```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix, precision_score, recall_score, f1_score
import matplotlib.pyplot as plt
import seaborn as sns

path="/content/drive/MyDrive/Dataset/Admission_prediction.csv"
knn_df=pd.read_csv(path)

# Display first few rows of the dataset
print("Dataset Preview:")
print(df.head())
```

```
Dataset Preview:
   Serial No.  GRE_Score  TOEFL_Score  University Rating  SOP  LOR
CGPA  \
0           1      337.0        118.0                4.0  4.5  4.5
9.65
1           2      324.0        107.0                4.0  4.0  4.5
8.87
2           3        NaN        104.0                3.0  3.0  3.5
8.00
3           4      322.0        110.0                3.0  3.5  2.5
8.67
4           5      314.0        103.0                2.0  2.0  3.0
8.21

   Research  Chance_of_Admit
0         1             0.92
1         1             0.76
2         1             0.72
3         1             0.80
4         0             0.65
```

```
knn_df.fillna(knn_df.median(), inplace=True)

# Define the features (X) and target (y)
X_knn = knn_df.drop('Chance_of_Admit', axis=1)
y_knn = (knn_df['Chance_of_Admit']>= 0.5).astype(int)
X_knn.shape,y_knn.shape
```

```
((500, 8), (500,))
```

```
# Split the dataset into training and testing sets
X_train_knn, X_test_knn, y_train_knn, y_test_knn =
train_test_split(X_knn, y_knn, test_size=0.2, random_state=42)
X_train_knn.shape,X_test_knn.shape,y_test_knn.shape,y_train_knn.shape
```

```
((400, 8), (100, 8), (100,), (400,))
```

```
# Standardize the features
scaler_knn = StandardScaler()
X_train_knn = scaler_knn.fit_transform(X_train_knn)
X_test_knn = scaler_knn.transform(X_test_knn)

# Train the KNN classifier
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train_knn, y_train_knn)
```

```
KNeighborsClassifier()
```

```
y_pred_knn = knn.predict(X_test_knn)
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/
validation.py:2732: UserWarning: X has feature names, but
KNeighborsClassifier was fitted without feature names
  warnings.warn(
```

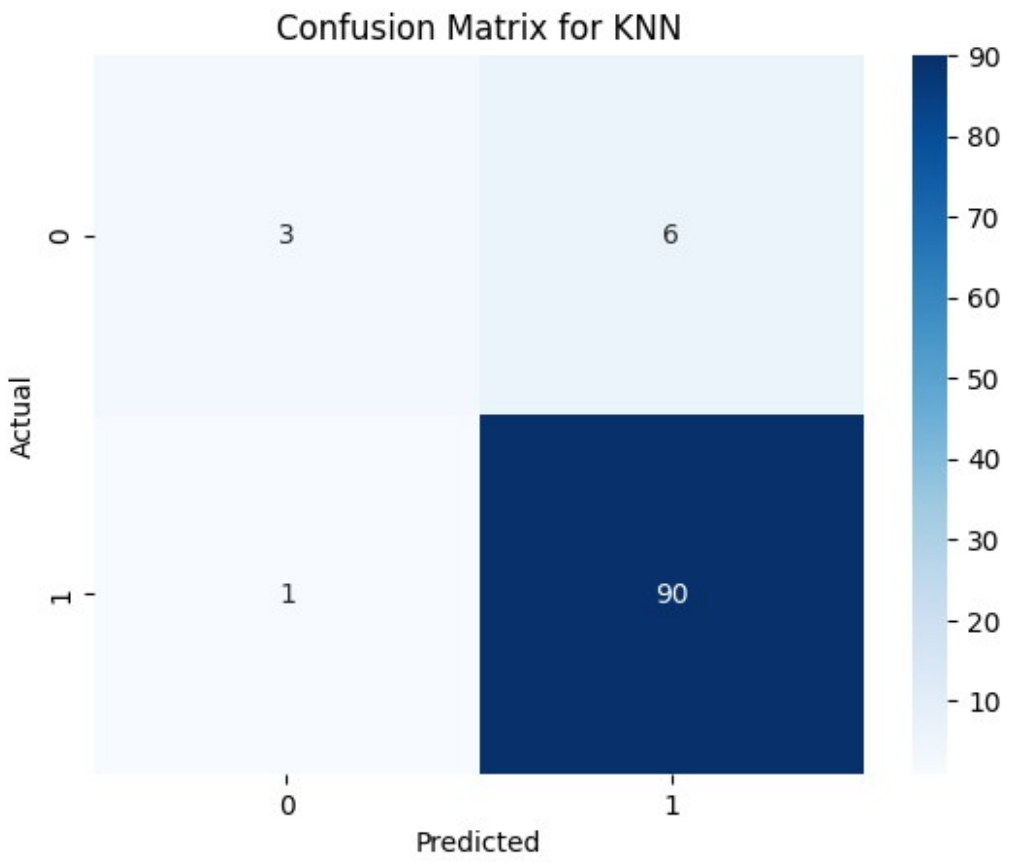
```
# Evaluate KNN
print("KNN Classification:")
print(f"Accuracy: {accuracy_score(y_test_knn, y_pred_knn)}")
print(f"Precision: {precision_score(y_test_knn, y_pred_knn)}")
print(f"Recall: {recall_score(y_test_knn, y_pred_knn)}")
print(f"F1 Score: {f1_score(y_test_knn, y_pred_knn)}")
print(classification_report(y_test_knn, y_pred_knn))
```

```
KNN Classification:
Accuracy: 0.93
Precision: 0.9375
Recall: 0.989010989010989
F1 Score: 0.9625668449197861
              precision    recall  f1-score   support

           0       0.75      0.33      0.46         9
           1       0.94      0.99      0.96        91

    accuracy                           0.93       100
   macro avg       0.84      0.66      0.71       100
weighted avg       0.92      0.93      0.92       100
```

```
# Confusion Matrix for KNN
cm_knn = confusion_matrix(y_test_knn, y_pred_knn)
sns.heatmap(cm_knn, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix for KNN')
plt.show()
```


Confusion Matrix for KNN
Actual
Predicted
0
1
3
6
1
90
90
80
70
60
50
40
30
20
10