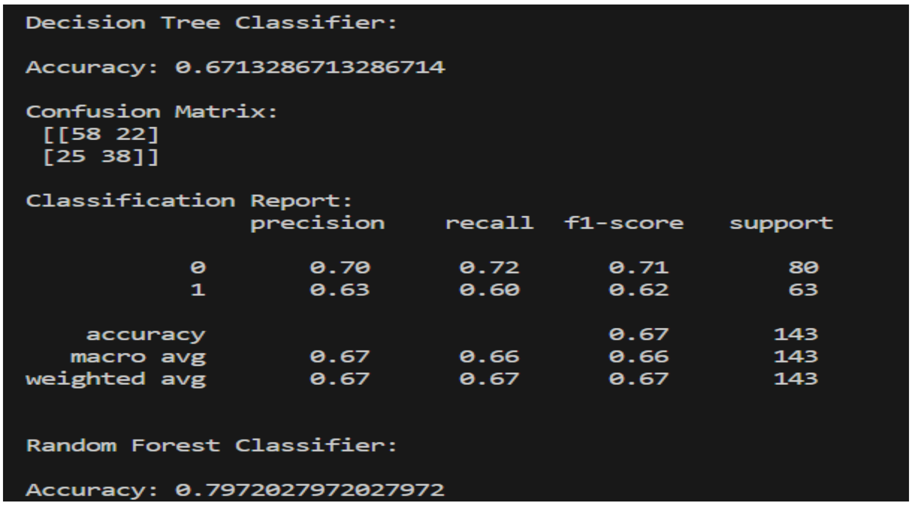
Ex 04

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

# Load the dataset
data = pd.read_csv('D:\\AML\\titanic dataset.csv')

# Display the first few rows of the dataset
print("First few rows of the dataset:")
print(data.head())

# Display information about the dataset
print("\nDataset info:")
print(data.info())

# Check for missing values
print("\nMissing values:")
print(data.isnull().sum())

# Handle missing values
data = data.drop(['Cabin'], axis=1)  # Drop 'Cabin' due to too many missing values
data = data.dropna()  # Drop other rows with missing values

# Encode categorical variables
le = LabelEncoder()
for column in data.select_dtypes(include=['object']).columns:
    data[column] = le.fit_transform(data[column])

# Define features and target variable
target_column = 'Survived'
features = data.drop(target_column, axis=1)
target = data[target_column]

# Normalize/scale the features
scaler = StandardScaler()
features_scaled = scaler.fit_transform(features)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features_scaled, target, test_size=0.2,
random_state=42)

# Initialize and train the Decision Tree Classifier
```

```
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)

# Predict and evaluate the Decision Tree Classifier
y_pred_dt = dt_model.predict(X_test)
accuracy_dt = accuracy_score(y_test, y_pred_dt)
conf_matrix_dt = confusion_matrix(y_test, y_pred_dt)
class_report_dt = classification_report(y_test, y_pred_dt)

print("\nDecision Tree Classifier:")
print("\nAccuracy:", accuracy_dt)
print("\nConfusion Matrix:\n", conf_matrix_dt)
print("\nClassification Report:\n", class_report_dt)

# Initialize and train the Random Forest Classifier
rf_model = RandomForestClassifier(random_state=42)
rf_model.fit(X_train, y_train)

# Predict and evaluate the Random Forest Classifier
y_pred_rf = rf_model.predict(X_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
conf_matrix_rf = confusion_matrix(y_test, y_pred_rf)
class_report_rf = classification_report(y_test, y_pred_rf)

print("\nRandom Forest Classifier:")
print("\nAccuracy:", accuracy_rf)
print("\nConfusion Matrix:\n", conf_matrix_rf)
print("\nClassification Report:\n", class_report_rf)

# Compare the performances
print("\nComparison of Models:")
print(f"Decision Tree Accuracy: {accuracy_dt}")
print(f"Random Forest Accuracy: {accuracy_rf}")
```

Output:



```
Decision Tree Classifier:

Accuracy: 0.6713286713286714

Confusion Matrix:
 [[58 22]
 [25 38]]

Classification Report:
               precision    recall  f1-score   support

           0       0.70      0.72      0.71        80
           1       0.63      0.60      0.62        63

    accuracy                           0.67       143
   macro avg       0.67      0.66      0.66       143
weighted avg       0.67      0.67      0.67       143


Random Forest Classifier:

Accuracy: 0.7972027972027972
```