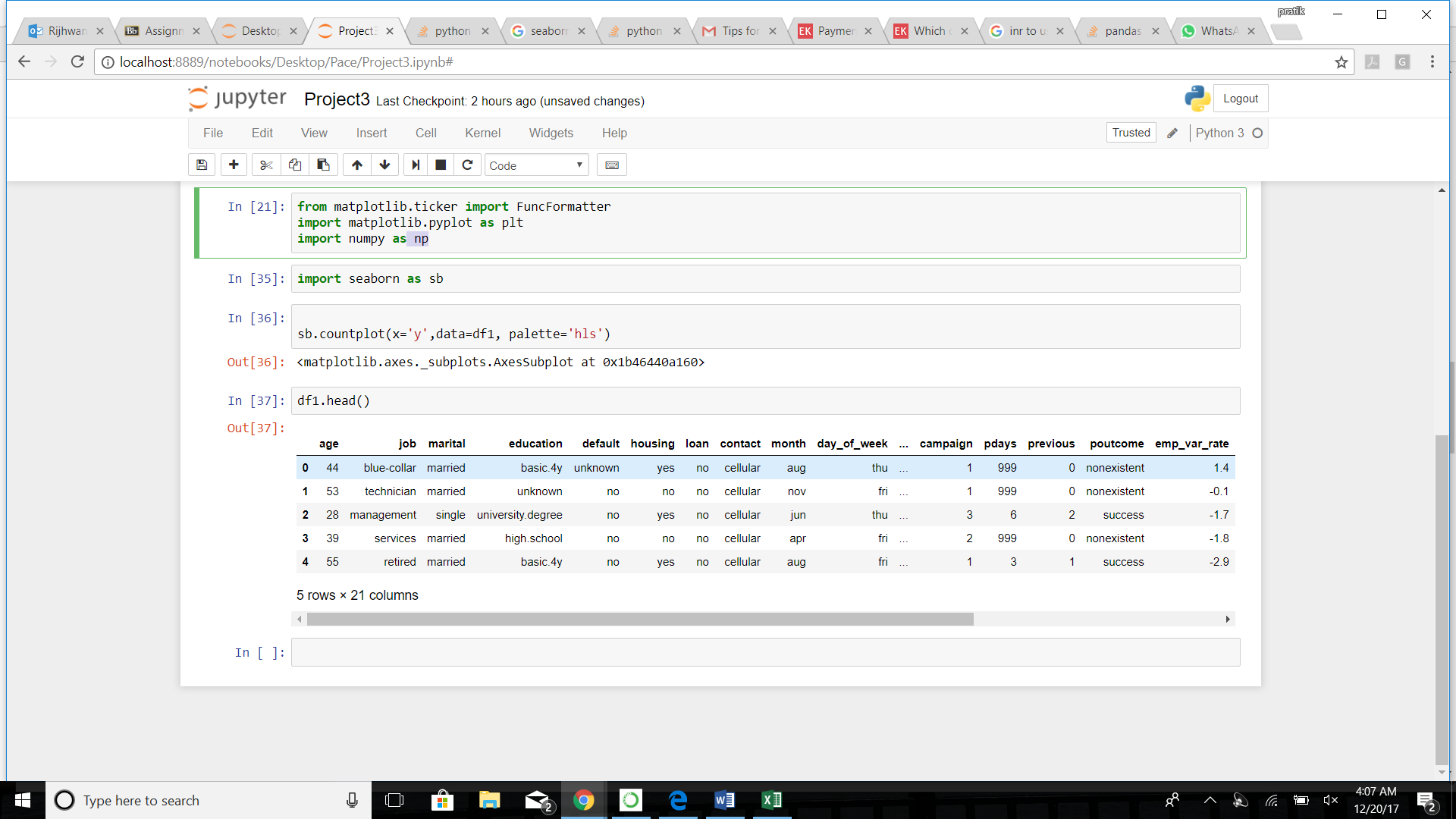
# IMPORT DATA

import pandas as pd

df1=pd.read\_csv("C:/Users/prati/Desktop/banking.csv")

print(df1)



from matplotlib.ticker import FuncFormatter

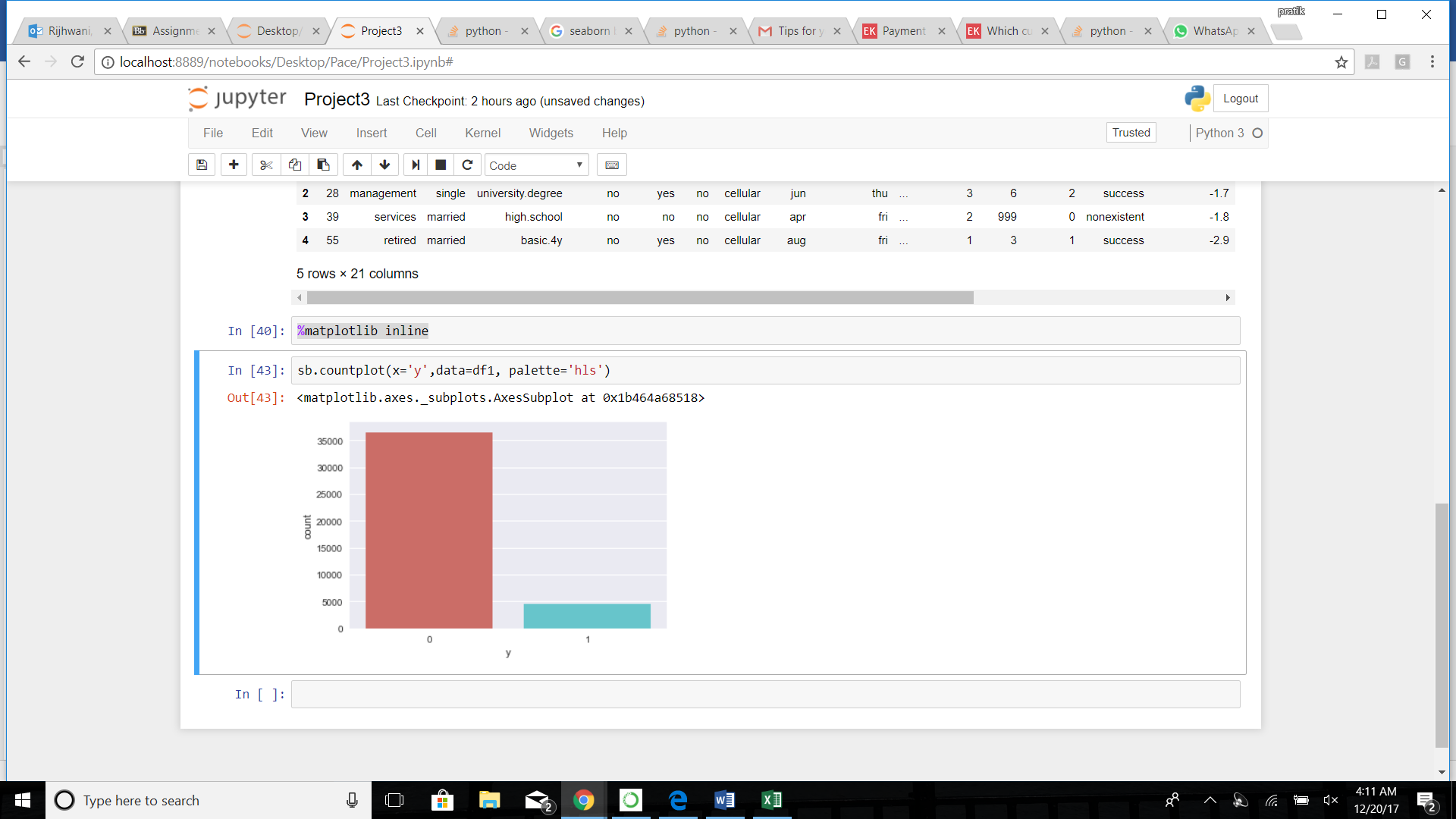
import matplotlib.pyplot as plt

import numpy as np

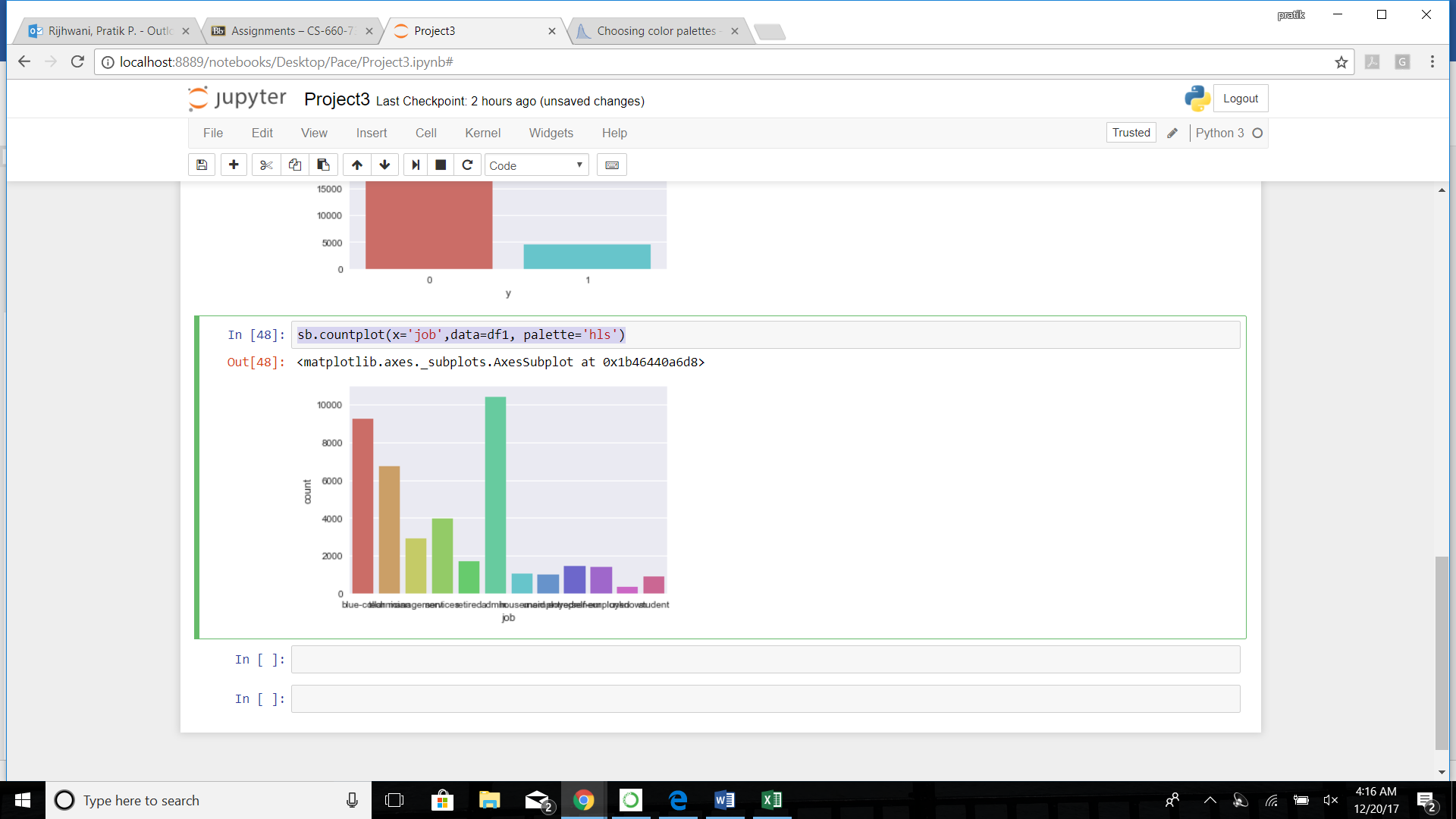
import seaborn as sb

%matplotlib inline

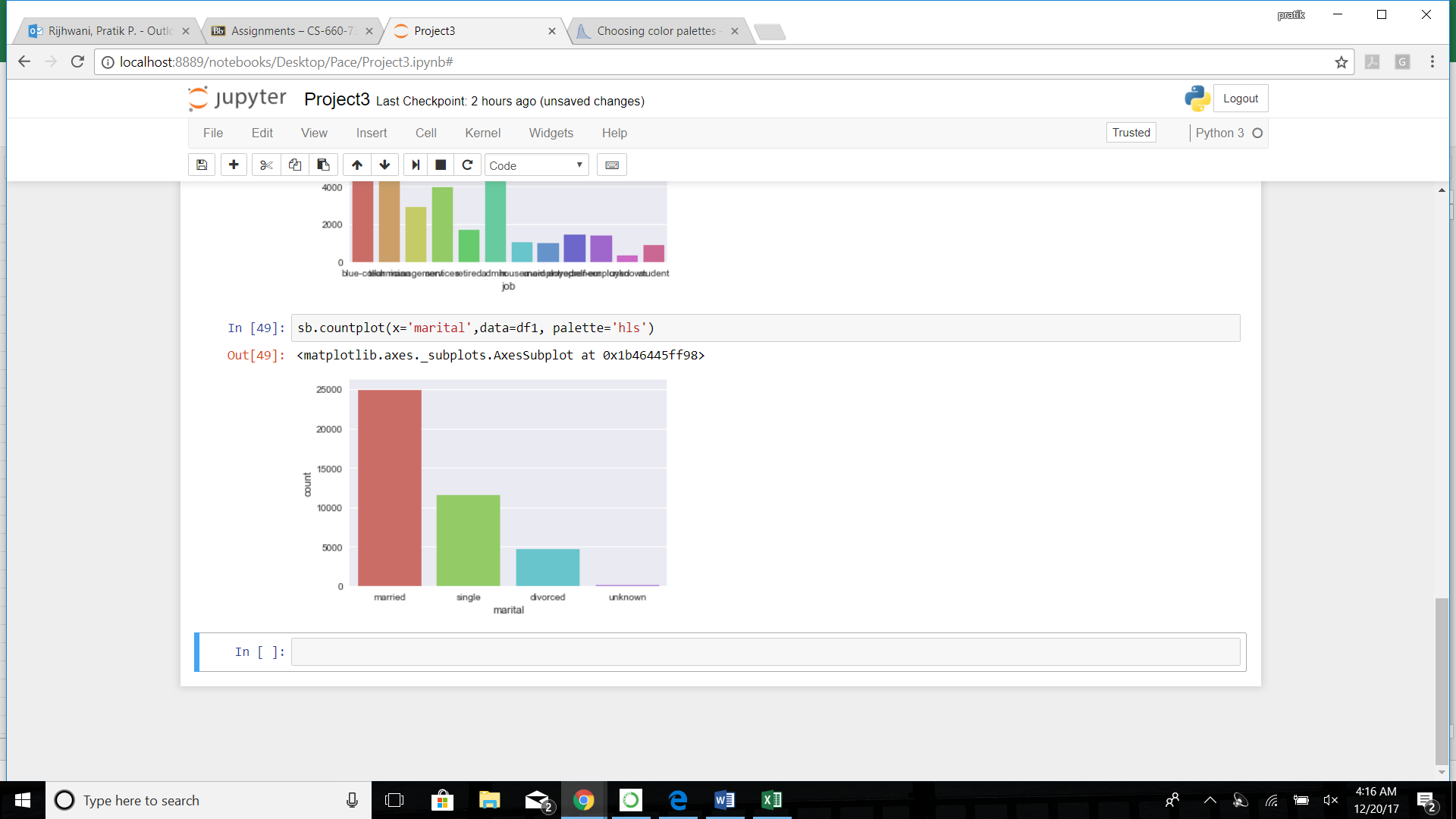
sb.countplot(x='y',data=df1, palette='hls')



sb.countplot(x='job',data=df1, palette='hls')

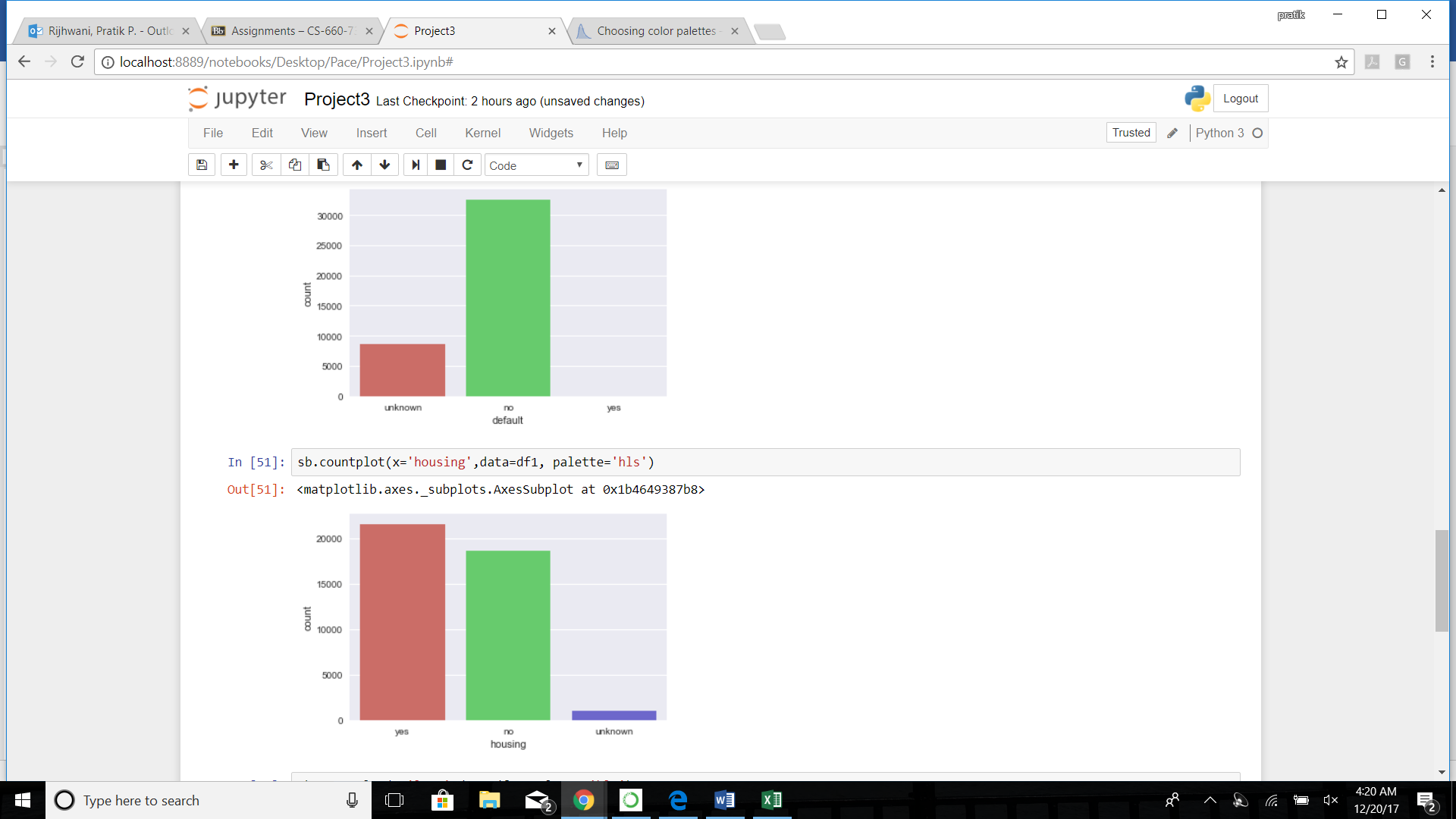


sb.countplot(x='marital',data=df1, palette='hls')



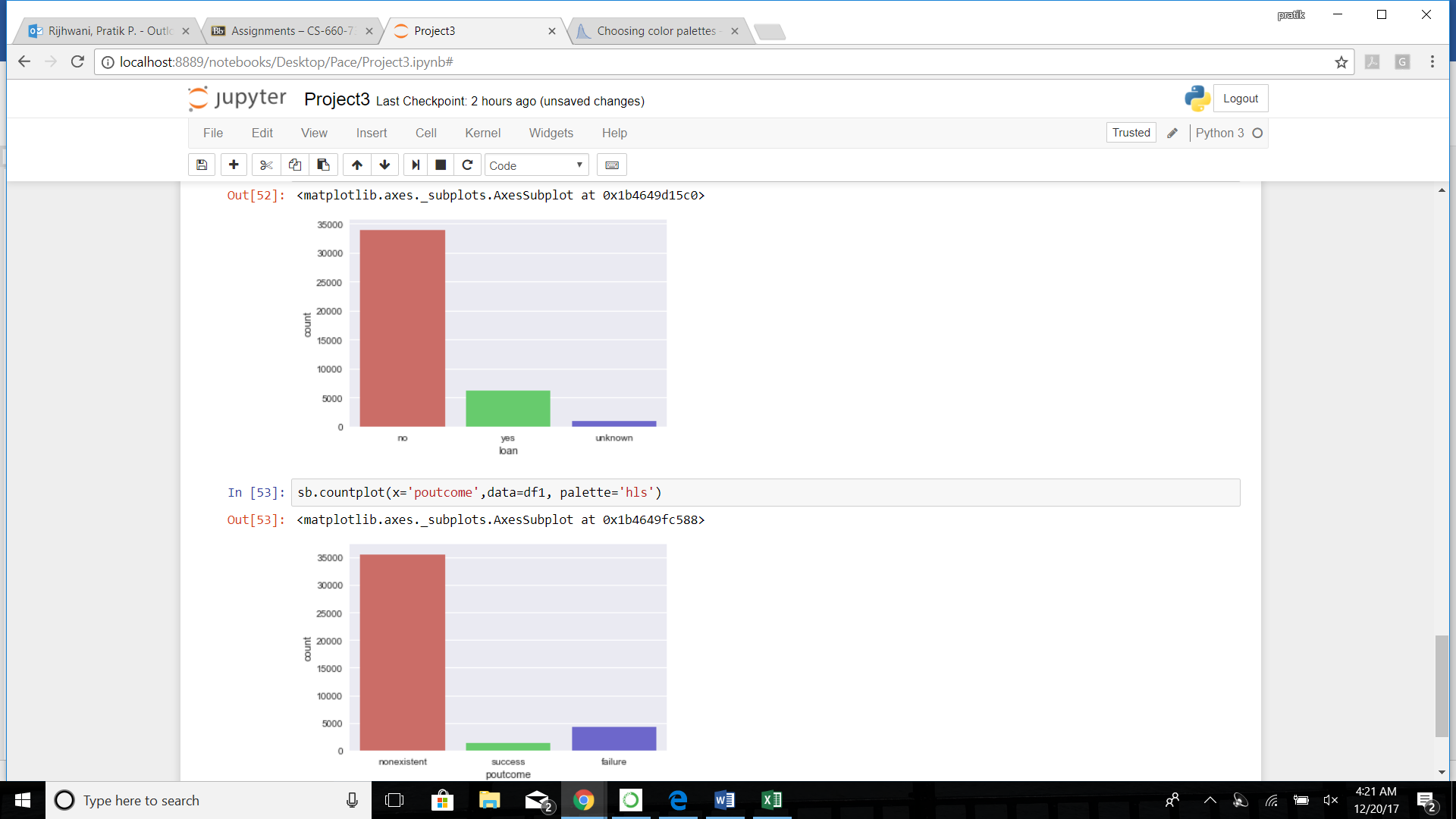
sb.countplot(x='default',data=df1, palette='hls')

sb.countplot(x='housing',data=df1, palette='hls')



sb.countplot(x='loan',data=df1, palette='hls')

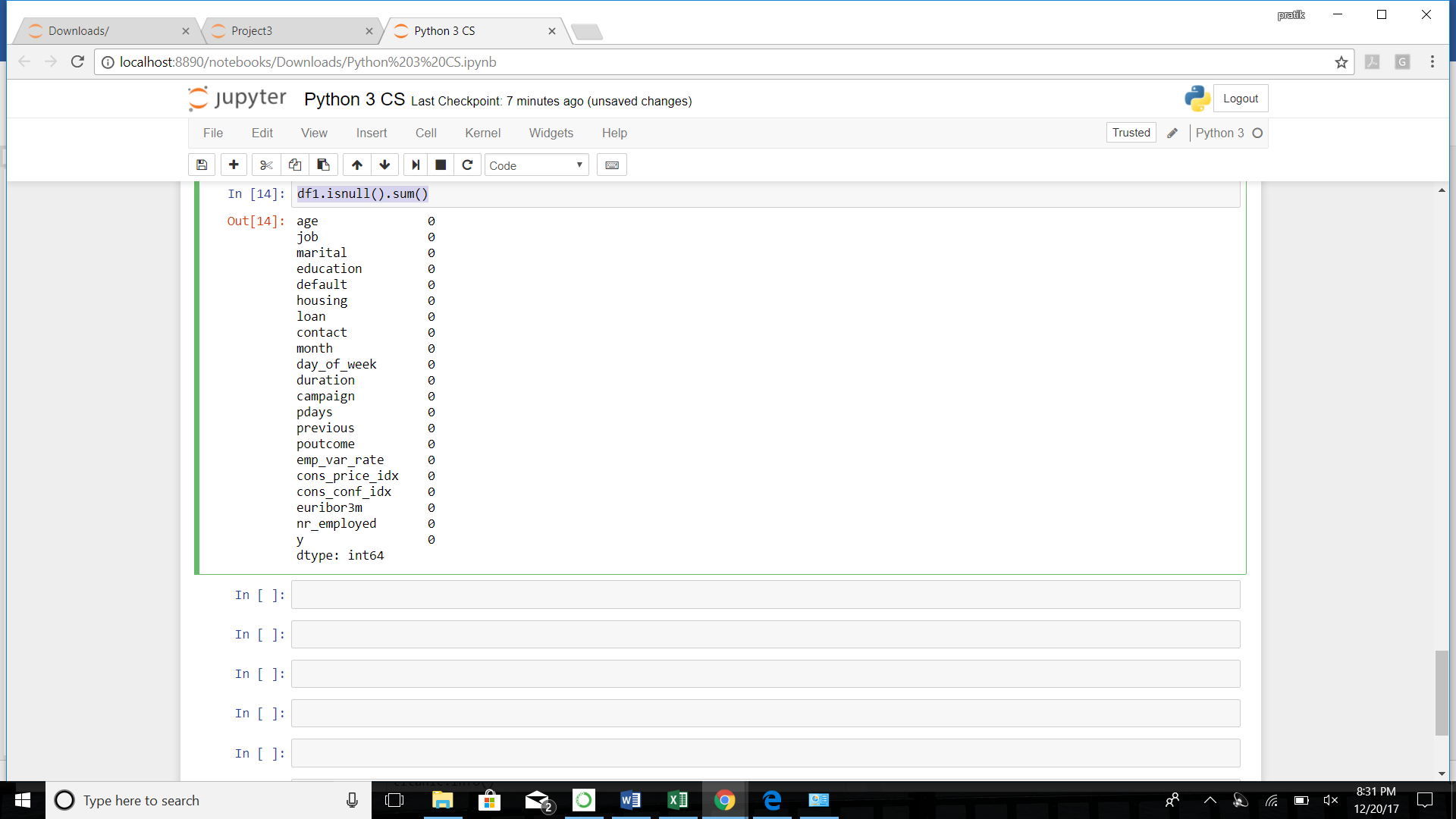
sb.countplot(x='poutcome',data=df1, palette='hls')



#check null values

df1.isnull().sum()

No null values



# create dummy variables

df\_job = pd.get\_dummies(df1['job'], drop\_first=True)

df\_marital = pd.get\_dummies(df1['marital'], drop\_first=True)

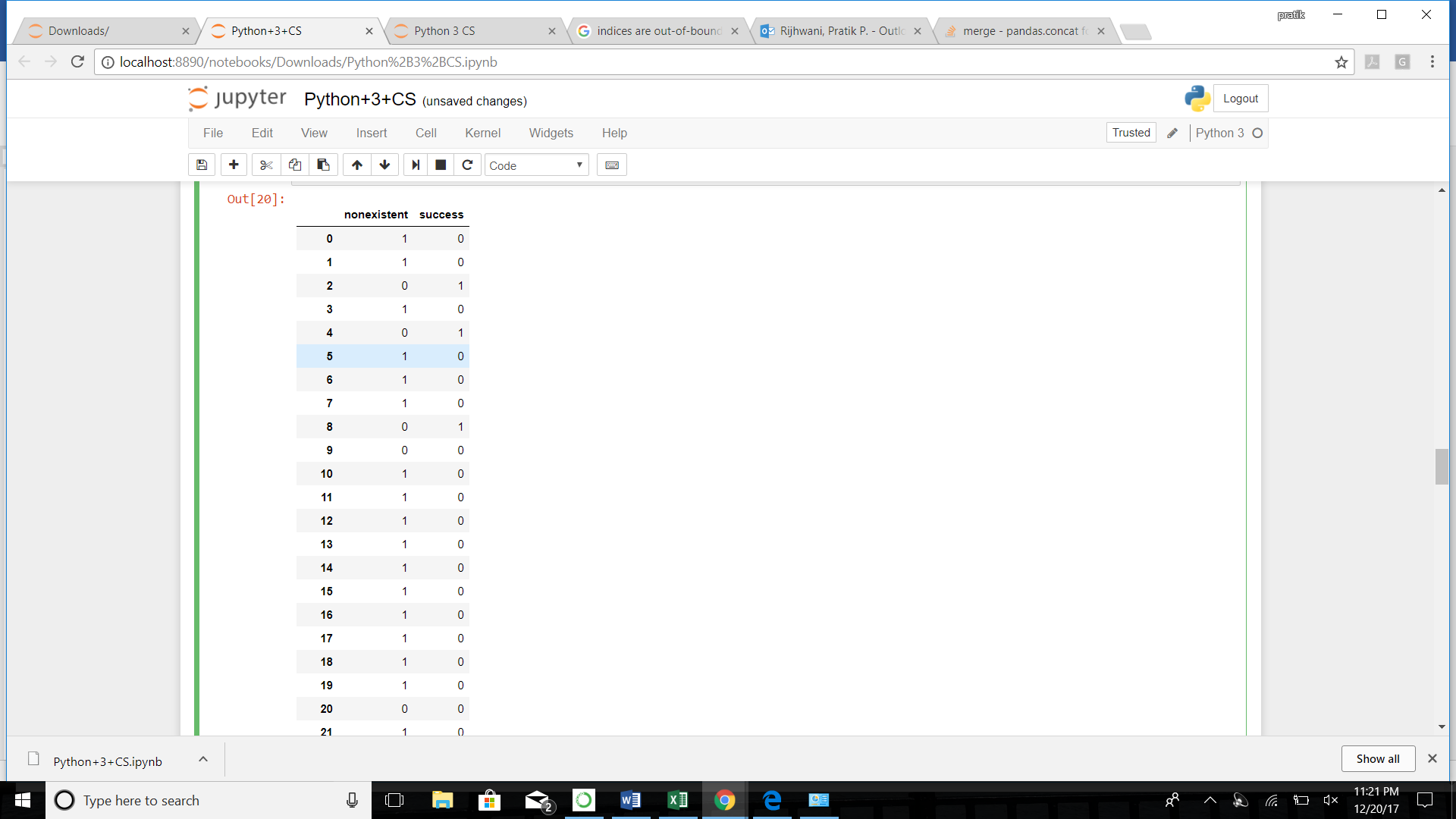
df\_default = pd.get\_dummies(df1['default'], drop\_first=True)

df\_housing = pd.get\_dummies(df1['housing'], drop\_first=True)

df\_loan = pd.get\_dummies(df1['loan'], drop\_first=True)

df\_pout=pd.get\_dummies(df1['poutcome'], drop\_first=True)

#sample poutcome dummies



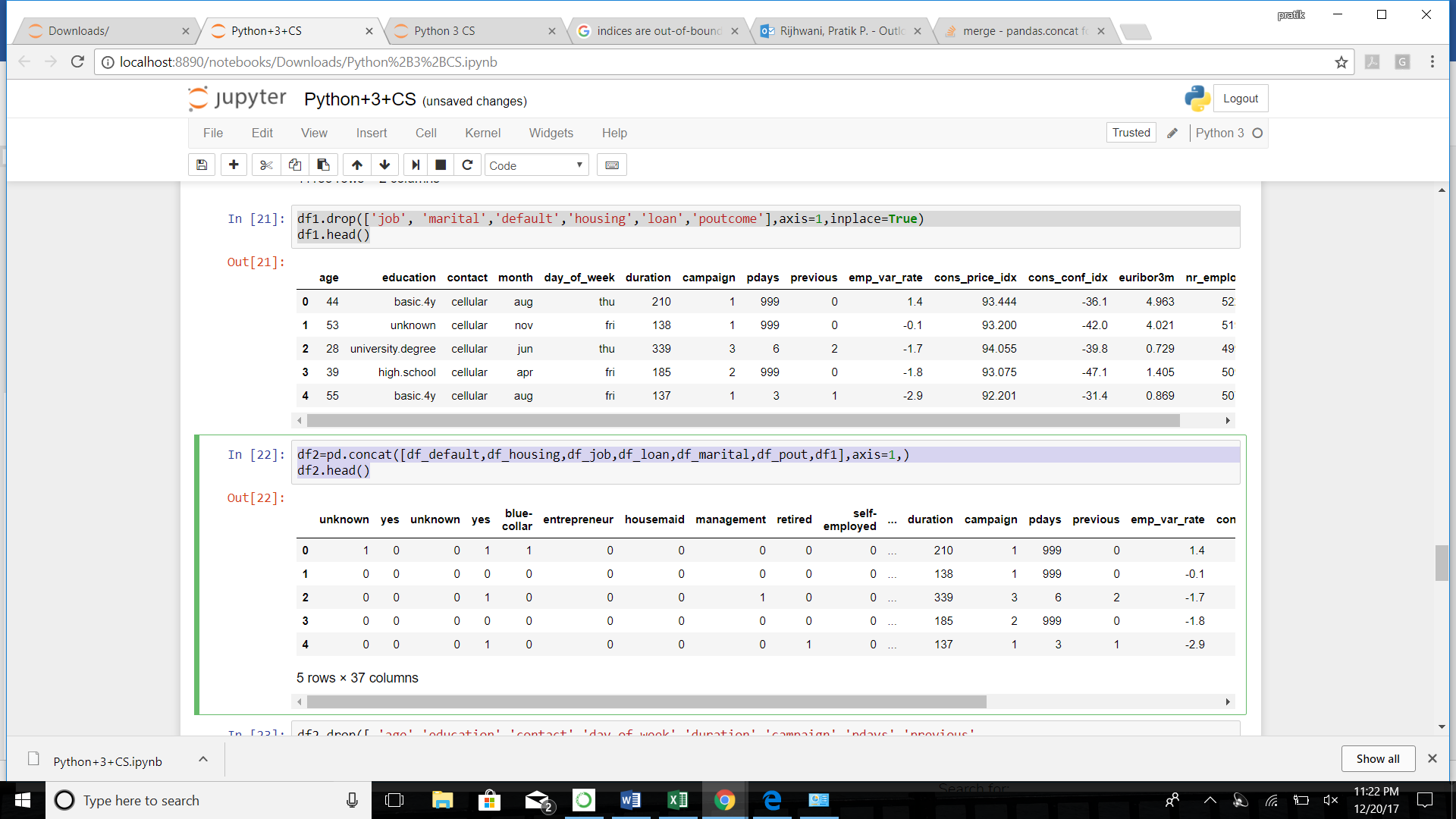
#droppping old columns and combining the dataframes

df1.drop(['job', 'marital','default','housing','loan','poutcome'],axis=1,inplace=True)

df1.head()

df2=pd.concat([df\_default,df\_housing,df\_job,df\_loan,df\_marital,df\_pout,df1],axis=1,)

df2.head()

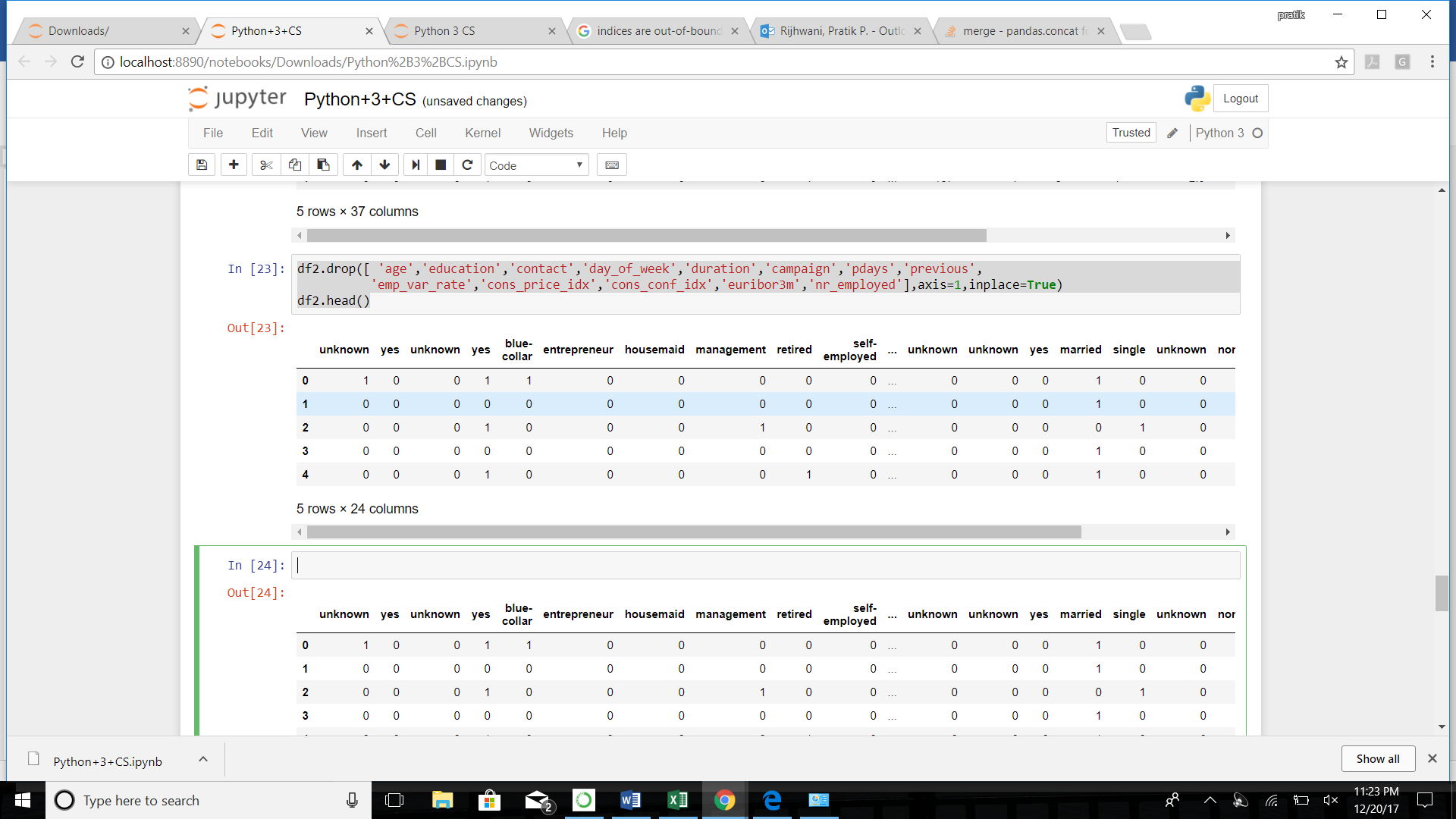


#dropping irrelevant columns

df2.drop([ 'age','education','contact','day\_of\_week','duration','campaign','pdays','previous',

'emp\_var\_rate','cons\_price\_idx','cons\_conf\_idx','euribor3m','nr\_employed'],axis=1,inplace=True)

df2.head()

\

# import packages

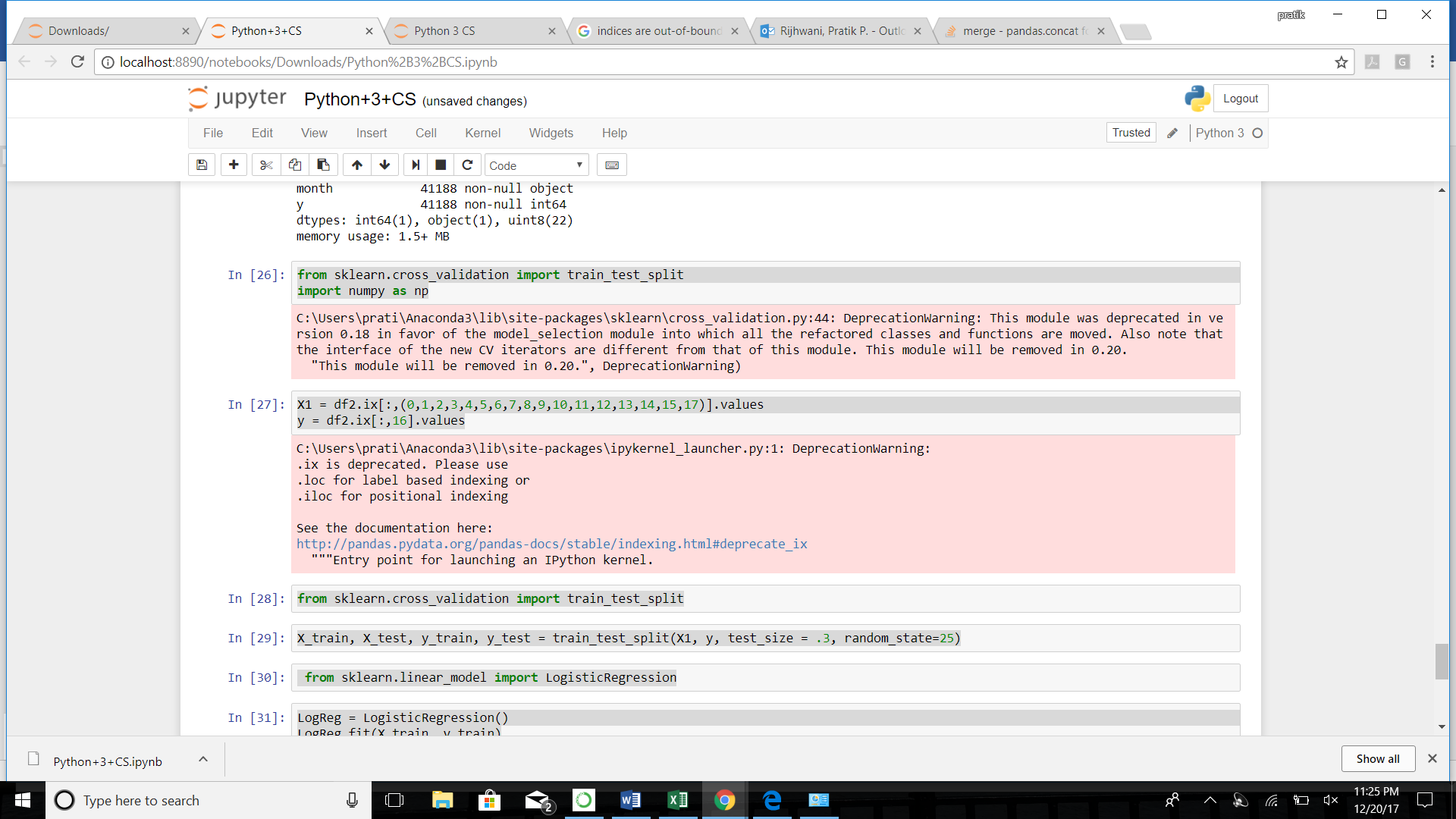
from sklearn.cross\_validation import train\_test\_split

import numpy as np

# deploying

X1 = df2.ix[:,(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,17)].values

y = df2.ix[:,16].values



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

# split

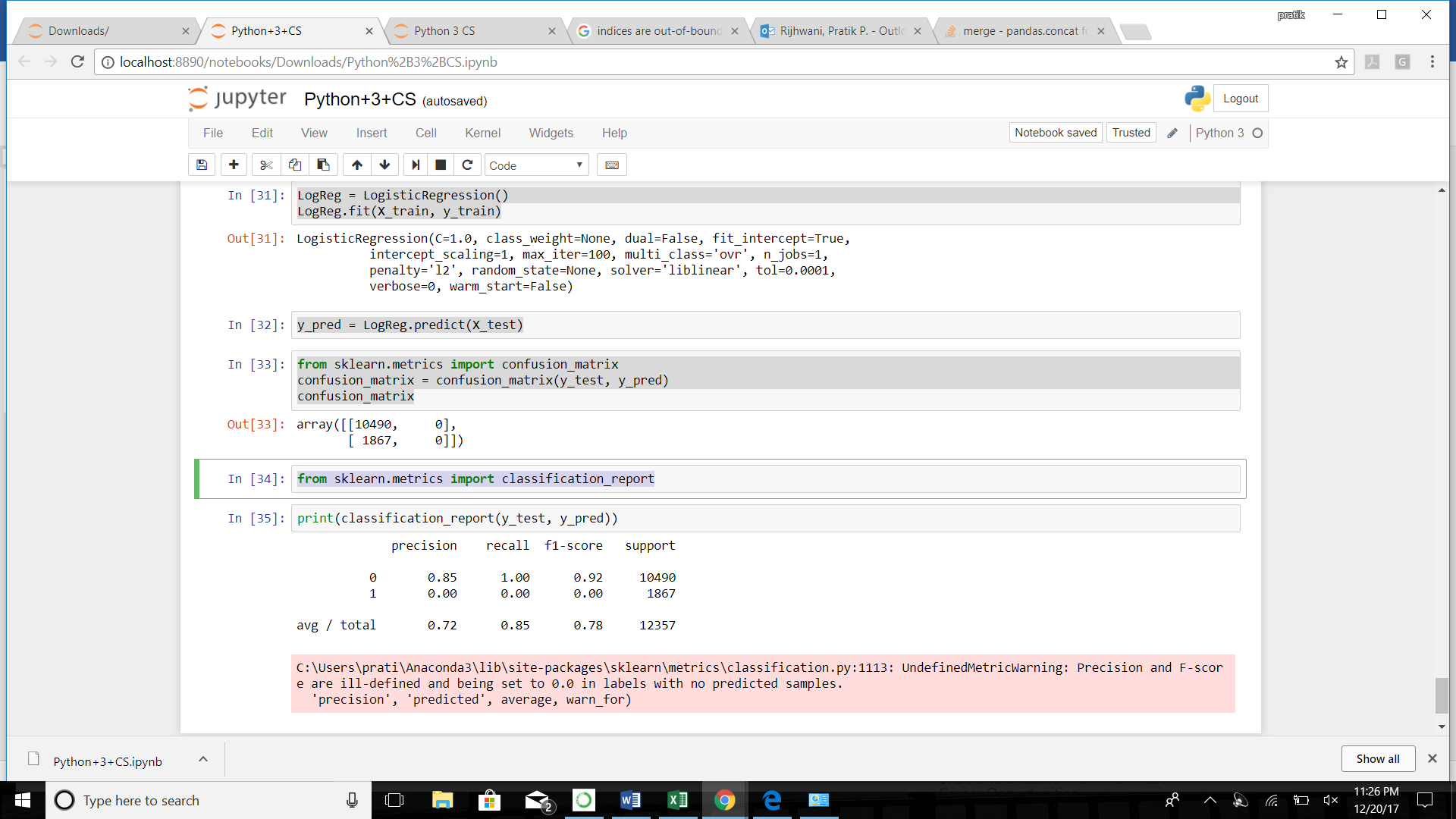
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X1, y, test\_size = .3, random\_state=25)

#logistic

from sklearn.linear\_model import LogisticRegression

LogReg = LogisticRegression()

LogReg.fit(X\_train, y\_train)

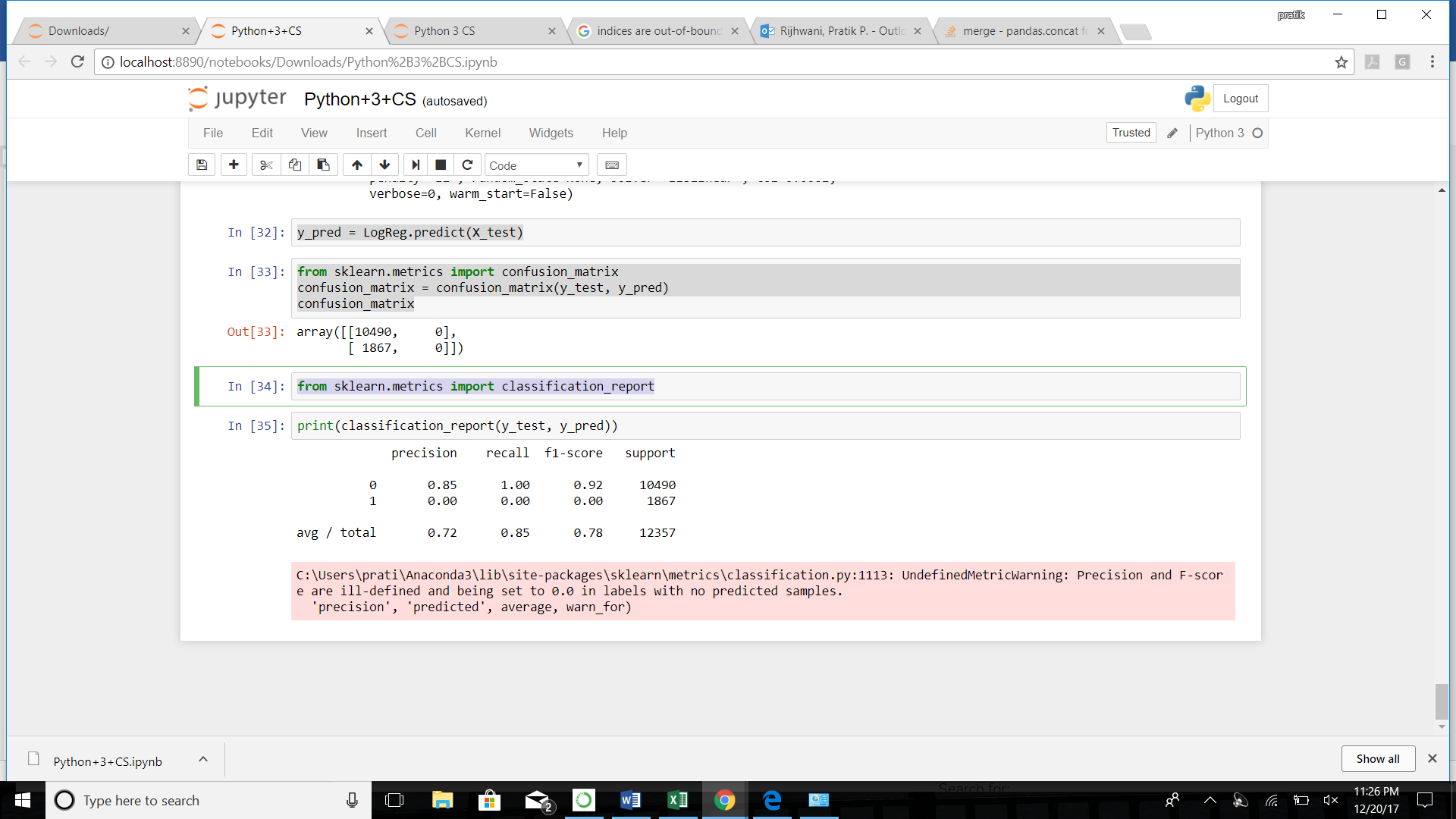


y\_pred = LogReg.predict(X\_test)

from sklearn.metrics import confusion\_matrix

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

confusion\_matrix



from sklearn.metrics import classification\_report

