**NAÏVE BAYES ALGORITHM (DAY 25)**

**“””Importing PANDAS and Dataset”””**

import pandas as pd

dataset=pd.read\_csv("train.csv")

dataset.columns

""" CPOLUMNS DISPLAYED

Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',

'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],

dtype='object')

"""

**""" IMporting all files"""**

from sklearn import preprocessing

#here cross\_validation not working so use alternative i.e model\_selection

""" from sklearn.cross\_validation import train\_test\_split

File "<ipython-input-7-d05cc6ee7707>", line 1, in <module>

from sklearn.cross\_validation import train\_test\_split

ModuleNotFoundError: No module named 'sklearn.cross\_validation'

"""

**“””Importing SKLEARN methods “””**

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

**""" preprocessing """**

#1)

le=preprocessing.LabelEncoder()

le.fit(dataset["Sex"])

""" Converting into numerical"""

dataset1=le.transform(dataset["Sex"])

dataset["Sex"]=dataset1

#2)

le=preprocessing.LabelEncoder()

le.fit(dataset["Embarked"])

""" Converting into numerical"""

dataset1=le.transform(dataset["Embarked"])

dataset["Embarked"]=dataset1

**""" Droping Uneccessary olumns """**

Real\_data=dataset.drop(["Cabin","Name","PassengerId","Ticket"],axis=1)

l=['Survived','Pclass','Sex','Age','SibSp','Parch','Fare','Embarked']

from sklearn.naive\_bayes import BernoulliNB

n=len(l)

**#LISTS to hold accuracy values and Confusion matrix values**

acc\_list=[]

conf\_list=[]

**#Iterator**

for x in range(n):

#print("x =",x)

str1=l[x]

#print(str1)

DV=Real\_data[str1]

#print(DV)

IDV=Real\_data.drop(str1,axis=1)

#print(IDV)

#spliting record of 70 training/ 30 testing

IDV\_train,IDV\_test,DV\_train,DV\_test=train\_test\_split(IDV,DV,test\_size=0.3,random\_state=0)

# Applyin naive bayes algorithm

clf=BernoulliNB()

DV\_pred=clf.fit(IDV\_train,DV\_train).predict(IDV\_test).T

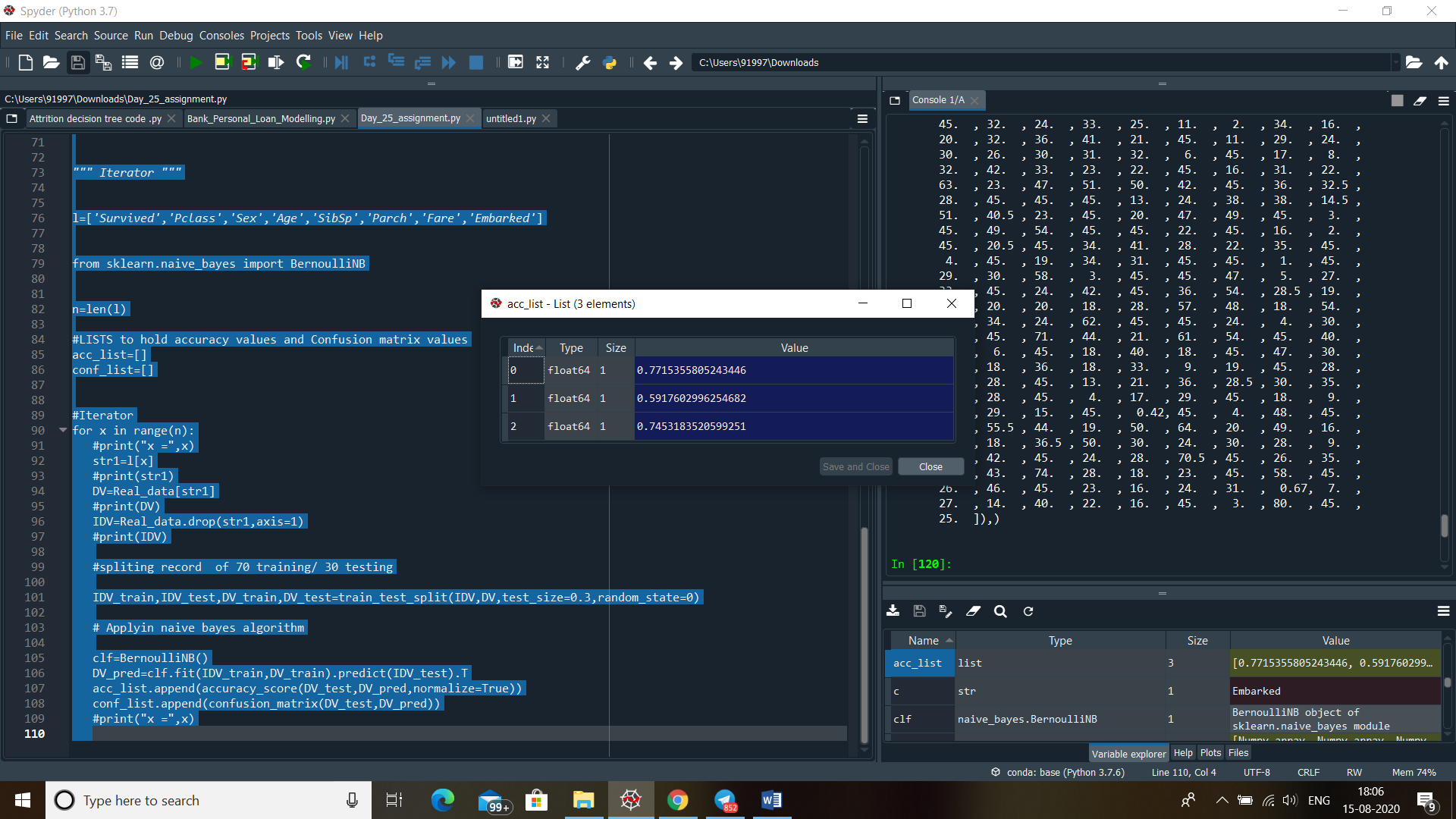
acc\_list.append(accuracy\_score(DV\_test,DV\_pred,normalize=True))

conf\_list.append(confusion\_matrix(DV\_test,DV\_pred))

#print("x =",x)

**RESULT:**

1. **ACCURACY**



1. **Confusion Matrix**

