Experiment No. 4
Apply Random Forest Algorithm on Adult Census Income Dataset and analyze the performance of the model
Date of Performance:
Date of Submission:

Aim: Apply Random Forest Algorithm on Adult Census Income Dataset and analyze the performance of the model.

Objective: Able to perform various feature engineering tasks, apply Random Forest Algorithm on the given dataset and maximize the accuracy, Precision, Recall, F1 score.

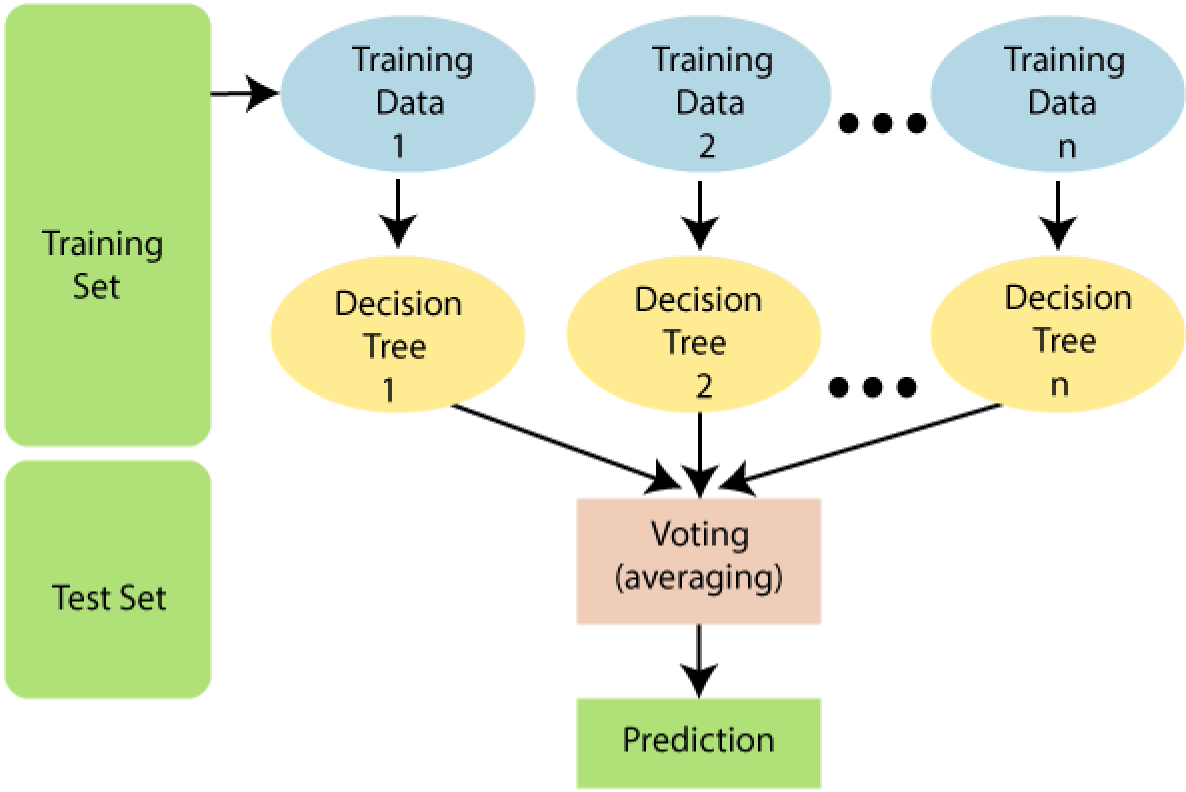
Theory:

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:



Dataset:

Predict whether income exceeds $50K/yr based on census data. Also known as "Adult" dataset.

Attribute Information:

Listing of attributes:

>50K, <=50K.

age: continuous.

workclass: Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.

fnlwgt: continuous.

education: Bachelors, Some-college, 11th, HS-grad, Prof-school, Assoc-acdm, Assoc-voc, 9th, 7th-8th, 12th, Masters, 1st-4th, 10th, Doctorate, 5th-6th, Preschool.

education-num: continuous.

marital-status: Married-civ-spouse, Divorced, Never-married, Separated, Widowed, Married-spouse-absent, Married-AF-spouse.

occupation: Tech-support, Craft-repair, Other-service, Sales, Exec-managerial, Prof-specialty, Handlers-cleaners, Machine-op-inspct, Adm-clerical, Farming-fishing, Transport-moving, Priv-house-serv, Protective-serv, Armed-Forces.

relationship: Wife, Own-child, Husband, Not-in-family, Other-relative, Unmarried.

race: White, Asian-Pac-Islander, Amer-Indian-Eskimo, Other, Black.

sex: Female, Male.

capital-gain: continuous.

capital-loss: continuous.

hours-per-week: continuous.

native-country: United-States, Cambodia, England, Puerto-Rico, Canada, Germany, Outlying-US(Guam-USVI-etc), India, Japan, Greece, South, China, Cuba, Iran, Honduras, Philippines, Italy, Poland, Jamaica, Vietnam, Mexico, Portugal, Ireland, France, Dominican-Republic, Laos, Ecuador, Taiwan, Haiti, Columbia, Hungary, Guatemala, Nicaragua, Scotland, Thailand, Yugoslavia, El-Salvador, Trinadad&Tobago, Peru, Hong, Holland-Netherlands.

Code & Output:

```
import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.preprocessing import LabelEncoder

from sklearn.metrics import classification_report, accuracy_score



df = pd.read_csv('adult.csv')



print(df.head())
```

```
   age  workclass  fnlwgt     education  educational-num      marital-status  \
0   25    Private  226802          11th                7       Never-married
1   38    Private   89814       HS-grad                9  Married-civ-spouse
2   28  Local-gov  336951    Assoc-acdm               12  Married-civ-spouse
3   44    Private  160323  Some-college               10  Married-civ-spouse
4   18          ?  103497  Some-college               10       Never-married

          occupation relationship   race  gender  capital-gain  capital-loss  \
0  Machine-op-inspct    Own-child  Black    Male             0             0
1    Farming-fishing      Husband  White    Male             0             0
2    Protective-serv      Husband  White    Male             0             0
3  Machine-op-inspct      Husband  Black    Male          7688             0
4                  ?    Own-child  White  Female             0             0

   hours-per-week native-country income
0              40  United-States  <=50K
1              50  United-States  <=50K
2              40  United-States   >50K
3              40  United-States   >50K
4              30  United-States  <=50K
```

```
df.columns = ['age', 'workclass', 'fnlwgt', 'education', 'education-num', 'marital-status',

                         'occupation', 'relationship', 'race', 'sex', 'capital-gain', 'capital-loss',

             'hours-per-week', 'native-country', 'income']
```

```
df.replace(' ?', pd.NA, inplace=True)

df.dropna(inplace=True)


categorical_columns = ['workclass', 'education', 'marital-status', 'occupation','relationship', 'race', 'sex', 'native-country', 'income']

label_encoders = {}

for col in categorical_columns:

   le = LabelEncoder()

   df[col] = le.fit_transform(df[col])

   label_encoders[col] = le


print(df)
```

```
       age  workclass  fnlwgt  education  education-num  marital-status  \
0       25          4  226802          1              7               4
1       38          4   89814         11              9               2
2       28          2  336951          7             12               2
3       44          4  160323         15             10               2
4       18          0  103497         15             10               4
...    ...        ...     ...        ...            ...             ...
48837   27          4  257302          7             12               2
48838   40          4  154374         11              9               2
48839   58          4  151910         11              9               6
48840   22          4  201490         11              9               4
48841   52          5  287927         11              9               2

       occupation  relationship  race  sex  capital-gain  capital-loss  \
0               7             3     2    1             0             0
1               5             0     4    1             0             0
2              11             0     4    1             0             0
3               7             0     2    1          7688             0
4               0             3     4    0             0             0
...           ...           ...   ...  ...           ...           ...
48837          13             5     4    0             0             0
48838           7             0     4    1             0             0
48839           1             4     4    0             0             0
48840           1             3     4    1             0             0
48841           4             5     4    0         15024             0
```

```
X = df.drop('income', axis=1)

y = df['income']


X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


rf = RandomForestClassifier(n_estimators=100, random_state=42)

rf.fit(X_train, y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
from sklearn.tree import export_graphviz

import pydotplus

from IPython.display import Image, display


y_pred = rf.predict(X_test)
```

```
print(f"Accuracy: {accuracy_score(y_test, y_pred)}")

print(f"ClassificationReport:\n{classification_report(y_test,
y_pred)}")
```

```
Accuracy: 0.8639574163169209
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.93      0.91      7479
           1       0.74      0.64      0.69      2290

    accuracy                           0.86      9769
   macro avg       0.82      0.79      0.80      9769
weighted avg       0.86      0.86      0.86      9769
```

Conclusion:

The Random Forest model achieved an accuracy of 0.864, indicating strong overall performance in predicting income levels. It exhibited high precision (0.89) and recall (0.93) for <=50K incomes, demonstrating its ability to accurately identify lower income brackets. Conversely, predictions for >50K incomes showed slightly lower precision (0.74) and recall (0.64), resulting in a moderate F1-score of 0.69. Overall, with balanced performance metrics across precision, recall, and F1-score, the model proved effective in leveraging demographic and socio-economic features to predict income levels reliably.