PATTERN RECOGNITION

NAME : PREM SHANDILYA
SAP ID : 590017213
BATCH : AI/ML (B2)

Description:

This experiment illustrates the use of logistic regression on the **Diabetes dataset**. The process involves training the model, generating predictions, and assessing its accuracy using essential metrics like the confusion matrix, precision, recall, sensitivity, and specificity. Furthermore, the ROC-AUC curve is plotted to analyze the balance between true positive and false positive rates, offering an in-depth evaluation of the classifier's effectiveness.

Aim:
The objective of this experiment is to implement logistic regression for binary classification using the **Diabetes dataset**. The study emphasizes assessing the model's performance through metrics such as precision, recall, sensitivity, and specificity. Additionally, the ROC-AUC curve is plotted to visualize the trade-off between true positives and false positives, helping to understand how well the model differentiates between the two classes.

Algorithm:

1.Load the Dataset:
Import the **Diabetes dataset** using sklearn.datasets.fetch_openml("diabetes"), then extract the features and target variable.

2.Preprocess the Data:
Prepare the dataset by scaling numerical features if needed and addressing any missing values (though the dataset is generally well-structured).

3.Split the Data:
Partition the dataset into training and test sets using train_test_split(), ensuring a fixed test proportion and random state for reproducibility.

4.Train the Model:
Fit a logistic regression model with suitable hyperparameters (e.g., solver type, maximum iterations) on the training dataset.

5. **Make Predictions:**
Use predict_proba() for probability estimates and predict() for binary class predictions on the test dataset.
6.

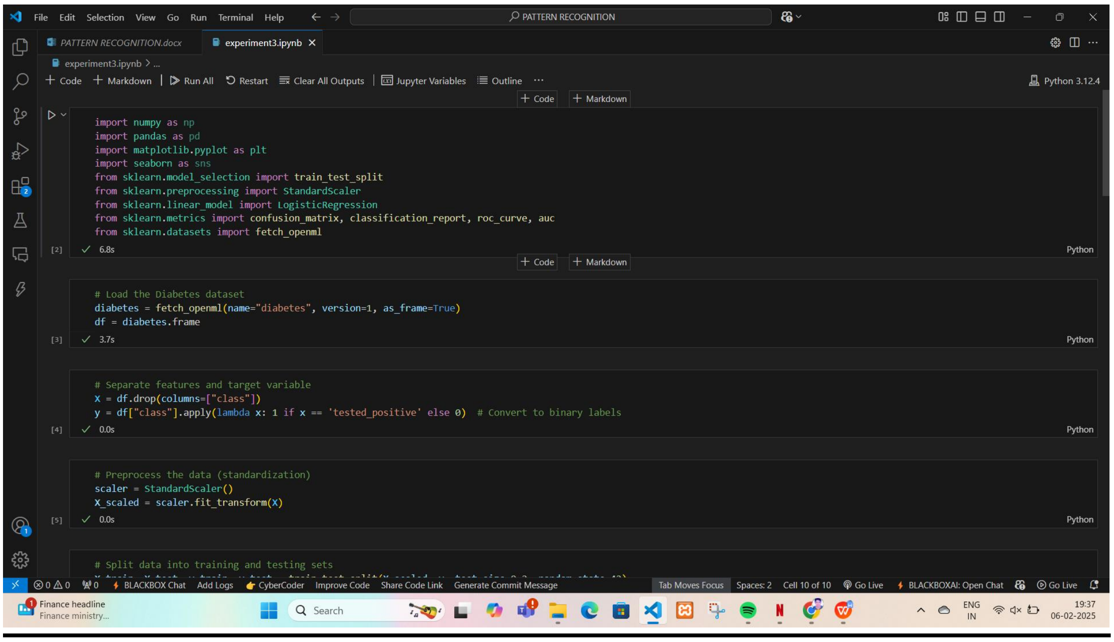
6.Evaluate Performance:
Compute the confusion matrix and derive key evaluation metrics:

- True Positive Rate (TPR)
- False Positive Rate (FPR)
- True Negative Rate (TNR)
- False Negative Rate (FNR)

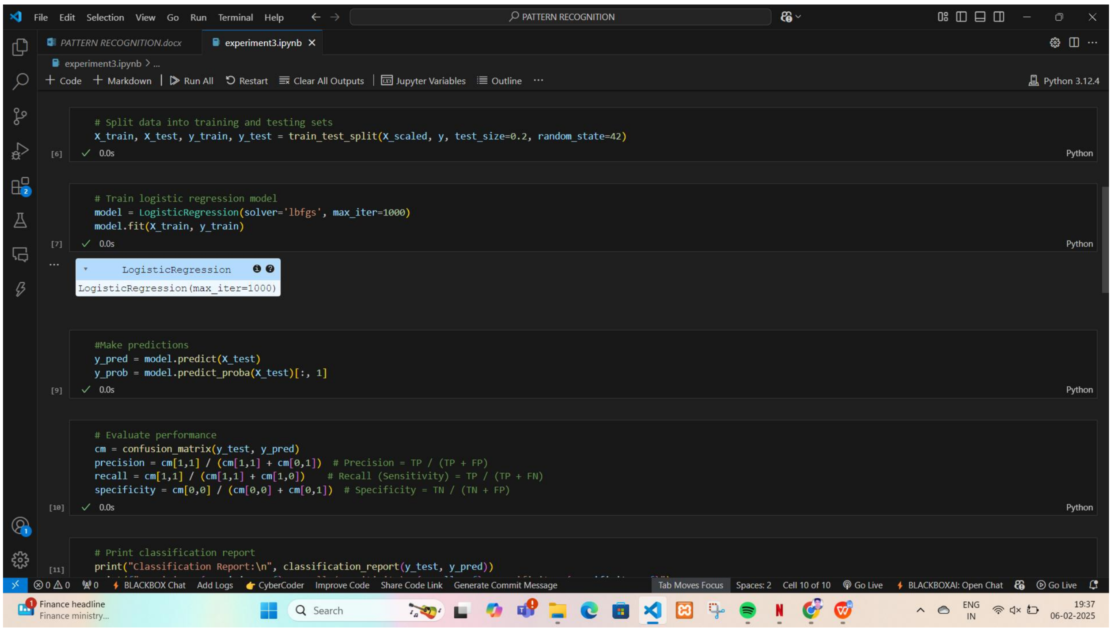
Also, calculate precision, recall, sensitivity, and specificity.

7.Plot ROC Curve:
Generate the ROC curve and determine the Area Under the Curve (AUC) to illustrate the relationship between TPR and FPR.

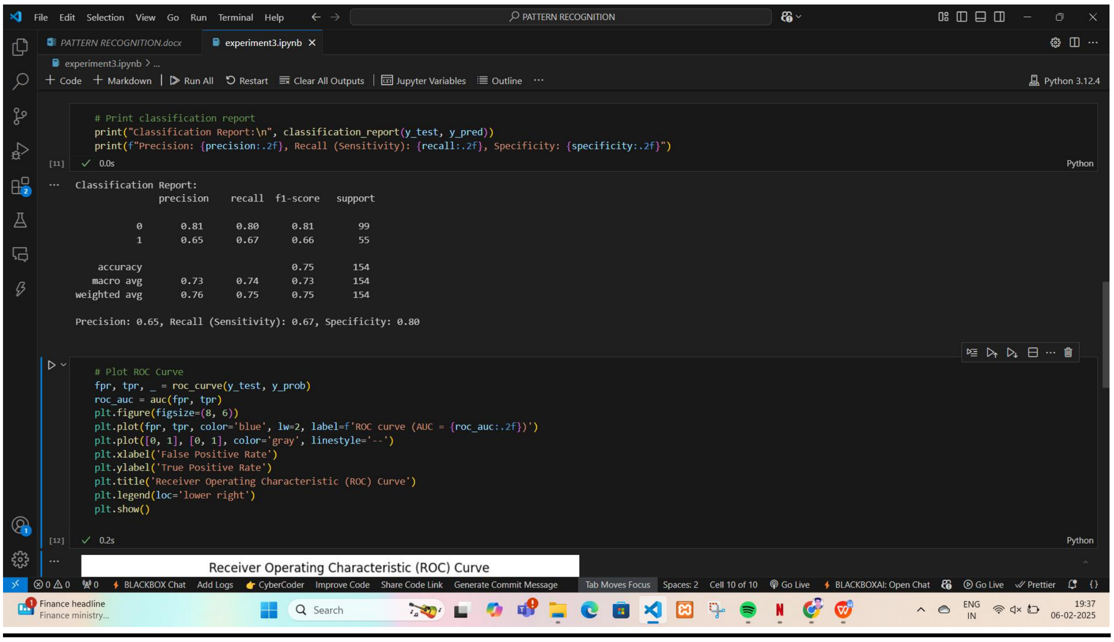
CODE



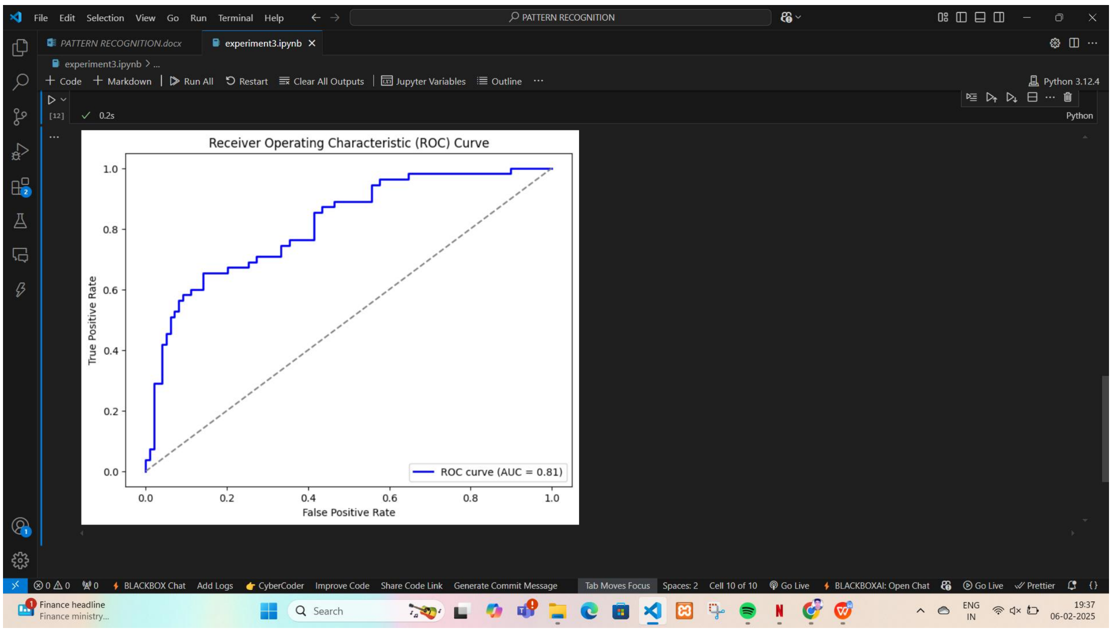
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, classification_report, roc_curve, auc
from sklearn.datasets import fetch_openml
# Load the Diabetes dataset
diabetes = fetch_openml(name="diabetes", version=1, as_frame=True)
df = diabetes.frame
# Separate features and target variable
X = df.drop(columns=["class"])
y = df["class"].apply(lambda x: 1 if x == 'tested_positive' else 0) # Convert to binary labels
# Preprocess the data (standardization)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
# Split data into training and testing sets



Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
# Train logistic regression model
model = LogisticRegression(solver='lbfgs', max_iter=1000)
model.fit(X_train, y_train)
LogisticRegression
LogisticRegression(max_iter=1000)
#Make predictions
y_pred = model.predict(X_test)
y_prob = model.predict_proba(X_test)[:, 1]
# Evaluate performance
cm = confusion_matrix(y_test, y_pred)
precision = cm[1,1] / (cm[1,1] + cm[0,1]) # Precision = TP / (TP + FP)
recall = cm[1,1] / (cm[1,1] + cm[1,0])   # Recall (Sensitivity) = TP / (TP + FN)
specificity = cm[0,0] / (cm[0,0] + cm[0,1]) # Specificity = TN / (TN + FP)
# Print classification report
print("Classification Report:\n", classification_report(y_test, y_pred))



# Print classification report
print("Classification Report:\n", classification_report(y_test, y_pred))
print(f"Precision: {precision:.2f}, Recall (Sensitivity): {recall:.2f}, Specificity: {specificity:.2f}")
Classification Report:
precision recall f1-score support
0 0.81 0.80 0.81 99
1 0.65 0.67 0.66 55
accuracy 0.75 154
macro avg 0.73 0.74 0.73 154
weighted avg 0.76 0.75 0.75 154
Precision: 0.65, Recall (Sensitivity): 0.67, Specificity: 0.80
Plot ROC Curve
fpr, tpr, _ = roc_curve(y_test, y_prob)
roc_auc = auc(fpr, tpr)
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='blue', lw=2, label=f'ROC curve (AUC = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='gray', linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc='lower right')
plt.show()
Receiver Operating Characteristic (ROC) Curve



Receiver Operating Characteristic (ROC) Curve
True Positive Rate
False Positive Rate
ROC curve (AUC = 0.81)