MANJARA CHARITABLE TRUST

RAJIV GANDHI INSTITUTE OF TECHNOLOGY, MUMBAI

Academic Year: 2022-2023 **Semester: VII**

Student Name: _____ Roll No:_____

Subject: **Blockchain** Class/Div: **BE/A**

Experiment No.:_____

Title of Experiment:

_____

Experiment Evaluation:

Experiment No.	Performance/ Coding (04)	Punctuality (03)	Question-Answer (03)	**Total (10)**
1				

Date of Experiment:

Date of Submission:

Staff Signature

Date: 11/09/23

Roll No: A762

EXPERIMENT NO: 06

Aim: To implement Random Forest Algorithm.

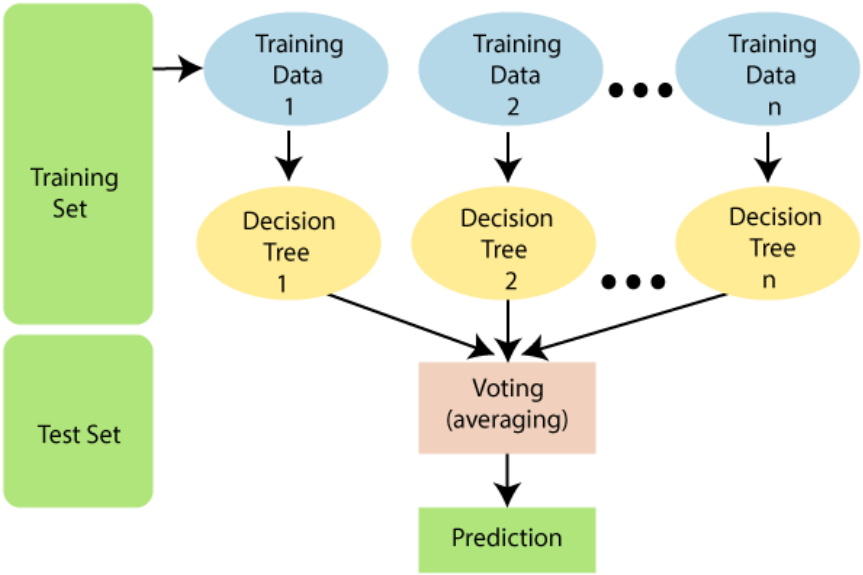
Theory:

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.*

As the name suggests, ***"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."*** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:



Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all

the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Why use Random Forest?

Below are some points that explain why we should use the Random Forest algorithm: ○ It takes less training time as compared to other algorithms. ○ It predicts output with high accuracy, even for the large dataset it runs efficiently. ○ It can also maintain accuracy when a large proportion of data is missing.

How does Random Forest algorithm work?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

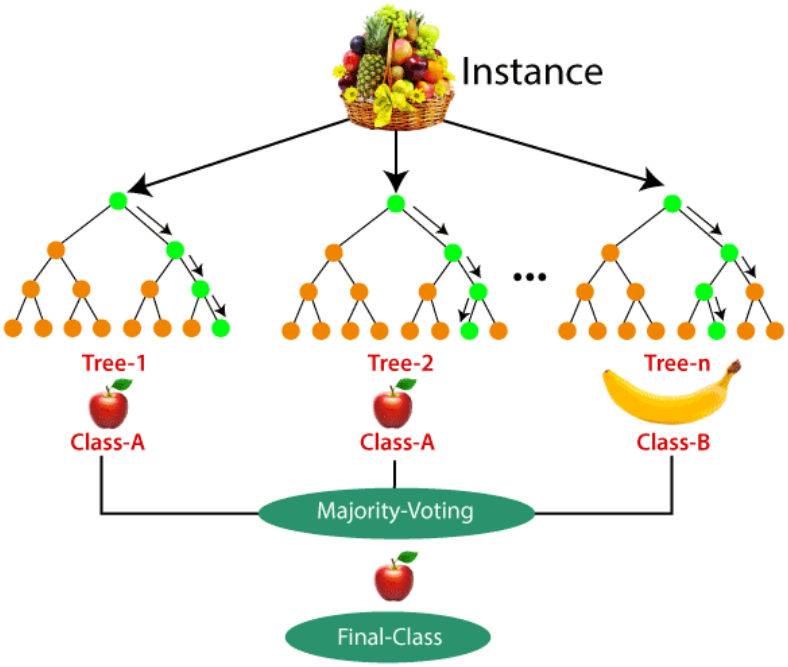
Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

The working of the algorithm can be better understood by the below example:

Example: Suppose there is a dataset that contains multiple fruit images. So, this dataset is given to the Random forest classifier. The dataset is divided into subsets and given to each decision tree. During the training phase, each decision tree produces a prediction result, and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision. Consider the below image:



Applications of Random Forest:

There are mainly four sectors where Random forest mostly used:

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

Advantages of Random Forest:

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality. ○ It enhances

the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest: ○ Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

Code:

```
#Import scikit-learn dataset library from
sklearn import datasets
#Load dataset iris=datasets.load_iris()
# print the label species (setosa, versicolor, virginica) print
(iris.target_names)
# print the names of the four features
print (iris.feature_names) # print the
iris data (top 5 records) print
(iris.data[0:5])
# print the iris labels (0:setosa, 1: versicolor, 2: virginica)
print (iris.target) import pandas as pd
data=pd.DataFrame({
'sepal length': iris.data[:,0],
'sepal width': iris.data[:,1],
'petal length': iris.data[:,2],
'petal width':iris.data[:,3],
'species': iris.target
}) data.head()
from sklearn.model_selection import train_test_split
x=data[['sepal length', 'sepal width', 'petal length', 'petal width']] # Features  y=data['species']
# Labels
#Split dataset into training set and test set
```

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3)
#Import Random Forest Model
from sklearn.ensemble import RandomForestClassifier
#Create a Gaussian Classifier
clf=RandomForestClassifier (n_estimators=100)
#Train the model using the training sets y_pred-clf.predict(X_test)
clf.fit(x_train, y_train) y_pred=clf.predict(x_test)
#Import scikit-learn metrics module for accuracy calculation  from
sklearn import metrics
# Model Accuracy, how often is the classifier correct?
print("Accuracy: ",metrics.accuracy_score (y_test, y_pred))  new_data
= [[3, 5, 4, 2]]  # New data point
new_data_df = pd.DataFrame(new_data, columns=['sepal length', 'sepal width', 'petal length',
'petal width'])
ans = clf.predict(new_data_df)
if ans[0] == 0:
print('setosa') elif ans[0] == 1:
   print('versicolor') else:
   print('virginica')
```

Output:

```
['setosa' 'versicolor' 'virginica']
['sepal length (cm)', 'sepal width (cm)', 'petal length (cm)', 'petal width (cm)']
[[5.1 3.5 1.4 0.2]
 [4.9 3.  1.4 0.2]
 [4.7 3.2 1.3 0.2]
 [4.6 3.1 1.5 0.2]
 [5.  3.6 1.4 0.2]]
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2
 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 2 2]
Accuracy:  0.9111111111111111
virginica
```

Conclusion:

Hence, we implemented the Random Forest Algorithm in Python.