Importing the Necessary Libraries

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

Loading the Dataset and Data Preprocessing

```
df = pd.read_csv('/content/Social_Network_Ads.csv')
```

```
df.head(1)
```

Splitting the data in train and test datasets

```
X = df[['Age', 'EstimatedSalary']]
y = df['Purchased']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Training the KNN model

```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train, y_train)
```

```
y_pred = knn.predict(X_test)
```

Evaluating model based on metrics

```
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", cm)
```

```
Confusion Matrix:
 [[48  4]
 [ 3 25]]
```

```
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.9125
```

```
error_rate = 1 - accuracy
print("Error Rate:", error_rate)
```

```
Error Rate: 0.08750000000000002
```

```
precision = precision_score(y_test, y_pred)
print("Precision:", precision)
```

```
Precision: 0.8620689655172413
```

```
recall = recall_score(y_test, y_pred)
print("Recall:", recall)
```

```
Recall: 0.8928571428571429
```

```
f1 = f1_score(y_test, y_pred)
print("F1 Score:", f1)
```

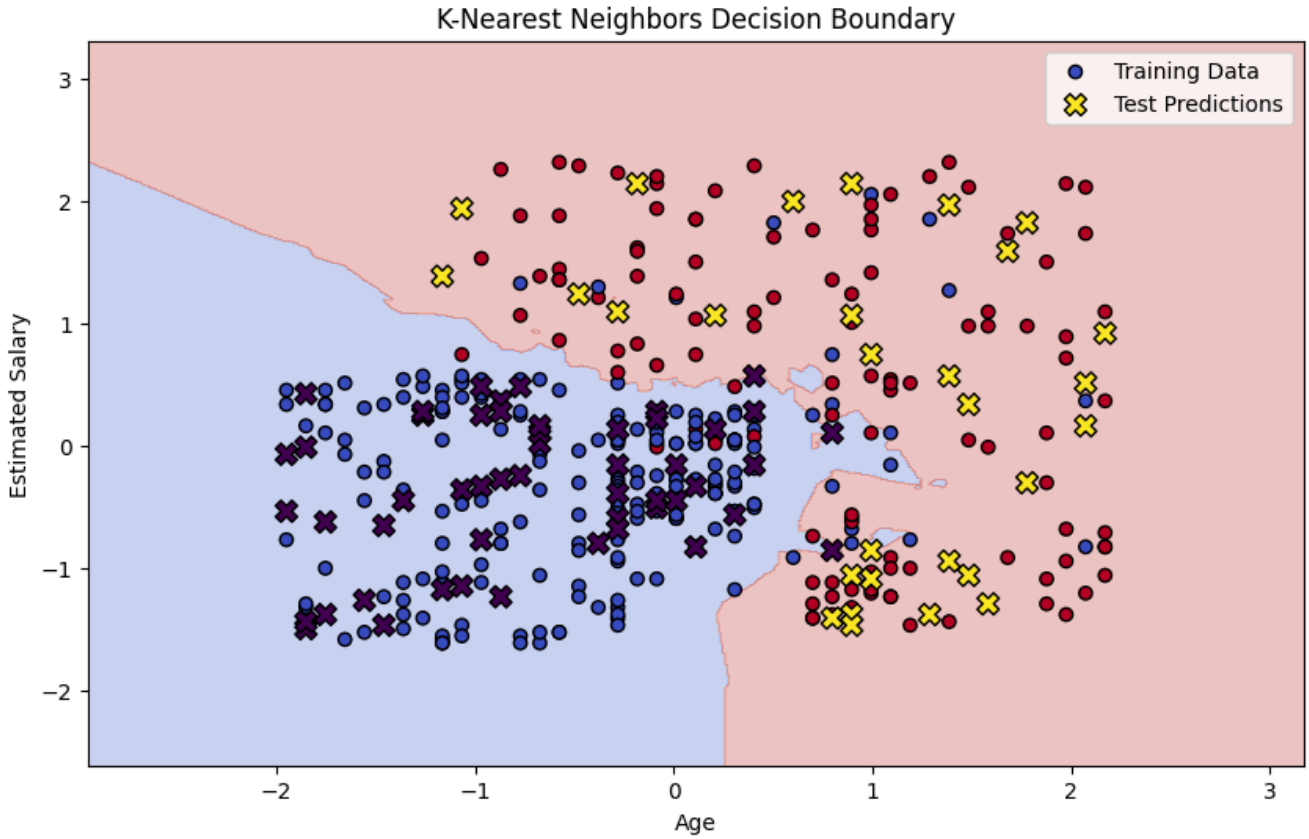
```
F1 Score: 0.8771929824561403
```

Data visualization

```
x_min, x_max = X_train[:, 0].min() - 1, X_train[:, 0].max() + 1
y_min, y_max = X_train[:, 1].min() - 1, X_train[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, 0.01), np.arange(y_min, y_max, 0.01))
```

```
Z = knn.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
```

```
plt.figure(figsize=(10, 6))
plt.contourf(xx, yy, Z, alpha=0.3, cmap='coolwarm')
plt.scatter(X_train[:, 0], X_train[:, 1], c=y_train, edgecolor='k', cmap='coolwarm', label='Training Data')
plt.scatter(X_test[:, 0], X_test[:, 1], c=y_pred, edgecolor='black', marker='X', s=100, label='Test Predicti
plt.title('K-Nearest Neighbors Decision Boundary')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```


K-Nearest Neighbors Decision Boundary
Training Data
Test Predictions
Estimated Salary
Age
3
2
1
0
−1
−2
−2
−1
0
1
2
3