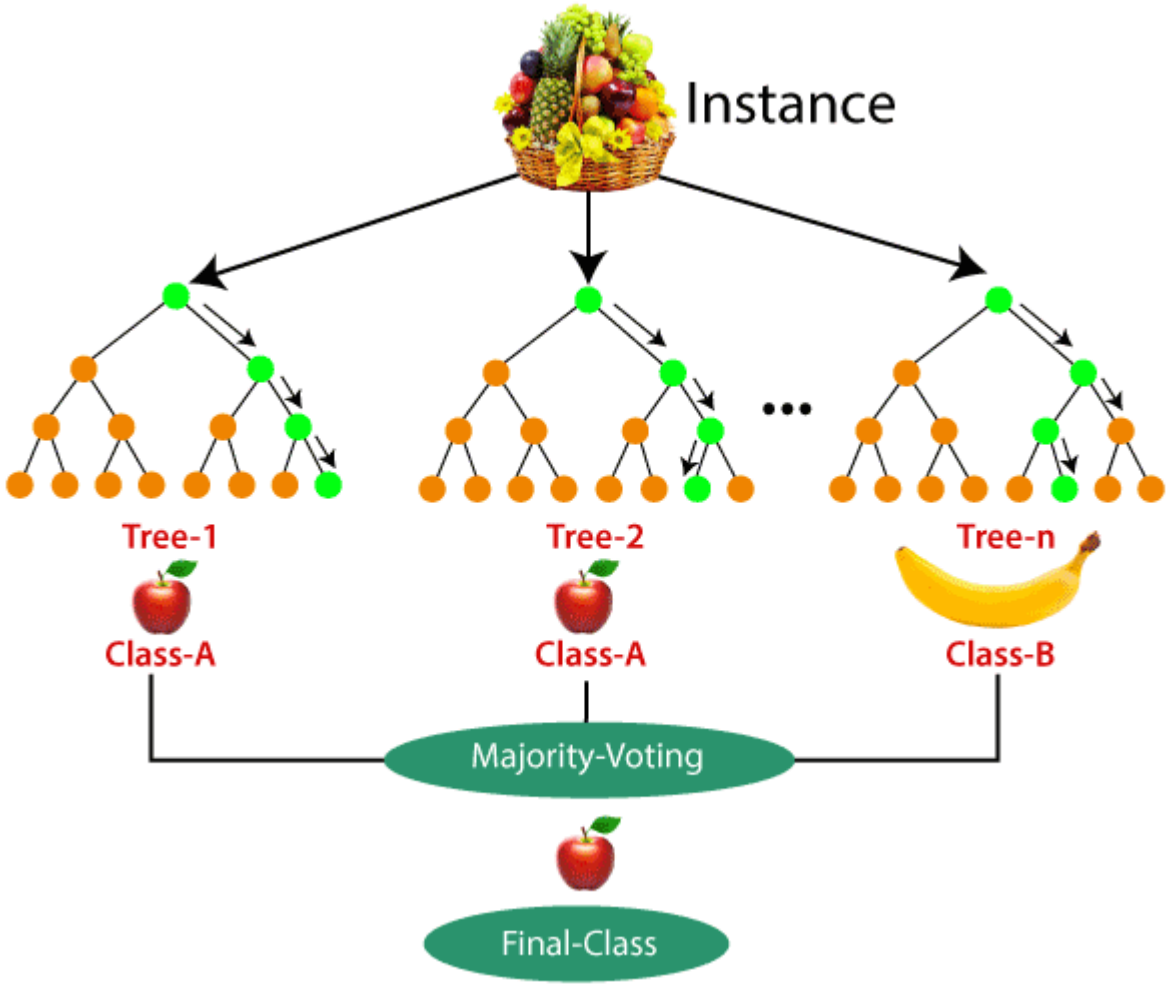


Working of Random Forest algorithm

- Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.
- The Working process can be explained
- **Phase-1**
- 1. Select random K data points from the training set.
- 2. Build the decision trees associated with the selected data points (Subsets).
- **3.** Choose the number N for decision trees that you want to build.
- 4. Repeat Step 1 & 2.

- **Phase -2**

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 Forests offer several advantages:

- They are effective in handling large and high-dimensional datasets.
- They provide robustness against overfitting, as the randomness introduced during the training process helps reduce variance.
- They can handle both numerical and categorical features without requiring extensive data preprocessing.
- They offer feature importance estimation, indicating the relevance of each feature in the prediction process.