In [4]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder

data = pd.read_csv(r'C:\Users\Student\Desktop\Bank.csv')
columns_to_drop = [col for col in data.columns if col not in ['Gender', 'Ag
data.drop(columns=columns_to_drop, inplace=True)

print(data.head())

# Label encoding Gender, Martial-Status, and IsLoanGiven columns
label_encoder = LabelEncoder()
data['Gender'] = label_encoder.fit_transform(data['Gender'])
data['Status'] = label_encoder.fit_transform(data['Status'])
data['IsLoanGiven'] = label_encoder.fit_transform(data['IsLoanGiven'])

# Dropping columns after 'IsLoanGiven'
data = data.iloc[:, :4]  # Assuming 'IsLoanGiven' is the fourth column, cha

# Separating features and target variable
x = data.drop('IsLoanGiven', axis=1)
y = data['IsLoanGiven']

# Splitting the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20)

# Scaling the features
scaler = StandardScaler()
scaler.fit(x_train)
x_train = scaler.transform(x_train)
x_test = scaler.transform(x_test)

# Display scaled features
print(x_train)
print(x_test)

# Using K-Nearest Neighbors classifier
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors=3)
classifier.fit(x_train, y_train)
result = classifier.predict(x_test)

# Display classification report and confusion matrix
from sklearn.metrics import confusion_matrix, classification_report
print(classification_report(y_test, result))
print(confusion_matrix(y_test, result))
```

```
  Gender  Age   Status IsLoanGiven
0      M   45   Single         Yes
1      F   50  Married          NO
2      M   35  Married          NO
3      M   30   Single          NO
4      F   43   Single         Yes
[[-1.58113883  1.63982104 -0.8660254 ]
 [ 0.63245553 -0.92779348 -0.8660254 ]
 [ 0.63245553 -1.38090193  1.15470054]
 [ 0.63245553  0.8846403   1.15470054]
 [-1.58113883  0.582568    1.15470054]
 [ 0.63245553 -0.62572119 -0.8660254 ]
 [ 0.63245553 -0.17261274 -0.8660254 ]]
[[ 0.63245553  0.43153185  1.15470054]
 [-1.58113883  2.09292948  1.15470054]]
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       1.0
           1       0.00      0.00      0.00       1.0

    accuracy                           0.00       2.0
   macro avg       0.00      0.00      0.00       2.0
weighted avg       0.00      0.00      0.00       2.0

[[0 1]
 [1 0]]
```

In [ ]: