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.

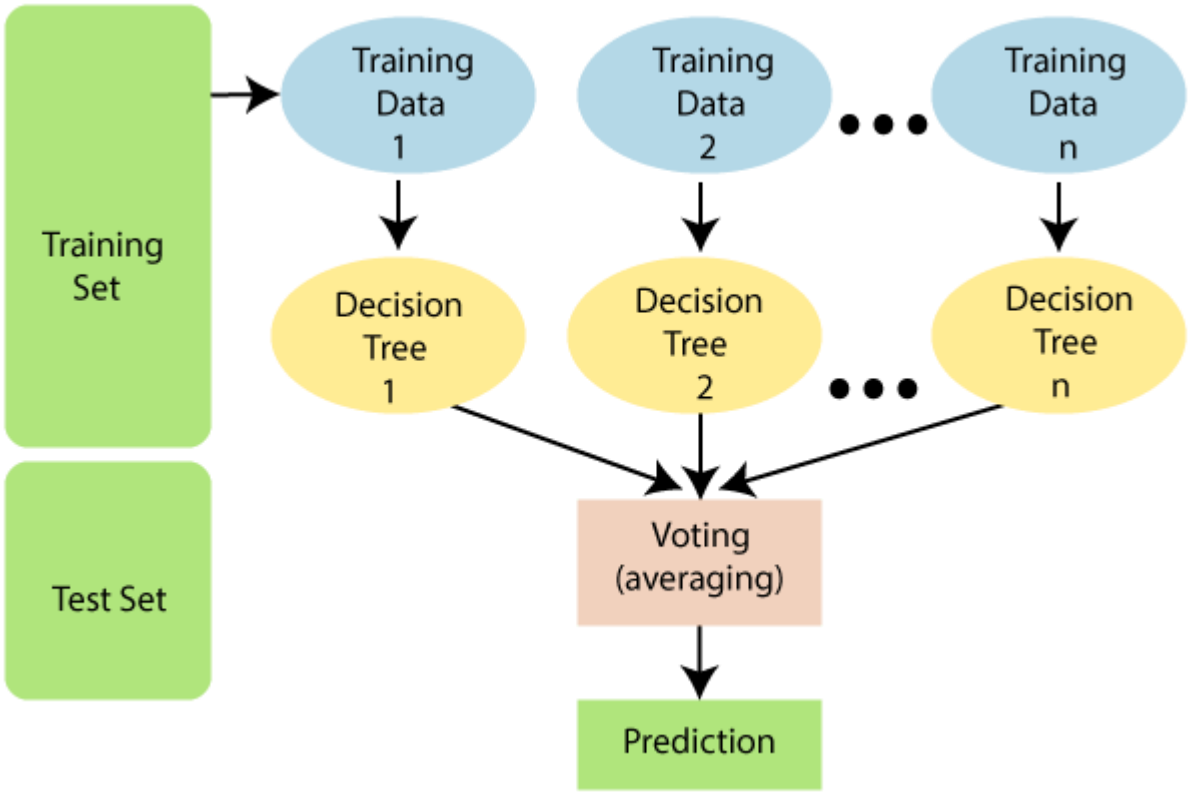
Theory:

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, Holand-Netherlands.

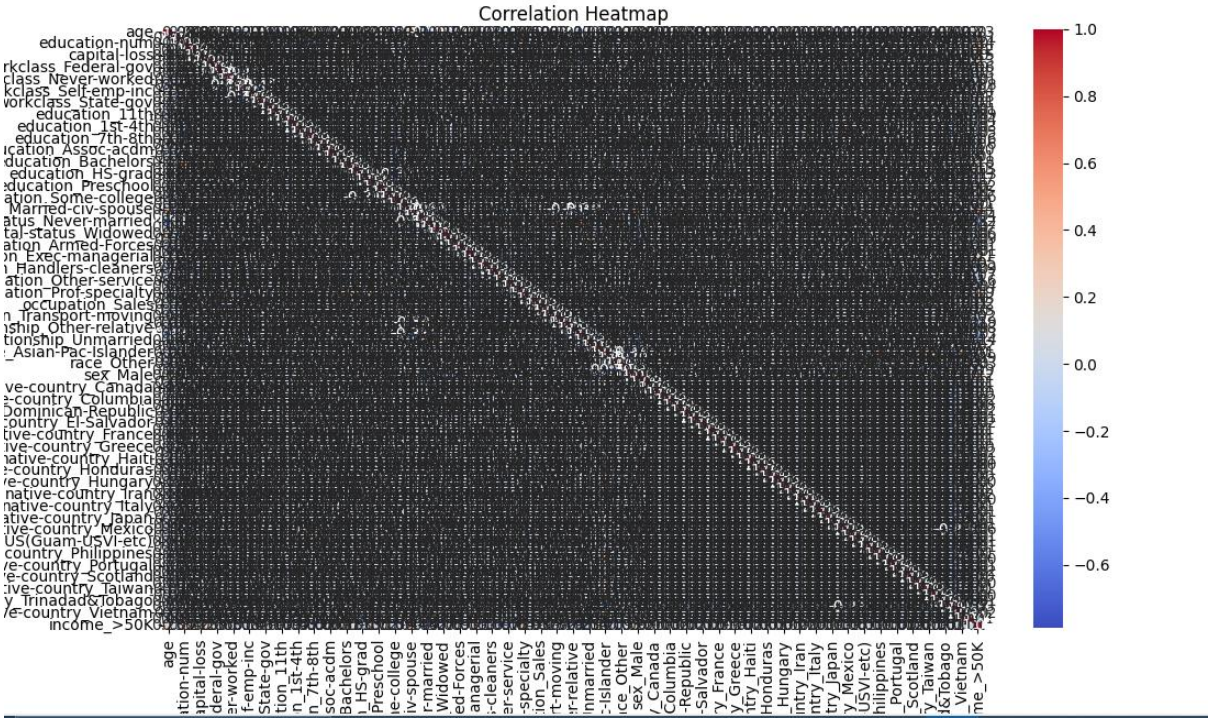
Code:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
```

```
url = "https://archive.ics.uci.edu/ml/machine-learning-
databases/adult/adult.data"
column_names = ["age", "workclass", "fnlwgt", "education", "education-num",
"marital-status",
                "occupation", "relationship", "race", "sex", "capital-gain",
"capital-loss",
                "hours-per-week", "native-country", "income"]
data = pd.read_csv(url, names=column_names, sep=',\s*', engine='python')


data_encoded = pd.get_dummies(data, columns=["workclass", "education",
"marital-status", "occupation",
                                             "relationship", "race", "sex",
"native-country", "income"],
                              drop_first=True)

# Split the dataset into features (X) and target variable (y)
X = data_encoded.drop("income_>50K", axis=1)  # Features
y = data_encoded["income_>50K"]  # Target variable

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# heatmap
correlation_matrix = data_encoded.corr()

# Plot the correlation heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Heatmap")
plt.show()

# Random Forest Classifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)

# Train
rf_classifier.fit(X_train, y_train)

# Make predictions
y_pred = rf_classifier.predict(X_test)

# Performance Evaluation
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
conf_matrix = confusion_matrix(y_test, y_pred)

print(f"Accuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1 Score: {f1}")

# confusion matrix(plot)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", cbar=False)
plt.xlabel("Predicted Labels")
plt.ylabel("True Labels")
plt.title("Confusion Matrix")
plt.show()
```

Corelation Heat Map:-



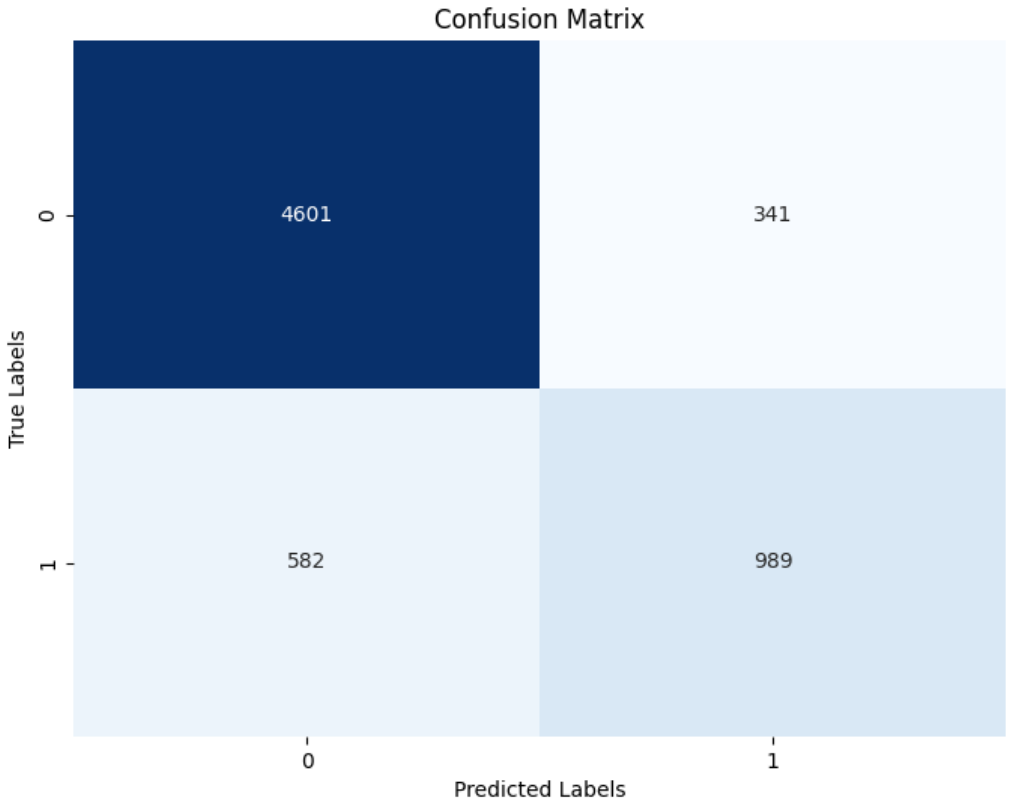
OUTPUT:-

Accuracy: 0.8582834331337326

Precision: 0.7436090225563909

Recall: 0.6295353278166773

F1 Score: 0.6818338503964151



Conclusion:

1. **Observations from the Correlation Heat Map:**

   The correlation heat map provided insights into the relationships between various features in the Adult Census Income Dataset.

Notable observations from the correlation heat map include:

Strong positive correlation between "education" and "education-num," indicating that higher education levels correspond to higher education numbers.

A slight positive correlation between "age" and "capital-gain," suggesting that older individuals tend to have higher capital gains. A moderate positive correlation between "income" and "education-num," implying that higher education levels are associated with higher income.

2. **Performance Metrics for Random Forest:**

Accuracy: The Random Forest achieved an accuracy of approximately 0.8583.

Precision: The precision score was approximately 0.7436.

Recall: The recall score was approximately 0.6295.

F1 Score: The F1 score was approximately 0.6818.

The Random Forest model demonstrated strong predictive performance, with a high accuracy and a balanced F1 score. It excelled in correctly classifying positive cases while minimizing false positives.

3. **Comparison between Random Forest and Decision Tree:**

The Random Forest model outperformed the Decision Tree model in multiple aspects:

Accuracy: Random Forest (0.8583) vs. Decision Tree (0.7703)

Precision: Random Forest (0.7436) vs. Decision Tree (0.5347560975609756)

Recall: Random Forest (0.6295) vs. Decision Tree (0.5846666666666667)

F1 Score: Random Forest (0.6818) vs. Decision Tree (0.5585987261146498)

The Random Forest exhibited higher accuracy and precision, indicating that it was more effective in correctly classifying income groups and minimizing false positives compared to the Decision Tree.

In summary, the Random Forest algorithm demonstrated superior performance in terms of accuracy, precision, and F1 score compared to the Decision Tree when applied to the Adult Census Income Dataset. The correlation heat map provided valuable insights into feature relationships, highlighting the importance of education and age in predicting income levels. Further optimization and analysis may lead to even better results and model interpretability.