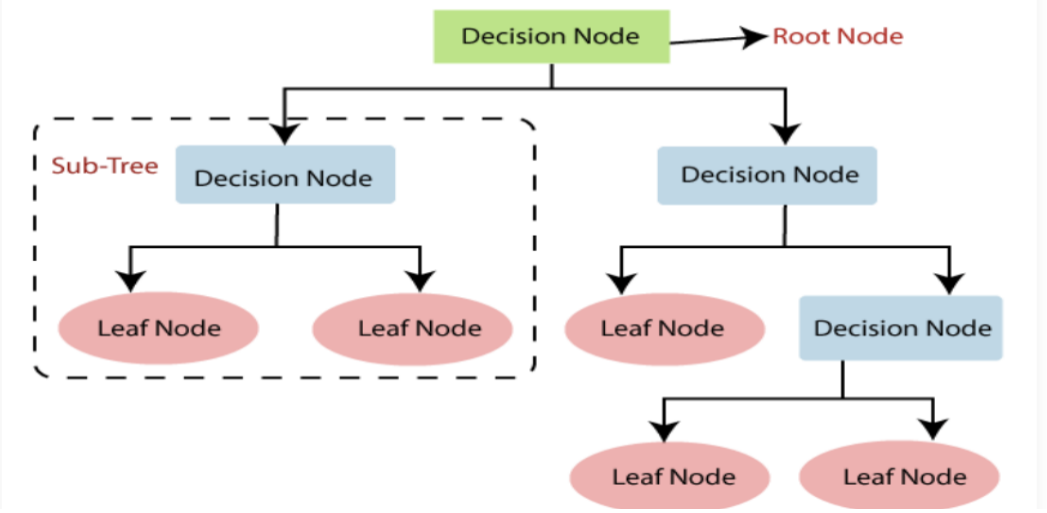
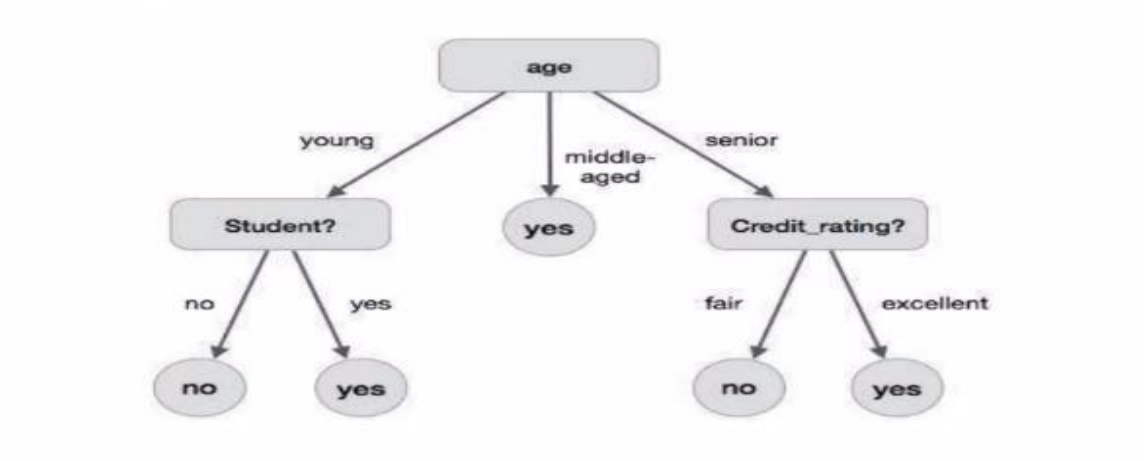
Decision Trees – Introduction

- Decision tree is a simple but powerful learning Paradigm.
- It is a type of classification algorithm for supervised learning.
- A decision tree is a tree in which each branch node represents a choice between a number of alternatives and each leaf node represents a decision.
- A node with outgoing edges is called an internal or test node. All other nodes are called leaves (also known as terminal or decision nodes).

- 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.
- *t is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.*
- A decision tree can contain categorical data (YES/NO) as well as numeric data.



Example:



Decision Tree Terminologies:

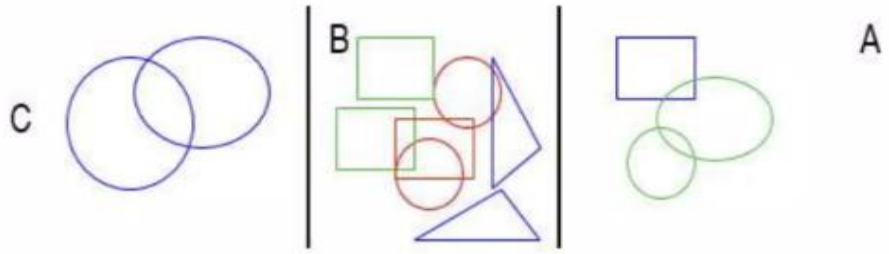
- Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- Splitting: Splitting 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.
- Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

Construction of decision tree

1) First test all attributes and select the one that will function as the best root.
2) Break-up the training set into subsets based on the branches of the root node.
3) Test the remaining attributes to see which ones fit best underneath the branches of the root node.
4) Continue this process for all other branches until

Decision Tree Example

- Given a collection of shape and colour example, learn a decision tree that represents it. Use this representation to classify new examples.



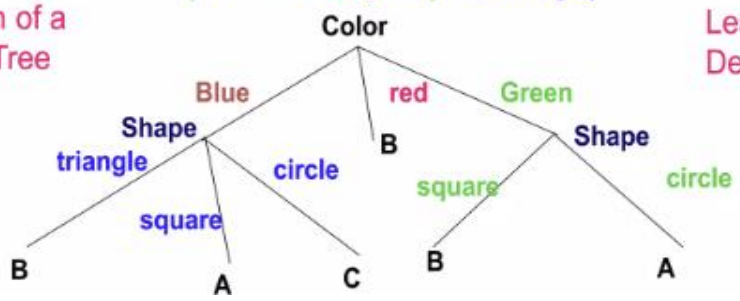
Decision Trees: The Representation

- Decision Trees are classifiers for instances represented as features vectors. (color = ;shape = ;label =)
- Nodes are tests for feature values;
- There is one branch for each value of the feature
- Leaves specify the categories (labels)
- Can categorize instances into multiple disjoint categories – multi-class

(color = RED ;shape = triangle)

Evaluation of a Decision Tree

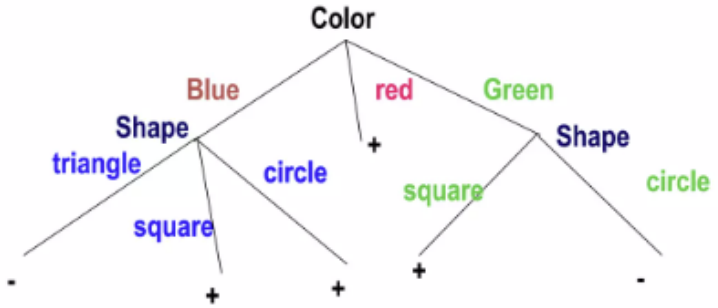
Learning a Decision Tree



Binary classified Decision Trees

They can represent any Boolean function

- Can be rewritten as rules in Disjunctive Normal Form (DNF)
- green ∧ square →positive
- blue ∧ circle → positive
- blue ∧ square → positive
- The disjunction of these rules is equivalent to the Decision Tree



Why use Decision Trees?

- There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. 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.

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:

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

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:

Gini Index= 1- ∑jPj2

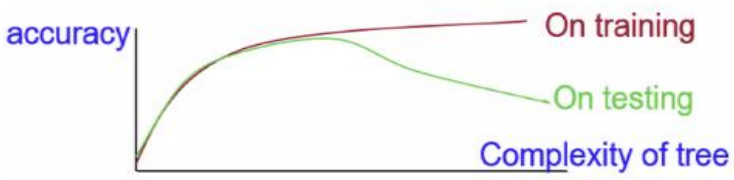
Decision Tree Regression:

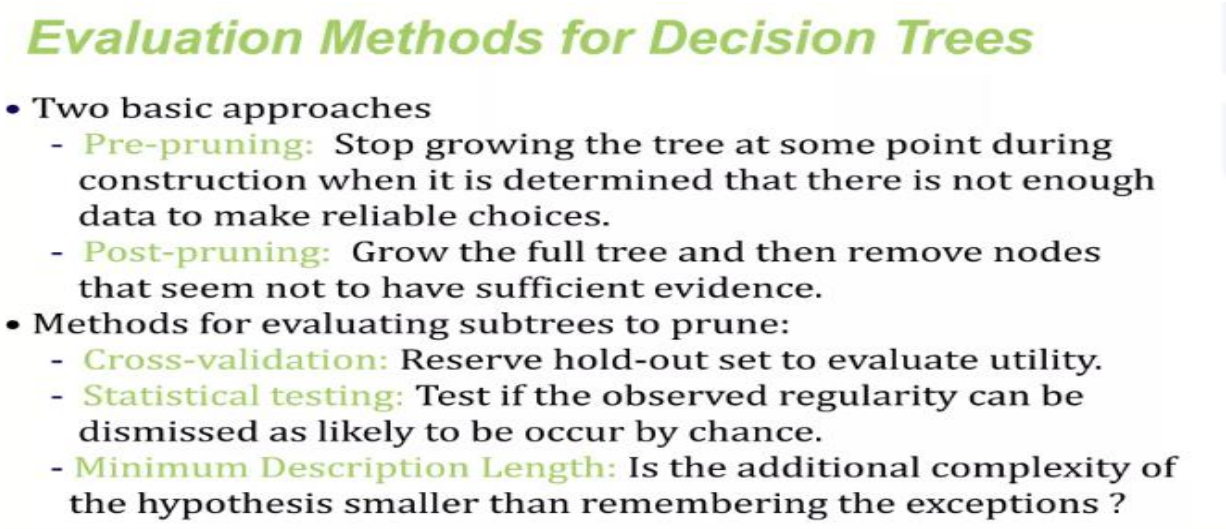
Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

Limitation of Decision Trees

- Learning a tree that classifies the training data perfectly may not lead to the tree with the best generalization performance.
  - There may be noise in the training data the tree is fitting
  - The algorithm might be making decisions based on very little data.
- A hypothesis h is said to overfit the training data if there is another hypothesis, h', such that 'h' has smaller error than h' on the training data but h has larger error on the test data than h'.

Decision Trees





Ensemble Learning:

- An ensemble is a composite model, combines a series of low performing classifiers with the aim of creating an improved classifier.
- Here, individual classifier vote and final prediction label returned that performs majority voting.
- Ensembles offer more accuracy than individual or base classifier.
- Ensemble methods can parallelize by allocating each base learner to different-different machines.
- Finally, you can say Ensemble learning methods are meta-algorithms that combine several machine learning methods into a single predictive model to increase performance.
- Ensemble methods can decrease variance using bagging approach, bias using a boosting approach, or improve predictions using stacking approach

Example:

- Let's take a real example to build the intuition.
- Suppose, you want to invest in a company XYZ. You are not sure about its performance though.
- So, you look for advice on whether the stock price will increase by more than 6% per annum or not?
- You decide to approach various experts having diverse domain experience

Ensemble Techniques:

Ensemble techniques in machine learning involve combining multiple models to improve performance. One common ensemble technique is bagging, which uses bootstrap sampling to create multiple datasets from the original data and trains a model on each dataset. Another technique is boosting, which trains models sequentially, each focusing on the previous models' mistakes. Random forests are a popular ensemble method that uses decision trees as base learners and combines their predictions to make a final prediction. Ensemble techniques are effective because they reduce overfitting and improve generalization, leading to more robust models.

Simple Ensemble Techniques:

Simple ensemble techniques combine predictions from multiple models to produce a final prediction. These techniques are straightforward to implement and can often improve performance compared to individual models.

Max Voting:

In this technique, the final prediction is the most frequent prediction among the base models. For example, if three base models predict the classes A, B, and A for a given sample, the final prediction using max voting would be class A, as it appears more frequently.

Averaging:

Averaging involves taking the average of predictions from multiple models. This can be particularly useful for regression problems, where the final prediction is the mean of predictions from all models. For classification, averaging can be applied to the predicted probabilities for a more confident prediction.

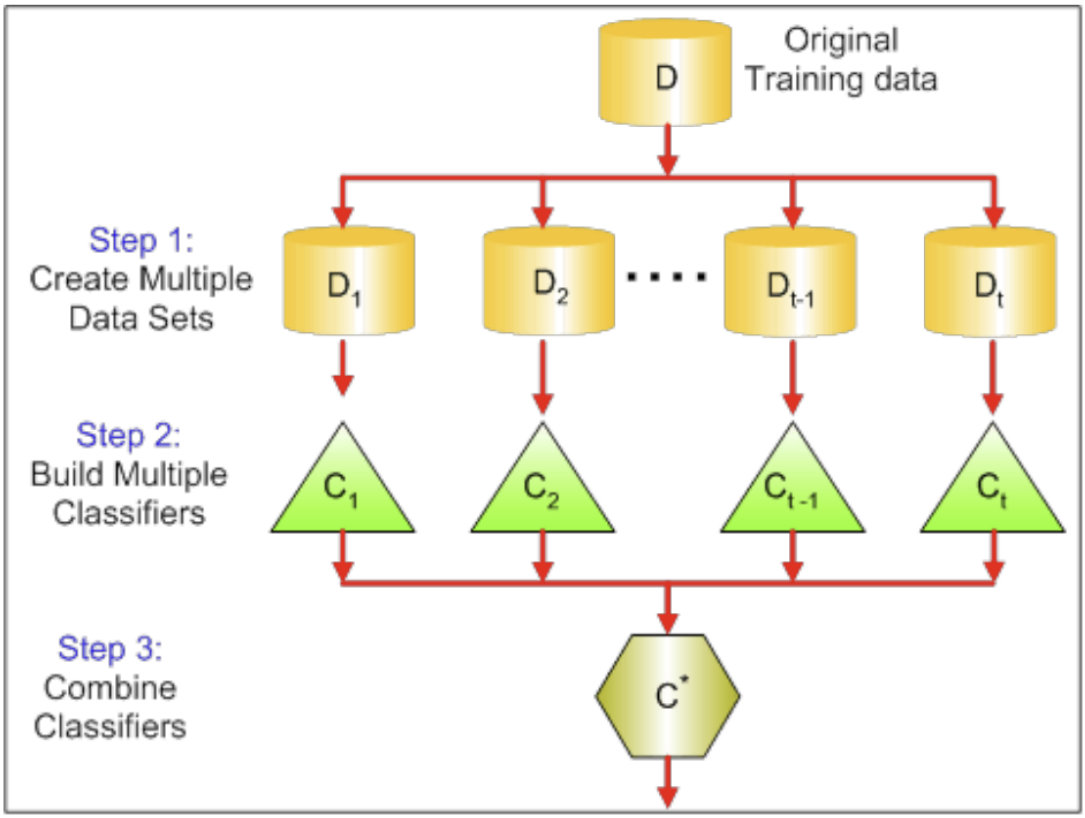
Weighted Averaging:

Weighted averaging is similar, but each model's prediction is given a different weight. The weights can be assigned based on each model's performance on a validation set or tuned using grid or randomized search techniques. This allows models with higher performance to have a greater influence on the final prediction.

Bagging (Bootstrap Aggregating)

- Bagging is a technique where multiple subsets of the dataset are created through bootstrapping (sampling with replacement).
- A base model (often a decision tree) is trained on each subset, and the final prediction is the average (for regression) or majority vote (for classification) of the individual predictions.
- Bagging helps reduce variance and overfitting, especially for unstable models.
- Bagging stands for bootstrap aggregation.
- It combines multiple learners in a way to reduce the variance of estimates.For example, random forest trains M Decision Tree, you can train M different trees on different random subsets of the data and perform voting for final prediction.

Example: – Random Forest – Extra Trees

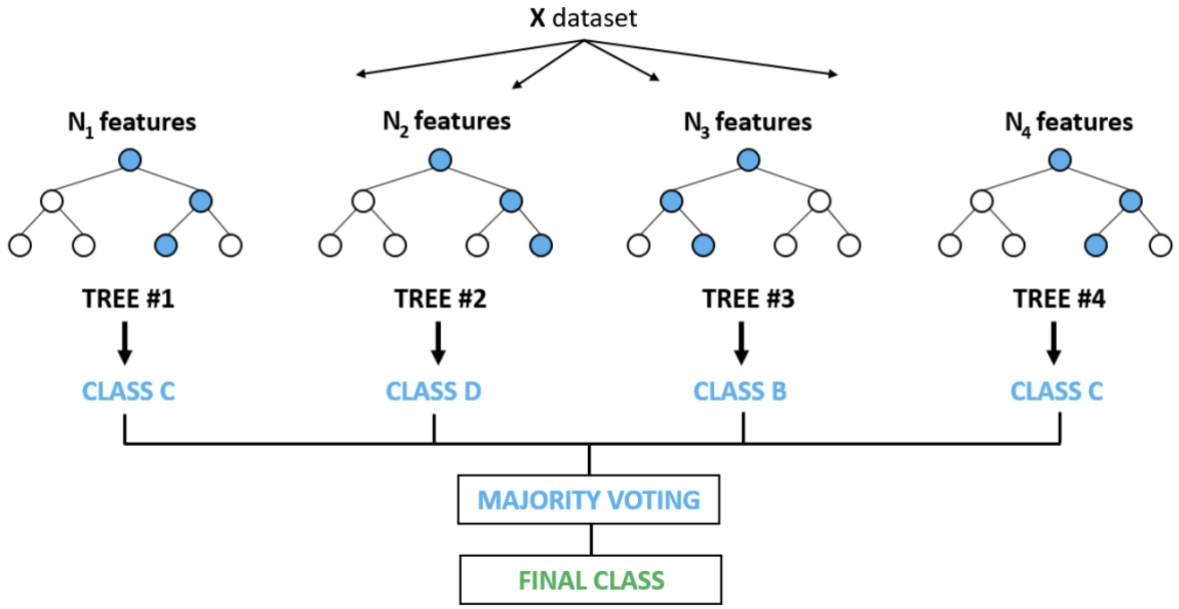


How it works ?

- Pick N random records from the dataset.
- Build a decision tree based on these N records.

- Choose the number of trees you want in your algorithm and repeat steps 1 and 2.
- In case of a regression problem, for a new record, each tree in the forest predicts a value for Y (output). The final value can be calculated by taking the average of all the values predicted by all the trees in forest. Or, in case of a classification problem, each tree in the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the majority vote.

Majority Voting:



Boosting:

- Boosting is an ensemble technique where base models are trained sequentially, with each subsequent model focusing on the mistakes of the previous ones.
- The final prediction is a weighted sum of the individual models' predictions, with higher weights given to more accurate models.
- Boosting algorithms like AdaBoost, Gradient Boosting, and XGBoost are popular because they improve model performance.

Boosting Models:

Models that are typically used in Boosting technique are: –

- XGBoost (Extreme Gradient Boosting)

- GBM (Gradient Boosting Machine)
- ADABoost (Adaptive Boosting

Adaboost Summary:

- Initially, Adaboost selects a training subset randomly.
- It iteratively trains the AdaBoost machine learning model by selecting the training set based on the accurate prediction of the last training.
- It assigns the higher weight to wrong classified observations so that in the next iteration these observations will get the high probability for classification.
- Also, It assigns the weight to the trained classifier in each iteration according to the accuracy of the classifier. The more accurate classifier will get high weight.
- This process iterate until the complete training data fits without any error or until reached to the specified maximum number of estimators.
- To classify, perform a "vote" across all of the learning algorithms you built.

How Adaboost Works?