Random Forest Algorithm

A powerful tree learning technique in Machine Learning.

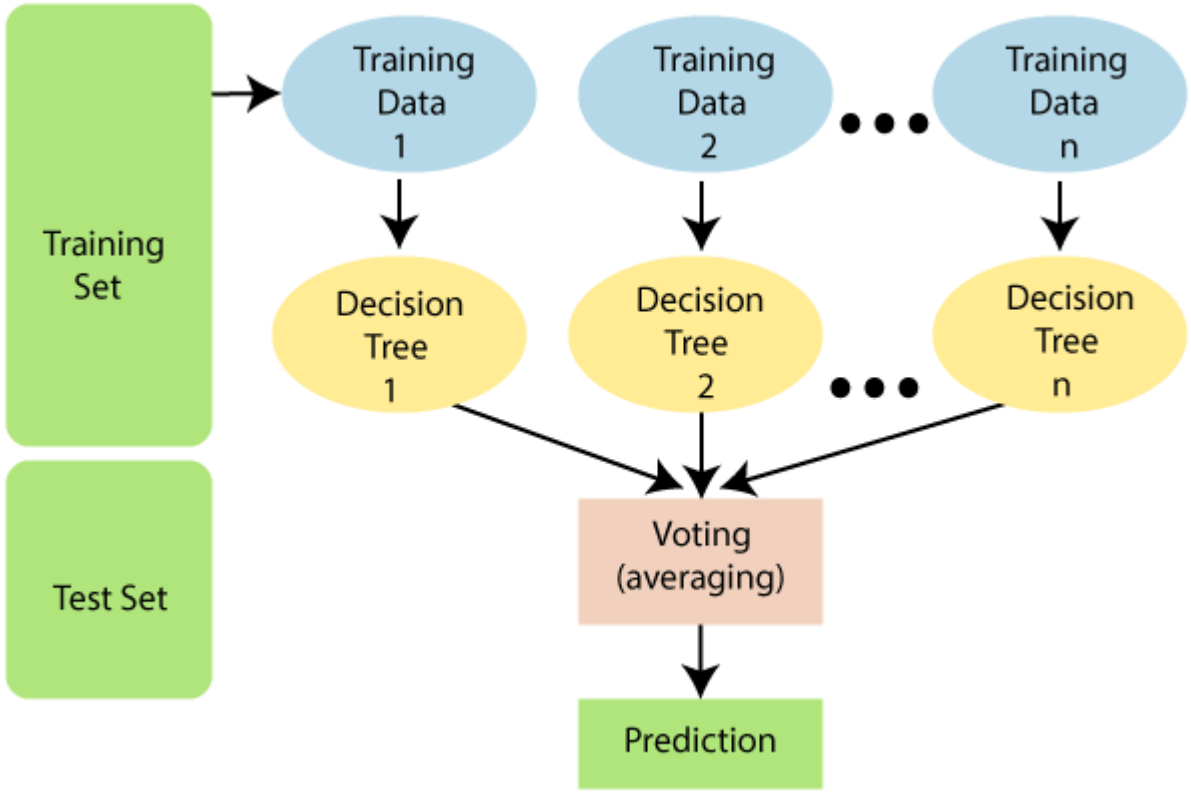
Random Forest Algorithm

- 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**.

Random Forest Algorithm

- Random Forest works by creating a **number of Decision Trees** during the training phase.
- Each tree is constructed using **a random subset of the data set** to measure a random subset of features in each partition.
- This randomness introduces **variability among individual trees**, reducing the **risk of overfitting** and improving **overall prediction performance.**

Random Forest Algorithm

- In prediction phase, the **algorithm aggregates the results of all trees**, either **by voting** (for classification tasks) or by **averaging** (for regression tasks).
- 

This collaborative decision-making process, supported by multiple trees with their insights, provides an example stable and precise results

What are Ensemble Learning models?

- In ensemble learning, different models, often of the same type or different types, team up to enhance predictive performance.
- Ensemble learning models work just like a group of diverse experts teaming up to make decisions
- It's all about leveraging the collective wisdom of the group to overcome individual limitations and make more informed decisions in various machine learning tasks.
- **Some popular ensemble models include-** XGBoost, AdaBoost, LightGBM, Random Forest, Bagging, Voting etc.

How Does Random Forest Work?

- **Random Forest works in two-phases**
  1. create the random forest by combining N decision tree,
  2. make predictions for each tree created in the first phase

How Does Random Forest Work?

- **Steps**

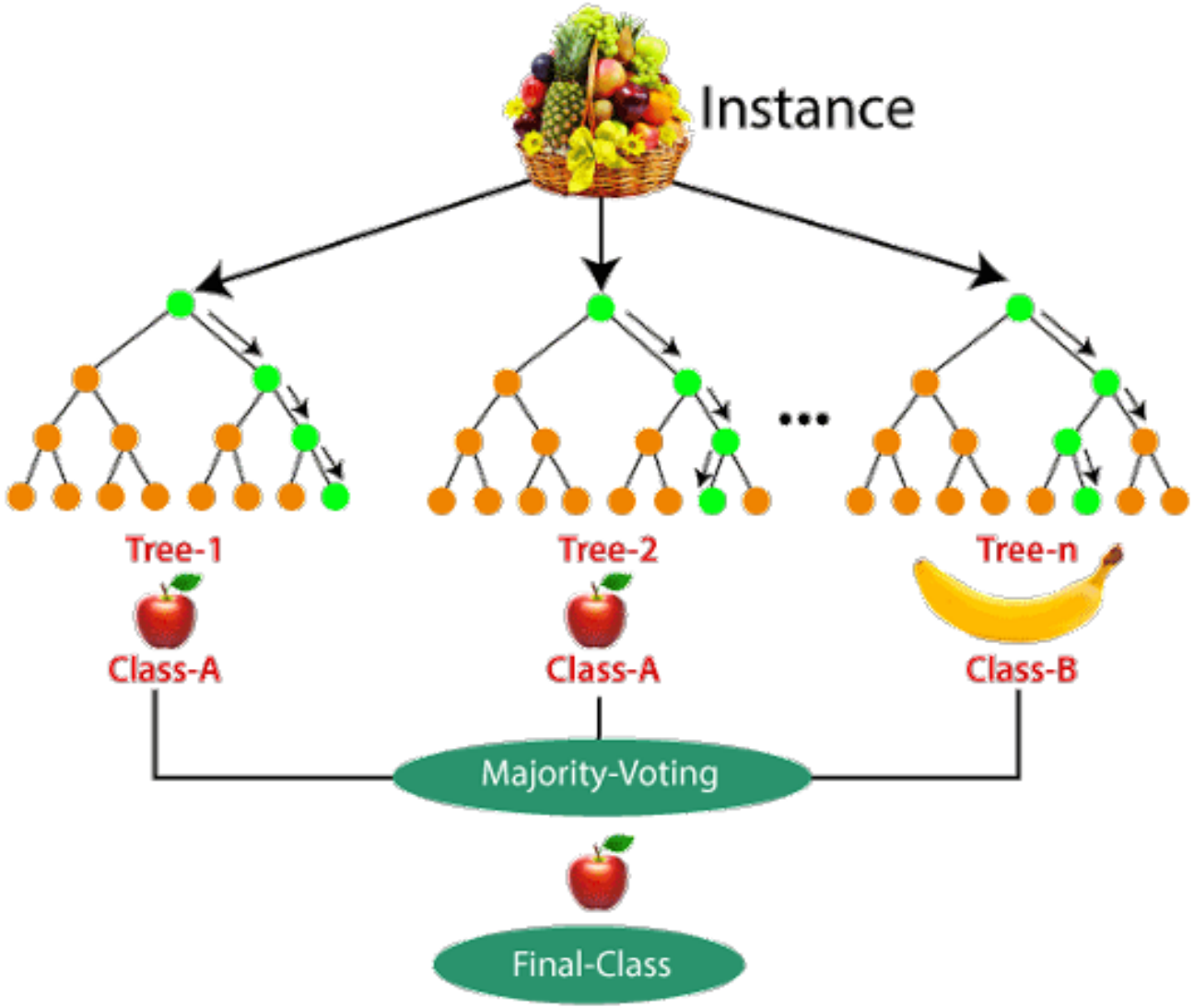
1. Select a subset of data points and a subset of features for constructing each decision tree. Simply put, **n random records** and **m features** are taken from the **data set having k number of records**.
2. Construct individual decision trees for each sample (subset)
3. Find the predictions of each decision tree for a new a data point(instance).
4. Final output is considered based on ***Majority Voting or Averaging*** for Classification and regression, respectively.

How Does Random Forest Work?

- **Example:** Consider the fruit basket as the data as shown in the figure below. Now n number of samples are taken from the fruit basket, and an individual decision tree is constructed for each sample. Each decision tree will generate an output, as shown in the figure. The final output is considered based on majority voting.

How Does Random Forest Work?

- **Example:**



Key Features of Random Forest(Advantages)

1. **Diversity:** Not all attributes/variables/features are considered while making an individual tree; each tree is different.
2. **Immune to the curse of dimensionality**: Since each tree does not consider all the features, the feature space is reduced.
3. **Parallelization:** Each tree is created independently out of different data and attributes. This means we can fully use the CPU to build random forests.

Key Features of Random Forest(Advantages)

4. **Stability/High Predictive Accuracy::** Stability arises because the result is based on majority voting/ averaging.
5. **Resistance to Overfitting:** Since each decision tree randomly selected data set for training, this prevents getting too caught up with the training data which makes the model less prone to overfitting.
6. **Large Datasets Handling:** As a team of decision trees is involved in decision making, each tree takes on a part of the dataset, ensuring that the expedition is not only thorough but also surprisingly quick.

Disadvantages of Random Forest

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

Decision Tree vs Random Forest

Aspect	Random Forest	Decision Tree
Nature	Ensemble of multiple decision trees	Single decision tree
Bias-Variance Trade-off	Lower variance, reduced overfitting	Higher variance, prone to overfitting
Predictive Accuracy	Generally higher due to ensemble	Prone to overfitting, may vary
Robustness	More robust to outliers and noise	Sensitive to outliers and noise
Training Time	Slower due to multiple tree construction	Faster as it builds a single tree
Interpretability	Less interpretable due to ensemble	More interpretable as a single tree
Feature Importance	Provides feature importance scores	Provides feature importance, but less reliable
Usage	Suitable for complex tasks, high-dimensional data	Simple tasks, easy interpretation

Important Hyperparameters in Random Forest

Hyperparameters to Increase the Predictive

- **max_features:** Maximum number of features random forest considers splitting a node.
- **mini_sample_leaf:** Determines the minimum number of leaves required to split an internal node.
- **criterion:** How to split the node in each tree? (Entropy/Gini impurity/Log Loss)
- **max_leaf_nodes:** Maximum leaf nodes in each tree

Important Hyperparameters in Random Forest

Hyperparameters to Increase the Speed.

- **n_jobs:** it tells the engine how many processors it is allowed to use. If the value is 1, it can use only one processor, but if the value is -1, there is no limit.
- **random_state:** controls randomness of the sample. The model will always produce the same results if it has a definite value of random state and has been given the same hyperparameters and training data.

Important Hyperparameters in Random Forest

Hyperparameters to Increase the Speed.

- **oob_score:** OOB means out of the bag. It is a random forest cross-validation method. In this, one-third of the sample is not used to train the data; instead used to evaluate its performance. These samples are called **out-of-bag samples**.

Applications of Random Forest

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

Python Implementation of Random Forest Algorithm

```
#Fitting Decision Tree classifier to the training set
from sklearn.ensemble import RandomForestClassifier
classifier= RandomForestClassifier(n_estimators= 10, criterion="entropy")
classifier.fit(x_train, y_train)
```

- **n_estimators=** The required number of trees in the Random Forest. The default value is 10. We can choose any number but need to take care of the overfitting issue.
- **criterion=** It is a function to analyze the accuracy of the split. Here we have taken "entropy" for the information gain.