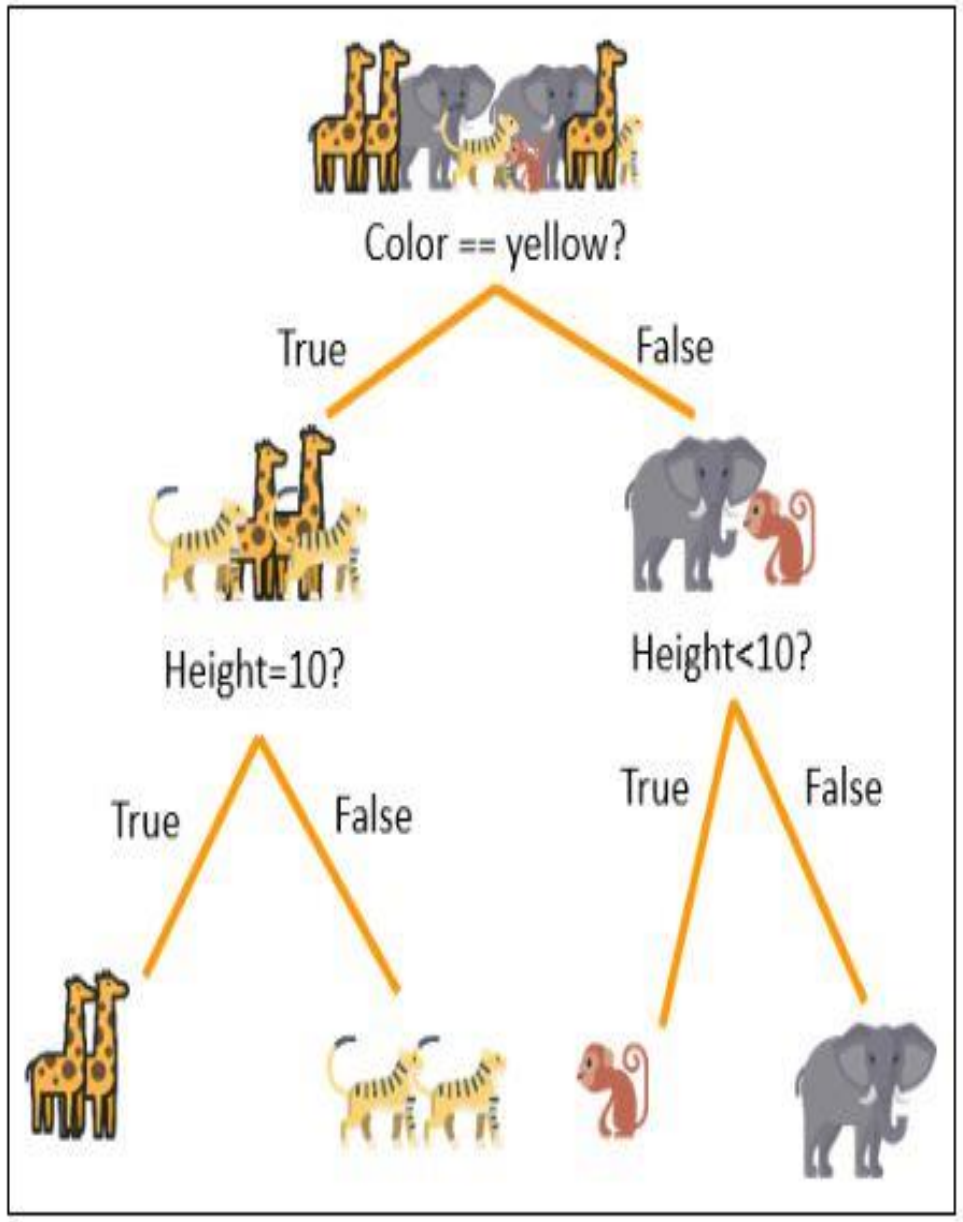
- **Decision Tree Classification Algorithm**

Dr. Jagendra Singh

Machine Learning

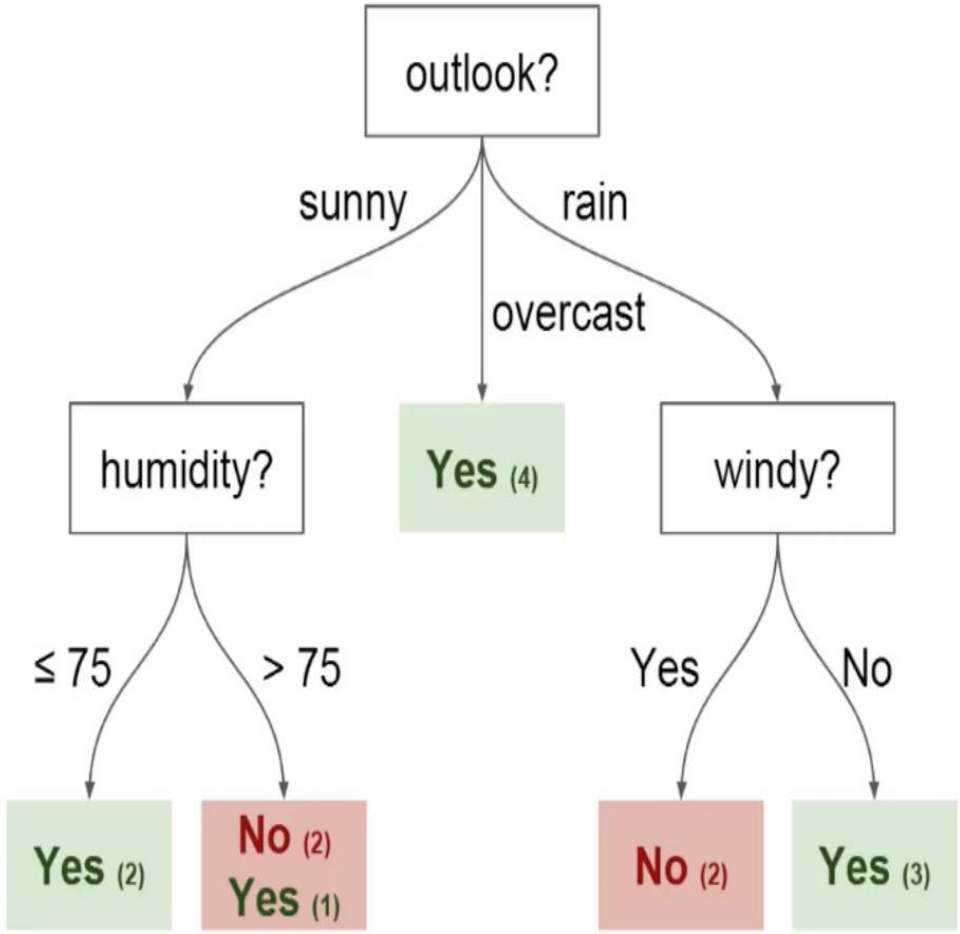


WHAT IS DECISION TREE CLASSIFICATION ALGORITHM?

- Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression 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.

WHAT IS DECISION TREE CLASSIFICATION ALGORITHM?

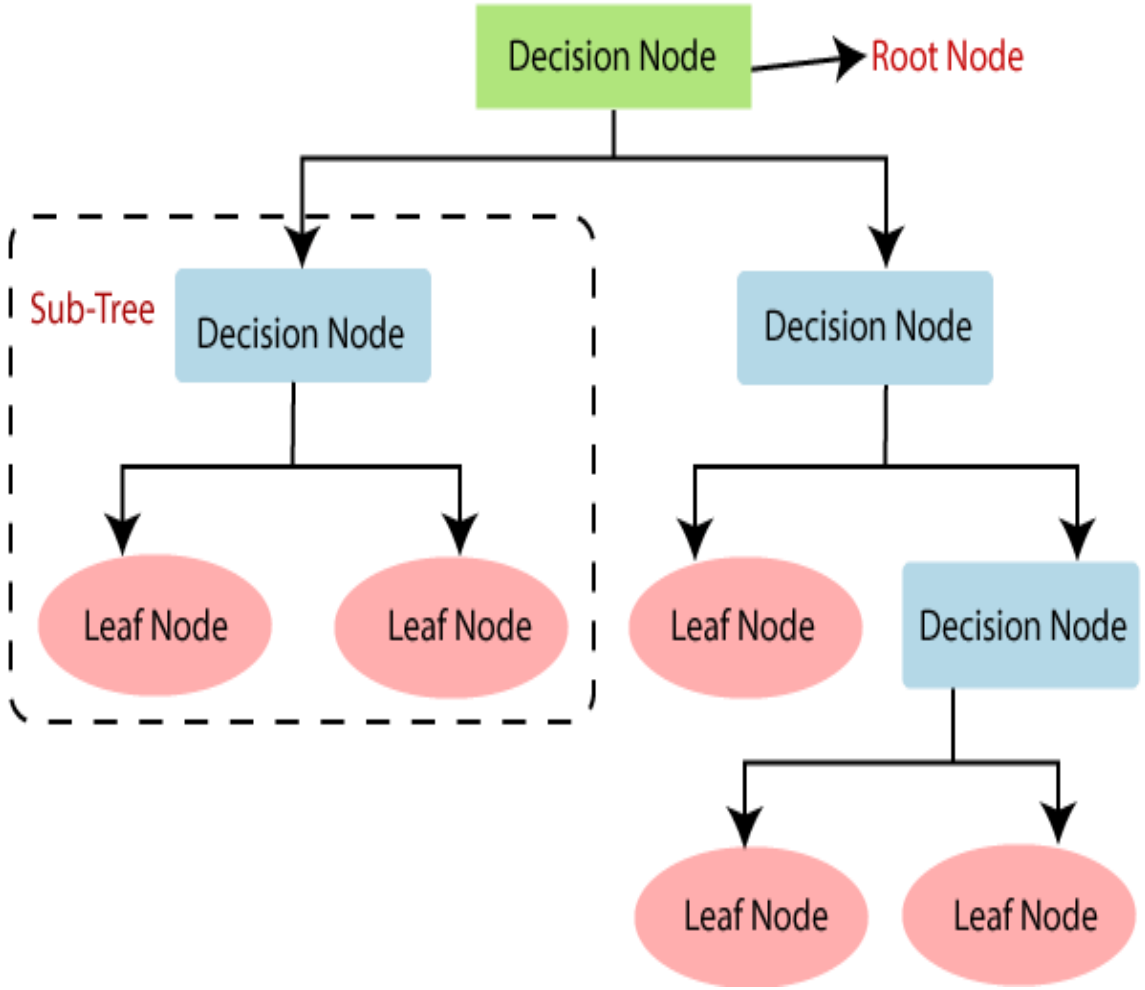
- The decisions or the test are performed on the basis of features of the given dataset.
- It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
- In order to build a tree, we use the **CART algorithm,** 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.



DECISION TREE CLASSIFICATION ALGORITHM

- This diagram explains the general structure of a decision tree:

- Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.



WHY USE DECISION TREES?

Below are the two reasons for using the Decision tree:

- 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.

HOW DOES THE DECISION TREE ALGORITHM WORK?

The complete process can be better understood using the below algorithm:

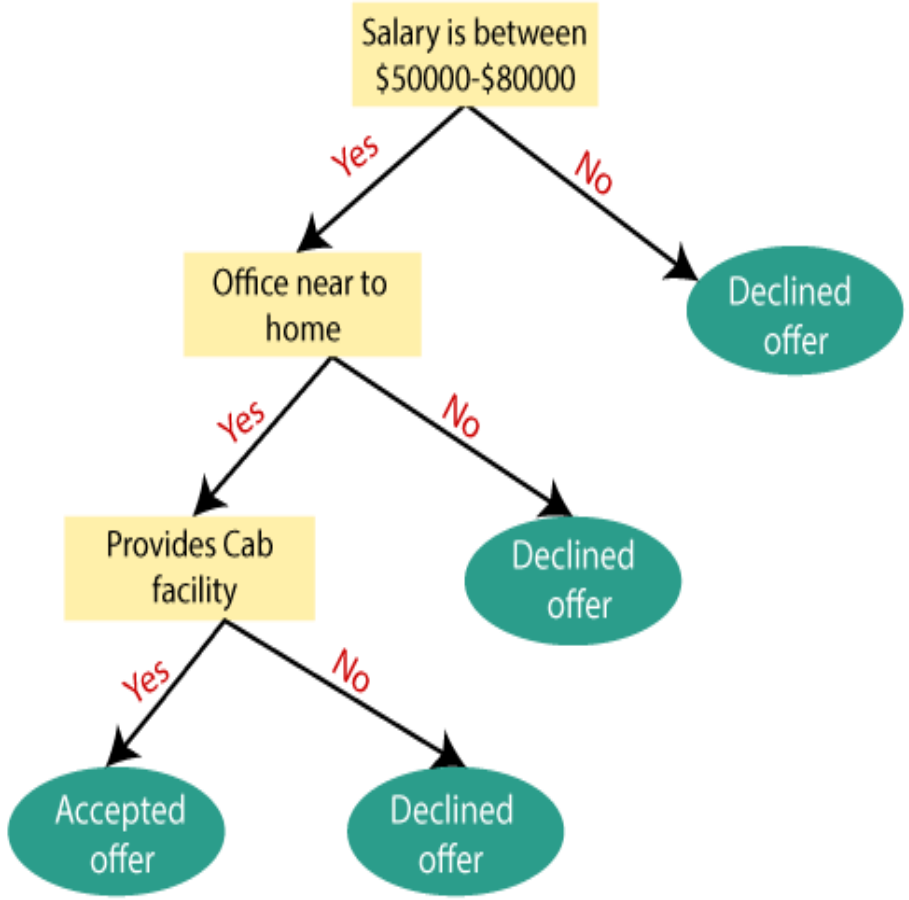
- **Step1:** the tree with the root node, says S, which contains the complete dataset
- **Step-2:** Find the best attribute in the dataset using **Attribute Selection Measure (ASM).**
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3.

Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example:

Suppose 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 ASM). 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).

Consider this diagram:



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**

INFORMATION GAIN

- 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)-
[(Weighted Avg) *Entropy(each feature)

INFORMATION GAIN

- **Entropy:** Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:
- `Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)`
- **Where,**
  - **S= Total number of samples**
  - **P(yes)= probability of yes**
  - **P(no)= probability of no**

GINI INDEX

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

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

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.

PYTHON IMPLEMENTATION OF DECISION TREE

- Now we will implement the Decision tree using Python. For this, we will use the dataset "**user_data.csv**,"

Steps will also remain the same, which are given below:

- Data Pre-processing step
- Fitting a Decision-Tree algorithm to the Training set
- Predicting the test result
- Test accuracy of the result(Creation of Confusion matrix)
- Visualizing the test set result.

THANK YOU