# Internship Project : Loan status Prediction

**SUBMITTED BY:**

**NAME: KAUSHAL JAIN**

**EMAIL ID:** [**jainkaushal899@gmail.com**](mailto:jainkaushal899@gmail.com)

**BATCH: ML WITH PYTHON**

**CERTIFICATION CODE: TCRIG02R65**

**GROUP: OWN**

**QUESTION:**

Please perform the below steps in a Google Colab or Jupyter Notebook as per your convenience:

**In this assignment, we need to predict Loan status detection.**

**Features:**

Company wants to automate the loan eligibility process (real time) based on customer detail provided while filling online application form. These details are Gender, Marital Status, Education, Number of Dependents, Income, Loan Amount, Credit History and others. To automate this process, they have given a problem to identify the customers segments, those are eligible for loan amount so that they can specifically target these customers. Here they have provided a partial data set.

We will create a model with the following steps:

● Import the relevant packages

● Download and explore the dataset

● Prepare the dataset for training

● Use any prediction algorithm based upon the EDA

● Train the model to fit the data

● Make predictions using the trained model

● Create a test case and generate a predicted result from the system

**SOURCE CODE AND OUTPUTS:**

# Importing Required Libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

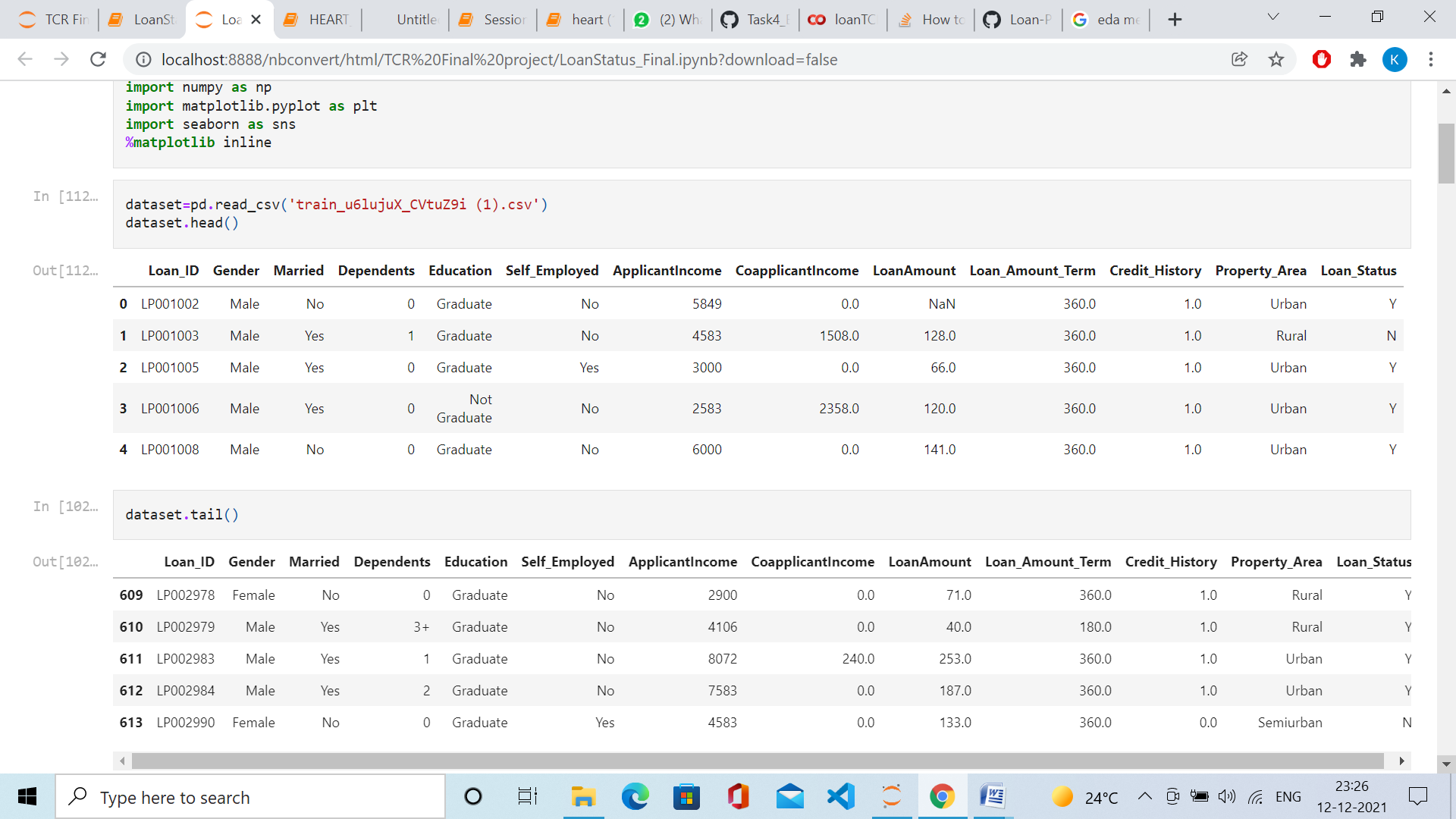
%matplotlib inline

import warnings

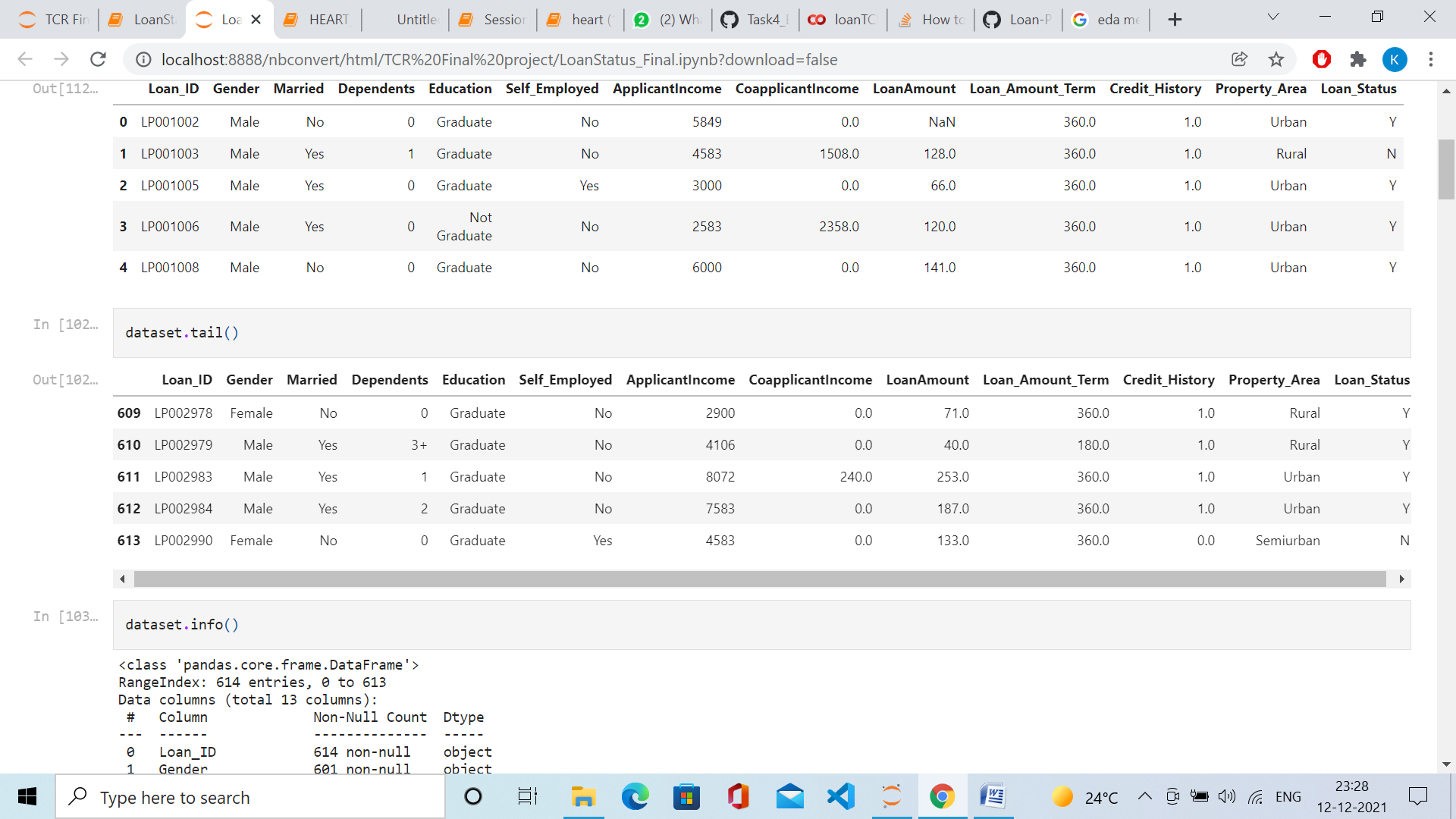
warnings.filterwarnings('ignore')

#Importing the Dataset

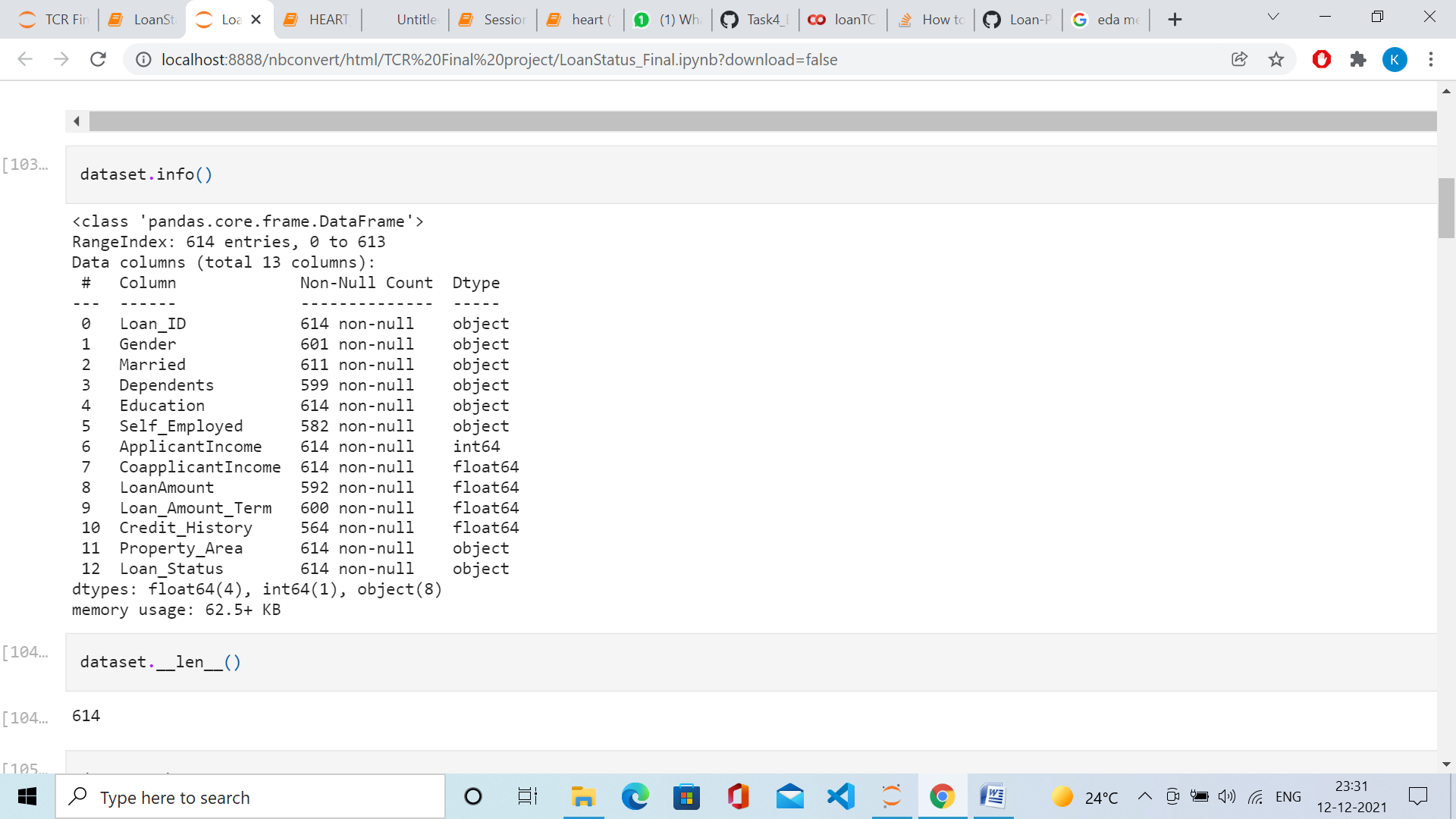
dataset=pd.read\_csv('train\_u6lujuX\_CVtuZ9i (1).csv')

dataset.head()

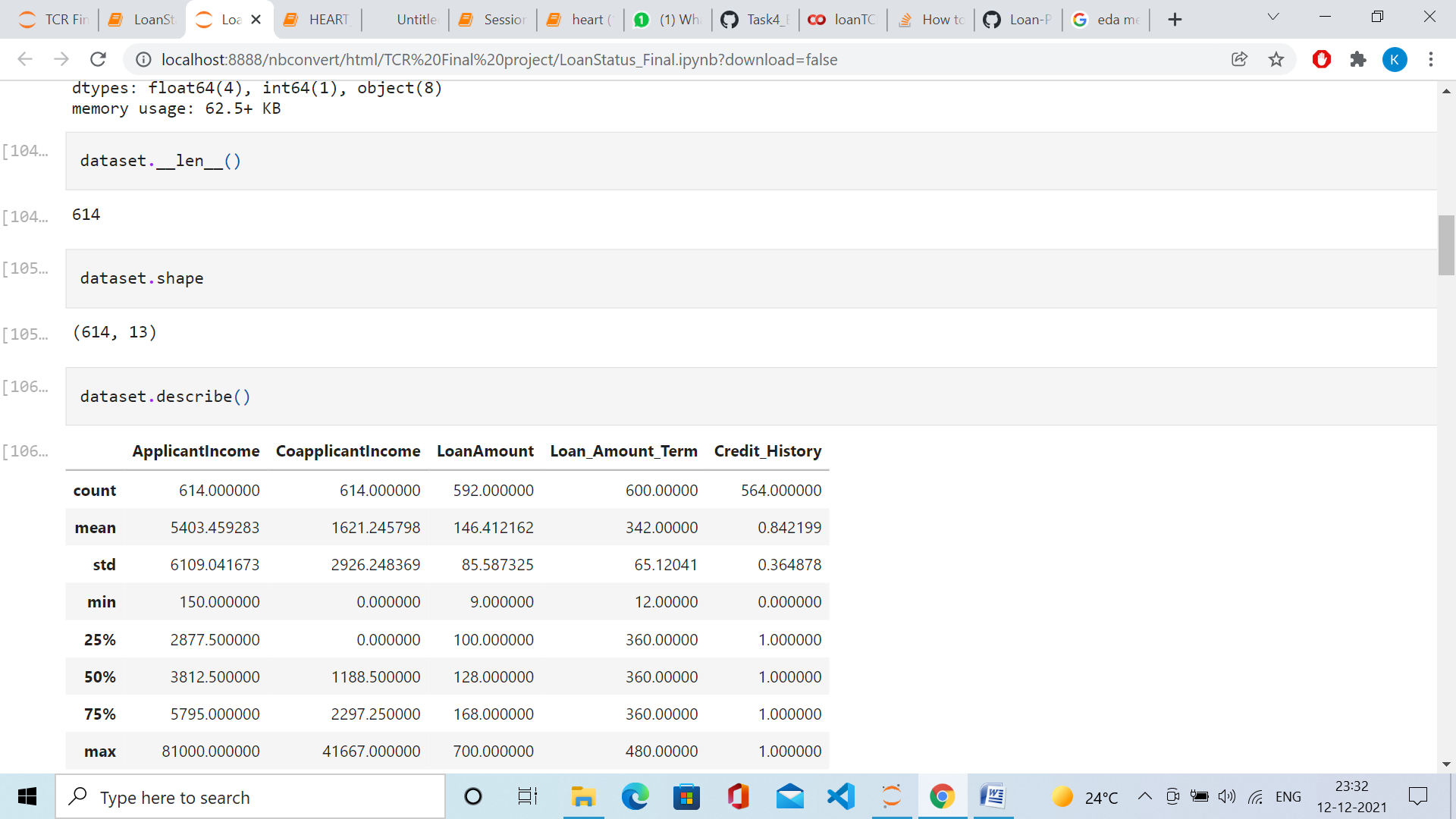
dataset**.**tail()



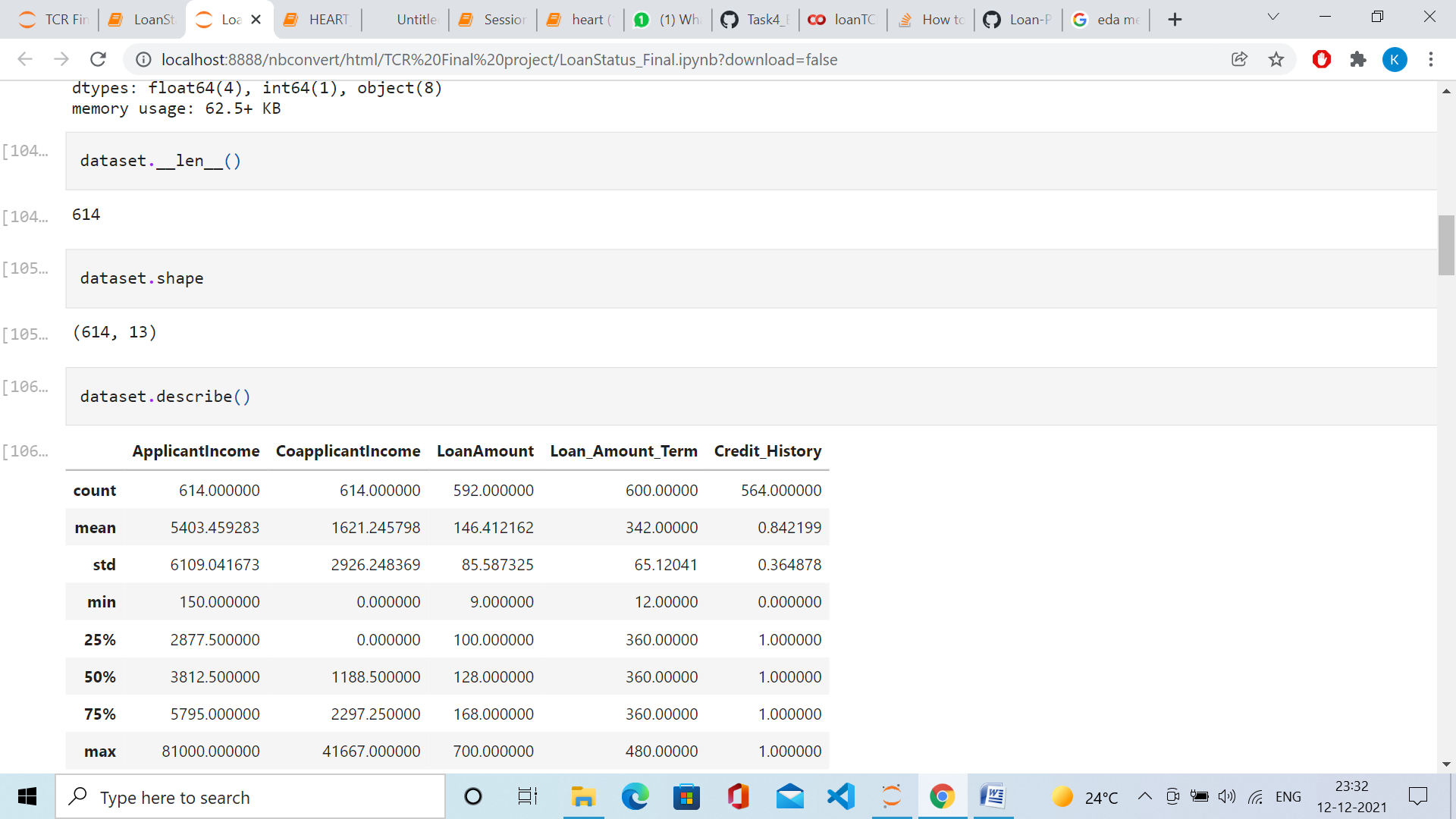
dataset.info()



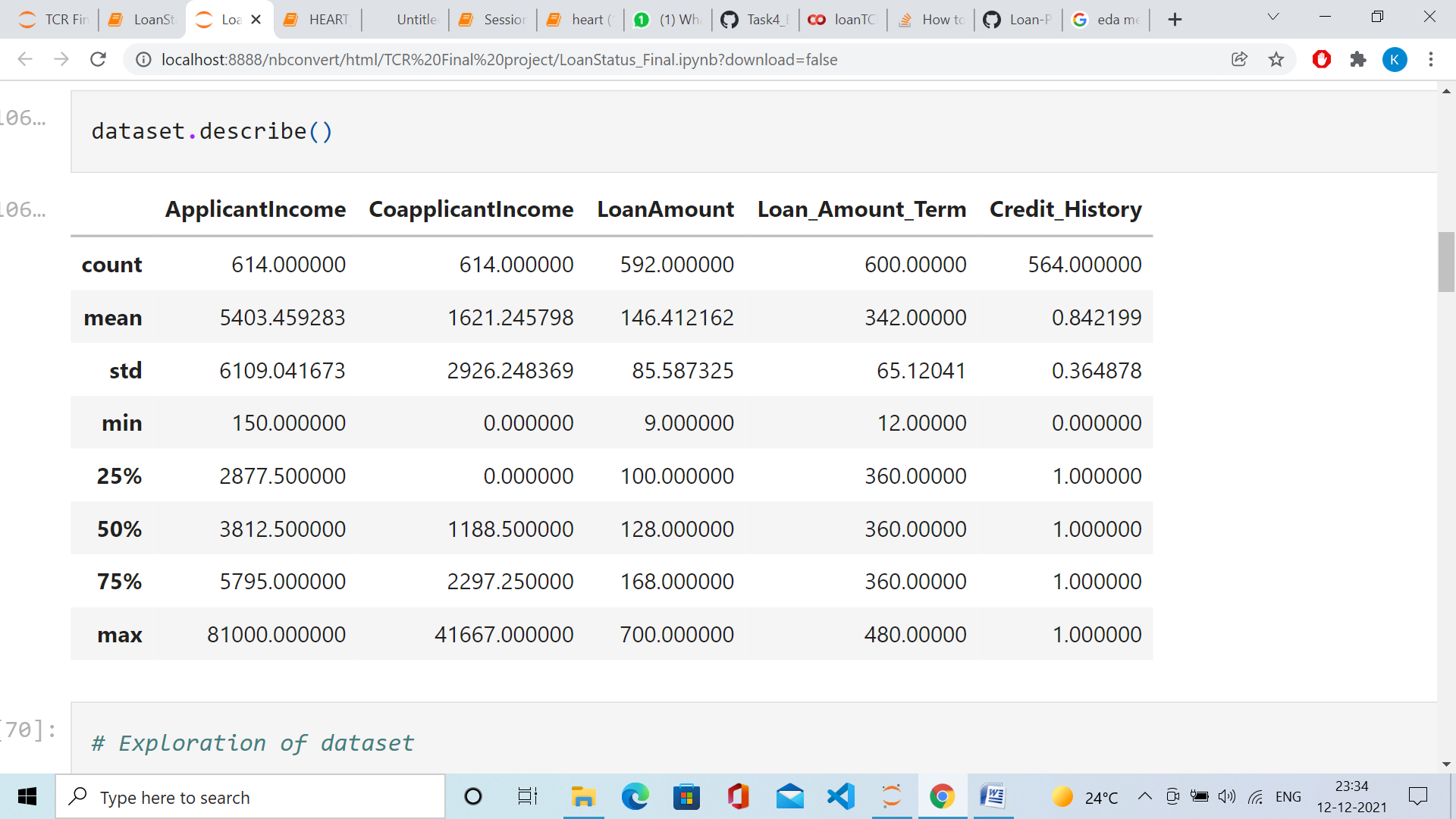
dataset.\_\_len\_\_()



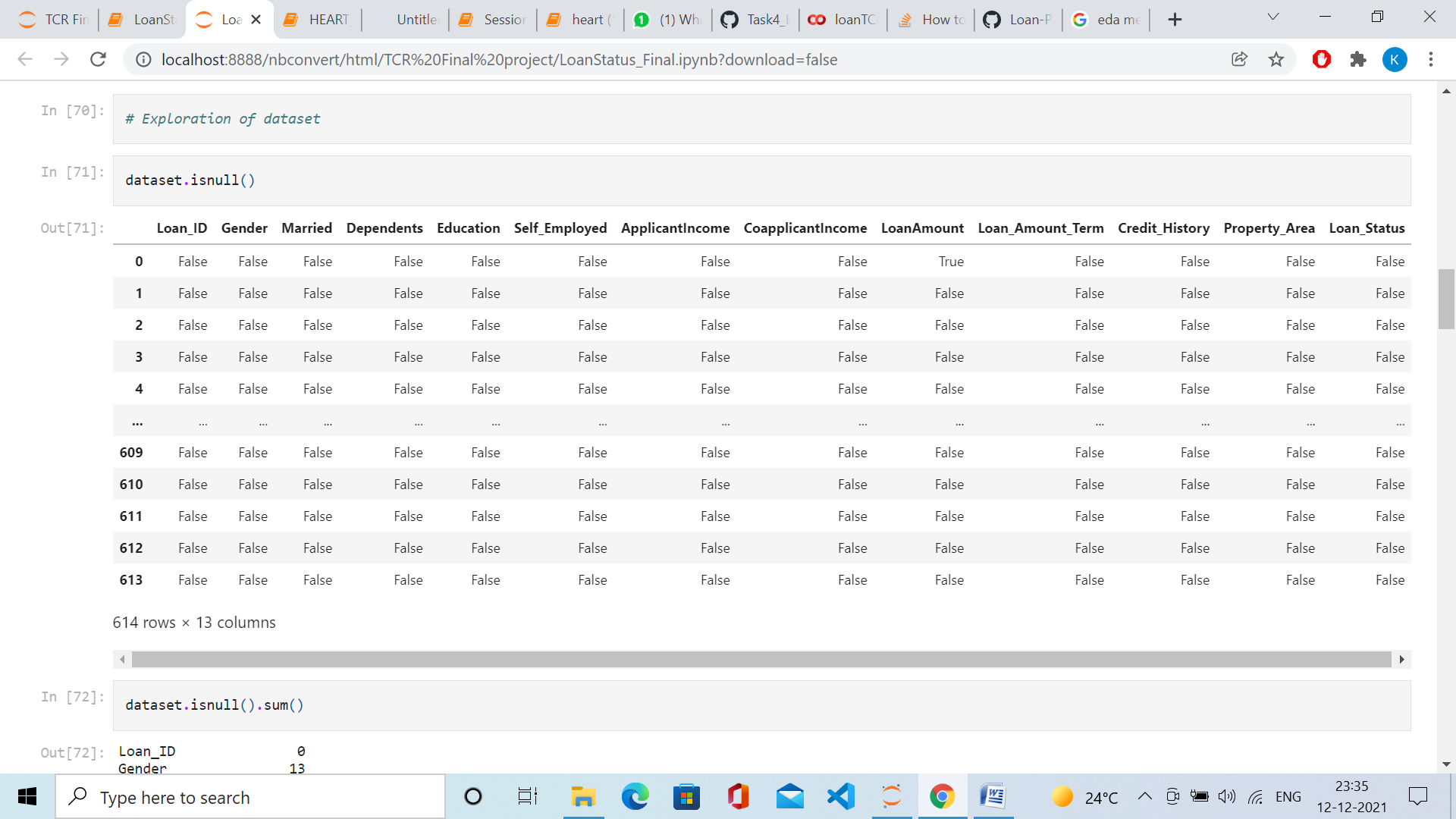
dataset.shape



dataset.describe()

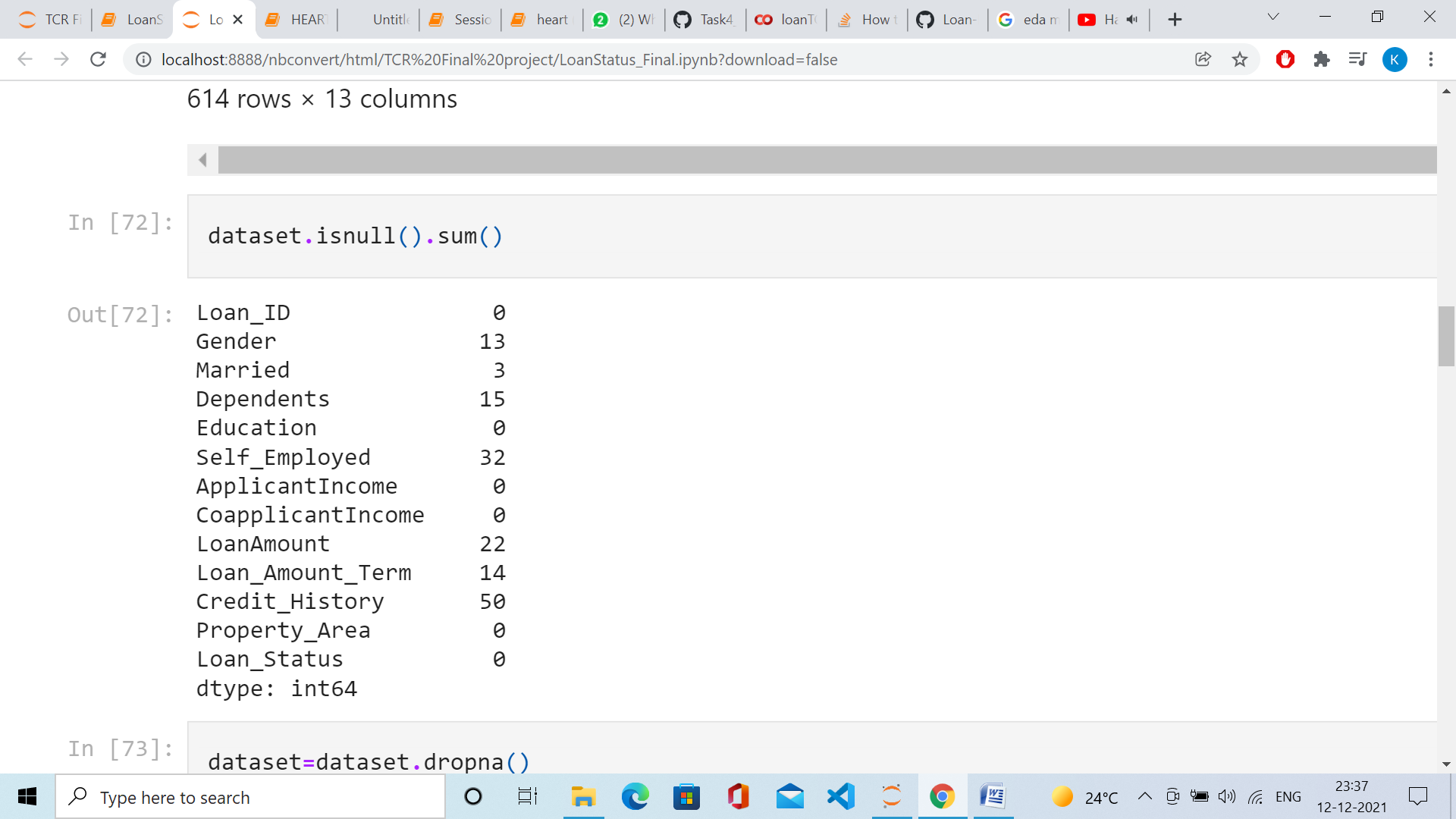


dataset.isnull()



# Checking total number of NULL values

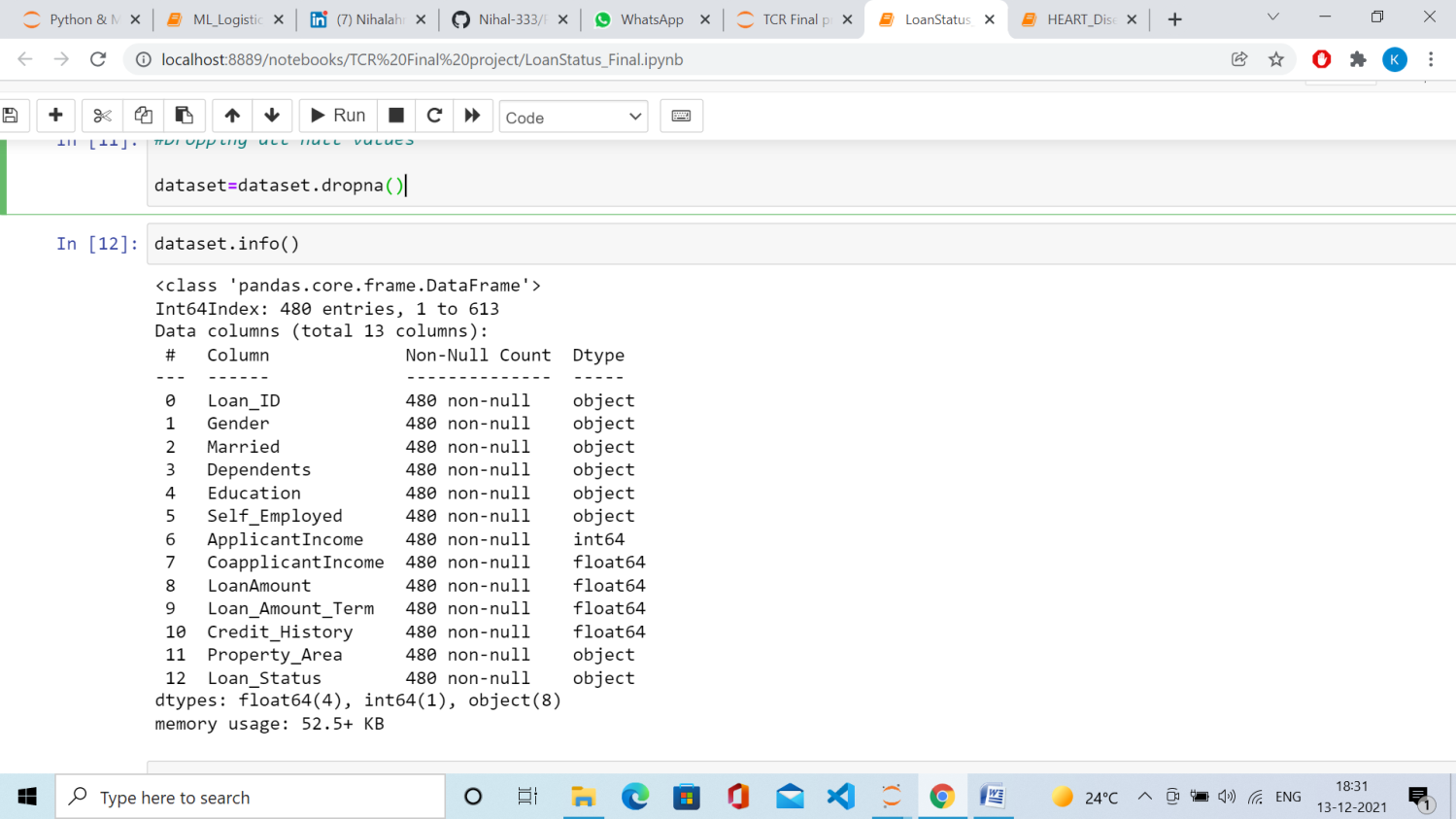
dataset.isnull().sum()



#Dropping all null values

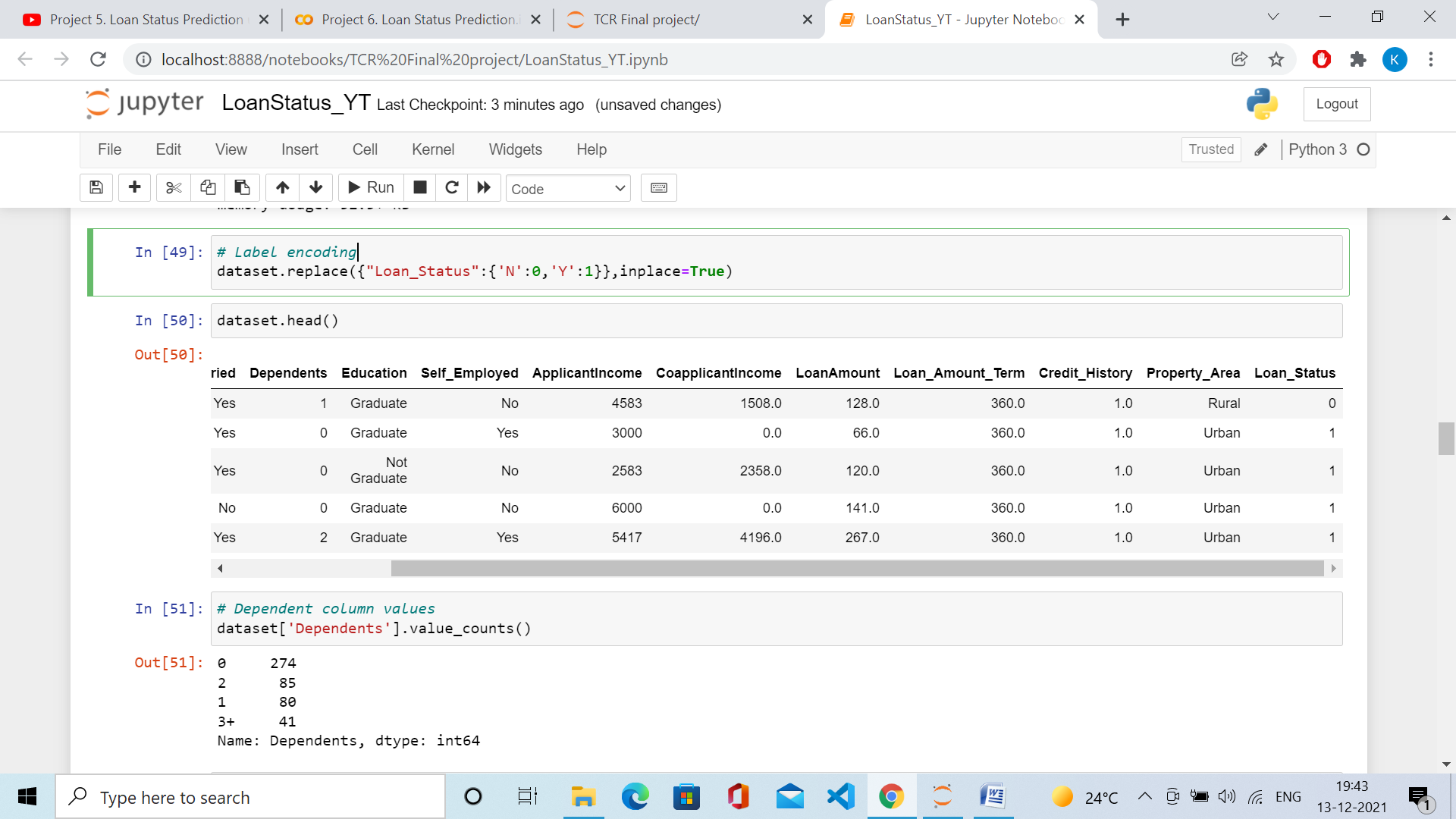
dataset=dataset.dropna()

dataset.info()



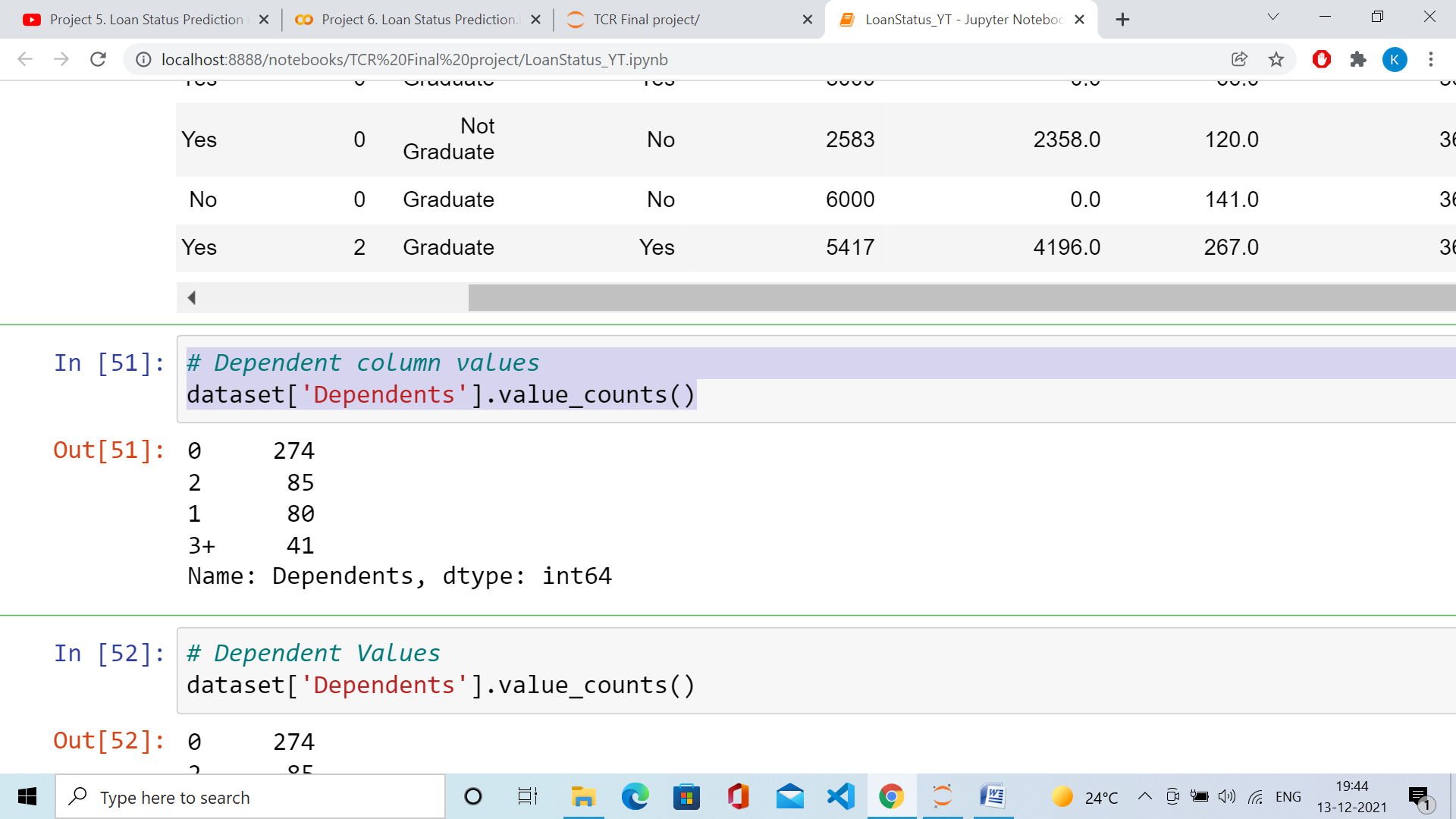
# Label encoding

dataset.replace({"Loan\_Status":{'N':0,'Y':1}},inplace=True)



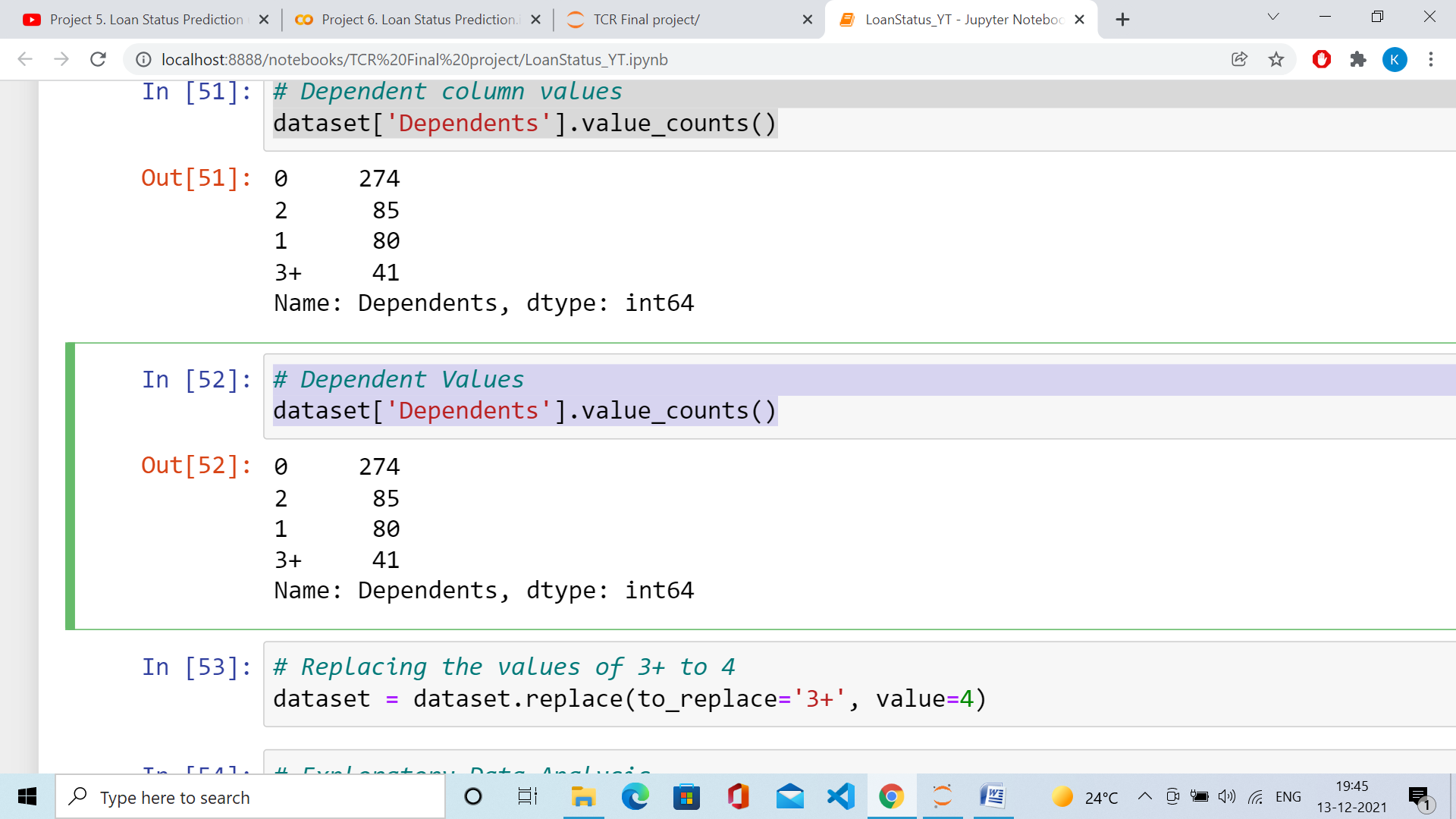
# Dependent column values

dataset['Dependents'].value\_counts()



# Dependent Values

dataset['Dependents'].value\_counts()



# Replacing the values of 3+ to 4

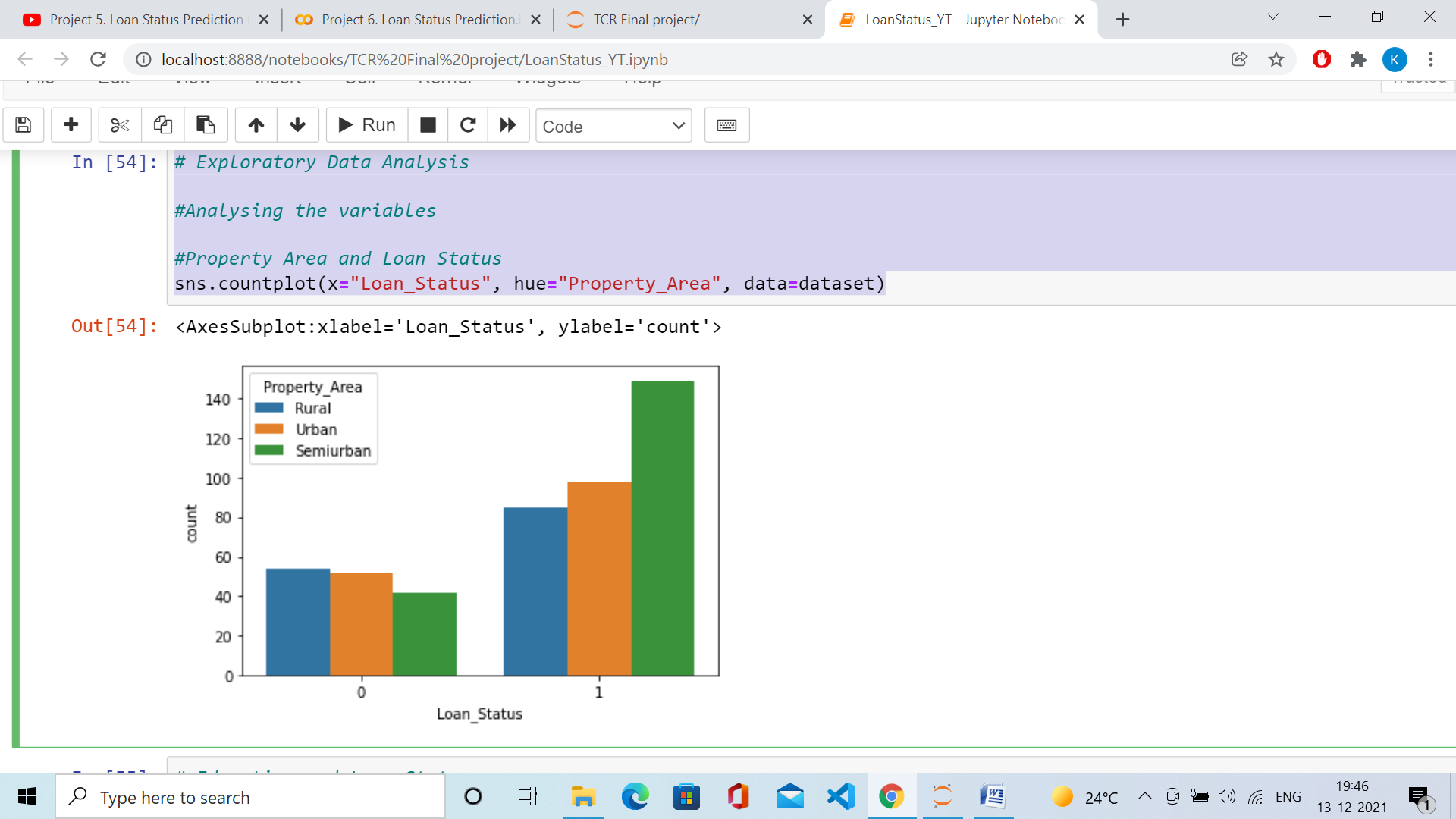
dataset = dataset.replace(to\_replace='3+', value=4)

# Exploratory Data Analysis

#Analysing the variables

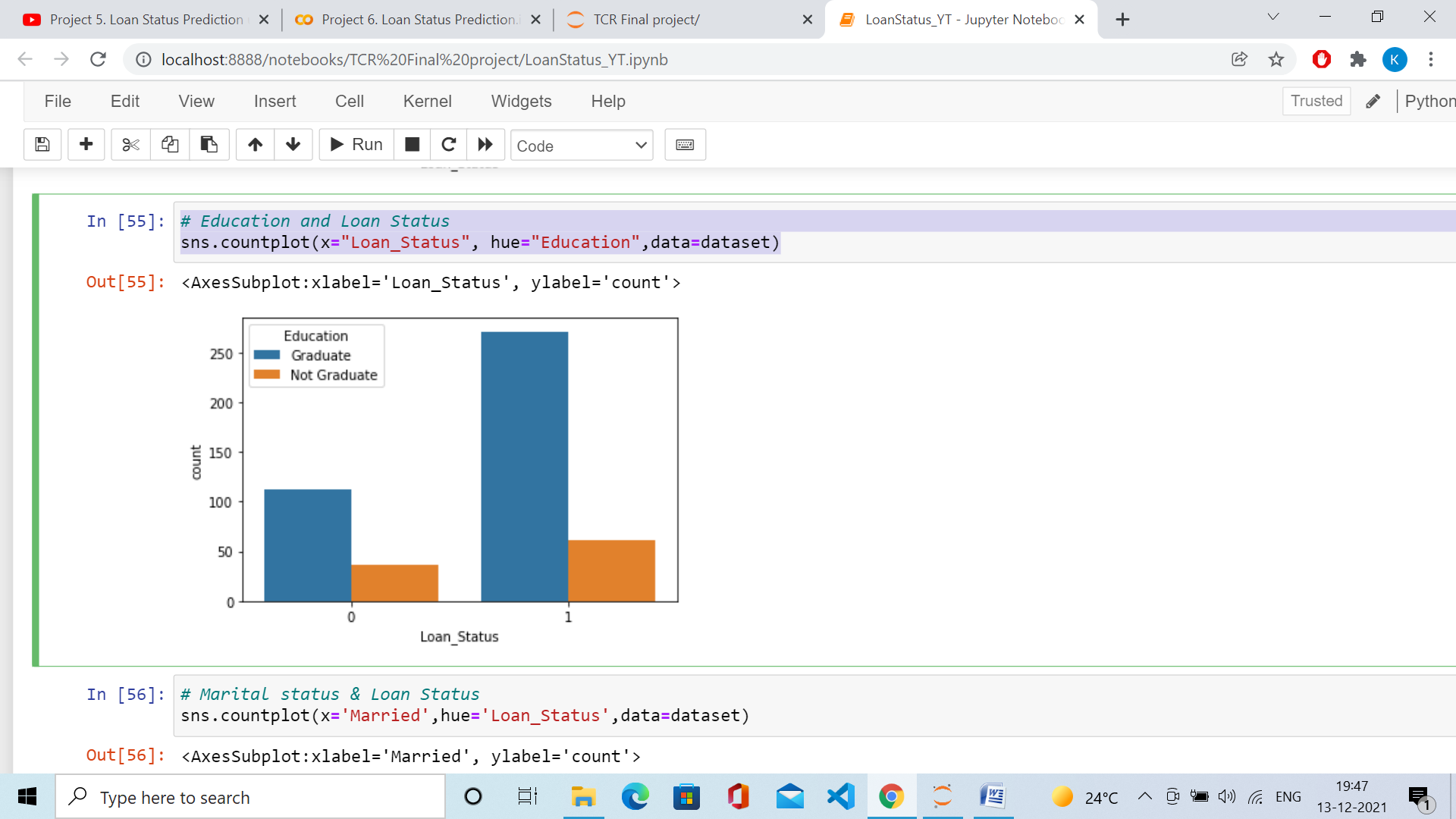
#Property Area and Loan Status

sns.countplot(x="Loan\_Status", hue="Property\_Area", data=dataset)



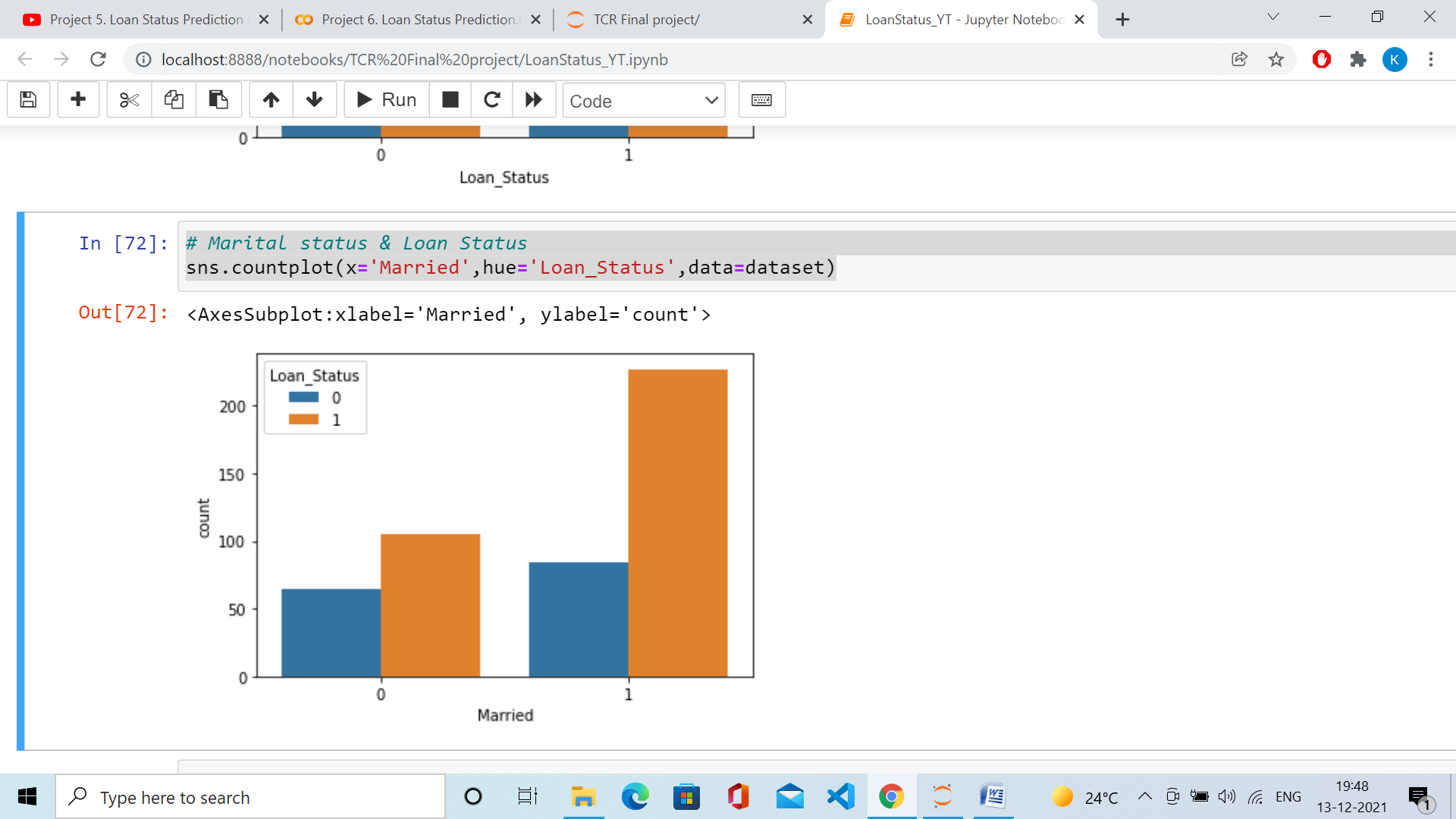
# Education and Loan Status

sns.countplot(x="Loan\_Status", hue="Education",data=dataset)



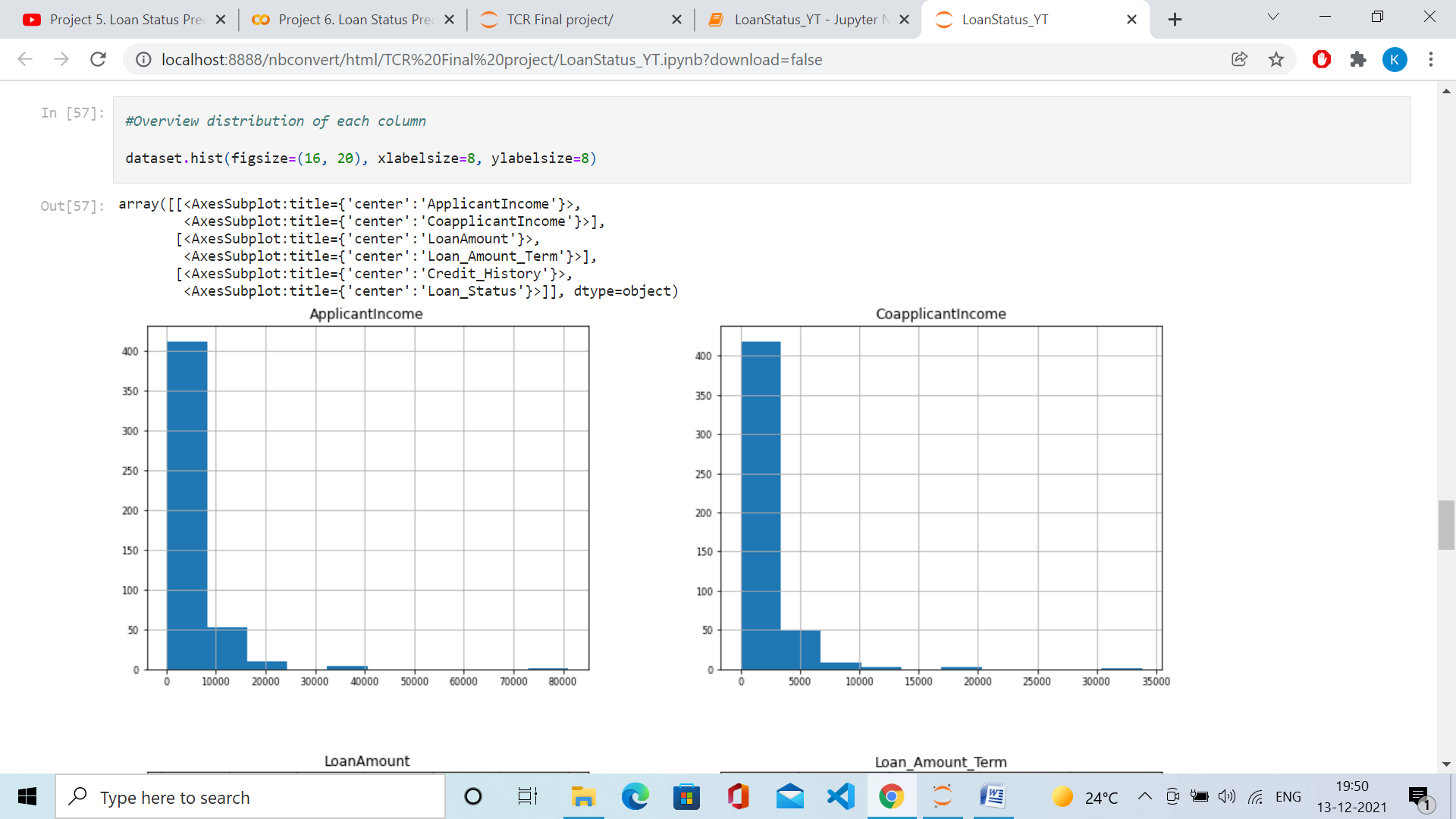
# Marital status & Loan Status

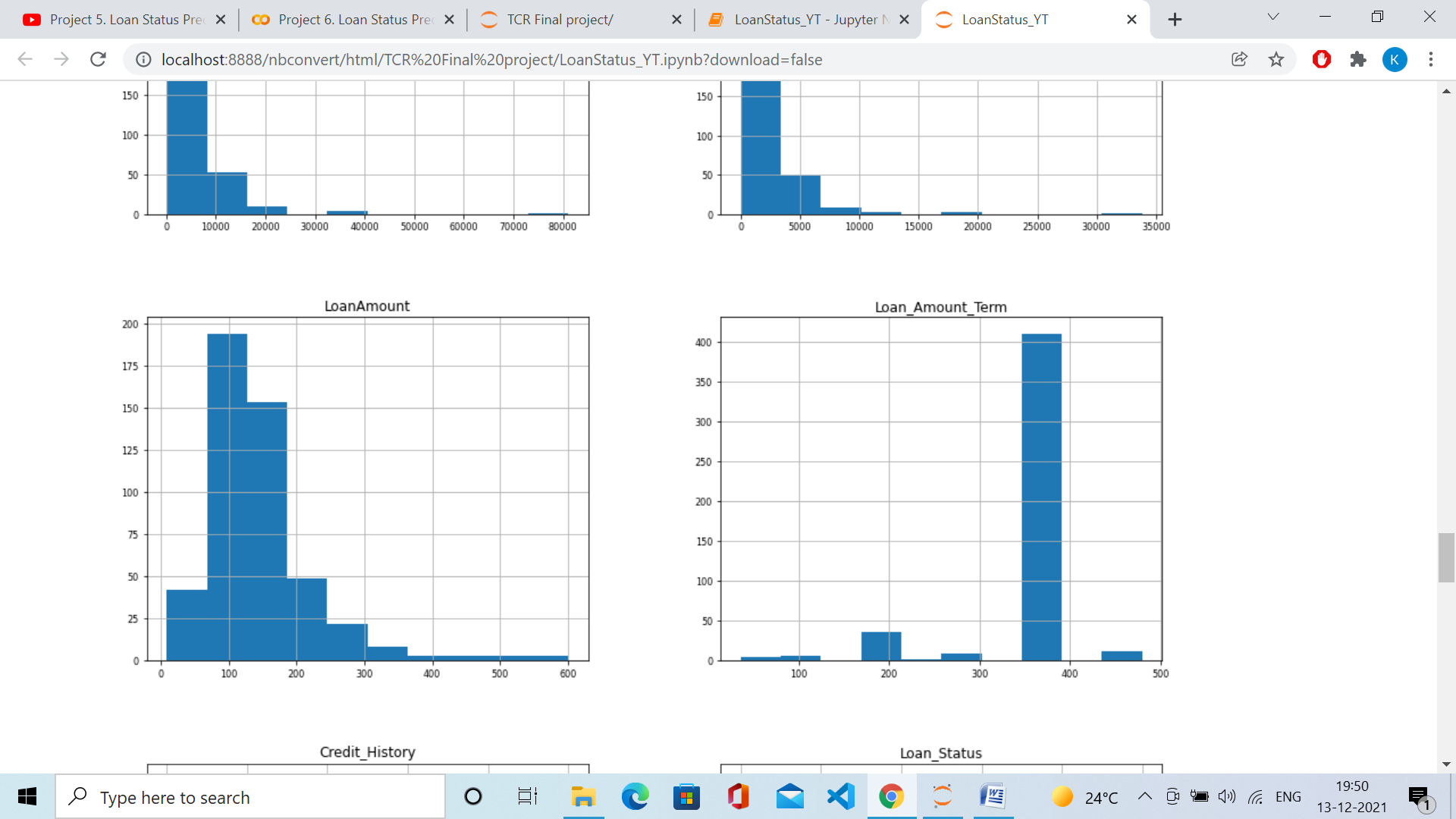
sns.countplot(x='Married',hue='Loan\_Status',data=dataset)

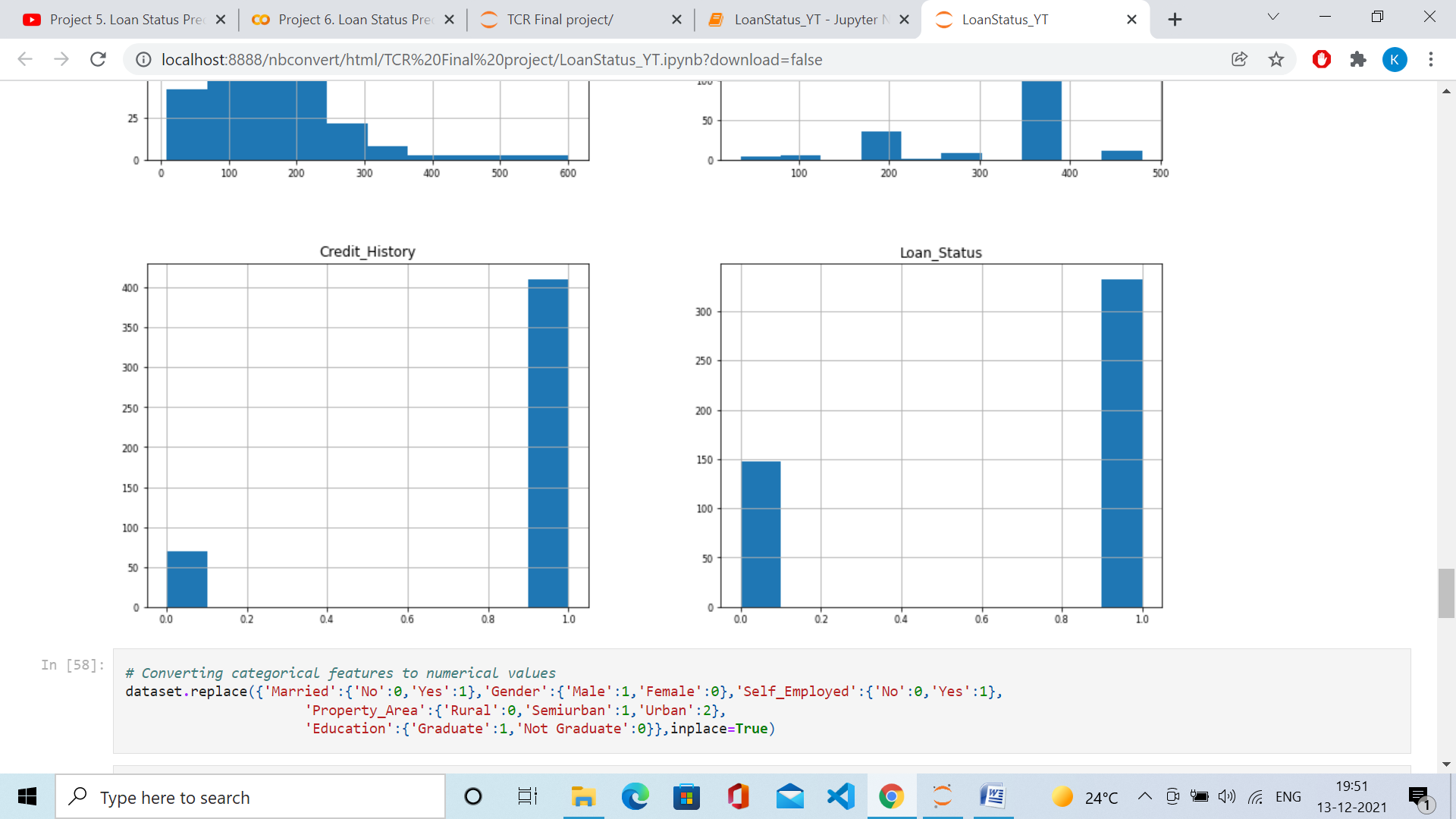


#Overview distribution of each column

dataset.hist(figsize=(16, 20), xlabelsize=8, ylabelsize=8)







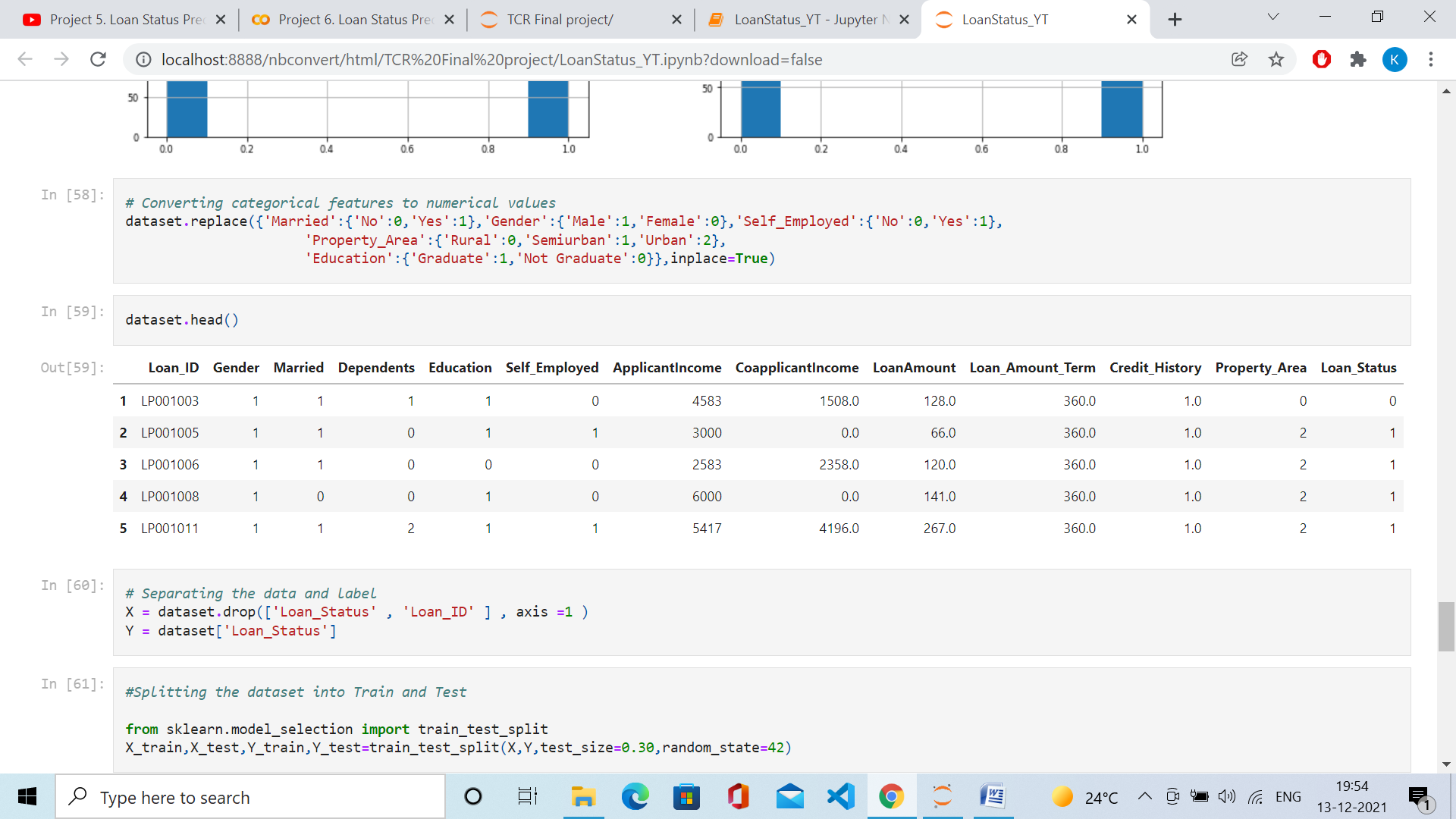
# Converting categorical features to numerical values

dataset.replace({'Married':{'No':0,'Yes':1},'Gender':{'Male':1,'Female':0},'Self\_Employed':{'No':0,'Yes':1},

'Property\_Area':{'Rural':0,'Semiurban':1,'Urban':2},

'Education':{'Graduate':1,'Not Graduate':0}},inplace=True)

dataset.head()



# Separating the data and label

X = dataset.drop(['Loan\_Status' , 'Loan\_ID' ] , axis =1 )

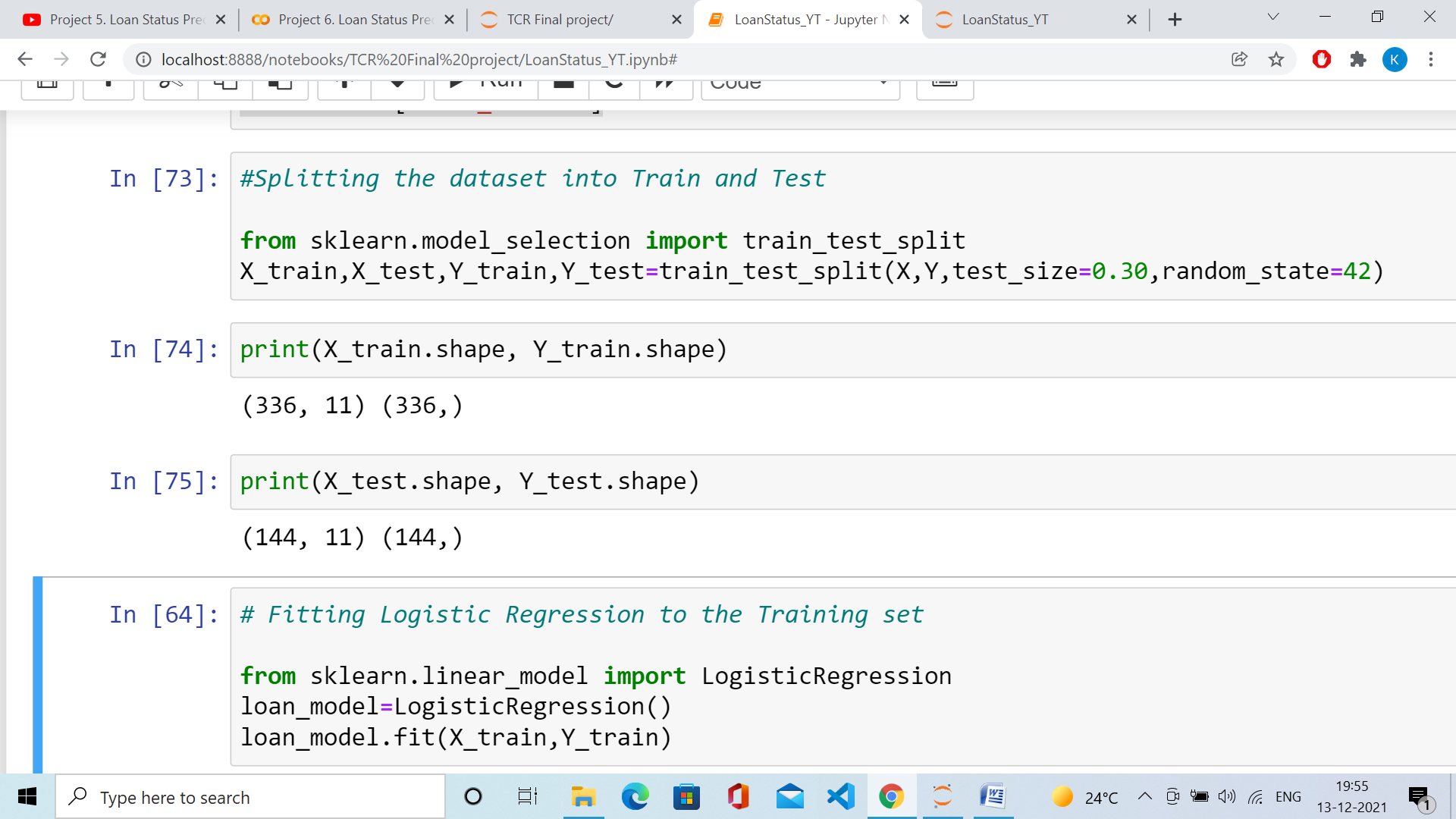
Y = dataset['Loan\_Status']

#Splitting the dataset into Train and Test

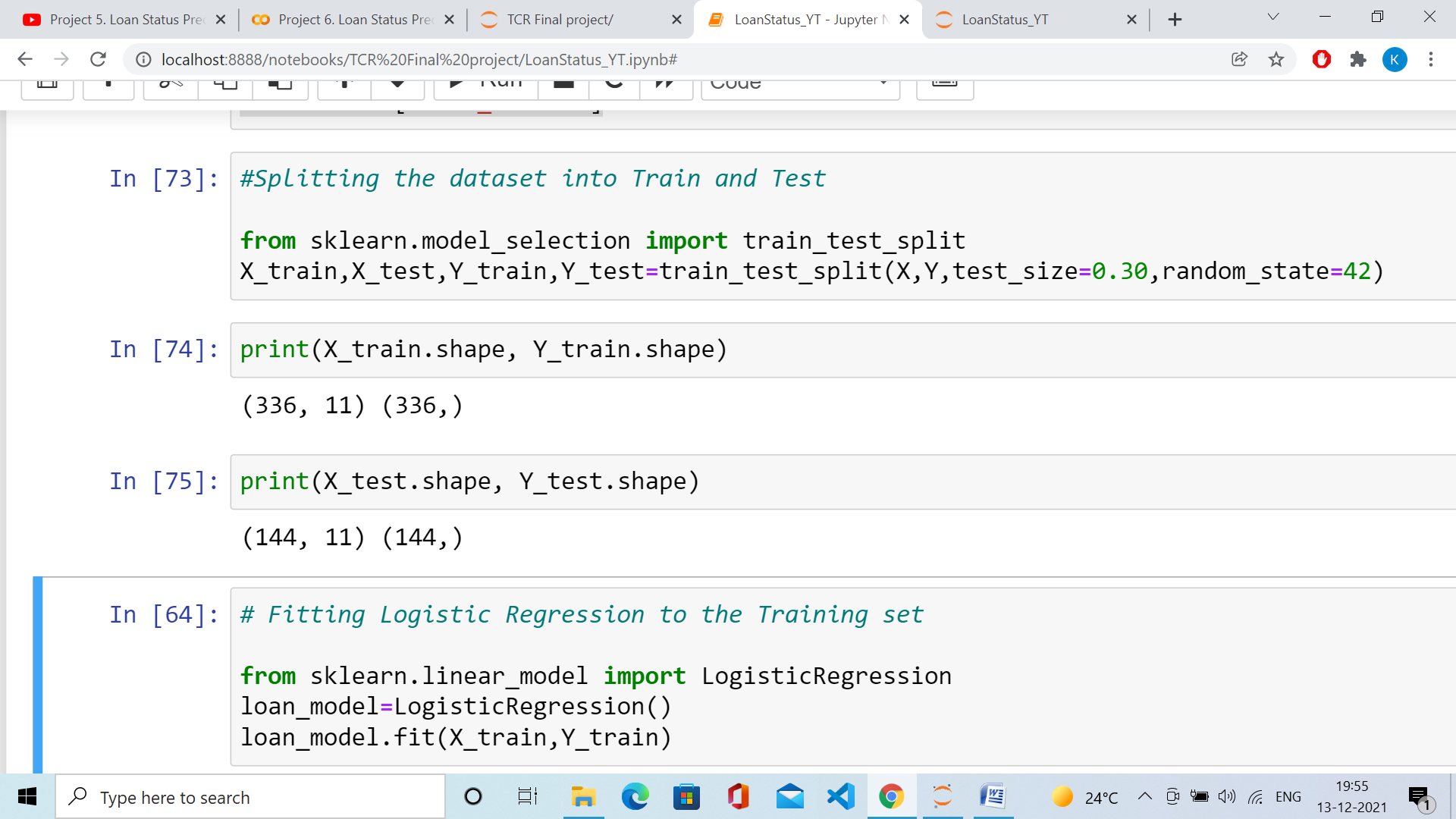
from sklearn.model\_selection import train\_test\_split

X\_train,X\_test,Y\_train,Y\_test=train\_test\_split(X,Y,test\_size=0.30,random\_state=42)

print(X\_train.shape, Y\_train.shape)



print(X\_test.shape, Y\_test.shape)



# Fitting Logistic Regression to the Training set

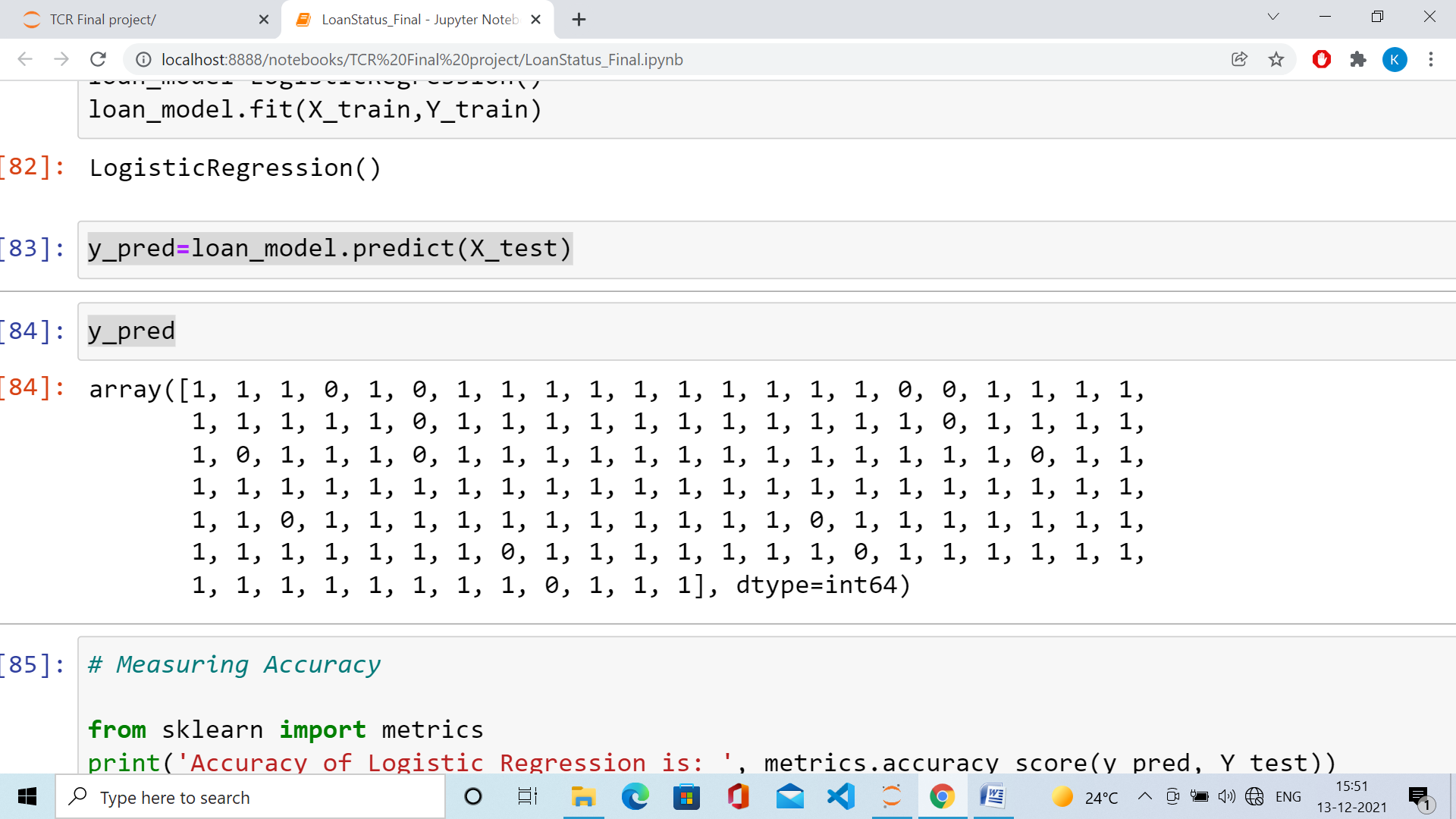
from sklearn.linear\_model import LogisticRegression

loan\_model=LogisticRegression()

loan\_model.fit(X\_train,Y\_train)

y\_pred=loan\_model.predict(X\_test)

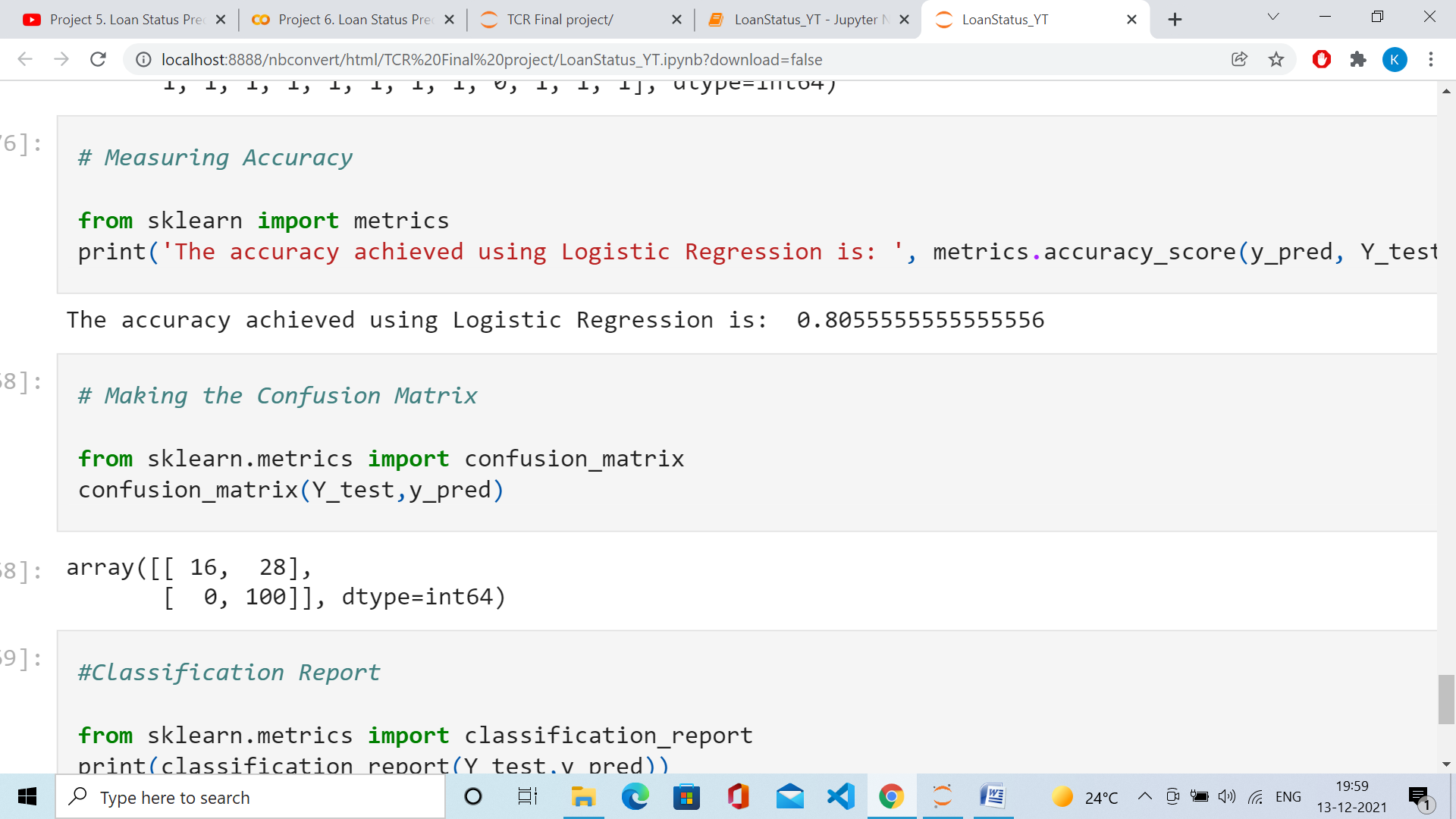
y\_pred



# Measuring Accuracy

from sklearn import metrics

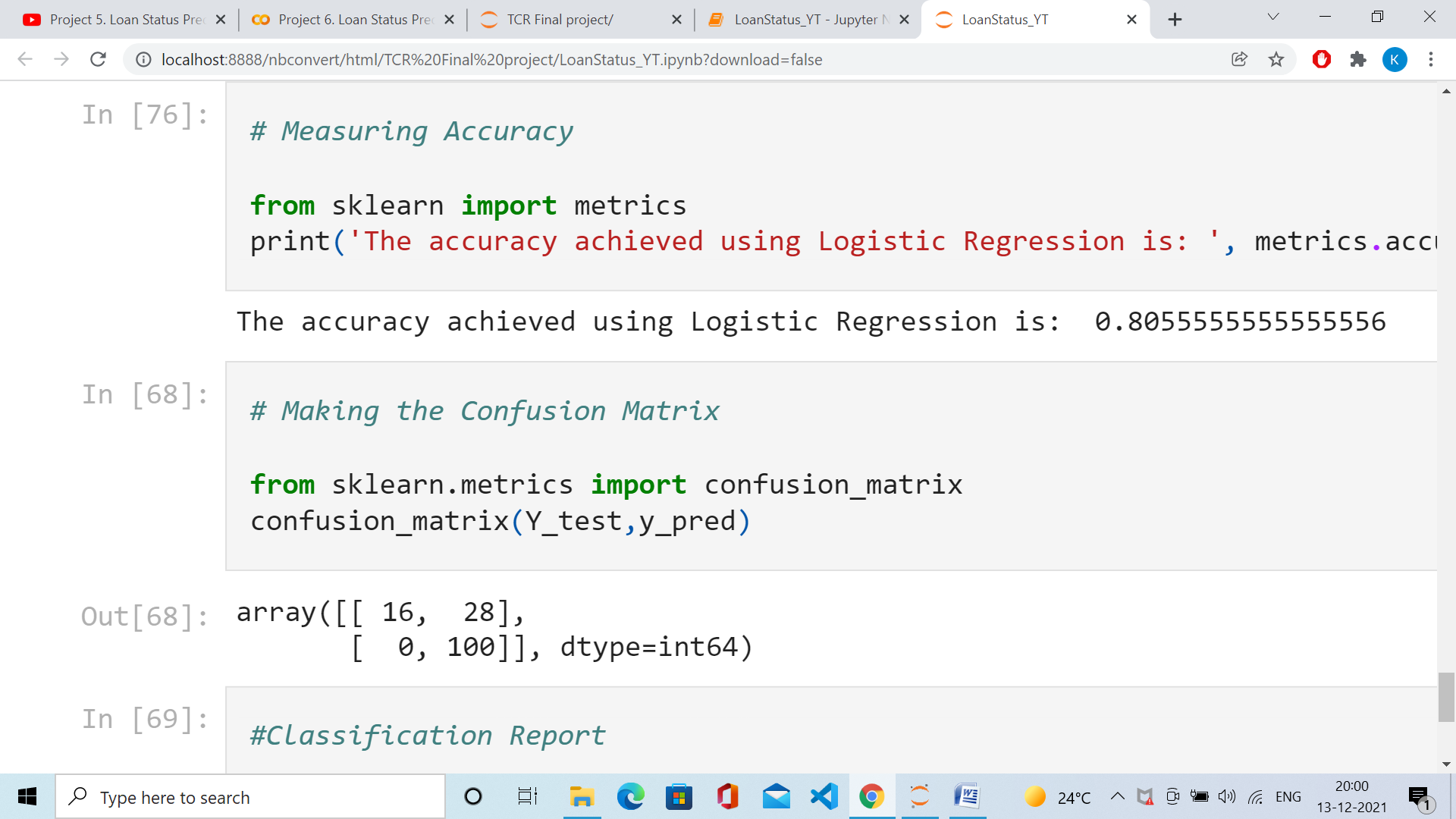
print('The accuracy achieved using Logistic Regression is: ', metrics.accuracy\_score(y\_pred, Y\_test))



# Making the Confusion Matrix

from sklearn.metrics import confusion\_matrix

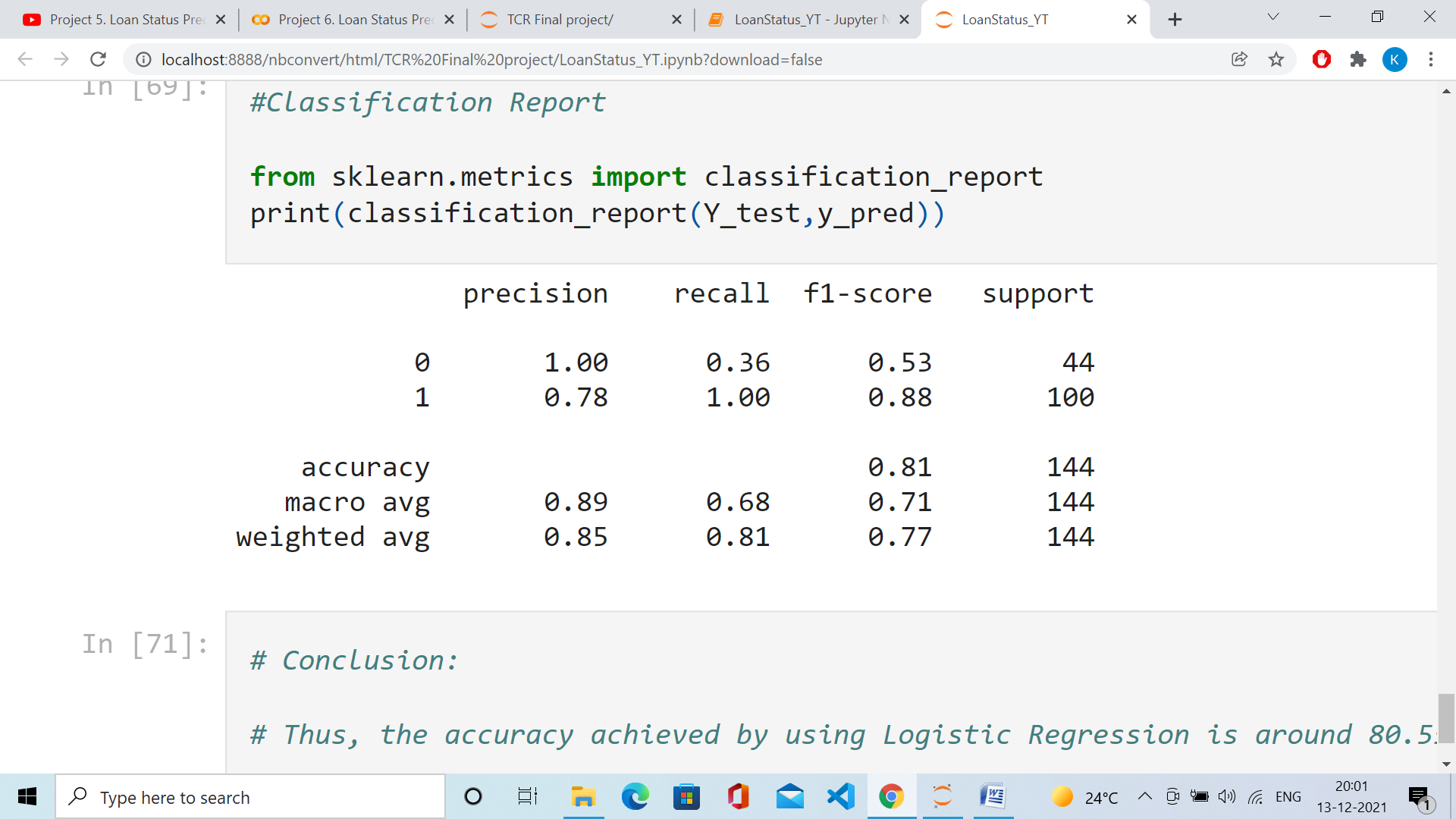
confusion\_matrix(Y\_test,y\_pred)



#Classification Report

from sklearn.metrics import classification\_report

print(classification\_report(Y\_test,y\_pred))



# Conclusion:

# Thus, the accuracy achieved by using Logistic Regression is around 80.55 %.