titanic-classification

July 22, 2023

[ ]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix,␣
↪classification_report
```

[ ]:
```
# Load the Titanic dataset
data = pd.read_csv('/content/titanic.csv')
```

[ ]:
```
# Preprocess the data
data = data.drop(['PassengerId', 'Name', 'Ticket', 'Cabin', 'Embarked'], axis=1)
data['Sex'] = data['Sex'].map({'female': 0, 'male': 1})
data['Age'].fillna(data['Age'].median(), inplace=True)
data['Fare'].fillna(data['Fare'].median(), inplace=True)
```

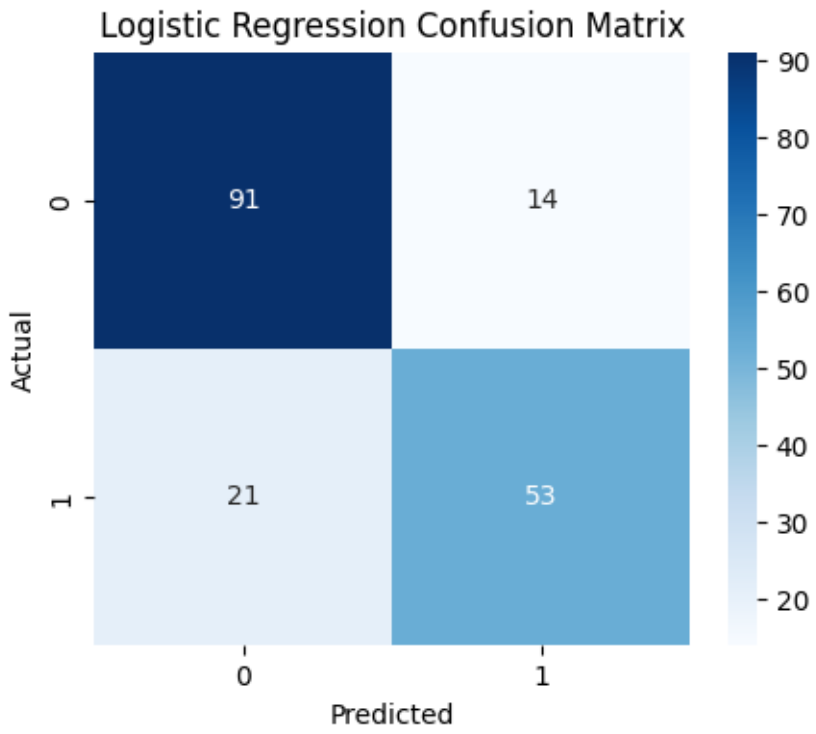
[ ]:
```
# Convert categorical features to one-hot encoding
data = pd.get_dummies(data, columns=['Pclass', 'SibSp', 'Parch'])
```

[ ]:
```
# Split the data into training and testing sets
X = data.drop('Survived', axis=1)
y = data['Survived']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

[ ]:
```
# Initialize and train multiple classification models
models = {
    'Logistic Regression': LogisticRegression(),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier()
}
```

[18]:
```python
for model_name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    # Calculate the accuracy of the model
    accuracy = accuracy_score(y_test, y_pred)
    print(f"{model_name} Accuracy:", accuracy,'\n')

    # Generate confusion matrix
    cm = confusion_matrix(y_test, y_pred)
    plt.figure(figsize=(5, 4))
    sns.heatmap(cm, annot=True, cmap='Blues', fmt='g')
    plt.title(f"{model_name} Confusion Matrix")
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
    plt.show()

    # Generate classification report
    report = classification_report(y_test, y_pred)
    print(f"{model_name} Classification Report:","\n")
    print(report,"\n")
    print("---------------+--------------------+------------------","\n")
```

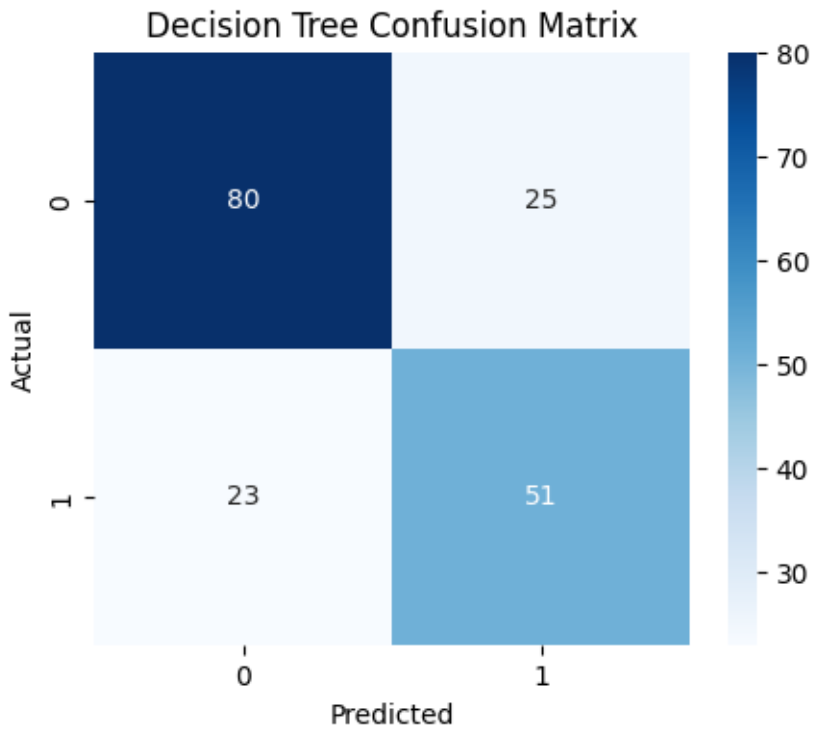
```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

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

Logistic Regression Accuracy: 0.8044692737430168
```



Logistic Regression Classification Report:

	precision	recall	f1-score	support
0	0.81	0.87	0.84	105
1	0.79	0.72	0.75	74
accuracy			0.80	179
macro avg	0.80	0.79	0.80	179
weighted avg	0.80	0.80	0.80	179

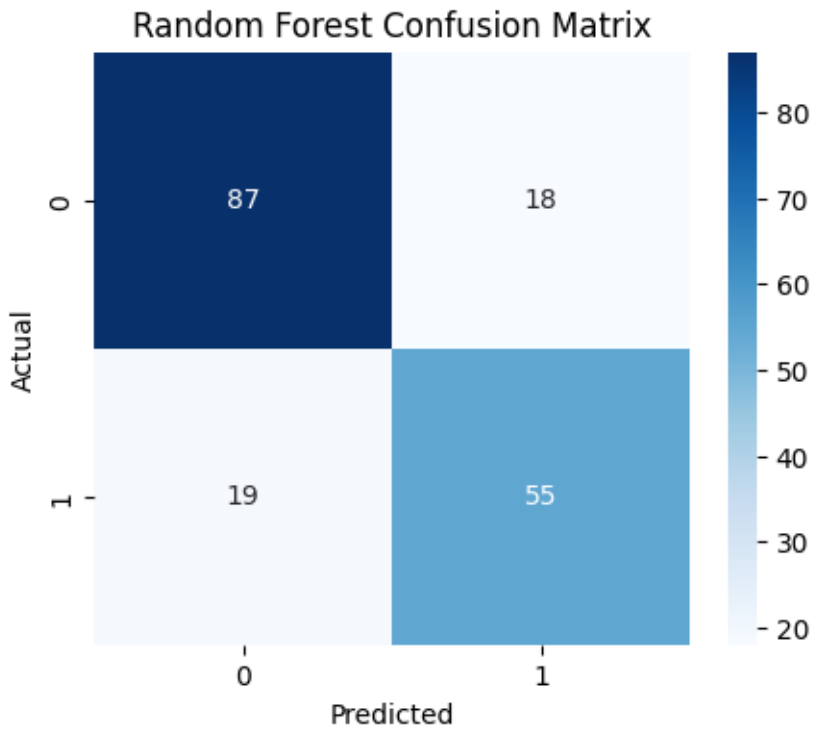
Decision Tree Accuracy: 0.7318435754189944



Decision Tree Classification Report:

	precision	recall	f1-score	support
0	0.78	0.76	0.77	105
1	0.67	0.69	0.68	74
accuracy			0.73	179
macro avg	0.72	0.73	0.72	179
weighted avg	0.73	0.73	0.73	179

Random Forest Accuracy: 0.7932960893854749




```
Random Forest Classification Report:

              precision    recall  f1-score   support

           0       0.82      0.83      0.82       105
           1       0.75      0.74      0.75        74

    accuracy                           0.79       179
   macro avg       0.79      0.79      0.79       179
weighted avg       0.79      0.79      0.79       179


---------------+--------------------+------------------
```

[ ]: