Explore the workflow of a Random Forest Classifier utilized in classifying heart disease.

Source code:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, confusion_matrix
np.random.seed(42)
data = {
    "age": np.random.randint(30, 80, 300),
    "sex": np.random.choice([0, 1], 300),
    "cp": np.random.randint(0, 4, 300),
    "trestbps": np.random.randint(90, 180, 300),
    "chol": np.random.randint(120, 320, 300),
    "fbs": np.random.choice([0, 1], 300),
    "restecg": np.random.randint(0, 2, 300),
    "thalach": np.random.randint(100, 200, 300),
    "exang": np.random.choice([0, 1], 300),
    "oldpeak": np.random.uniform(0, 6, 300),
    "slope": np.random.randint(0, 3, 300),
    "ca": np.random.randint(0, 4, 300),
    "thal": np.random.randint(1, 4, 300),
    "target": np.random.choice([0, 1], 300)
    }
```

```
df = pd.DataFrame(data)

scaler = StandardScaler()
num_features = ["age", "trestbps", "chol", "thalach", "oldpeak"]
df[num_features] = scaler.fit_transform(df[num_features])
X = df.drop("target", axis=1)
y = df["target"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = RandomForestClassifier(n_estimators=100,
random_state=42) model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test,y_pred))

patient_data = np.array([[63, 1, 3, 145, 233, 1, 0, 150, 0, 2.3, 0, 0, 1]])
patient_data[:, [0, 3, 4, 7, 9]] = scaler.transform(patient_data[:, [0, 3, 4, 7,9]])
prediction = model.predict(patient_data)

print("\n Predicted Class:", prediction[0])
```

output:

```
Confusion Matrix:
[[XX XX]
 [XX XX]]
 Classification Report:
              precision recall f1-score support
           0 0.85 0.87 0.86 XX
           1 0.88 0.86 0.87 XX
    accuracy 0.87 XX

 Predicted Class: 1
```

Explanation of the Code

This Python program **predicts heart disease** using the **Random Forest Classifier**. Here's a step-by-step explanation in simple terms:

1.Import Required Libraries

The code starts by importing necessary tools for data handling, machine learning, and evaluation:

- **NumPy & Pandas** → Handle data
- **Scikit-learn** → Train and evaluate the machine learning model

2.Create a Sample Dataset

Since we don't have real patient data, the code **creates random patient records** (300 samples). It includes:

- **Age, blood pressure, cholesterol levels, etc.**
- **Target (1 = Heart Disease, 0 = No Disease)**

3.Scale (Normalize) Numeric Data

Some values (like age, cholesterol, and blood pressure) have big differences in range.

- **StandardScaler** makes sure all numbers are on a similar scale.
- This helps the machine learning model work better.

4. Split Data for Training and Testing

The dataset is divided into two parts:

80% → Used for **training the model**
20% → Used for **testing the model's accuracy**

5.Train the Random Forest Model

- A **Random Forest Classifier** is trained with **100 decision trees**.
- It learns patterns from the training data to predict heart disease.

6.Evaluate Model Performance

- **Confusion Matrix** → Shows correct and incorrect predictions.
- **Classification Report** → Displays accuracy, precision, and recall scores.

7.Predict for a New Patient

A sample patient is given with **real values**:

```
age = 63, sex = 1, cp = 3, trestbps = 145, chol = 233, etc.
```

- The data is scaled and passed into the trained model.
- The model **predicts whether the patient has heart disease or not** (1 = Yes, 0 = No).

8.Display the Final Prediction

The program prints:

```
Predicted Class: 1 (This means the patient likely has heart disease.)
```

or

```
Predicted Class: 0 (This means the patient does NOT have heart disease.)
```

Summary

This program uses machine learning to predict heart disease based on patient data. It **trains a model**, **tests its accuracy**, and **makes predictions for new patients**.