1. Implement logistic regression using Python/R to perform classification on

Social\_Network\_Ads.csv dataset.

1. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall

On the given dataset.

import pandas as pd

import numpy as np

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, accuracy\_score, precision\_score, recall\_score, f1\_score

dataset = pd.read\_csv('/home/admin-com/Downloads/Social\_Network\_Ads.csv')

print(dataset.head())

print(dataset.tail())

User ID Gender Age EstimatedSalary Purchased

0 15624510 Male 19 19000 0

1 15810944 Male 35 20000 0

2 15668575 Female 26 43000 0

3 15603246 Female 27 57000 0

4 15804002 Male 19 76000 0

User ID Gender Age EstimatedSalary Purchased

395 15691863 Female 46 41000 1

396 15706071 Male 51 23000 1

397 15654296 Female 50 20000 1

398 15755018 Male 36 33000 0

399 15594041 Female 49 36000 1

X = dataset[['Age','EstimatedSalary']]

Y = dataset['Purchased']

X\_train,X\_test,Y\_train,Y\_test = train\_test\_split(X,Y,test\_size=0.25,random\_state=0)

scaler = StandardScaler()

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

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

classifier = LogisticRegression(random\_state=0)

classifier.fit(X\_train,Y\_train)

Y\_pred = classifier.predict(X\_test)

cm = confusion\_matrix(Y\_test,Y\_pred)

tp = cm [1,1]

fp = cm [0,1]

tn = cm [0,0]

fn = cm [1,0]

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

error\_rate = 1- accuracy

precision = precision\_score(Y\_test,Y\_pred)

recall= recall\_score(Y\_test,Y\_pred)

f1 = f1\_score(Y\_test,Y\_pred)

print("Confusion Matrix:")

print(cm)

print ("True Positive: {fp}")

print("False Positive: {p}")

print("True Negative: (tn)")

print("False Negative: (fn)")

print("Accuracy: (accuracy: 4f)")

print("Error Rate: (error rate: 41}")

print("Precision: (precision: 4f)")

print("Recall: (recall: 4f)")

print("F1 Score: (11:41)")

Confusion Matrix:

[[65 3]

[ 8 24]]

True Positive: {fp}

False Positive: {p}

True Negative: (tn)

False Negative: (fn)

Accuracy: (accuracy: 4f)

Error Rate: (error rate: 41}

Precision: (precision: 4f)

Recall: (recall: 4f)

F1 Score: (11:41)

plt.figure(figsize=(6,6))

sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",

xticklabels=["Predict No", "Predict Yes"],

yticklabels=["Actual No", "Actual Yes"])

plt.title("Confusion Matrix")

plt.xlabel("Predicted")

plt.ylabel("Actual")

plt.show()

Confusion Matrix:

[[65 3]

[ 8 24]]

True Positive: {fp}

False Positive: {p}

True Negative: (tn)

False Negative: (fn)

Accuracy: (accuracy: 4f)

Error Rate: (error rate: 41}

Precision: (precision: 4f)

Recall: (recall: 4f)

F1 Score: (11:41)

