import numpy as np

from sklearn import preprocessing

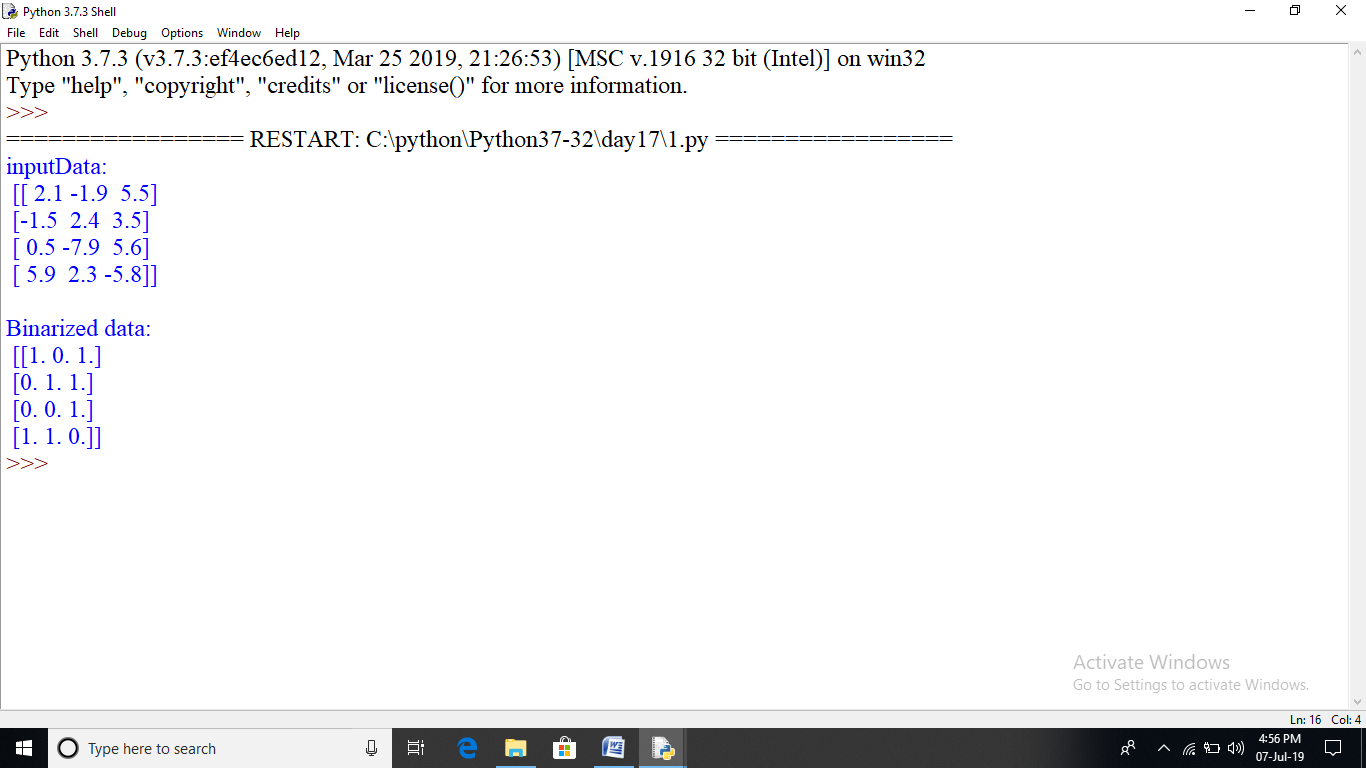
input\_data=np.array([[2.1,-1.9,5.5],[-1.5,2.4,3.5],[0.5,-7.9,5.6],[5.9,2.3,-5.8]])

print("inputData:\n",input\_data)

#binarization

data\_binarized=preprocessing.Binarizer(threshold=1.5).transform(input\_data)

print("\nBinarized data:\n",data\_binarized)



import numpy as np

from sklearn import preprocessing

input\_data=np.array([[2.1,-1.9,5.5],[-1.5,2.4,3.5],[0.5,-7.9,5.6],[5.9,2.3,-5.8]])

print("inputData:\n",input\_data)

#meanRemoval

print("mean=",input\_data.mean(axis=0))

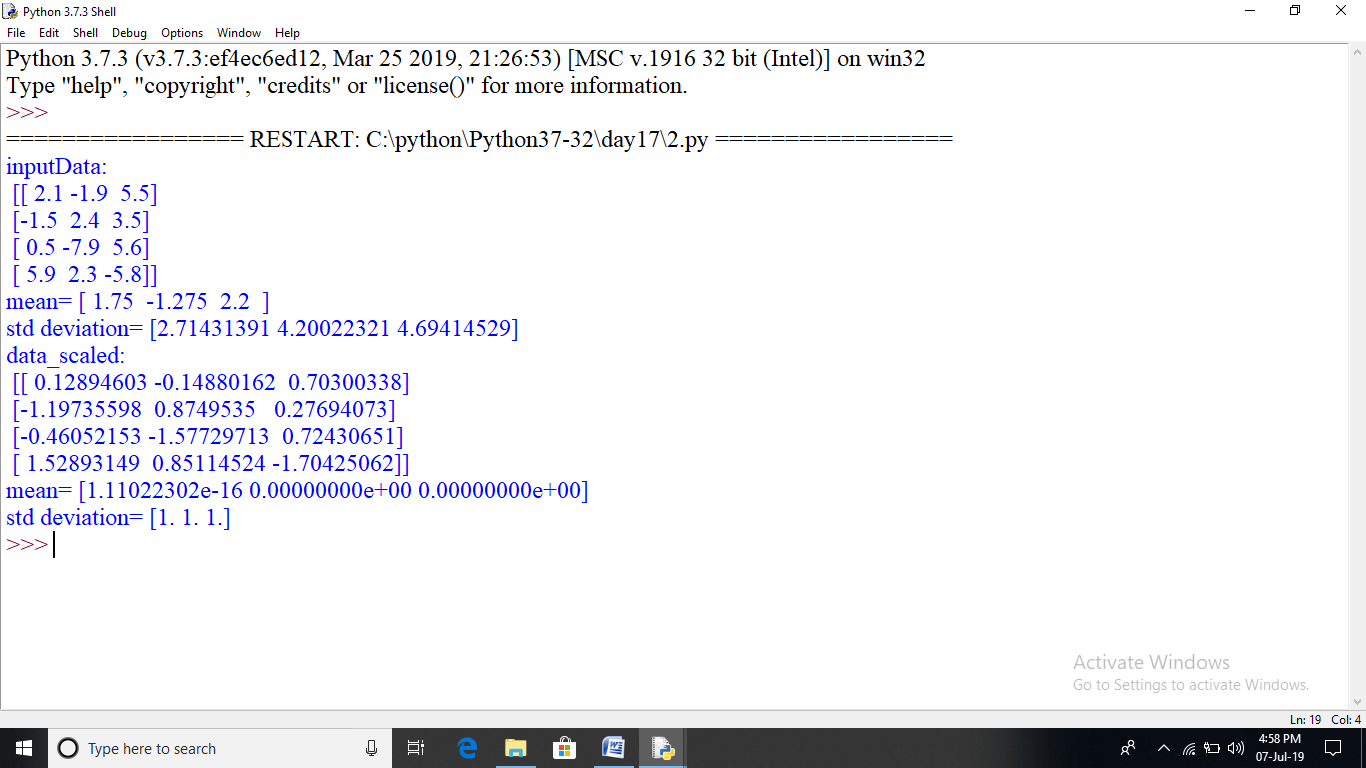
print("std deviation=",input\_data.std(axis=0))

data\_scaled=preprocessing.scale(input\_data)

print("data\_scaled:\n",data\_scaled)

print("mean=",data\_scaled.mean(axis=0))

print("std deviation=",data\_scaled.std(axis=0))



import matplotlib.pyplot as plt

xs=[2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25]

ys=[10,12,20,22,21,25,30,21,32,34,35,30,50,45,55,60,66,64,67,72,74,80,79,84]

print(len(xs),len(ys))

import numpy as np

def slope\_intercept(x\_val,y\_val):

x=np.array(x\_val)

y=np.array(y\_val)

m=(((np.mean(x)\*np.mean(y))-np.mean(x\*y))/

(np.mean(x)\*\*2-np.mean(x\*\*2)))

m=round(m,2)

b=(np.mean(y)-np.mean(x)\*m)

return m,b

m,b=slope\_intercept(xs,ys)

reg\_line=[(m\*x)+b for x in xs]

plt.scatter(xs,ys,color="red")

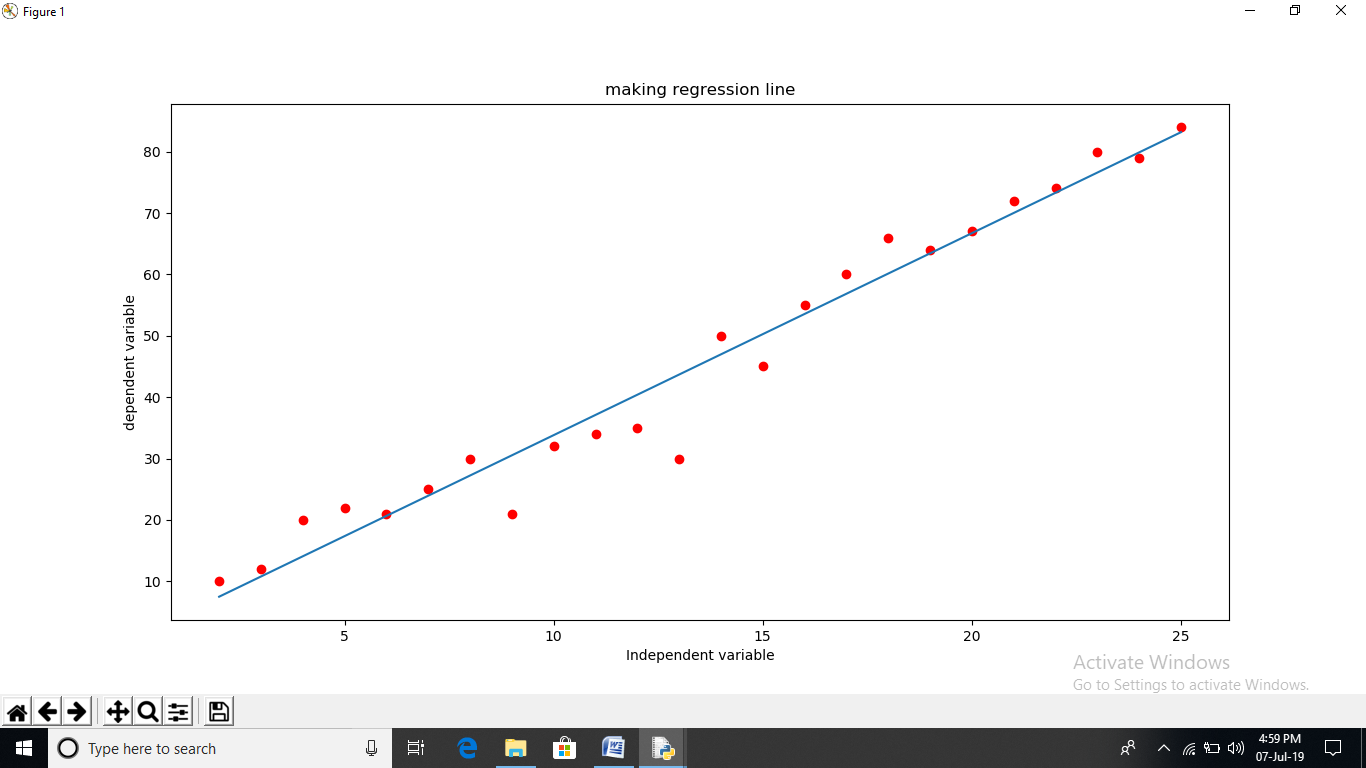
plt.plot(xs,reg\_line)

plt.ylabel("dependent variable")

plt.xlabel("Independent variable")

plt.title("making regression line")

plt.show()



from sklearn import tree

clf=tree.DecisionTreeClassifier()

#[height,hair,length,voice-pitch]

X=[[180,15,0],

[167,42,1],

[136,35,1],

[174,15,0],

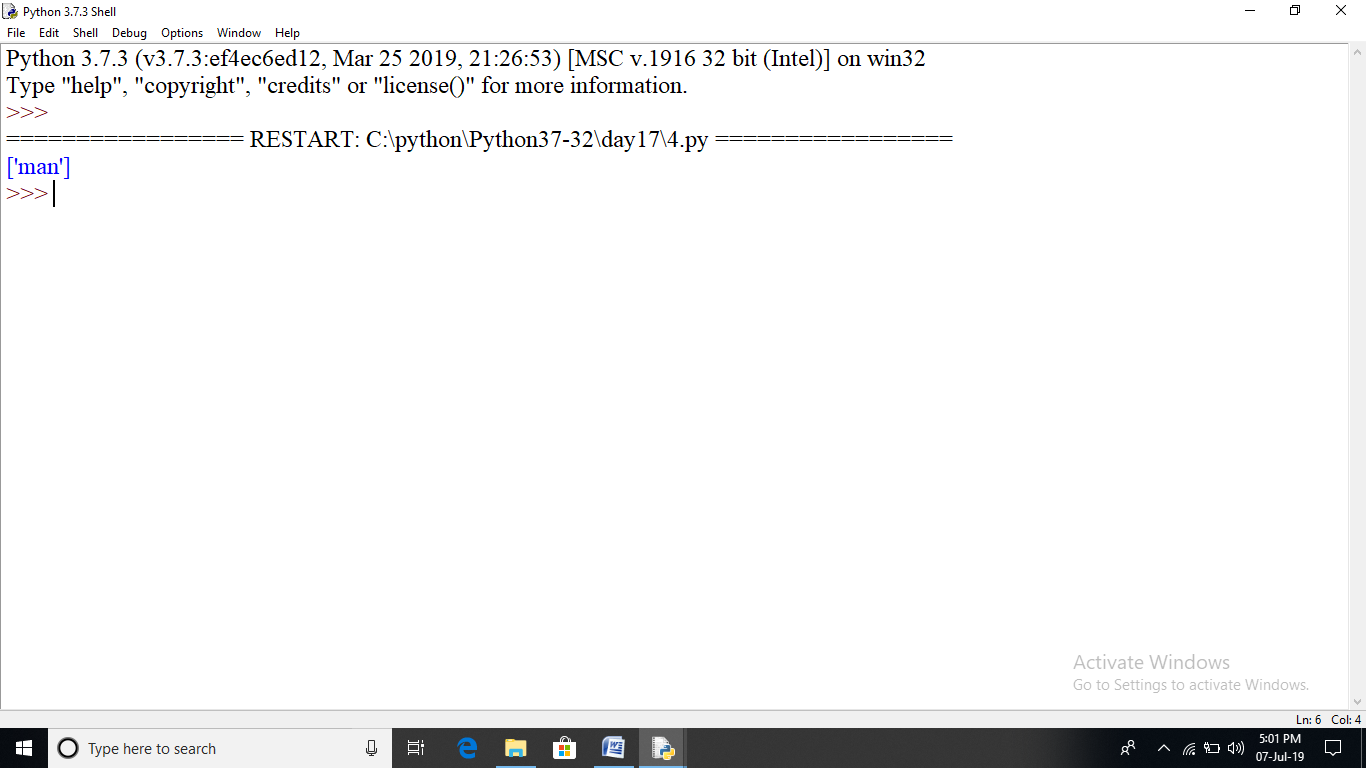
[141,28,1]]

Y=['man','woman','woman','man','woman']

clf=clf.fit(X,Y)

prediction=clf.predict([[120,10,1]])

print(prediction)



import sklearn

from sklearn.datasets import load\_breast\_cancer

data=load\_breast\_cancer()

label\_names=data['target\_names']

labels=data['target']

feature\_names=data['feature\_names']

features=data['data']

print("label names:",label\_names)

print(labels[0])

print(feature\_names[0])

print(features[0])

#forganizing data into sets

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

train,test,train\_labels,test\_labels=train\_test\_split(features,labels,test\_size=0.40,random\_state=42)

#building the model

#native bayes algo

from sklearn.naive\_bayes import GaussianNB

gnb=GaussianNB()

model=gnb.fit(train,train\_labels)

#evaluating the model and its accuracy

preds=gnb.predict(test)

print(preds)

from sklearn.metrics import accuracy\_score

print(accuracy\_score(test\_labels,preds))

