```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
import matplotlib.pyplot as plt
import seaborn as sns

def load_credit_data():
    # Sample hardcoded data (can be replaced by real dataset)
    data = {
        'Income': [40000, 60000, 25000, 50000, 70000, 100000, 120000, 15000,
35000, 80000],
        'Age': [25, 45, 35, 40, 29, 55, 33, 60, 48, 30],
        'LoanAmount': [1000, 2000, 500, 1500, 1800, 2500, 3000, 700, 1100,
2300],
        'CreditScore': [0, 1, 0, 1, 1, 1, 1, 0, 0, 1]  # 1 = Good, 0 = Bad
    }
    return pd.DataFrame(data)

def preprocess_data(df):
    X = df[['Income', 'Age', 'LoanAmount']]
    y = df['CreditScore']
    scaler = StandardScaler()
    X_scaled = scaler.fit_transform(X)
    return train_test_split(X_scaled, y, test_size=0.3, random_state=42)

def train_classifier(X_train, y_train):
    model = LogisticRegression()
    model.fit(X_train, y_train)
    return model

def evaluate_classifier(model, X_test, y_test):
    y_pred = model.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    print("\n--- Evaluation Results ---")
    print(f"Accuracy: {acc:.2f}")
    print("Classification Report:\n", classification_report(y_test, y_pred))
    return y_test.values, y_pred, confusion_matrix(y_test, y_pred)

def plot_results(y_test, y_pred, cm):
    # Confusion Matrix
    sns.heatmap(cm, annot=True, fmt='d', cmap="YlGnBu", xticklabels=["Bad",
"Good"], yticklabels=["Bad", "Good"])
    plt.title("Confusion Matrix")
    plt.xlabel("Predicted")
    plt.ylabel("Actual")
    plt.show()

    # Actual vs Predicted
    x = np.arange(len(y_test))
    plt.figure()
    plt.bar(x - 0.2, y_test, width=0.4, label='Actual', color='gray')
    plt.bar(x + 0.2, y_pred, width=0.4, label='Predicted', color='orange')
    plt.xticks(x)
    plt.xlabel("Test Sample Index")
    plt.ylabel("Credit Score Class")
    plt.title("Actual vs Predicted Credit Scores")
    plt.legend()
    plt.show()

def main():
```

```
    print("=== Credit Score Classification Application ===")
    df = load_credit_data()
    X_train, X_test, y_train, y_test = preprocess_data(df)
    model = train_classifier(X_train, y_train)
    y_test, y_pred, cm = evaluate_classifier(model, X_test, y_test)
    plot_results(y_test, y_pred, cm)

if __name__ == "__main__":
    main()
```