**PROGRAM 1 (CHAPTER EXERCISE)**

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

x = np.array([5, 5.25, 5.5, 5.75, 6, 6.5, 7, 7.5, 8, 8.5, 8.75, 9])

y = np.array([0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0])

x = x.reshape(-1,1)

from sklearn.linear\_model import LogisticRegression

classifier = LogisticRegression(random\_state = 0)

classifier.fit(x, y)

classifier.predict([[5.5]])

classifier.predict([[5.75]])

y\_pred = classifier.predict(x)

print(np.concatenate((y.reshape(len(y),1), (y\_pred.reshape(len(y\_pred),1))),1))

from sklearn.metrics import confusion\_matrix, accuracy\_score, f1\_score, precision\_score, recall\_score

cm = confusion\_matrix(y, y\_pred)

print(cm)

accuracy\_score(y, y\_pred)

precision\_score(y, y\_pred)

recall\_score(y, y\_pred)

f1\_score(y, y\_pred)

**PROGRAM 2**

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

df = pd.read\_csv('Social\_Network\_Ads.csv')

df

x = df.iloc[:, :-1].values

y = df.iloc[:, -1].values

**splitting the dataset**

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

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size = 0.2, random\_state=0)

print(x\_train)

print(x\_test)

**Feature Scaling**

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

x\_train = sc.fit\_transform(x\_train)

x\_test = sc.transform(x\_test)

print(x\_train)

print(x\_test)

**Training**

from sklearn.linear\_model import LogisticRegression

classifier = LogisticRegression(random\_state = 0)

classifier.fit(x\_train, y\_train)

**Predicting a result**

print(classifier.predict(sc.transform([[30,87000]])))

**Predicting test set results**

y\_pred = classifier.predict(x\_test)

print(np.concatenate((y\_test.reshape(len(y\_test),1), y\_pred.reshape(len(y\_pred),1)),1))

**confusion matrix**

from sklearn.metrics import confusion\_matrix, accuracy\_score, precision\_score, recall\_score, f1\_score

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

print(cm)

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

precision\_score(y\_test, y\_pred)

recall\_score(y\_test, y\_pred)

f1\_score(y\_test, y\_pred)