In [1]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
```

In [2]:
```
# Load the Titanic dataset
url = "https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv"
titanic_data = pd.read_csv(url)
```

In [3]:
```
# Drop unnecessary columns
titanic_data = titanic_data.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1)
```

In [4]:
```
# Handle missing values
titanic_data['Age'].fillna(titanic_data['Age'].median(), inplace=True)
titanic_data['Embarked'].fillna(titanic_data['Embarked'].mode()[0], inplace=True)
```

In [5]:
```
# Convert categorical variables to numerical
titanic_data['Sex'] = titanic_data['Sex'].map({'male': 0, 'female': 1})
titanic_data = pd.get_dummies(titanic_data, columns=['Embarked'], drop_first=True)
```

In [6]:
```
# Split the data into features (X) and target variable (y)
X = titanic_data.drop('Survived', axis=1)
y = titanic_data['Survived']
```

In [7]:
```
# Split the data 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
```

In [8]:
```
# Logistic Regression
logreg = LogisticRegression()
logreg.fit(X_train, y_train)
```

```
C:\Users\squir\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:460: Conve
rgenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.or
g/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (http
s://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[8]:
```
LogisticRegression()
```

In [9]:
```
# Predictions on the test set
y_pred = logreg.predict(X_test)
```

In [10]:
```
# Model evaluation
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
classification_report_str = classification_report(y_test, y_pred)

print("Logistic Regression Results:")
print(f"Accuracy: {accuracy}")
print("Confusion Matrix:")
print(conf_matrix)
print("Classification Report:")
print(classification_report_str)
```

```
Logistic Regression Results:
Accuracy: 0.8044692737430168
Confusion Matrix:
[[90 15]
 [20 54]]
Classification Report:
              precision    recall  f1-score   support

           0       0.82      0.86      0.84       105
           1       0.78      0.73      0.76        74

    accuracy                           0.80       179
   macro avg       0.80      0.79      0.80       179
weighted avg       0.80      0.80      0.80       179
```

In [11]:
```
# Regression Model (Random Forest Regressor)
regressor = RandomForestRegressor()
regressor.fit(X_train, y_train)
```

Out[11]:
```
▾ RandomForestRegressor

RandomForestRegressor()
```

In [12]:
```
# Predictions on the test set
y_pred_reg = regressor.predict(X_test)
```

In [14]:
```
# Model evaluation
accuracy = accuracy_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred_reg)
print("\nRandom Forest Regressor Results:")
print(f"Accuracy: {accuracy}")
print(f"Mean Squared Error: {mse}")
```

```
Random Forest Regressor Results:
Accuracy: 0.8044692737430168
Mean Squared Error: 0.14472265498079054
```