What is Random forest?

A Random Forest is like a group decision-making team in machine learning.

It combines the opinions of many "trees" (individual models) to make better predictions, creating a more robust and accurate overall model.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables, as in the case of regression, and categorical variables, as in the case of classification.

It performs better for classification and regression tasks

Working of Random Forest Algorithm

Before understanding the working of the random forest algorithm in machine learning, we must look into the ensemble learning technique.

Ensemble simplymeans combining multiple models. Thus a collection of models is used to make predictions rather than an individual model.

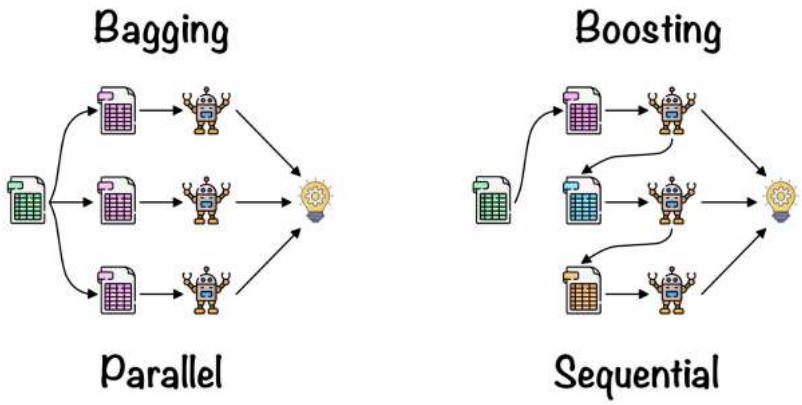
Ensemble uses two types of methods:

Bagging

It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest.

Boosting

It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.



Bagging Bagging, also known as Bootstrap Aggregation, serves as the ensemble technique in the Random Forest algorithm.

Here are the steps involved in Bagging:

Selection of Subset:

Bagging starts by choosing a random sample, or subset, from the entire dataset.

Bootstrap Sampling:

Each model is then created from these samples, called Bootstrap Samples, which are taken from the original data with replacement. This process is known as row sampling.

Bootstrapping:

The step of row sampling with replacement is referred to as bootstrapping.

Independent Model Training: Each model is trained independently on its corresponding Bootstrap Sample.

This training process generates results for each model.

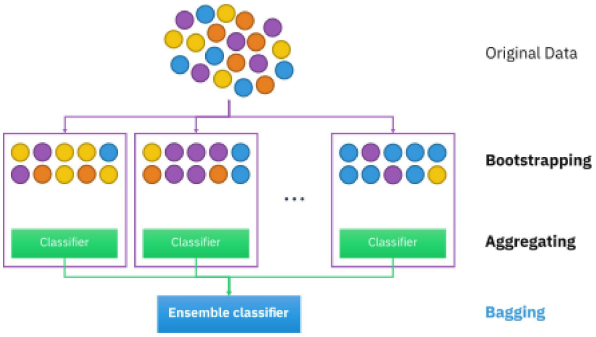
Majority Voting:

The final output is determined by combining the results of all models through majority voting.

The most commonly predicted outcome among the models is selected.

Aggregation:

This step, which involves combining all the results and generating the final output based on majority voting, is known as aggregation.



Importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
df = pd.read_csv('/content/audi.csv')
X = df.iloc[:,[0,1,3,4,5,6,7,8]].values
Y = df.iloc[:,[2]].values
```

```
print(X)
```

```
print(Y)
```

Data Preprocessing

Label Encoding

```
from sklearn.preprocessing import LabelEncoder
le1 = LabelEncoder()
X[:,0] = le1.fit_transform(X[:,0])
le2 = LabelEncoder()
X[:,-4] = le2.fit_transform(X[:,-4])
```

```
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
ct = ColumnTransformer(transformers = [('encoder',OneHotEncoder(),[2])],remainder='passthrough')
X = ct.fit_transform(X)
```

```
print(X)
```

Feature Scaling

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
```

```
print(X)
```

```
[[-0.58326752  1.2007284  -0.71233307 ...  0.35714729  0.35755001
  -0.88021837]
 [ 1.71447913 -0.83282781 -0.71233307 ... -1.57832278  1.03713001
   0.11492465]
 [-0.58326752  1.2007284  -0.71233307 ... -1.42944047  0.35755001
  -0.88021837]
 ...
 [-0.58326752  1.2007284  -0.71233307 ...  0.35714729 -0.09035499
  -1.54364705]
 [ 1.71447913 -0.83282781 -0.71233307 ...  0.35714729 -0.22163749
  -0.88021837]
 [-0.58326752  1.2007284  -0.71233307 ...  0.35714729 -0.22163749
  -0.88021837]]
```

Splitting Dataset into Training set and Test set

```
from sklearn.model_selection import train_test_split
(X_train,X_test,Y_train,Y_test) = train_test_split(X,Y,test_size=0.2,random_state=0)
```

Training Model

```
from sklearn.ensemble import RandomForestRegressor
regression = RandomForestRegressor(random_state=0)
regression.fit(X_train,Y_train)
```

```
/usr/local/lib/python3.12/dist-packages/sklearn/base.py:1389: DataConversionWarning: A column-vector y was passed when a 1d array w
  return fit_method(estimator, *args, **kwargs)
```

RandomForestRegressor

```
RandomForestRegressor(random_state=0)
```

```
y_pred = regression.predict(X_test)
```

Testing result

```
print(np.concatenate((y_pred.reshape(len(y_pred),1),Y_test.reshape(len(Y_test),1)),1))
```

```
[[14337.15 14998.  ]
 [23450.35 21950.  ]
 [27330.07 28990.  ]
 ...
 [46275.18 45995.  ]
 [31359.   30500.  ]
 [ 9929.62  8400.  ]]
```

Calculating Accuracy

```
from sklearn.metrics import r2_score,mean_absolute_error
r2_score(Y_test, y_pred)
```

```
0.9536134841307546
```

```
mean_absolute_error(Y_test,y_pred)
```

```
1538.730980670462
```

Double-click (or enter) to edit

```
print(y_pred)
```

```
[14337.15 23450.35 27330.07 ... 46275.18 31359.    9929.62]
```

```
print(Y_test)
```

```
[[14998]
 [21950]
 [28990]
 ...
 [45995]
 [30500]
 [ 8400]]
```

```
y_pred = np.reshape(y_pred,(-1,1))
```

Making Pandas DataFrame

```
mydata = np.concatenate((Y_test,y_pred),axis=1)
dataframe = pd.DataFrame(mydata,columns=['Real Price','Predicted Price'])
```

```
print(dataframe)
```

```
      Real Price  Predicted Price
0        14998.0         14337.15
1        21950.0         23450.35
2        28990.0         27330.07
3        25489.0         27200.98
4        30950.0         32250.05
...          ...              ...
2129     23700.0         39147.77
2130     18000.0         16679.95
2131     45995.0         46275.18
2132     30500.0         31359.00
2133      8400.0          9929.62

[2134 rows x 2 columns]
```

Decision trees	**Random Forest**
1. Decision trees normally suffer from the problem of overfitting if it's allowed to grow without any control.	1. Random forests are created from subsets of data, and the final output is based on average or majority ranking; hence the problem of overfitting is taken care of.
2. A single decision tree is faster in computation.	2. It is comparatively slower.
3. When a data set with features is taken as input by a decision tree, it will formulate some rules to make predictions.	3. Random forest randomly selects observations, builds a decision tree, and takes the average result. It doesn't use any set of formulas.