**Assignment No : 4**

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split, GridSearchCV

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

# Load dataset (Replace 'banknote\_data.csv' with your actual dataset file)

df = pd.read\_csv("/content/sample\_data/data\_banknote\_authentication.csv")

# Display the first few rows of the dataset

print("\nFirst 5 rows of the dataset:")

print(df.head())

**Output:-**

First 5 rows of the dataset:

3.6216 8.6661 -2.8073 -0.44699 0

0 4.54590 8.1674 -2.4586 -1.46210 0

1 3.86600 -2.6383 1.9242 0.10645 0

2 3.45660 9.5228 -4.0112 -3.59440 0

3 0.32924 -4.4552 4.5718 -0.98880 0

4 4.36840 9.6718 -3.9606 -3.16250 0

# Splitting features and target variable

X = df.iloc[:, :-1]

y = df.iloc[:, -1]

# Splitting dataset into training (80%) and testing (20%) sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42, stratify=y)

# Standardize the features (important for logistic regression)

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

# Train Logistic Regression model

model = LogisticRegression()

model.fit(X\_train, y\_train)

# Predictions

y\_pred = model.predict(X\_test)

# Generate Confusion Matrix BEFORE accuracy

conf\_matrix = confusion\_matrix(y\_test, y\_pred)

print("\nConfusion Matrix:")

print(conf\_matrix)

**Output:**

Confusion Matrix:

[[148 5]

[ 0 122]]

plt.figure(figsize=(5, 4))

sns.heatmap(conf\_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=["Fake", "Authentic"], yticklabels=["Fake", "Authentic"])

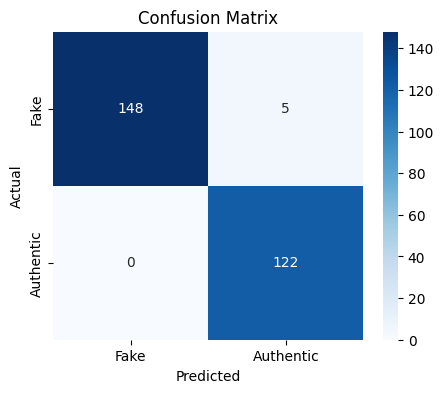
plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.title("Confusion Matrix")

plt.show()

**Output:**



# Classification report

print("\nClassification Report:")

print(classification\_report(y\_test, y\_pred))

**Output:**

Classification Report:

precision recall f1-score support

0 1.00 0.97 0.98 153

1 0.96 1.00 0.98 122

accuracy 0.98 275

macro avg 0.98 0.98 0.98 275

weighted avg 0.98 0.98 0.98 275

# Now, calculate model accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"\nModel Accuracy: {accuracy:.2f}")

**Output:**

Model Accuracy: 0.98