Name : Cheruvathoor Abin Anto

Roll No : TIT2425008

**Implementation of Naive Bayes Classification in Python**

**Step 1: Import libraries & Step 2: Import data**

# Import libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler

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

from sklearn.naive\_bayes import GaussianNB

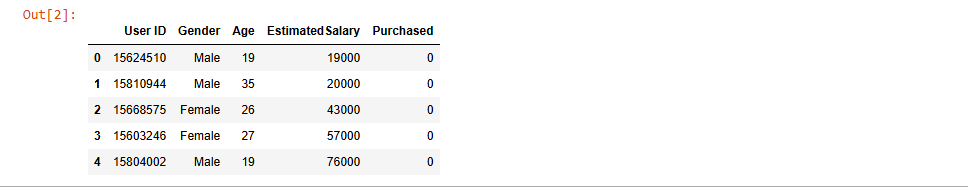
from sklearn.metrics import accuracy\_score

from sklearn.metrics import confusion\_matrix

# Read dataset

df\_net = pd.read\_csv('D:/Aby/IDS/P6/Social\_Network\_Ads.csv')

df\_net.head()

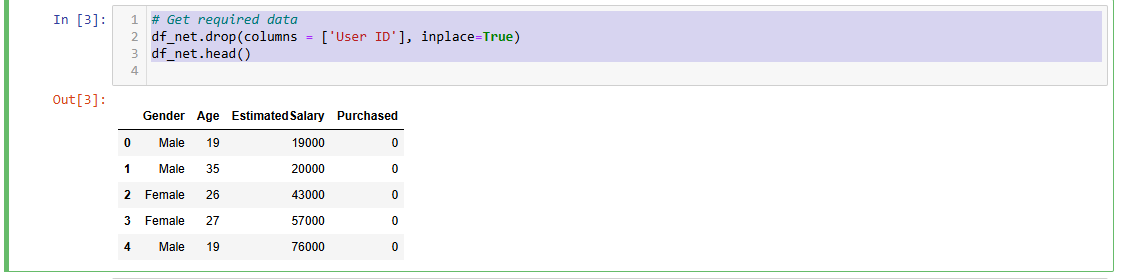


# Step 3: Data Analysis / Preprocessing

# Get required data

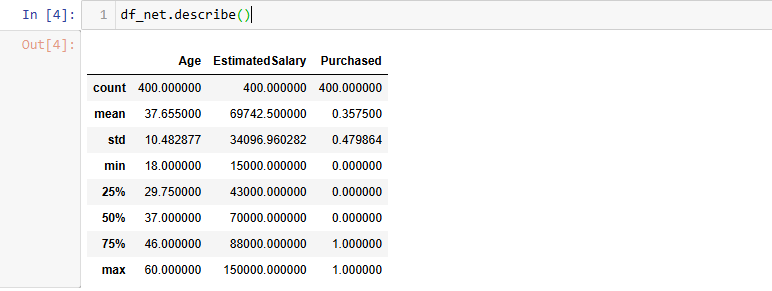
df\_net.drop(columns = ['User ID'], inplace=True)

df\_net.head()



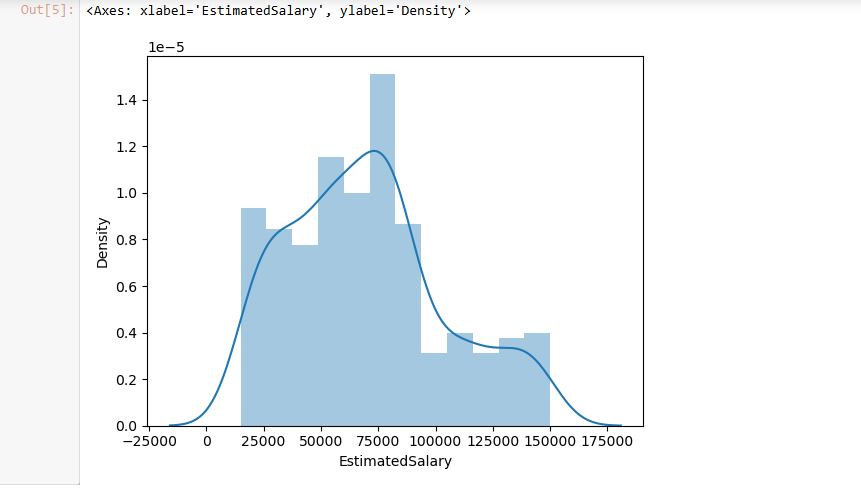
**Describe data**

df\_net.describe()



**Distribution of Data**

sns.distplot(df\_net['EstimatedSalary'])

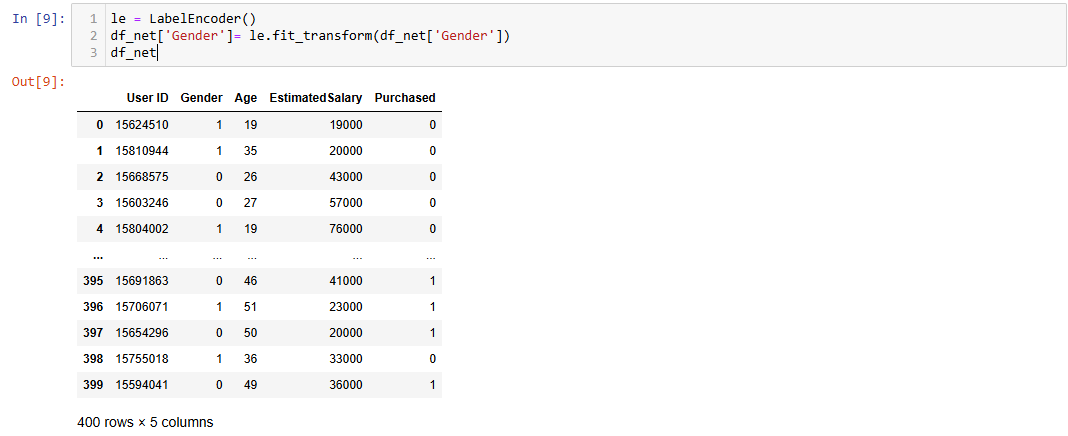


**Label encoding**

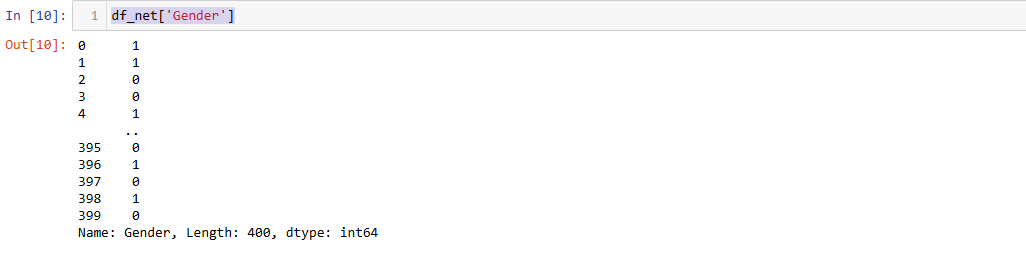
le = LabelEncoder()

df\_net['Gender']= le.fit\_transform(df\_net['Gender'])

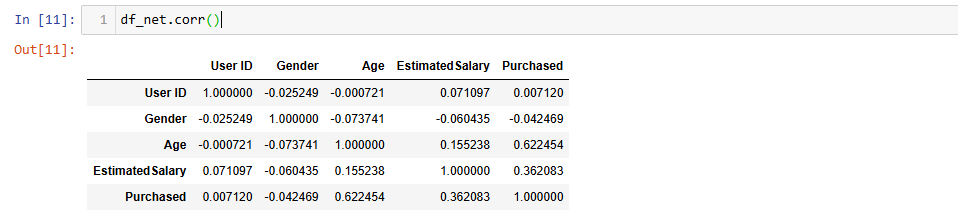
df\_net



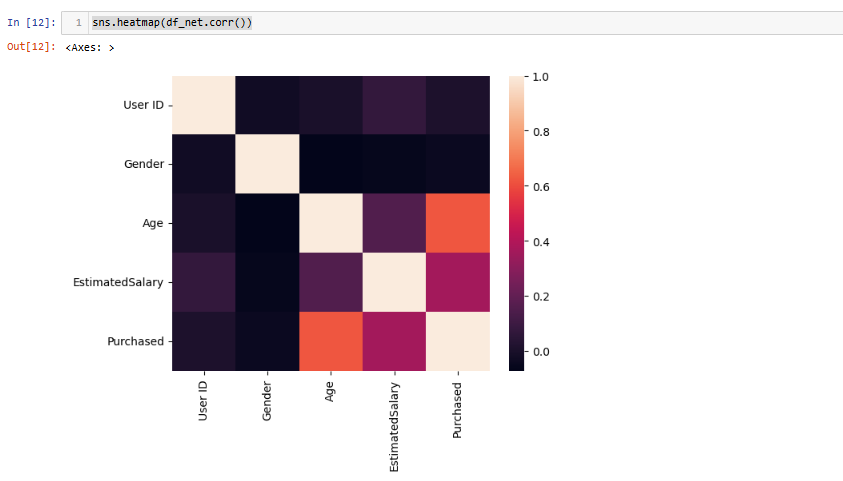
df\_net['Gender']



df\_net.corr()

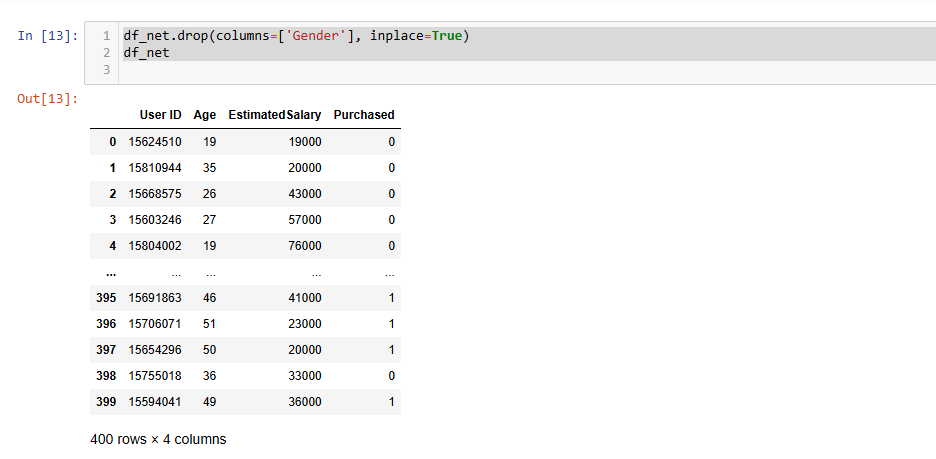


sns.heatmap(df\_net.corr())



df\_net.drop(columns=['Gender'], inplace=True)

df\_net



# Step 4: Split data

# X = df\_net.iloc[:, :-1].values

# # all rows and all columns except the last one(-1)

# X

# 

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

# # Selects the last column

# y

# 

# This is important because if the model is trained and tested on the same data, it may over-fit the data and perform poorly on new, unseen data.

# We have split the data into 75% for training and 25% for testing.

# X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = True)

# Step 5: Feature scaling

# sc = StandardScaler()

# X\_train = sc.fit\_transform(X\_train)

# X\_test = sc.transform(X\_test)

# X\_train

# 

# 

# classifier = GaussianNB()

# classifier.fit(X\_train, y\_train)

# 

# Step 7: Predict result / Score model

# y\_pred = classifier.predict(X\_test)

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

# Step 8: Evaluate model

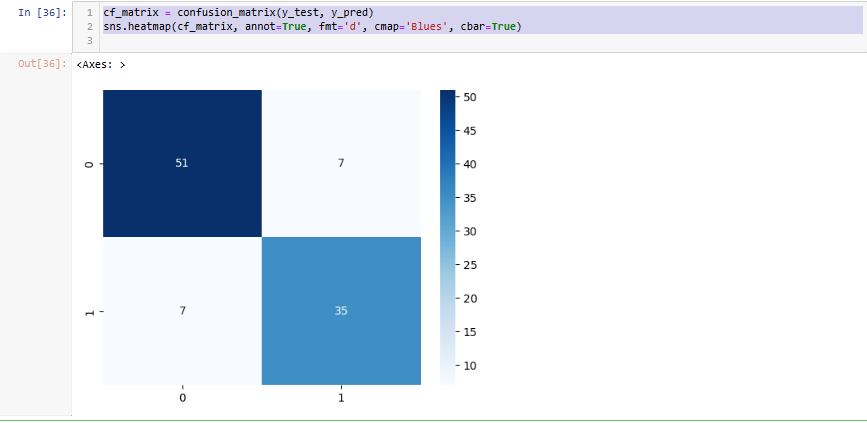
# accuracy\_score(y\_test, y\_pred)

# 

**Confusion matrix**

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

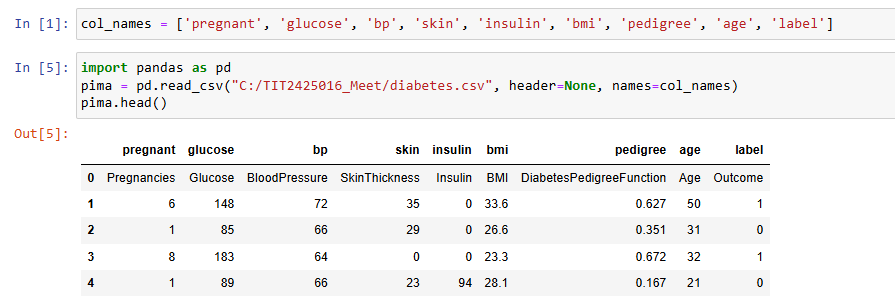
sns.heatmap(cf\_matrix, annot=True, fmt='d', cmap='Blues', cbar=True)



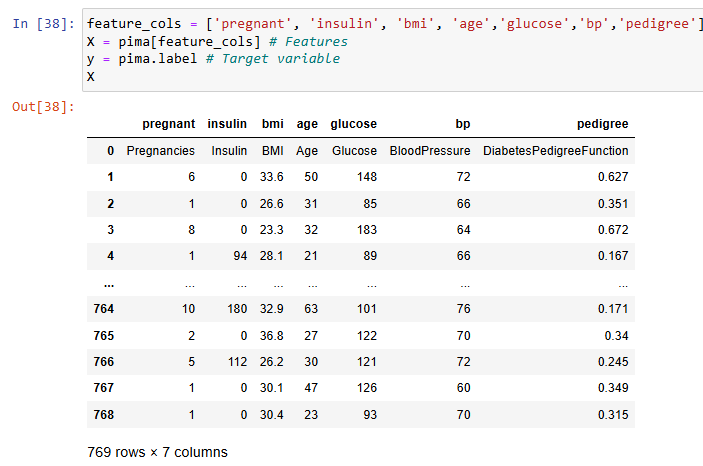
print(classifier.predict(sc.transform([[45, 97000, 0]])))



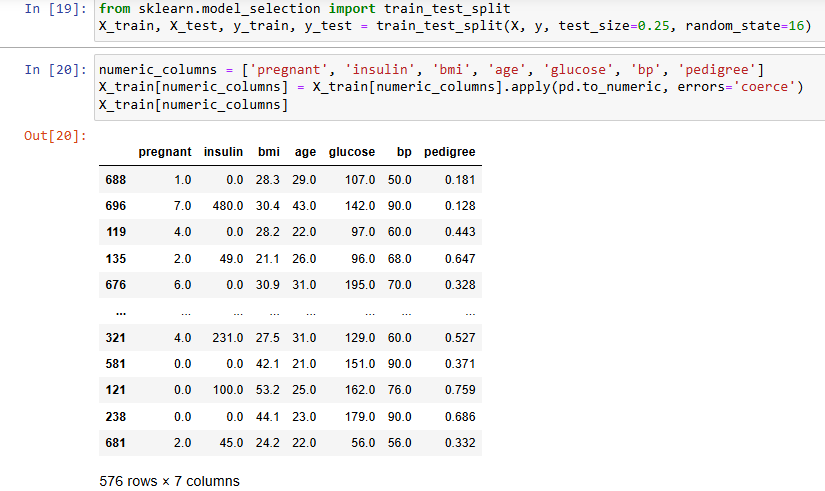
***Ch. 5 ) Logical Regression***

**1 )   
 **

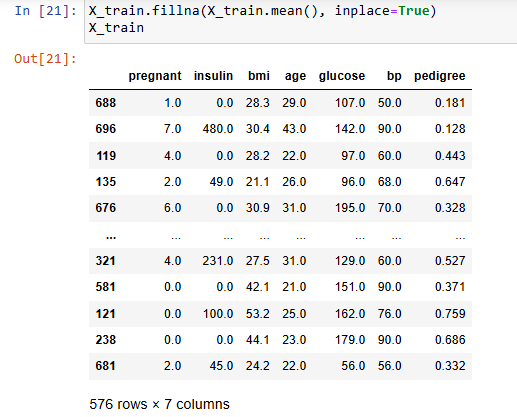
**2 )**

****

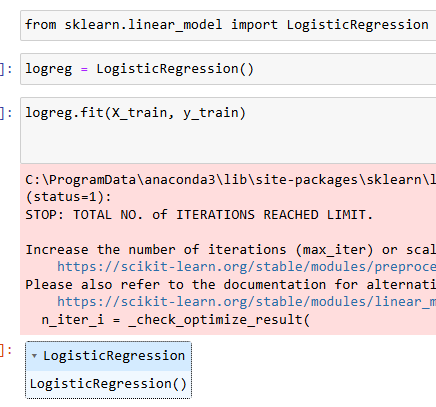
**3)**

****

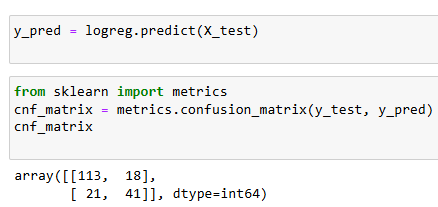
**4)**

****

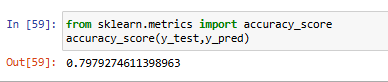
**5)**

****

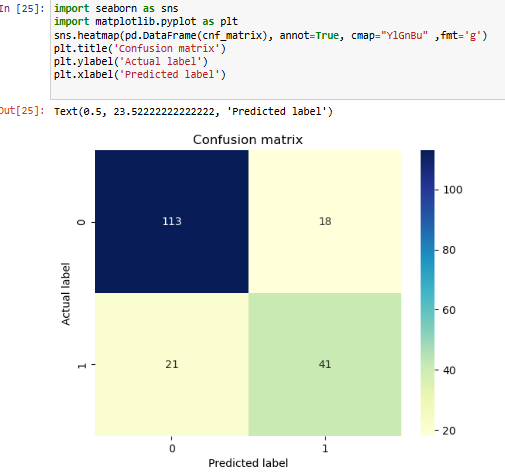
**6)**

****

**7)**

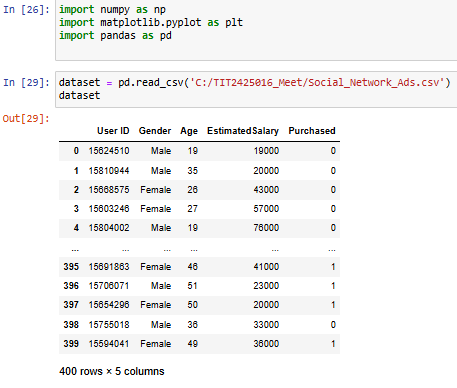
****

**8)**

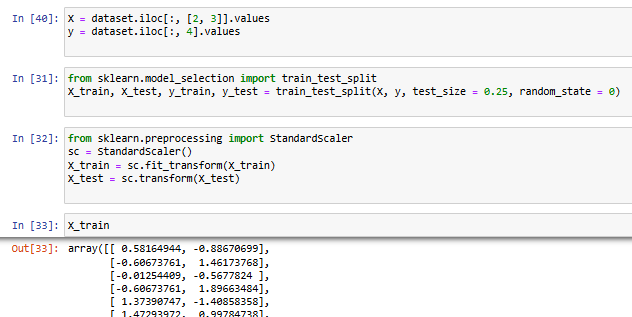
****

**Ch. 5 )** **Support Vector Machine using python**

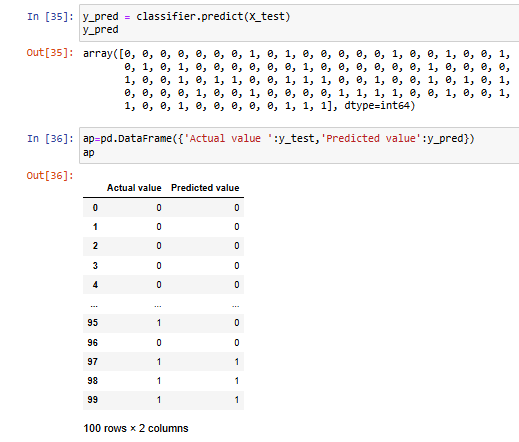
**1)**

****

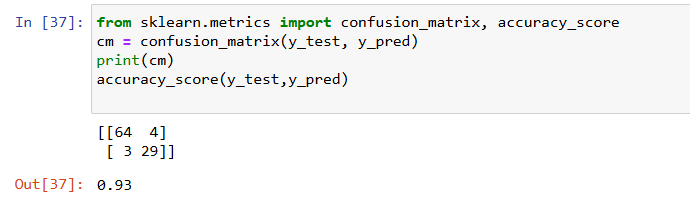
**2)**

****

**3)**

****

**4)**

****