```
"""
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

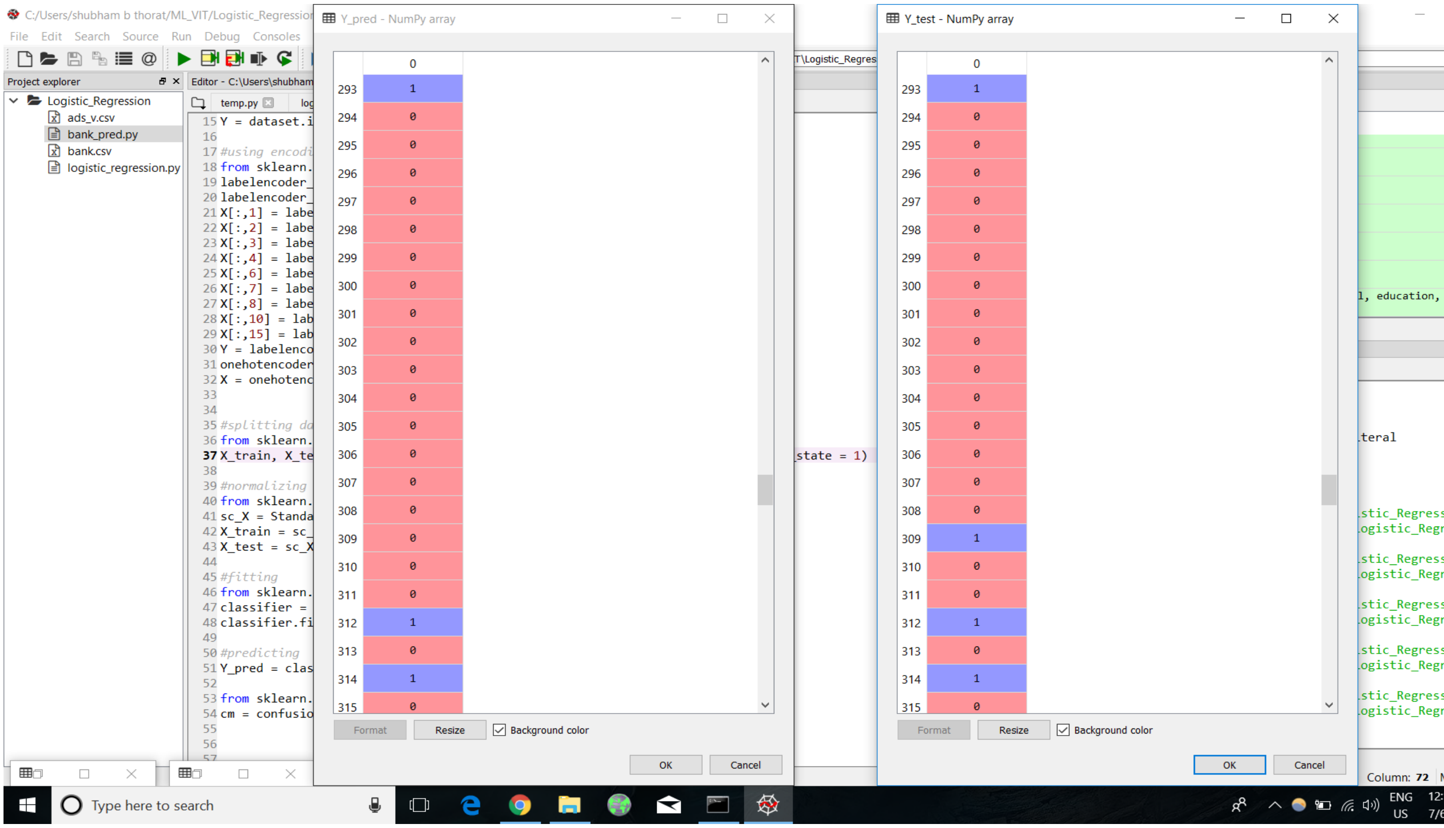
X_test = np.append(arr = np.ones((453,1)).astype(int),values = X_test , axis = 1)

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()

"""



```
Y = dataset.i

#using encodi
from sklearn.
labelencoder_
labelencoder_
X[:,1] = labe
X[:,2] = labe
X[:,3] = labe
X[:,4] = labe
X[:,6] = labe
X[:,7] = labe
X[:,8] = labe
X[:,10] = lab
X[:,15] = lab
Y = labelenco
onehotencoder
X = onehotenc


#splitting da
from sklearn.
X_train, X_te

#normalizing
from sklearn.
sc_X = Standa
X_train = sc_
X_test = sc_X

#fitting
from sklearn.
classifier =
classifier.fi

#predicting
Y_pred = clas

from sklearn.
cm = confusio
```

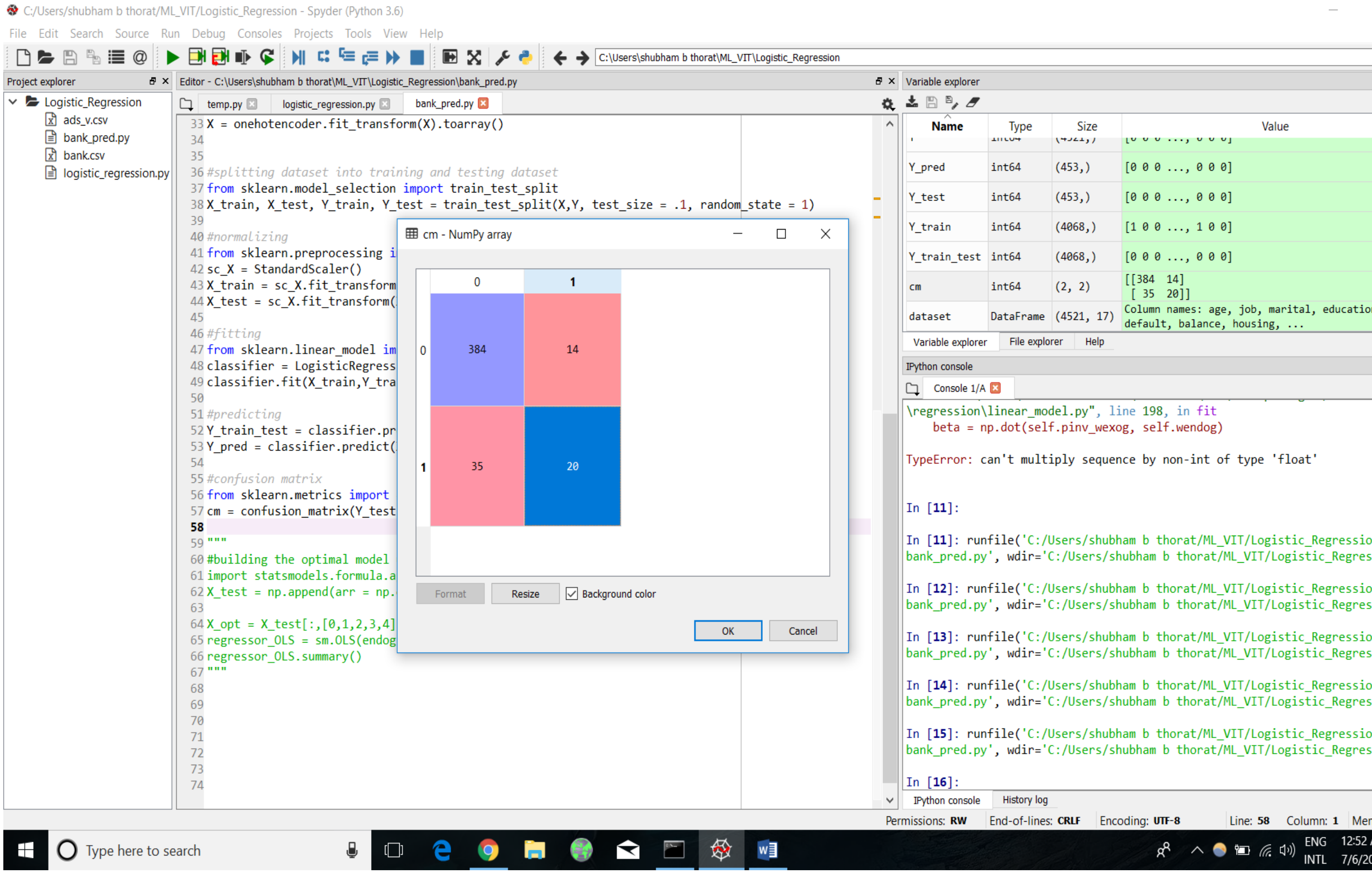
Y_pred - NumPy array

	0
293	1
294	0
295	0
296	0
297	0
298	0
299	0
300	0
301	0
302	0
303	0
304	0
305	0
306	0
307	0
308	0
309	0
310	0
311	0
312	1
313	0
314	1
315	0

Y_test - NumPy array

	0
293	1
294	0
295	0
296	0
297	0
298	0
299	0
300	0
301	0
302	0
303	0
304	0
305	0
306	0
307	0
308	0
309	1
310	0
311	0
312	1
313	0
314	1
315	0

Confusion_matrix



C:/Users/shubham b thorat/ML_VIT/Logistic_Regression - Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\shubham b thorat\ML_VIT\Logistic_Regression
Project explorer
Logistic_Regression
ads_v.csv
bank_pred.py
bank.csv
logistic_regression.py
Editor - C:\Users\shubham b thorat\ML_VIT\Logistic_Regression\bank_pred.py
temp.py
logistic_regression.py
bank_pred.py

```python
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 i
sc_X = StandardScaler()
X_train = sc_X.fit_transform
X_test = sc_X.fit_transform(

#fitting
from sklearn.linear_model im
classifier = LogisticRegress
classifier.fit(X_train,Y_tra

#predicting
Y_train_test = classifier.pr
Y_pred = classifier.predict(

#confusion matrix
from sklearn.metrics import
cm = confusion_matrix(Y_test

"""
#building the optimal model
import statsmodels.formula.a
X_test = np.append(arr = np.

X_opt = X_test[:,[0,1,2,3,4]
regressor_OLS = sm.OLS(endog
regressor_OLS.summary()
"""
```

	0	1
0	384	14
1	35	20

Name	Type	Size	Value
Y_pred	int64	(453,)	[0 0 0 ..., 0 0 0]
Y_test	int64	(453,)	[0 0 0 ..., 0 0 0]
Y_train	int64	(4068,)	[1 0 0 ..., 1 0 0]
Y_train_test	int64	(4068,)	[0 0 0 ..., 0 0 0]
cm	int64	(2, 2)	[[384 14] [35 20]]
dataset	DataFrame	(4521, 17)	Column names: age, job, marital, educatio default, balance, housing, ...

```
\regression\linear_model.py", line 198, in fit
    beta = np.dot(self.pinv_wexog, self.wendog)

TypeError: can't multiply sequence by non-int of type 'float'


In [11]:

In [11]: runfile('C:/Users/shubham b thorat/ML_VIT/Logistic_Regressio
bank_pred.py', wdir='C:/Users/shubham b thorat/ML_VIT/Logistic_Regres

In [12]: runfile('C:/Users/shubham b thorat/ML_VIT/Logistic_Regressio
bank_pred.py', wdir='C:/Users/shubham b thorat/ML_VIT/Logistic_Regres

In [13]: runfile('C:/Users/shubham b thorat/ML_VIT/Logistic_Regressio
bank_pred.py', wdir='C:/Users/shubham b thorat/ML_VIT/Logistic_Regres

In [14]: runfile('C:/Users/shubham b thorat/ML_VIT/Logistic_Regressio
bank_pred.py', wdir='C:/Users/shubham b thorat/ML_VIT/Logistic_Regres

In [15]: runfile('C:/Users/shubham b thorat/ML_VIT/Logistic_Regressio
bank_pred.py', wdir='C:/Users/shubham b thorat/ML_VIT/Logistic_Regres

In [16]:
```