"""

Created on Thu Jul 5 23:42:36 2018

@author: shubham b thorat

classification to check if a customer invests in the mutual fund or not

"""

#importing libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

#importing datasets

dataset = pd.read\_csv("bank.csv", sep = ";")

X = dataset.iloc[:,:-1].values

Y = dataset.iloc[:,16].values

#using encoding

from sklearn.preprocessing import LabelEncoder ,OneHotEncoder

labelencoder\_X = LabelEncoder()

X[:,1] = labelencoder\_X.fit\_transform(X[:,1])

X[:,2] = labelencoder\_X.fit\_transform(X[:,2])

X[:,3] = labelencoder\_X.fit\_transform(X[:,3])

X[:,4] = labelencoder\_X.fit\_transform(X[:,4])

X[:,6] = labelencoder\_X.fit\_transform(X[:,6])

X[:,7] = labelencoder\_X.fit\_transform(X[:,7])

X[:,8] = labelencoder\_X.fit\_transform(X[:,8])

X[:,10] = labelencoder\_X.fit\_transform(X[:,10])

X[:,15] = labelencoder\_X.fit\_transform(X[:,15])

Y = labelencoder\_X.fit\_transform(Y)

onehotencoder = OneHotEncoder(categorical\_features= [1,2,3,4,6,7,8,10,15])

X = onehotencoder.fit\_transform(X).toarray()

#splitting dataset into training and testing dataset

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

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X,Y, test\_size = .1, random\_state = 1)

#normalizing

from sklearn.preprocessing import StandardScaler

sc\_X = StandardScaler()

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

X\_test = sc\_X.fit\_transform(X\_test)

#fitting

from sklearn.linear\_model import LogisticRegression

classifier = LogisticRegression(random\_state = 0)

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

#predicting

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

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

#confusion matrix

from sklearn.metrics import confusion\_matrix

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

"""

#building the optimal model using backward elimination

import statsmodels.formula.api as sm

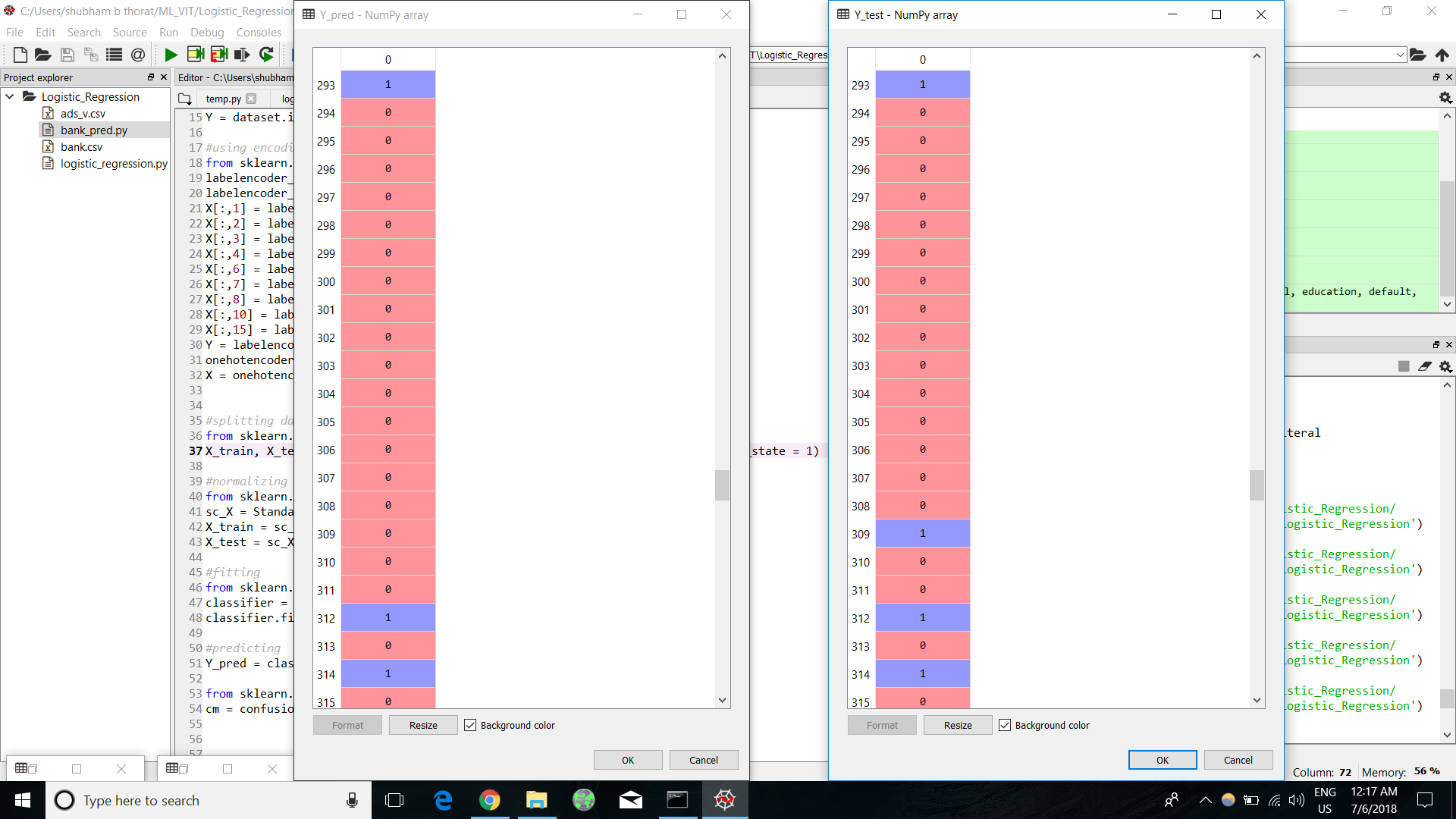
X\_test = np.append(arr = np.ones((453,1)).astype(int),values = X\_test , axis = 1)

X\_opt = X\_test[:,[0,1,2,3,4]]

regressor\_OLS = sm.OLS(endog = Y\_test , exog = X\_opt).fit() #ordinary least square(min\_value)

regressor\_OLS.summary()

"""



Confusion\_matrix

