What is a Decision Tree?

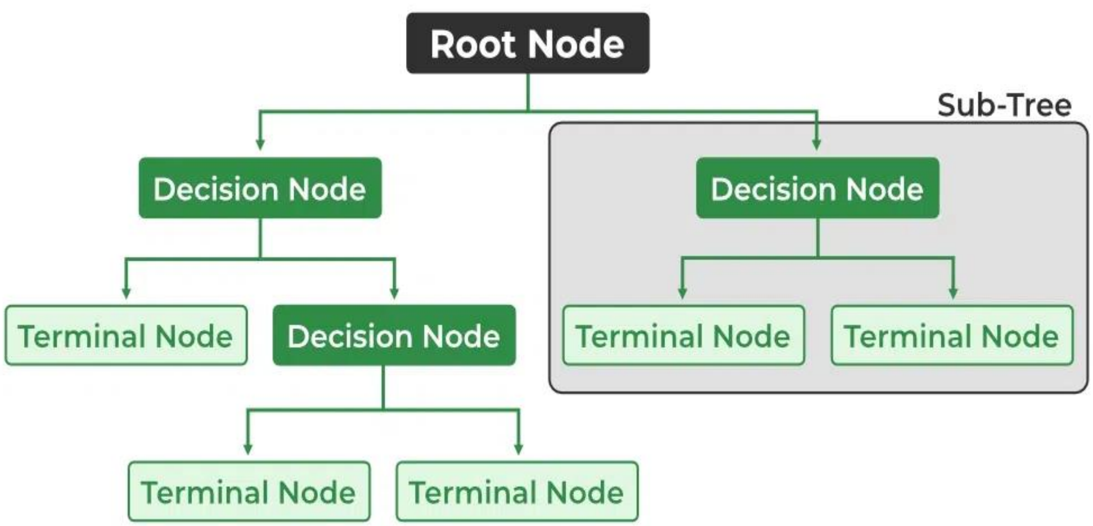
Decision Tree: A decision tree is a flowchart-like structure where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label. It is a powerful tool of supervised learning algorithms used for both classification and regression tasks.

Decision Tree Terminologies

Some of the common Terminologies used in Decision Trees are as follows:

- **Root Node:** It is the topmost node in the tree, which represents the complete dataset. It is the starting point of the decision-making process.
- Decision/Internal Node: A node that symbolizes a choice regarding an input feature. Branching off of internal nodes connects them to leaf nodes or other internal nodes.
- **Leaf/Terminal Node:** A node without any child nodes that indicates a class label or a numerical value.
- **Splitting:** The process of splitting a node into two or more sub-nodes using a split criterion and a selected feature.
- **Branch/Sub-Tree:** A subsection of the decision tree starts at an internal node and ends at the leaf nodes.
- **Parent Node:** The node that divides into one or more child nodes.
- **Child Node:** The nodes that emerge when a parent node is split.
- **Impurity**: A measurement of the target variable's homogeneity in a subset of data. It refers to the degree of randomness or uncertainty in a set of examples. The **Gini index** and **entropy** are two commonly used impurity measurements in decision trees for classifications task

- **Variance**: Variance measures how much the predicted and the target variables vary in different samples of a dataset. It is used for regression problems in decision trees. **Mean squared error, Mean Absolute Error, friedman_mse, or Half Poisson deviance** are used to measure the variance for the regression tasks in the decision tree.
- **Information Gain:** Information gain is a measure of the reduction in impurity achieved by splitting a dataset on a particular feature in a decision tree. The splitting criterion is determined by the feature that offers the greatest information gain, It is used to determine the most informative feature to split on at each node of the tree, with the goal of creating pure subsets
- **Pruning**: The process of removing branches from the tree that do not provide any additional information or lead to overfitting.



Entropy: **(To Check the purity of columns values (categories))**

Entropy is the measure of the degree of randomness or uncertainty in the dataset. In the case of classifications, It measures the randomness based on the distribution of class labels in the dataset.

The entropy for a subset of the original dataset having K number of classes for the i_{th} node can be defined as:

Where,

- S is the dataset sample.
- k is the particular class from K classes
- p(k) is the proportion of the data points that belong to class k to the total number of data points in dataset sample

  S.
- Here p(i,k) should not be equal to zero.

Important points related to Entropy:

1. The entropy is 0 when the dataset is completely homogeneous, meaning that each instance belongs to the same class. It is the lowest entropy indicating no uncertainty in the dataset sample.
2. when the dataset is equally divided between multiple classes, the entropy is at its maximum value. Therefore, entropy is highest when the distribution of class labels is even, indicating maximum uncertainty in the dataset sample.
3. Entropy is used to evaluate the quality of a split. The goal of entropy is to select the attribute that minimizes the entropy of the resulting subsets, by splitting the dataset into more homogeneous subsets with respect to the class labels.
4. The highest information gain attribute is chosen as the splitting criterion (i.e., the reduction in entropy after splitting on that attribute), and the process is repeated recursively to build the decision tree.

Gini Impurity or index: **(is same of entropy which show of purity of columns values)**

Gini Impurity is a score that evaluates how accurate a split is among the classified groups. The Gini Impurity evaluates a score in the range between 0 and 1, where 0 is when all observations belong to one class, and 1 is a random distribution of the elements within classes. In this case, we want to have a Gini index score as low as possible. Gini Index is the evaluation metric we shall use to evaluate our Decision Tree Model.

Here,

- p_i is the proportion of elements in the set that belongs to the i_{th} category.

Information Gain: **(To check how much information a column is gaining)**

Information gain measures the reduction in entropy or variance that results from splitting a dataset based on a specific property. It is used in decision tree algorithms to determine the usefulness of a feature by partitioning the dataset into more homogeneous subsets with respect to the class labels or target variable. The higher the information gain, the more valuable the feature is in predicting the target variable.
The information gain of an attribute A, with respect to a dataset S, is calculated as follows:

where

- A is the specific attribute or class label
- |H| is the entropy of dataset sample S
- $|H_V|$ is the number of instances in the subset S that have the value v for attribute A

Information gain measures the reduction in entropy or variance achieved by partitioning the dataset on attribute A. The attribute that maximizes information gain is chosen as the splitting criterion for building the decision tree.
Information gain is used in both classification and regression decision trees. In classification, entropy is used as a measure of impurity, while in regression, variance is used as a measure of impurity. The information

gain calculation remains the same in both cases, except that entropy or variance is used instead of entropy in the formula.

How does the Decision Tree algorithm Work?

The decision tree operates by analyzing the data set to predict its classification. It commences from the tree's root node, where the algorithm views the value of the root attribute compared to the attribute of the record in the actual data set. Based on the comparison, it proceeds to follow the branch and move to the next node.

The algorithm repeats this action for every subsequent node by comparing its attribute values with those of the sub-nodes and continuing the process further. It repeats until it reaches the leaf node of the tree. The complete mechanism can be better explained through the algorithm given below.

- Step-1: Begin 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 nodeClassification and Regression Tree algorithm.

In dataset:

Independent column = input column = input features they are same they are can be many

Similarly Target column = output column = dependent feature are the same they are only one

Advantages of the Decision Tree:

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

Disadvantages of the Decision Tree:

1. The decision tree contains lots of layers, which makes it complex.
2. It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
3. For more class labels, the computational complexity of the decision tree may increase.

Python Code:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

# Load the iris dataset
iris = load_iris()
X = iris.data
y = iris.target

# Split the dataset into a training set and a test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Create a decision tree classifier object
```

```
clf = DecisionTreeClassifier(criterion='entropy')

# Train the decision tree classifier
clf.fit(X_train, y_train)

# Predict the labels of the test set
y_pred = clf.predict(X_test)

# Print the accuracy of the classifier
print("Accuracy:", accuracy_score(y_test, y_pred))
```