**Machine Learning Report on**

**Loan Application Status Prediction**

**Introduction**

Now a days, ever wondered, how great would it be, if we could predict, whether our request for a loan, will be approved or not, simply by the use of machine learning, from the ease and comfort of your home? Sounds fascinating right? Well, in this article, I will be guiding you through that!

This will not only be a practical project, which is applicable in the current times but also will add more to your knowledge of how the system of Loan Approval works.

You see, any bank, approves a loan based on the two most vital points:

* How risky is the borrower currently, (This is the factor, on which the interest rate of the borrower will depend)?
* Should they lend the money to the borrower at the given risks?

If both of these conditions give an affirmatory result, the bank proceeds with the loan approval.

**Algorithms, that we are going to use are:**

* Logistic Regression
* GaussianNB
* DecisionTreeClassifier
* RandomForestClassifier
* KNeighborsClassifier

You might think, why I am using different types of algorithms, rather than choosing any one algorithm for the model building.

The reason is simple, just to check which algorithm among them will give the best accuracy to predict the desired output.

**Dataset**

**Step 1**

**To know the Problem Statement:**

This dataset includes details of applicants who have applied for loan. The dataset includes details like credit history, loan amount, their income, dependents etc.

Independent Variables:

* Loan\_ID
* Gender
* Married
* Dependents
* Education
* Self\_Employed
* ApplicantIncome
* CoapplicantIncome
* Loan\_Amount
* Loan\_Amount\_Term
* Credit History
* Property\_Area

Dependent Variable (Target Variable):

* Loan\_Status: You have to build a model that can predict whether the loan of the applicant will be approved or not on the basis of the details provided in the dataset.

Note: The link of the dataset is below.

Download Files:

• https://github.com/dsrscientist/DSData/blob/master/loan\_prediction.csv

**Let’s code!**

For this project, I have used Python. You can use any python editing environment that you like, e.g., PyCharm, Jupyter Notebook, Sublime, Atom, VSCode, Google Colab Notebook, etc.

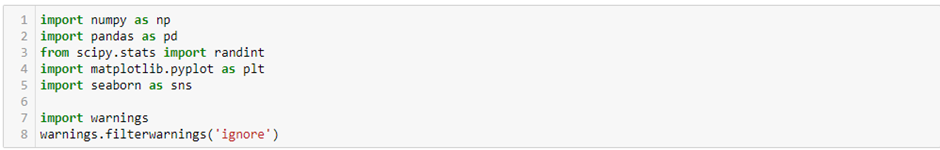
What I have used is a Jupyter Notebook however there is no dearth on availability of such tools.

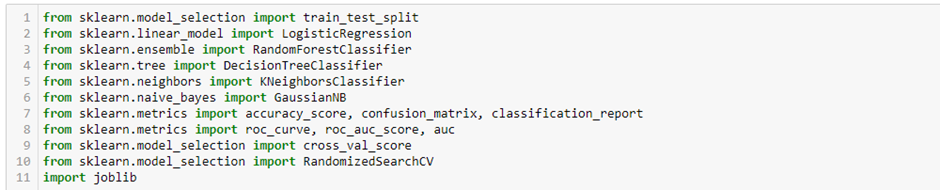
The benefit of using a Jupyter notebook is that I already had installed it in my system so that I could run the code locally at any point of time. Even I don’t need Internet Connection to run the code.

**Importing Libraries**

**Step 2**

Let us first import all the modules and libraries that we are going to use in the future while making the project. The dependencies that we will be using are numpy, pandas, seaborn, and ScikitLearn.

****



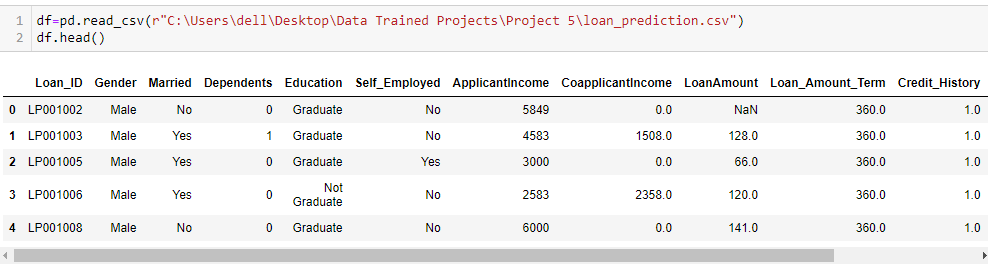
**Data Collection and Processing**

Simply downloading the dataset will not do. We need to link the dataset to our code so that our code can read the data from the table. The dataset will be in the format of a CSV. Thus, to read it, we will be taking the help of the pandas method, called read\_csv()

We are strong the dataset in the variable called “**df**”. We can thus now refer to the entire dataset by this variable name.

**Step 3**

To get a glance at the first 5 rows of the dataset, use the head() method and shows the following output as shown below.



**Step 4**

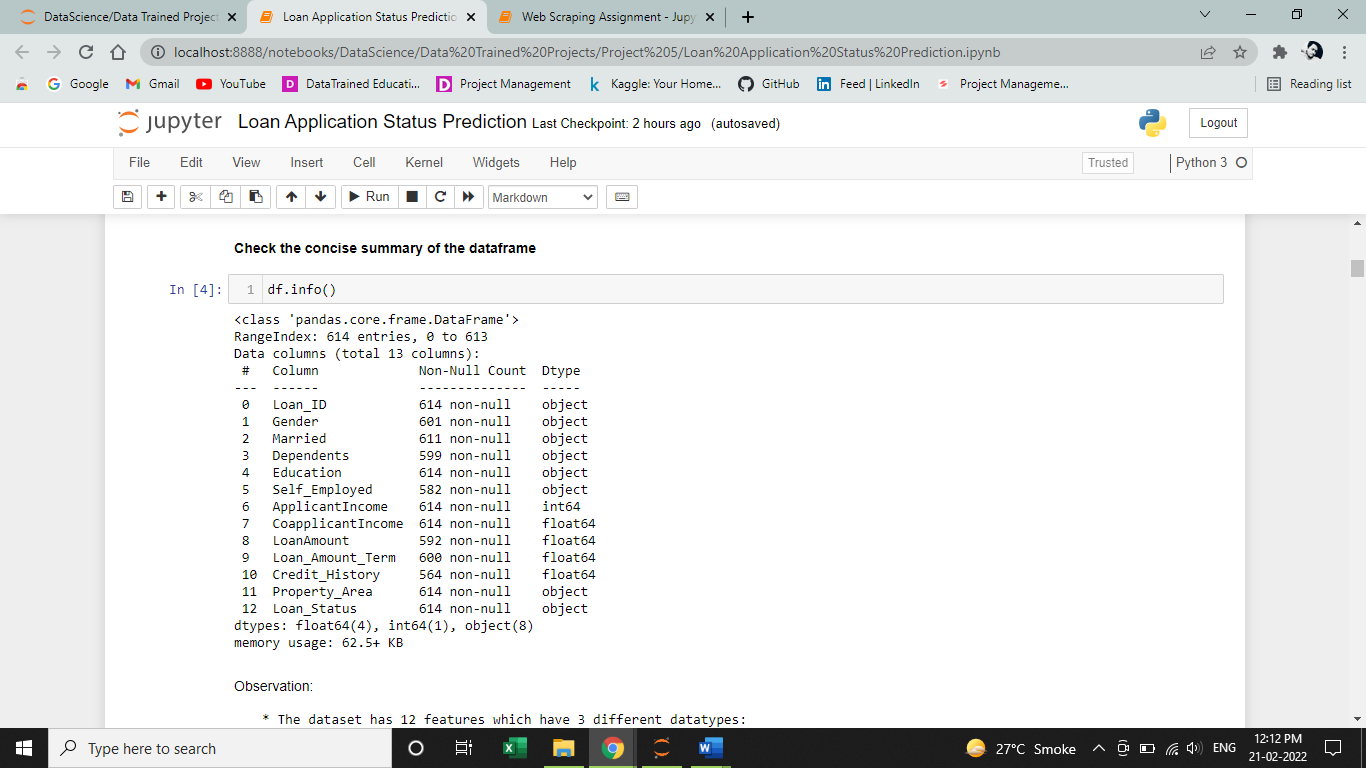
Now, let us check the number of rows and columns in the dataset. We run the command and see the output:



The shape of the dataset **df** is – 614 rows x 13 columns

**Step 5**

Next, let us check the cosine summary or description of the dataset **df**



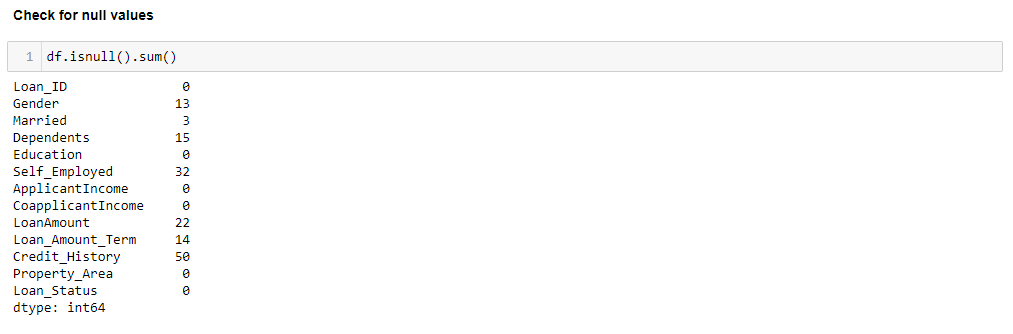
**Observation:**

The dataset has 12 features which have 3 different datatypes:

* float64 = 4 features
* int64 = 1 feature
* object = 7 feature
* object = 1 label

**Step 6**

Next step is to check for null values



We can see lots of null values present in the dataset **df**

**Step 7**

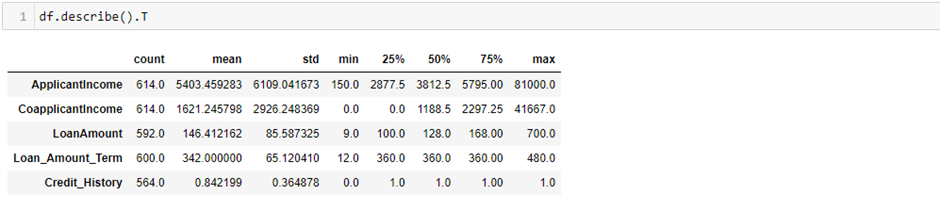
Check for duplicate variables



We, can observe that there are no duplicates in the dataset **df.**

**Step 8**

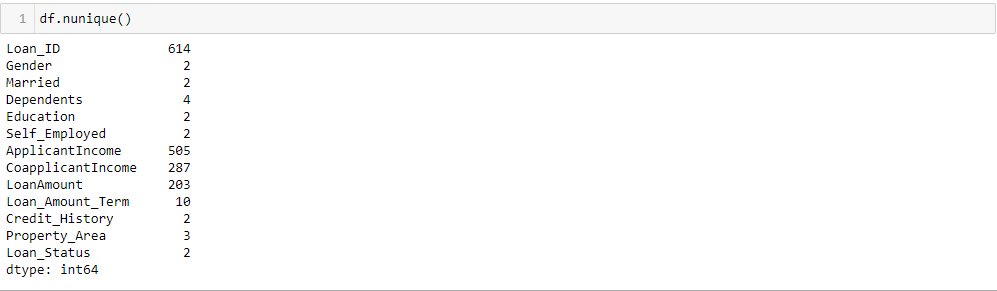
The next step is to check the statistical summary of the dataset **df**



As there are 5 numerical features, so its shows five statistical summary of the dataset. Here we can find the mean. standard deviation, minimum and maximum values and the Ist, IInd, IIIrd quantile of the features.

**Step 9**

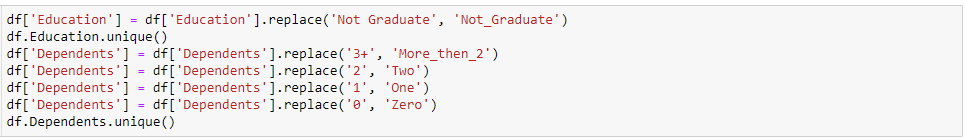
Check for uniqueness in feature columns



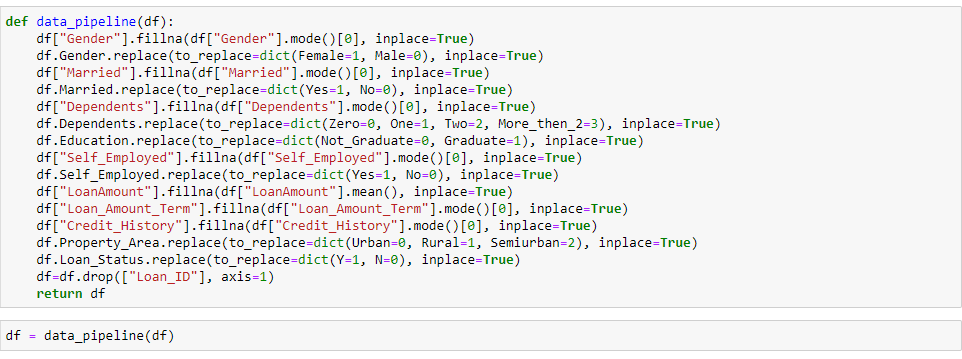
We found out that there are some features having uniqueness in them.

**Step 10**

First, lets change some of the class name for ease handling



After changing name, now let’s treat the missing values and encode the features after observing the dataset **df**.



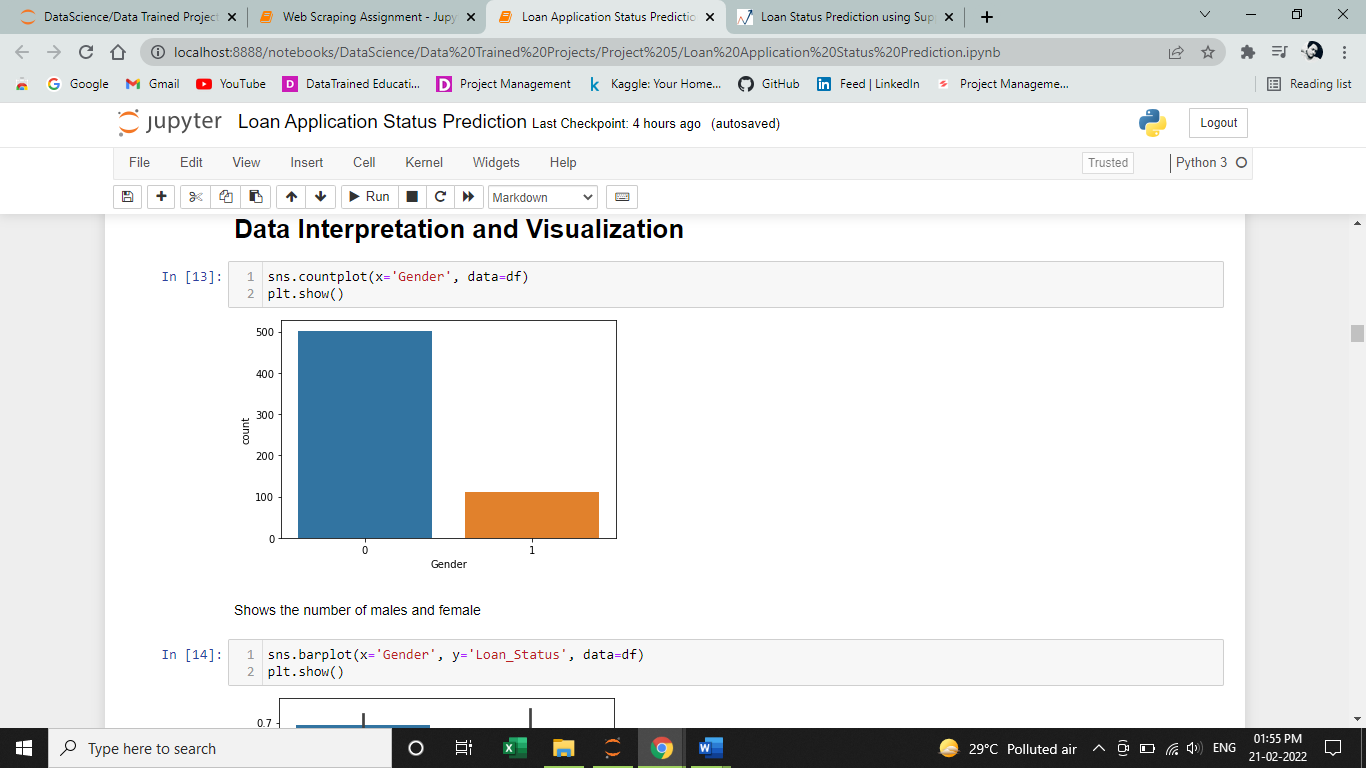
We have treated –

* Filled the null values of 'Gender' by mode. Encoded Female = 1, Male = 0
* Filled the null values of 'Married' by mode. Encoded Yes = 1, No = 0
* Filled the null values of 'Dependents' by mode.
* In Feature 'Education', encoded Not\_Graduate = 0, Graduate = 1
* Filled the null values of 'Self\_Employed' by mode. Encoded Yes = 1, No = 0
* Filled the null values of 'Loan\_Amount' by mean.
* Filled the null values of 'Loan\_Amount\_Term' by mode.
* Filled the null values of 'Credit\_History' by mode.
* In Feature 'Property\_Area', encoded Urban=0, Rural=1, Semiurban=2
* In feature 'Loan\_Status', encoded Y=1, N=0
* Dropped the column 'Loan\_ID', which have no such correlation with 'Loan\_Status'

**Data Interpretation and Visualization**

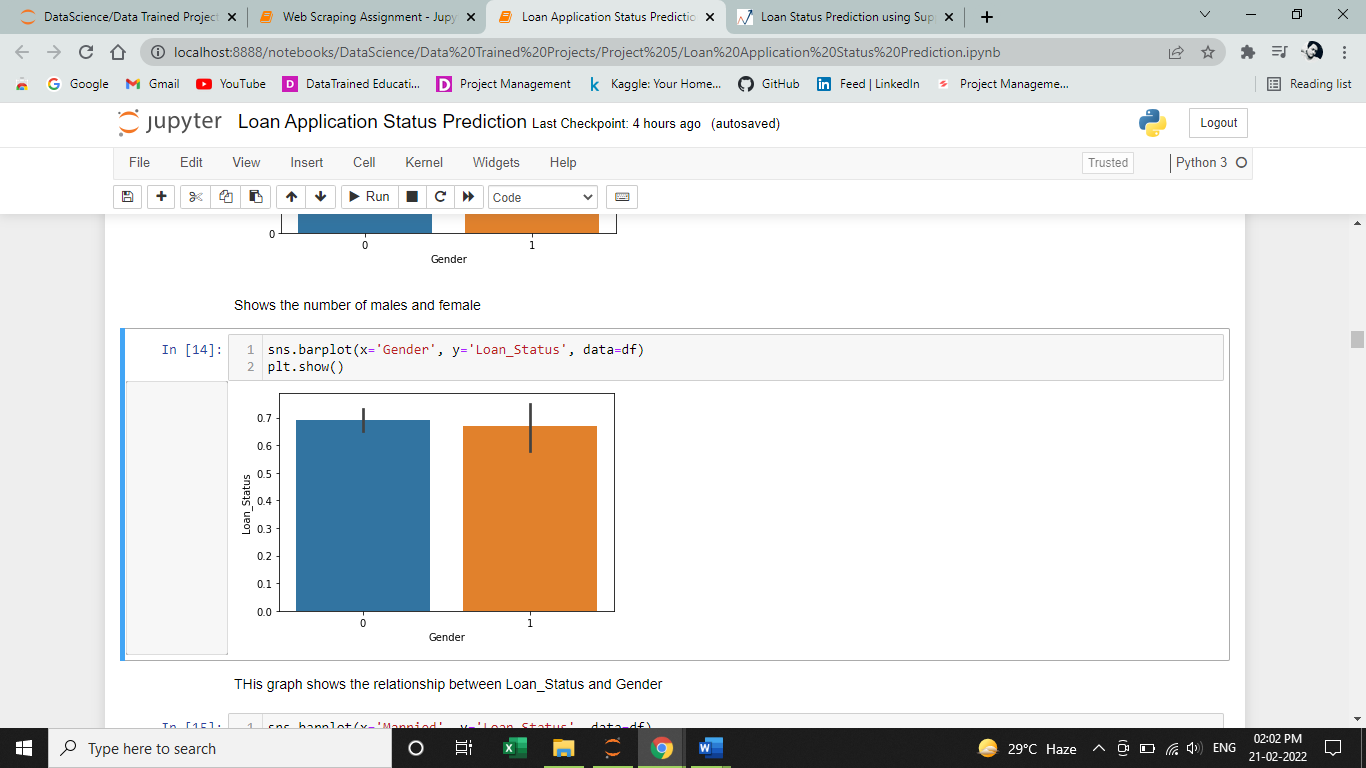
**Step 11**

Plotting a count plot for ‘Gender’ feature in the dataset.

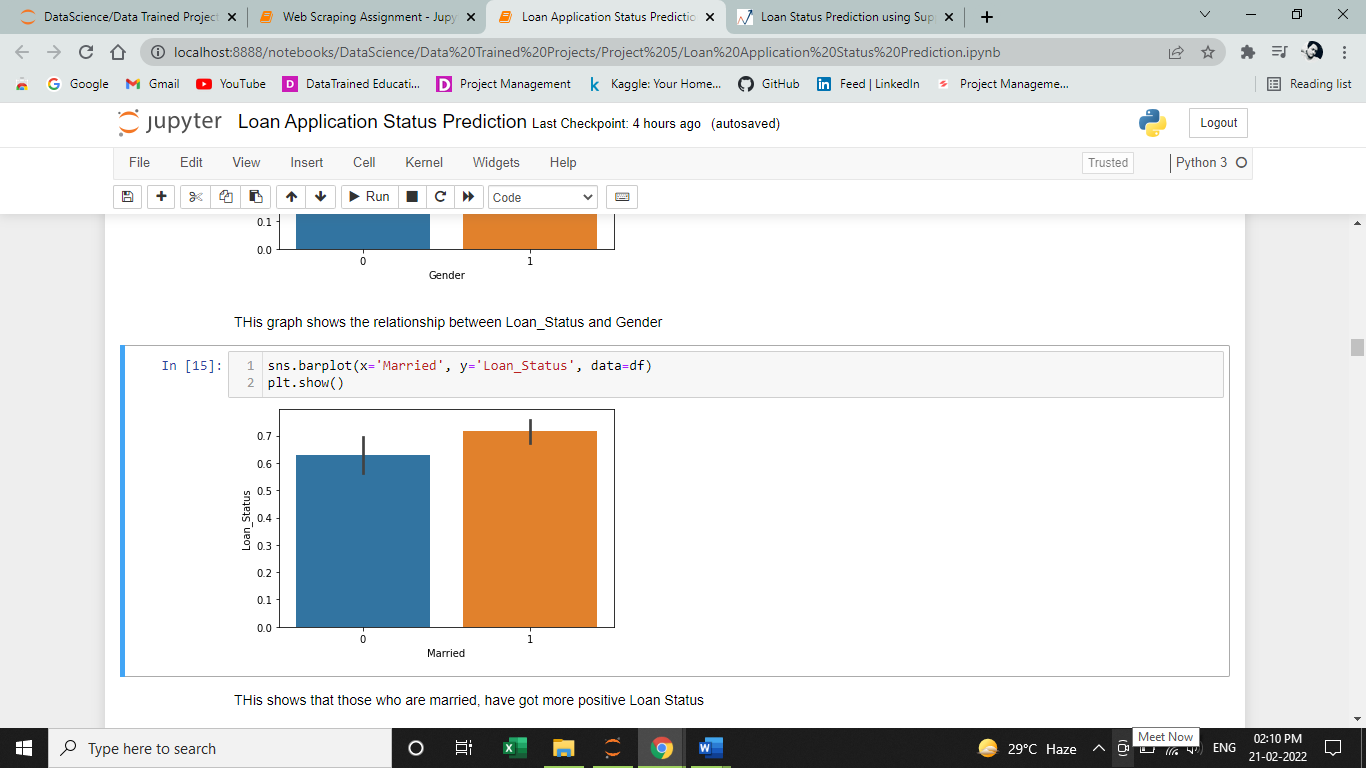
****

The above figure shows the number of males and females

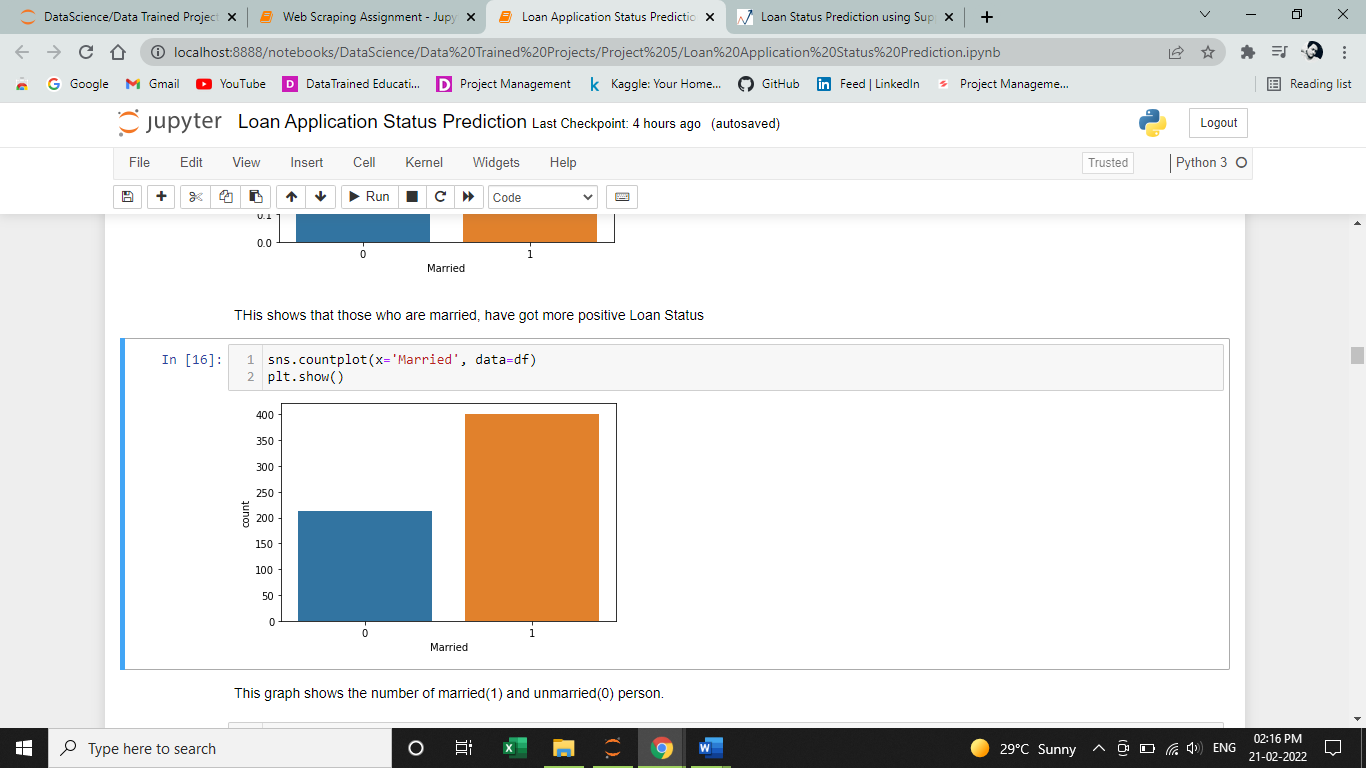
Female = 1, Male = 0



The above graph shows the relation between feature ‘Gender’ with the label ‘Loan\_Status’.

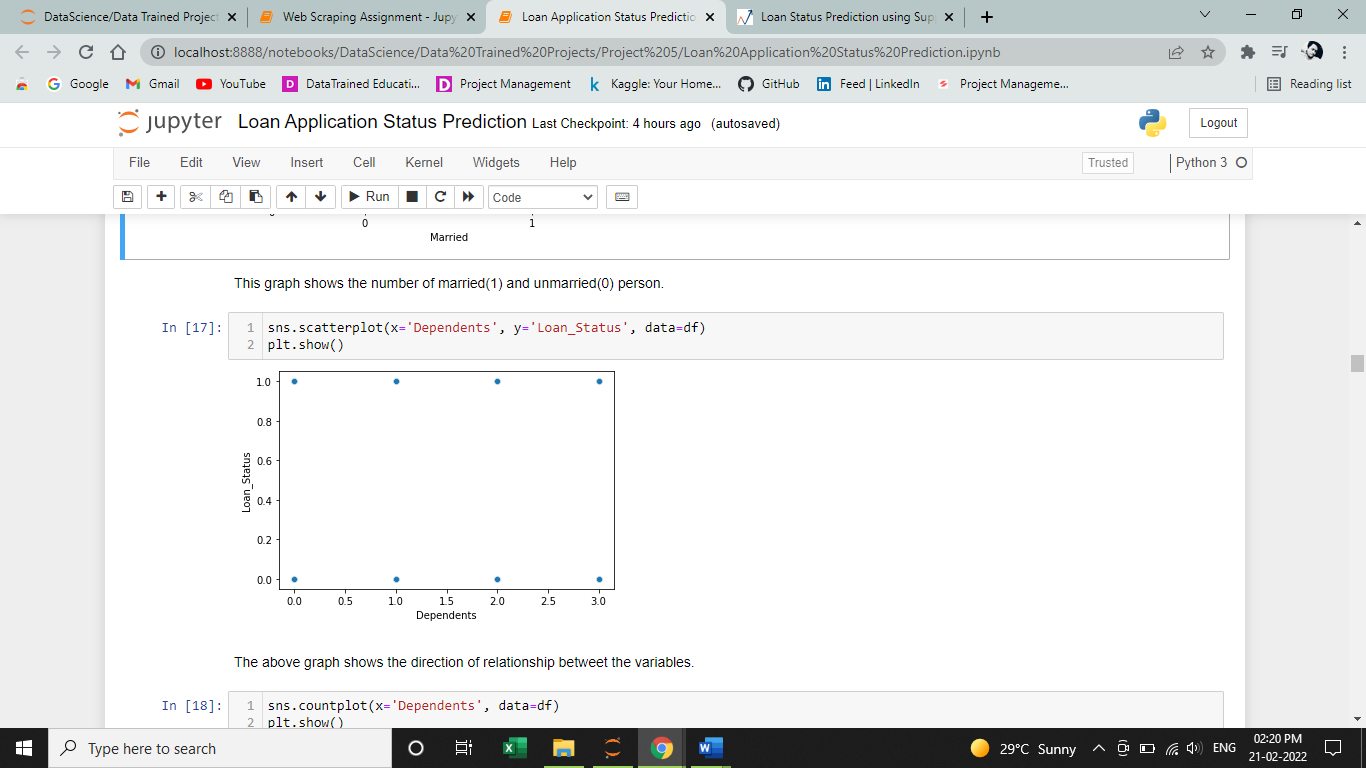


The above graph shows the relations between ‘Married’ feature and the label ‘Loan\_Status’.

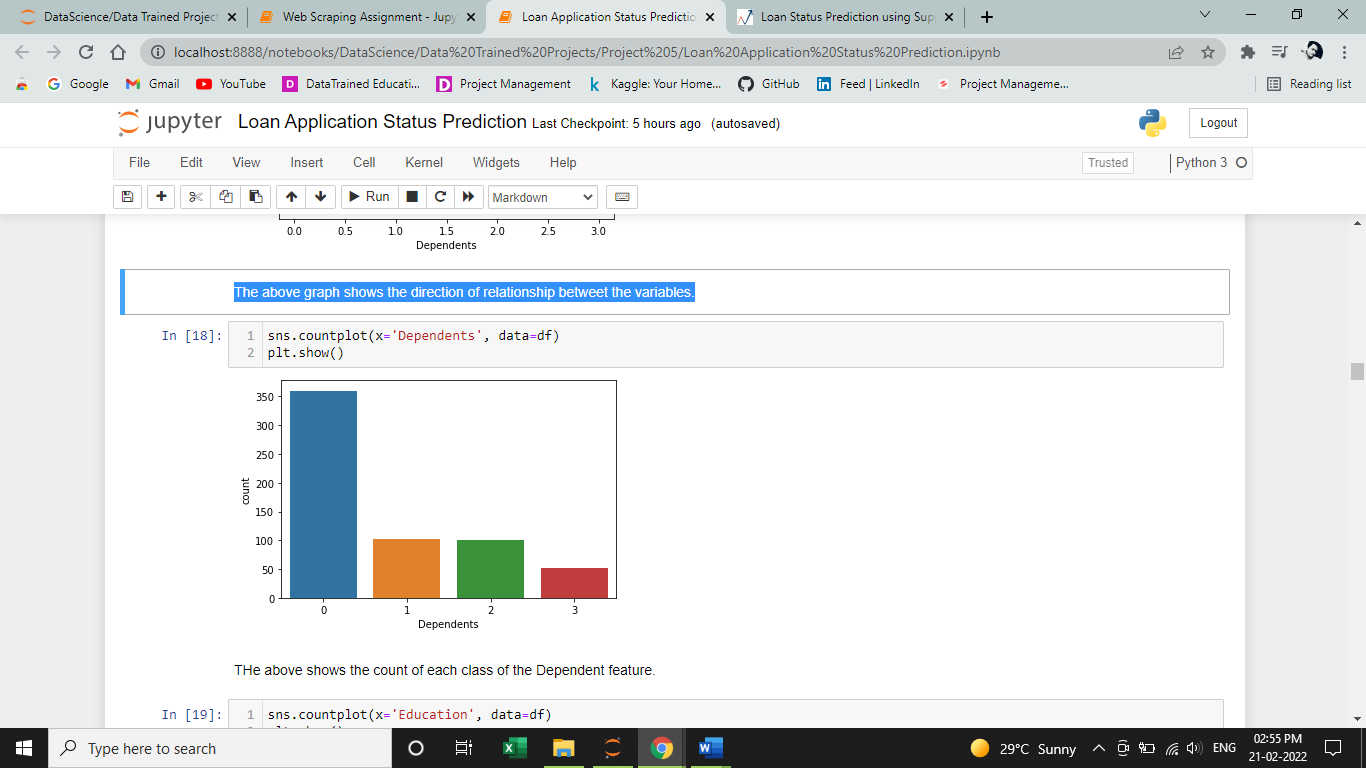


The above shows the number of Married peoples in feature ‘Married’.

Here, married = 1, unmarried = 0

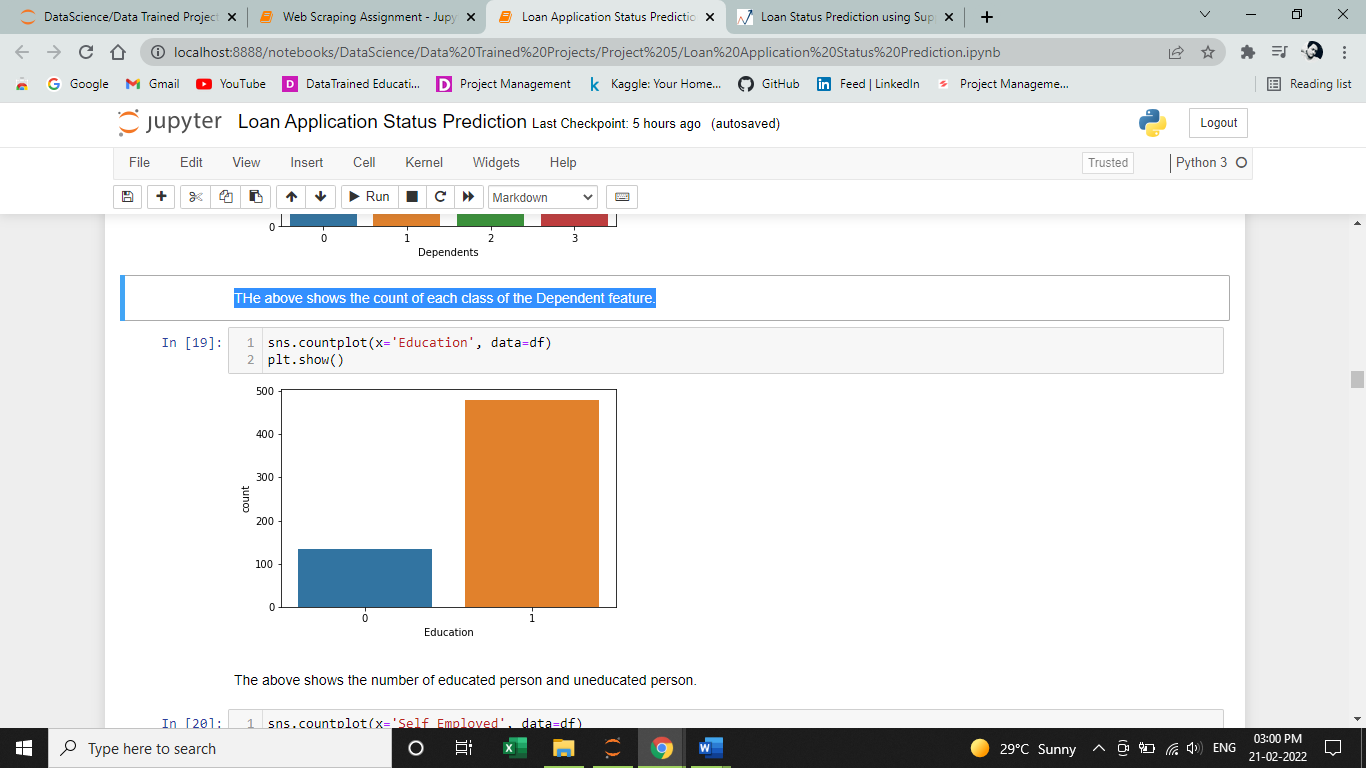


The above graph shows the direction of relationship between the variables.



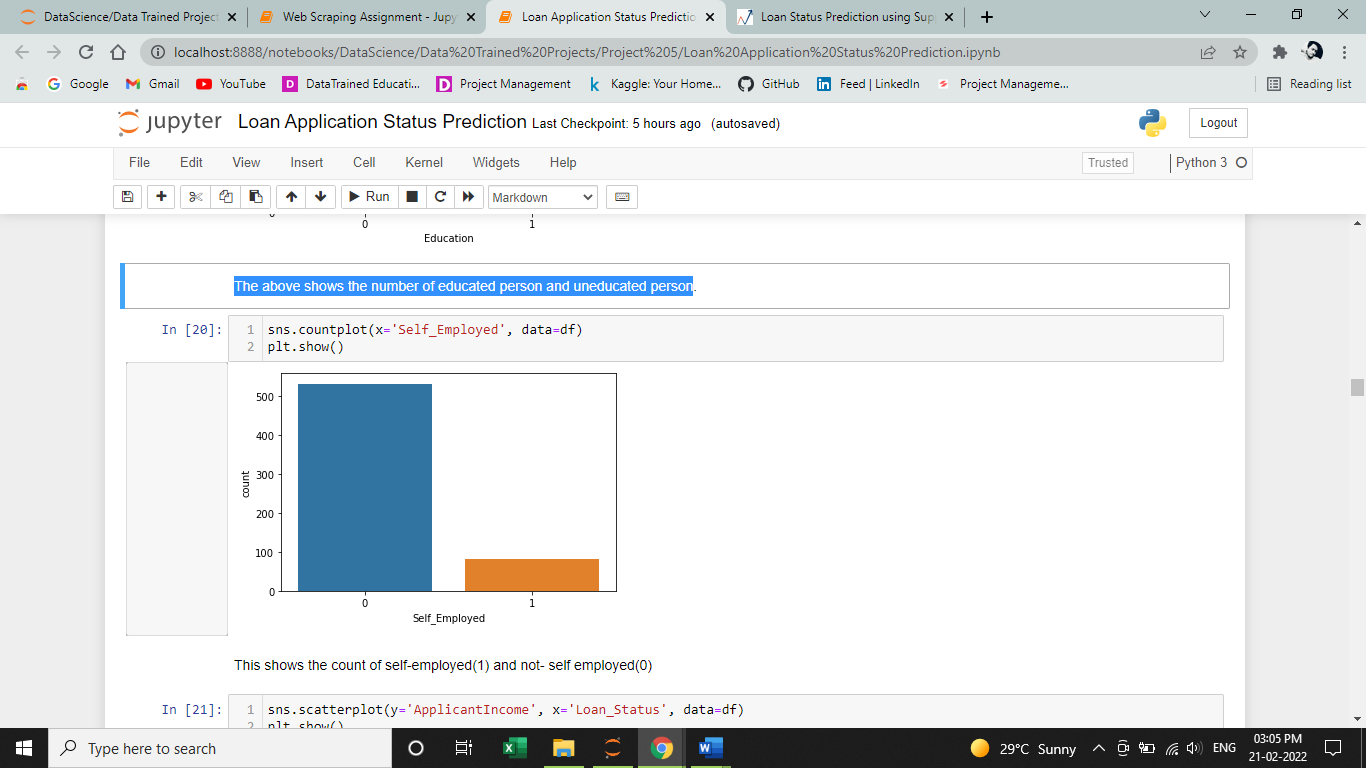
The above shows the count of each class of the Dependent feature.

Here, **0** = Zero, **1** = One, **2** = Two and **3** = 3 or more than 3

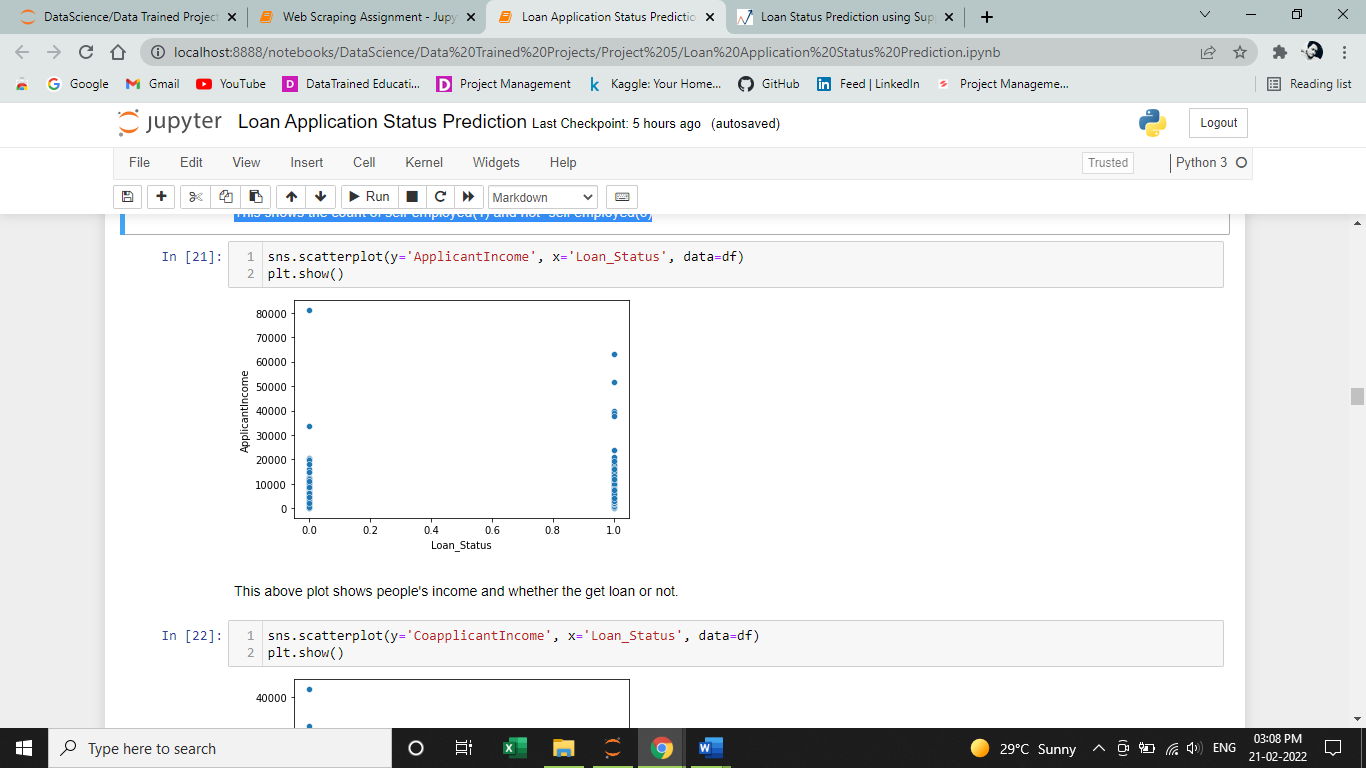


The above shows the number of educated person and uneducated person.

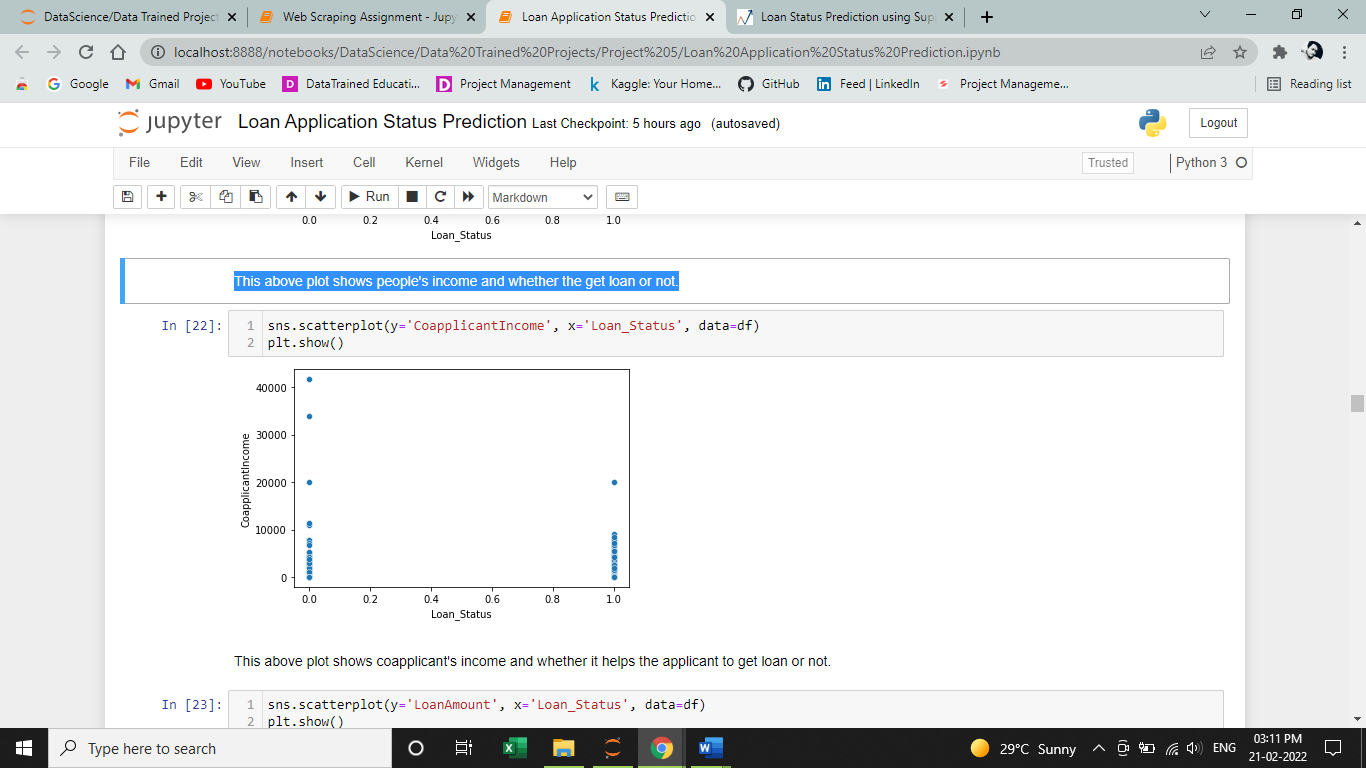
Here, 0 = Not Graduate and 1 = Graduate



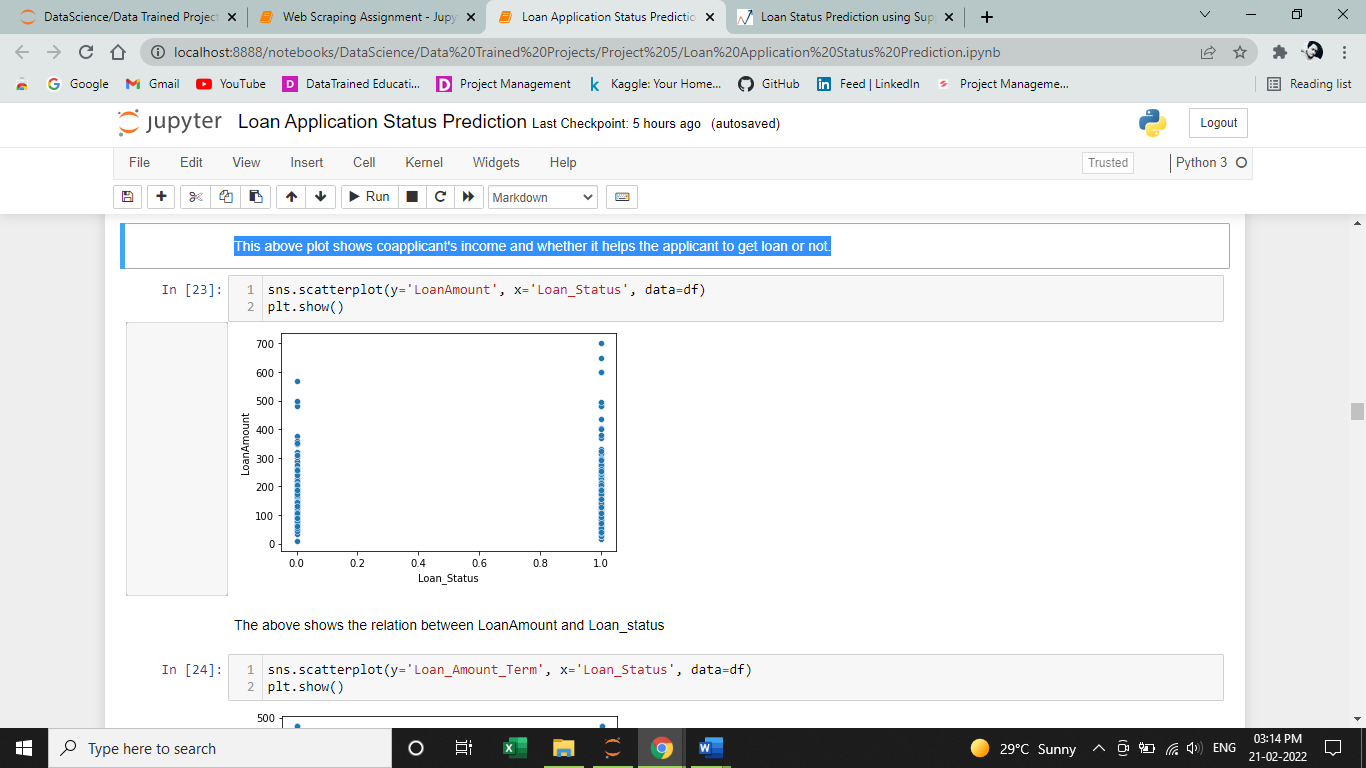
This shows the count of self-employed = 1 and not self-employed = 0



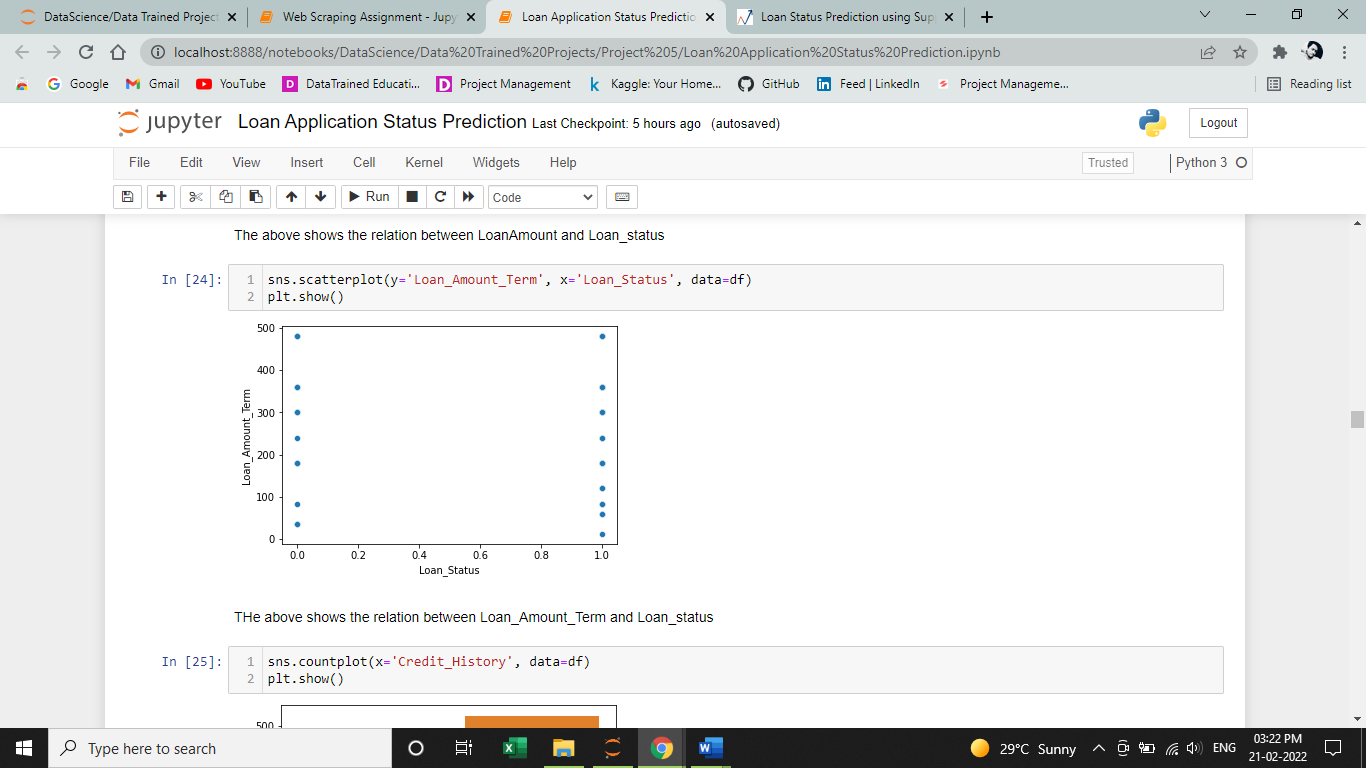
This above plot shows people's income and whether the get loan or not, and we can also observe that their income varies a lot in both the cases.



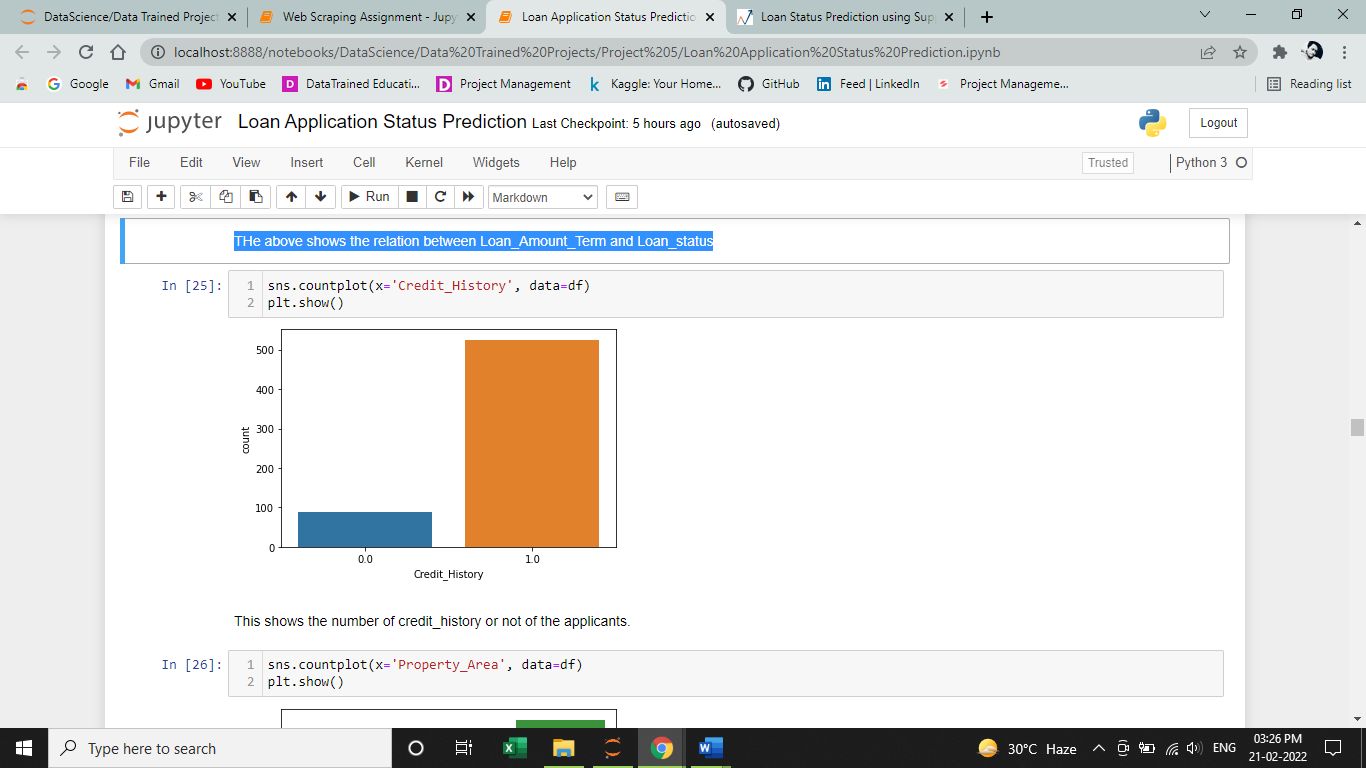
This above plot shows coapplicant's income and whether it helps the applicant to get loan or not.



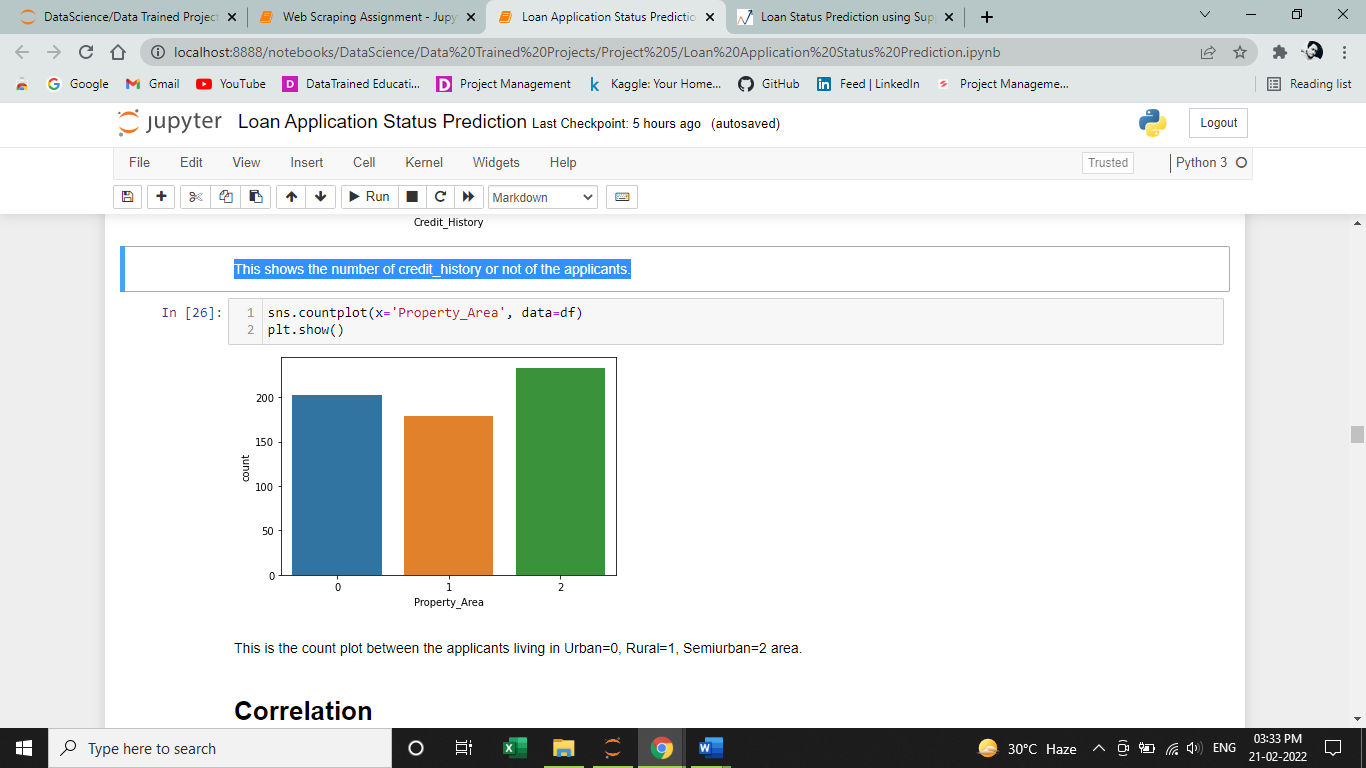
The above shows the relation between LoanAmount and Loan\_status and also the count of positive or negative Loan\_Status.



The above shows the relation between Loan\_Amount\_Term and Loan\_status.



The above graph shows the number of Credit\_History or not of the applicants.



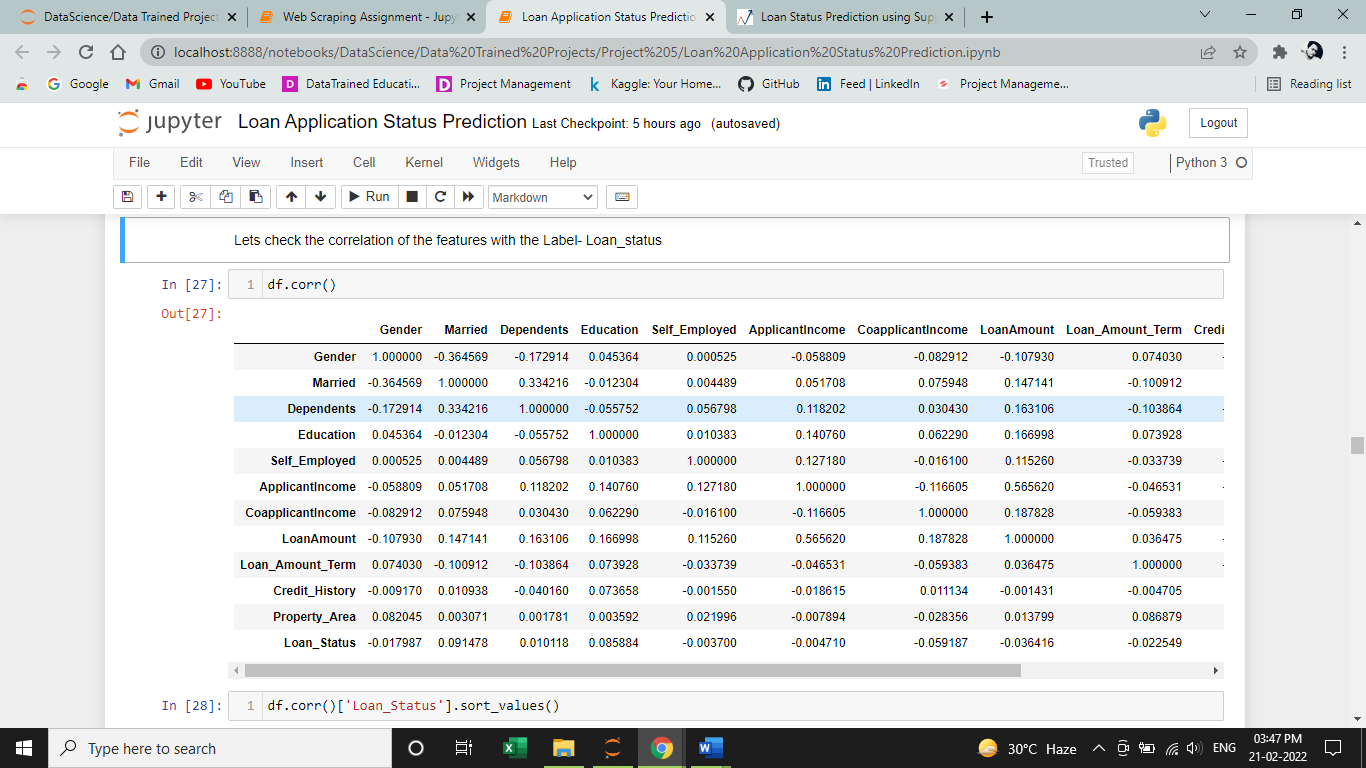
This is the count plot between the applicants living in Urban=0, Rural=1, Semiurban=2 area.

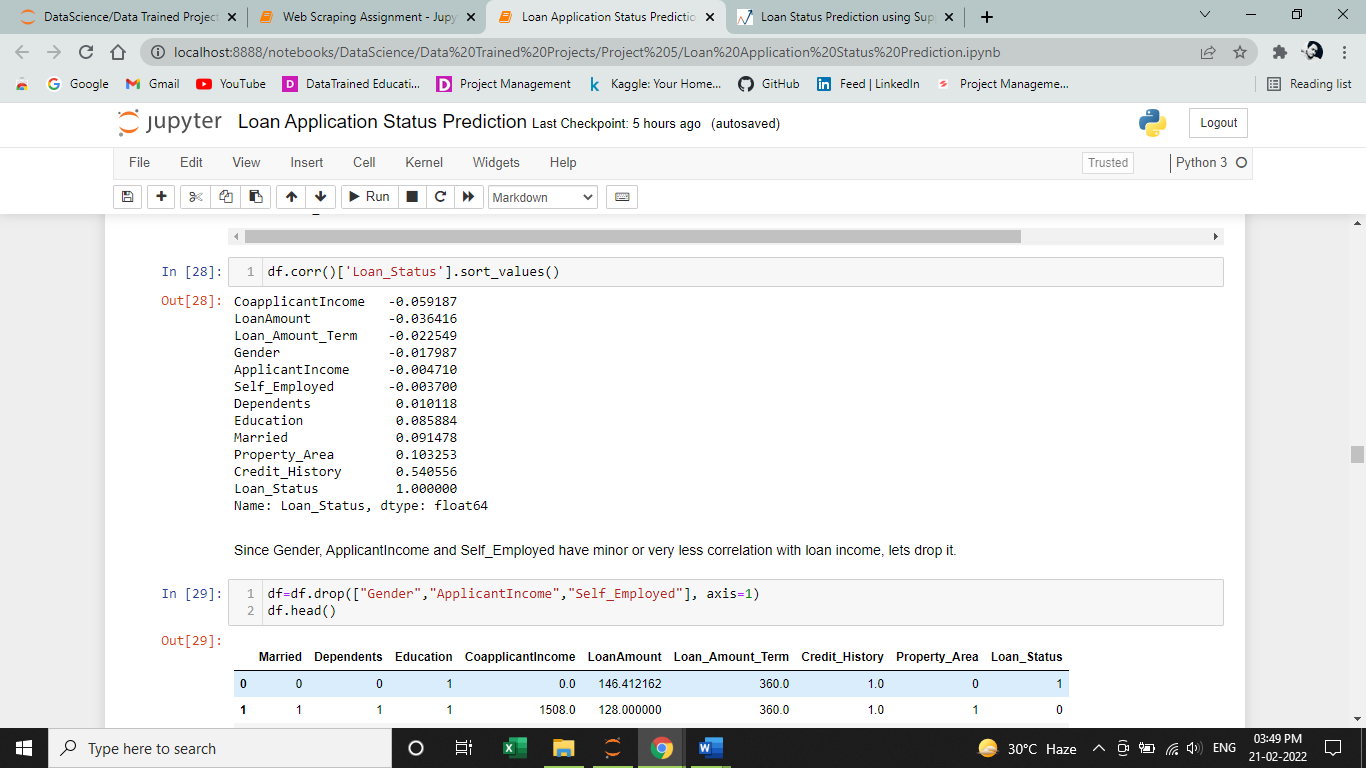
In the above section, I have tried to plot different possible graphs with each other and see their relations with each other.

**Step 12**

**Finding Correlation**

Let’s check the correlation of the features with other features and the Label - ‘Loan\_status’.

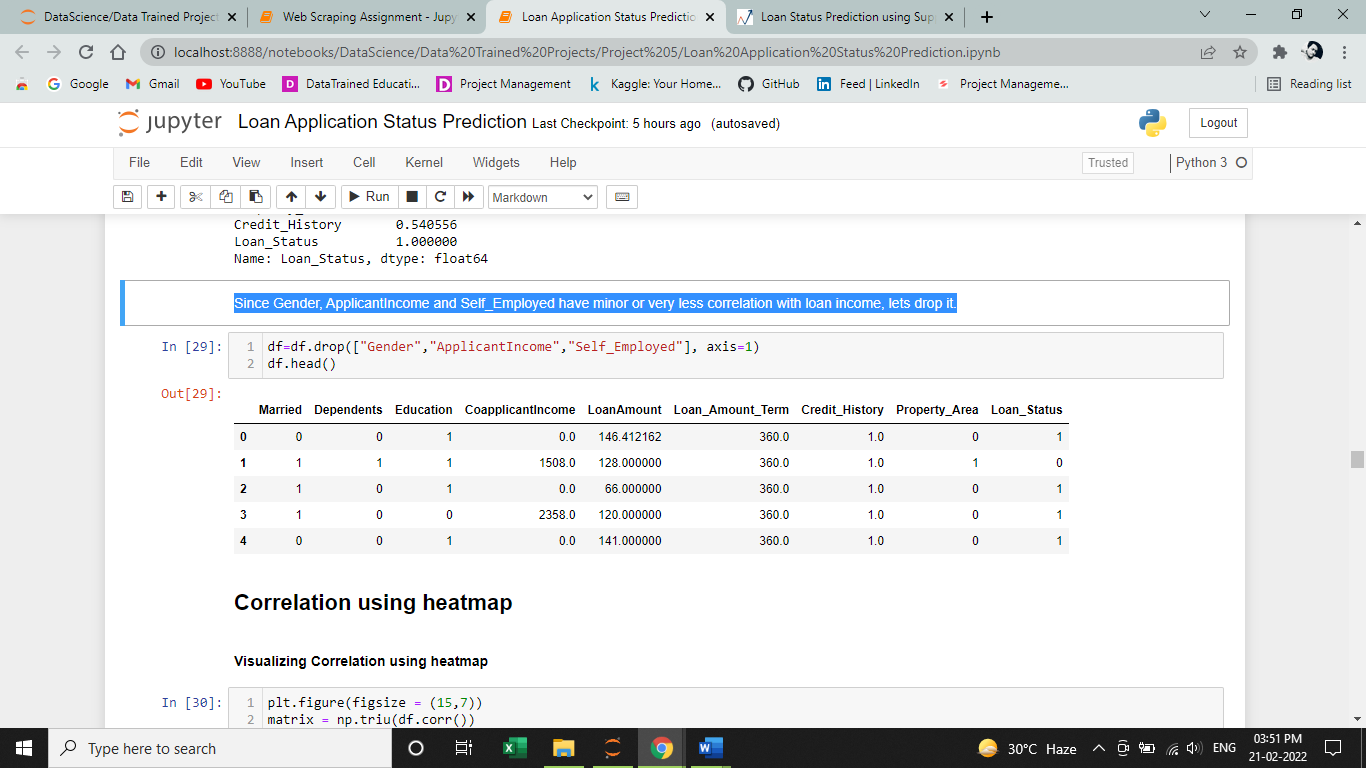




**Step 13**

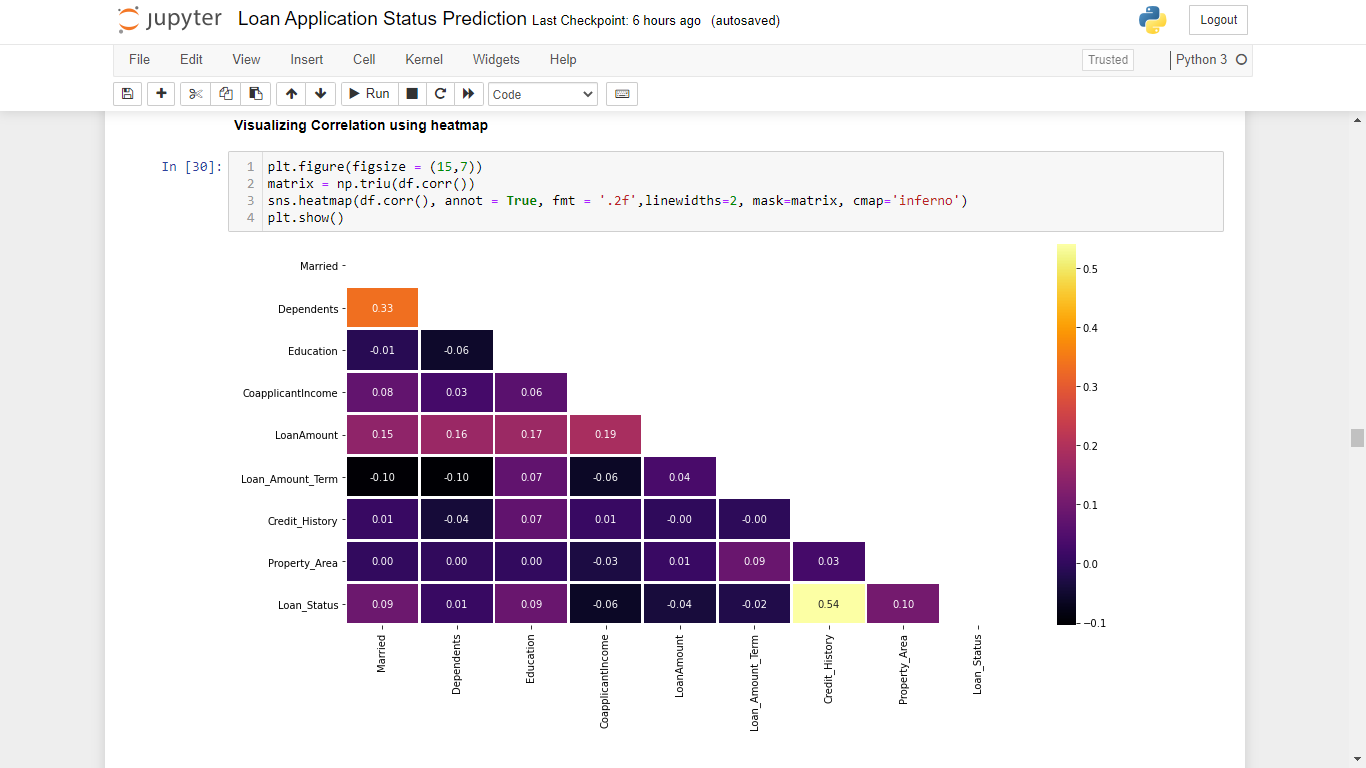
**Drop features which have zero or very less correlation with each other.**

Since Gender, ApplicantIncome and Self\_Employed have minor or very less correlation with loan income, lets drop it.



**Step 14**

**Visualizing Correlation matrix using heatmap**

****

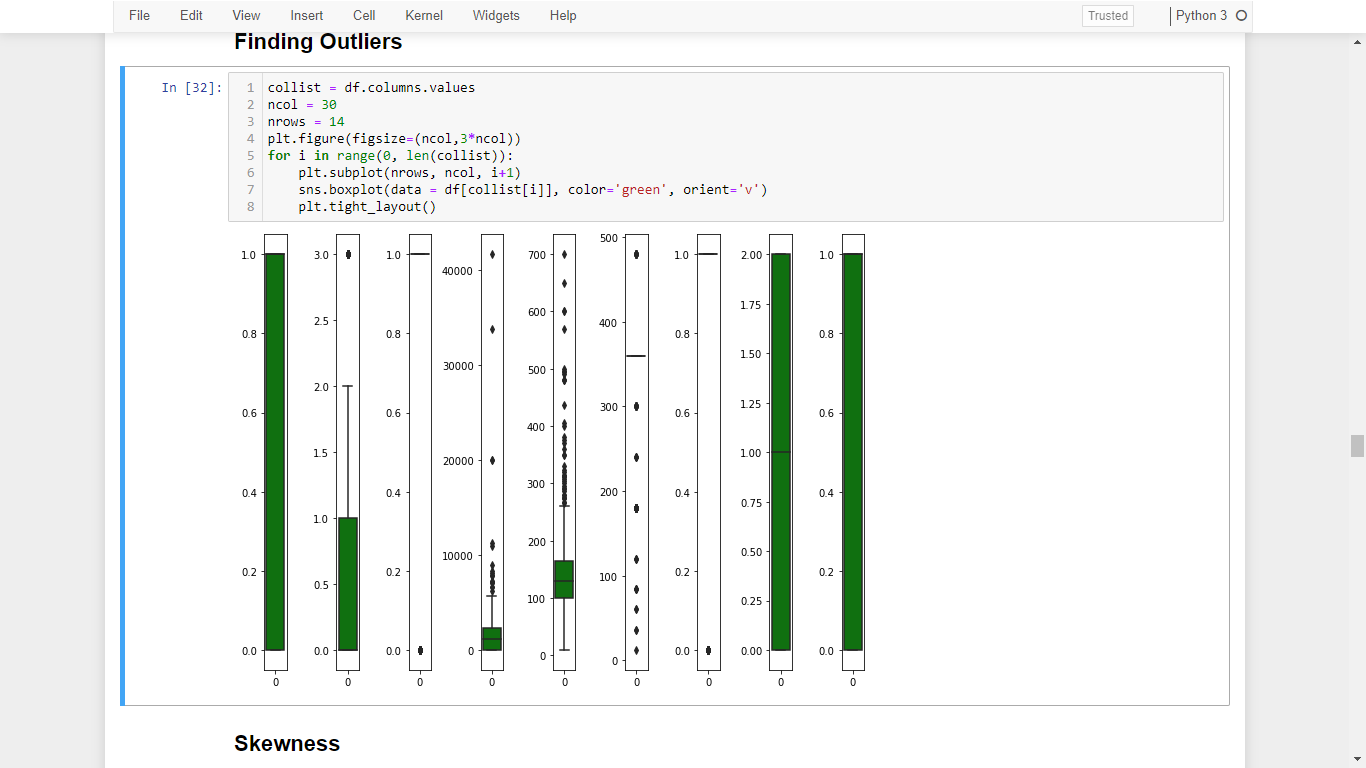
**Step 15**

**Visualizing Variables summary using Heat map**

****

**Step 16**

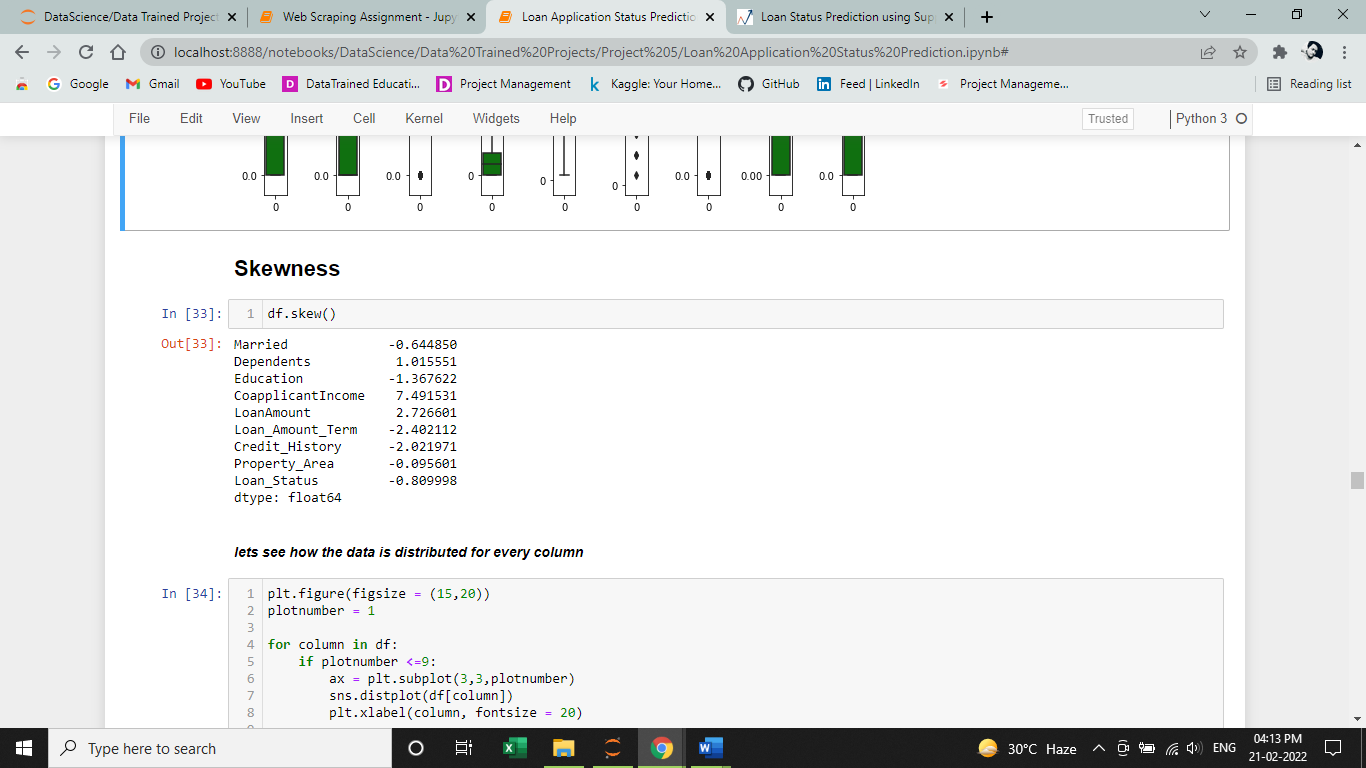
**Finding Outliers**



The above plot shows the outliers present in some features in the dataset **df.**

**Step 17**

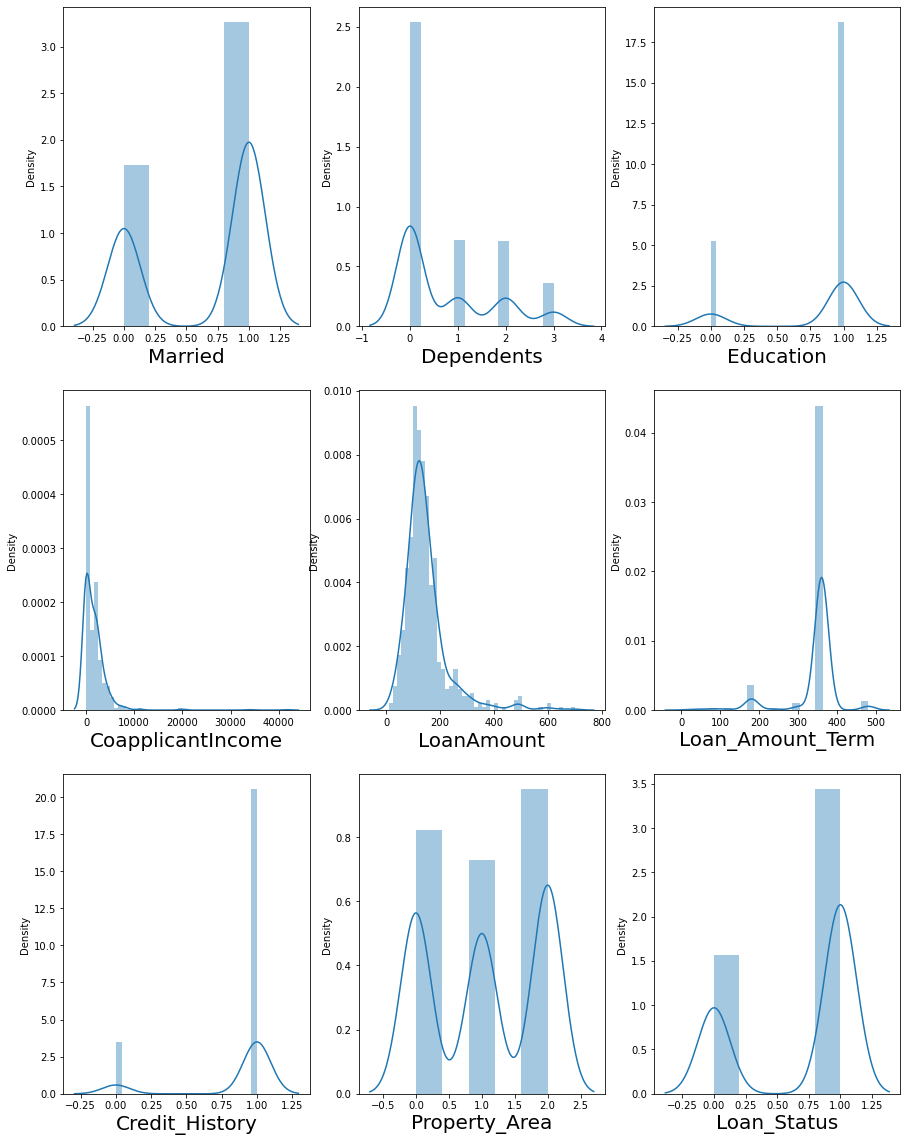
**Checking Skewness in the features**

****

We found that the dataframe **df**, have skewness in almost all the features.

**Step 18**

**Let’s visualize how the data is distributed for every column**

****

**Feature Engineering**

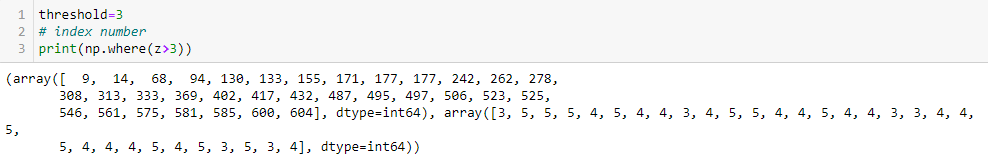
**Step 19**

**Removing Outliers using Z-Score Technique**

The Z-score is the signed number of standard deviations by which the value of an observation or data point is above the mean value of what is being observed or measured.

****

As, it is difficult to say which data point is an outlier. Let’s try and define a threshold to identify an outlier.



Don’t be confused by the results. The first array contains the list of row numbers and second array respective column numbers, which mean z[9][1] have a Z-score higher than 3.

Since, we found the outliers, now it’s time to remove the outliers by the below simple codes.

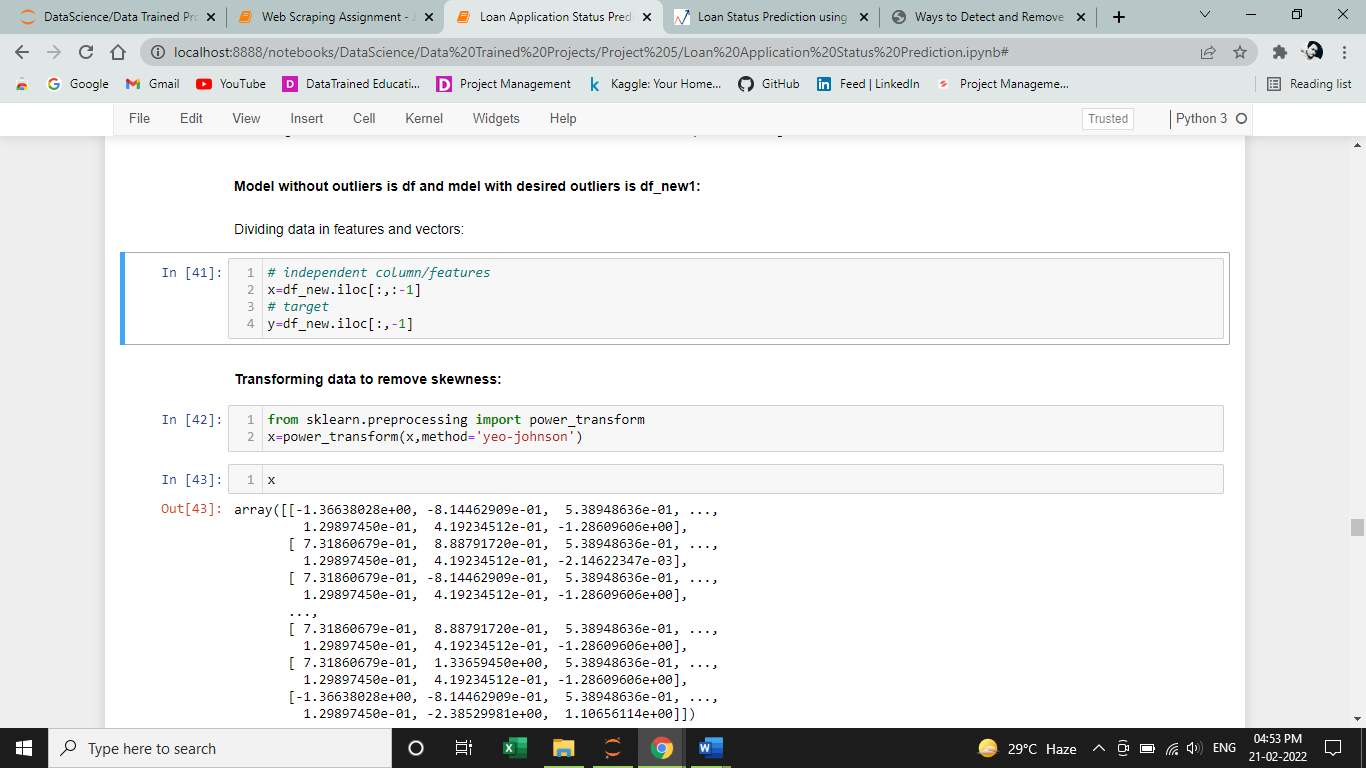


Let’s, check the total data loss percentage



**Step 20**

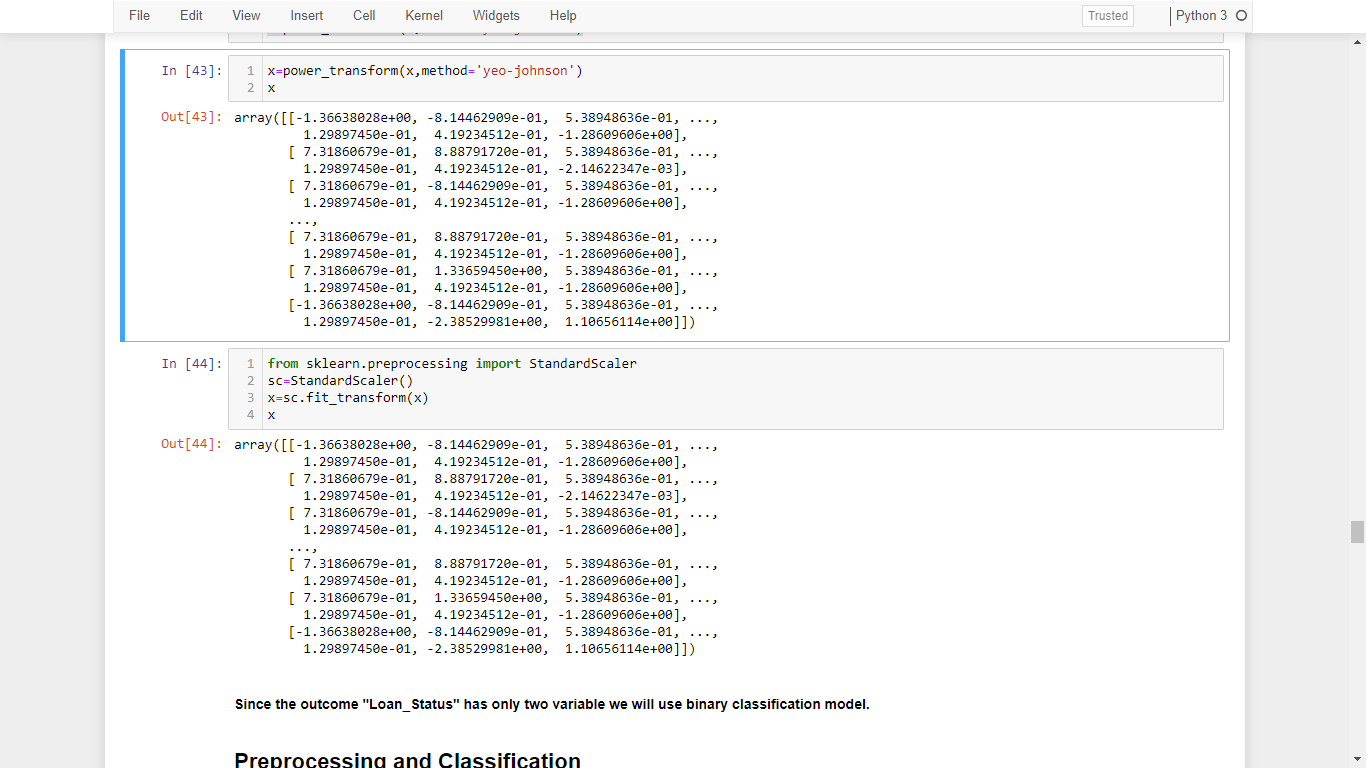
**Dividing data in features and vectors**

****

Above step, I have separated the features with the label into x and y.

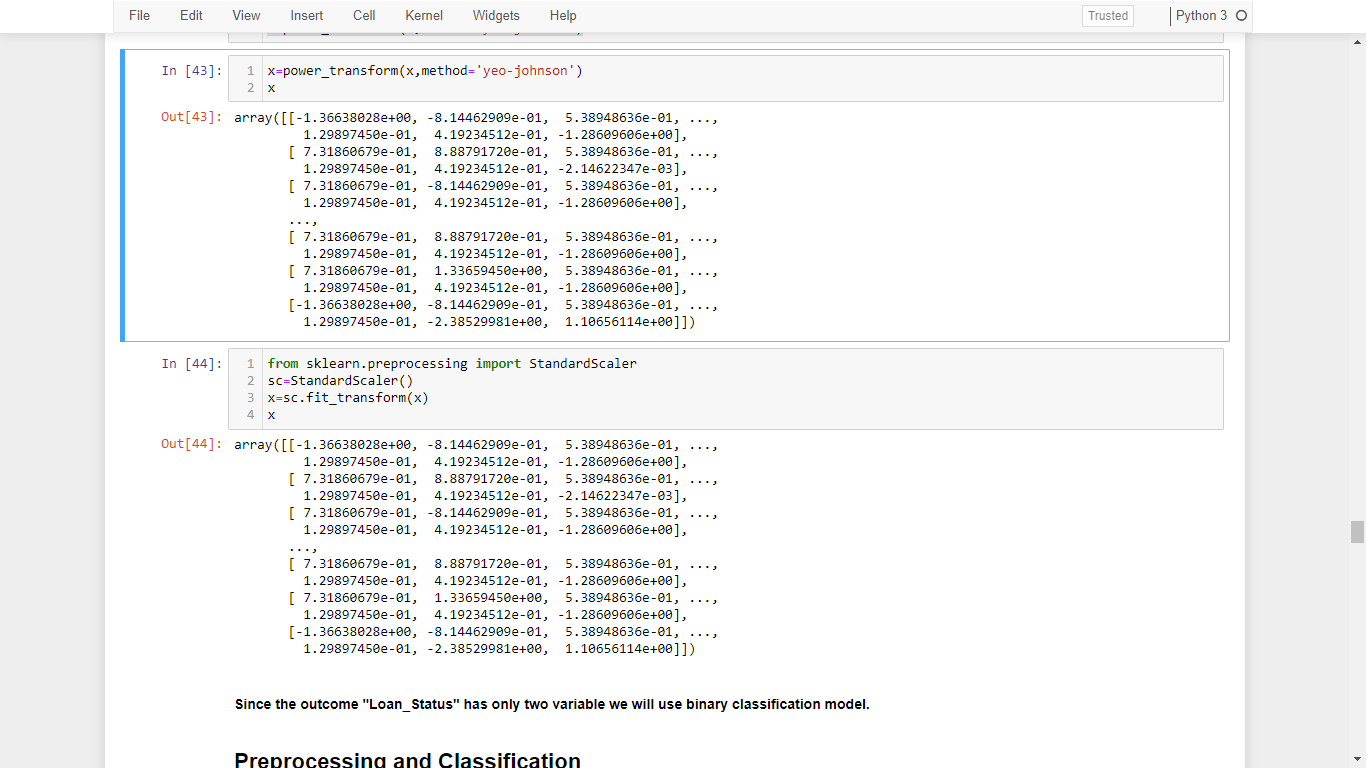
**Step 21**

**Transforming data to remove skewness using power\_transform**

****

**Step 22**

**Standardizing the data using StandardScaler**

****

I have used both power transform and standard scaler to remove skewness from the dataset and standardizing the data.

Since the outcome "Loan\_Status" has only two variable we will use binary classification model.

**Pre-processing and Classification**

**Step 23**

Split the “**Loan\_Status”** column from the other columns.



I found that the label is not equally distributed equally, so I use “**SMOTE”** to make the “Loan\_Status” column distribution equal as shown below.

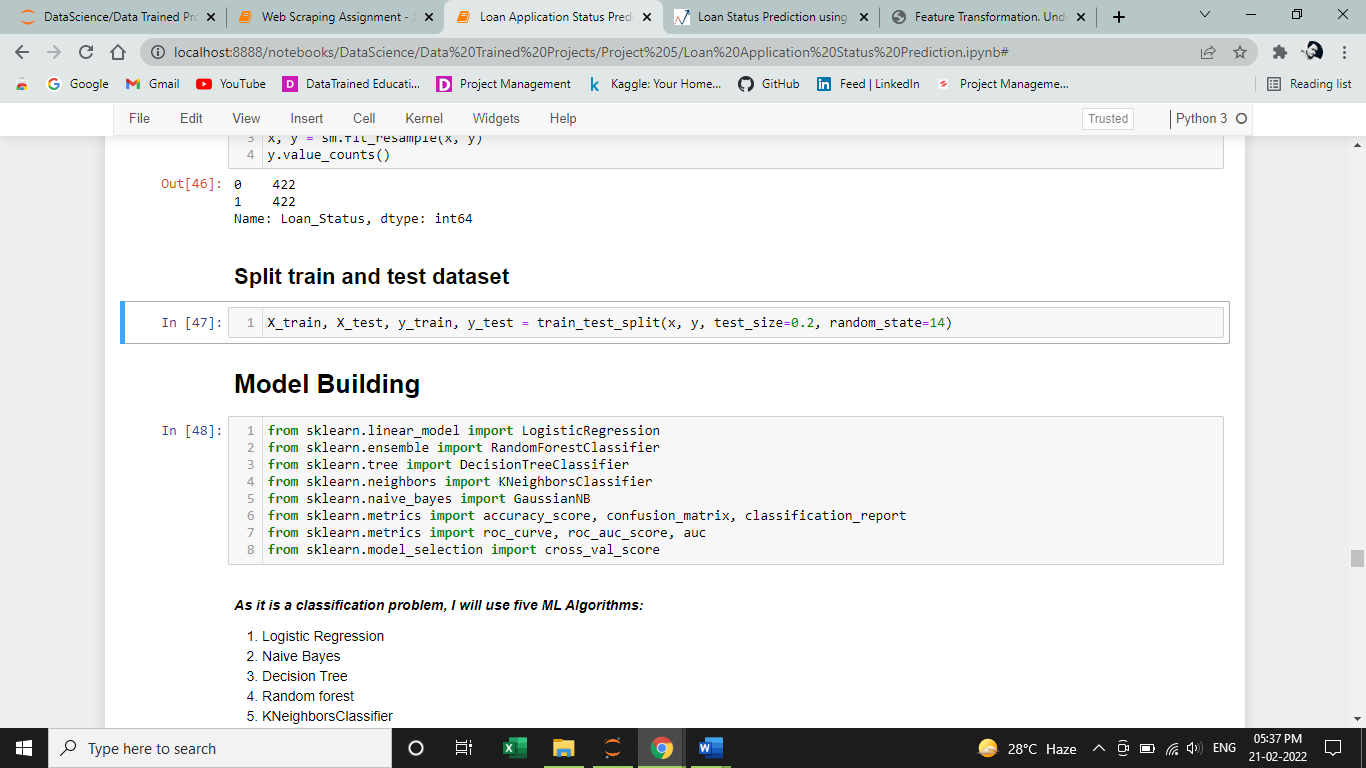
**Step 24**



Equally distributed the label i.e., 0 = 422 and 1 = 422.

**Split train and test Dataset**

**Step 25**

****

In this step, I have split the dataset into X\_train, X\_test, y\_train and y\_test with test\_size = 0.2 and random\_state = 14.

Let’s understand each of the variables by knowing what type of values they will be storing :

**X\_train**: contains a random set of values from variable ‘ X ‘

**y\_train**: contains the output (the Loan Status) of the corresponding value of X\_train.

**X\_test**: contains a random set of values from variable ‘ X ‘, excluding the ones already present in X\_train( as they are already taken).

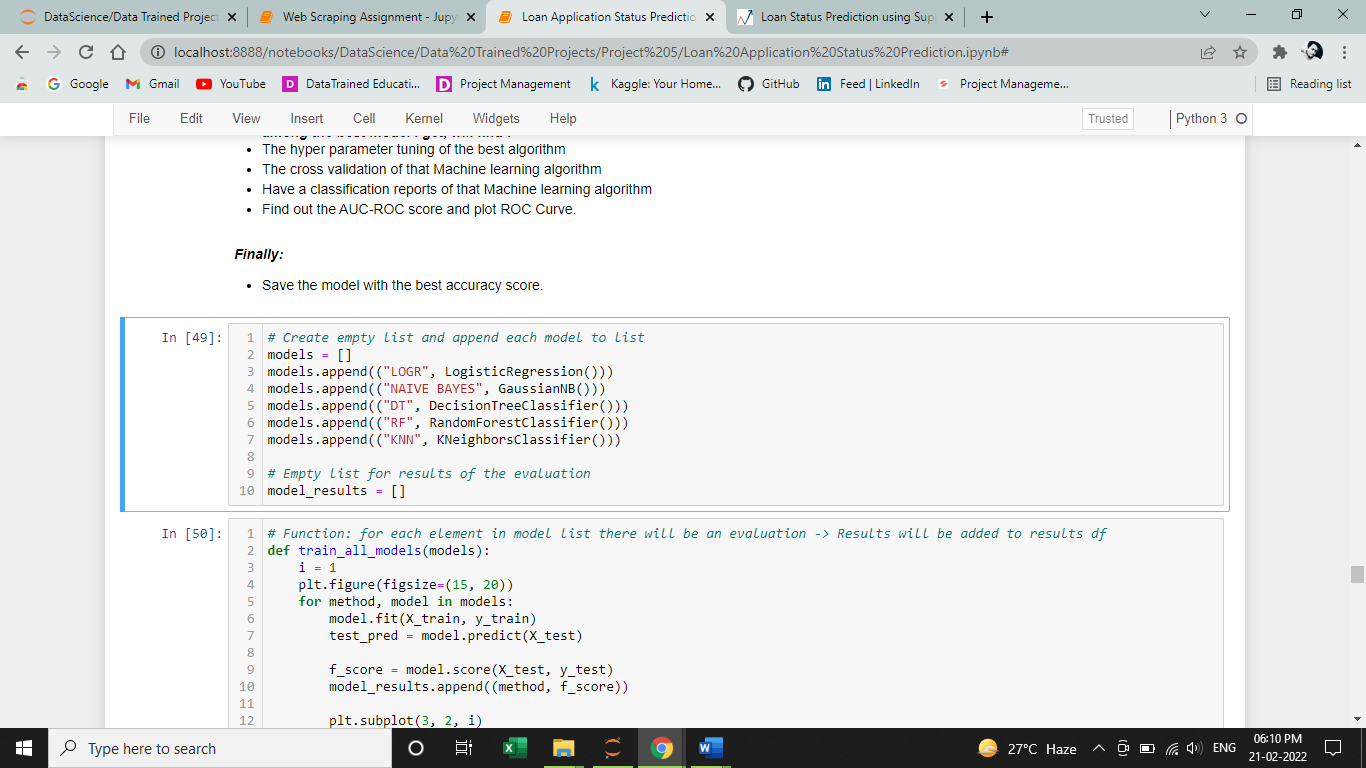
**Y\_test**: contains the output (the Loan Status) of the corresponding value of X\_test.

test\_size: represents the ratio of how the data is distributed among X\_train and X\_test (Here 0.2 means that the data will be segregated in the X\_train and X\_test variables in an 80:20 ratio). You can use any value you want. A value < 0.3 is preferred.

random\_state: Controls the shuffling applied to the data before applying the split.

**Model Building**

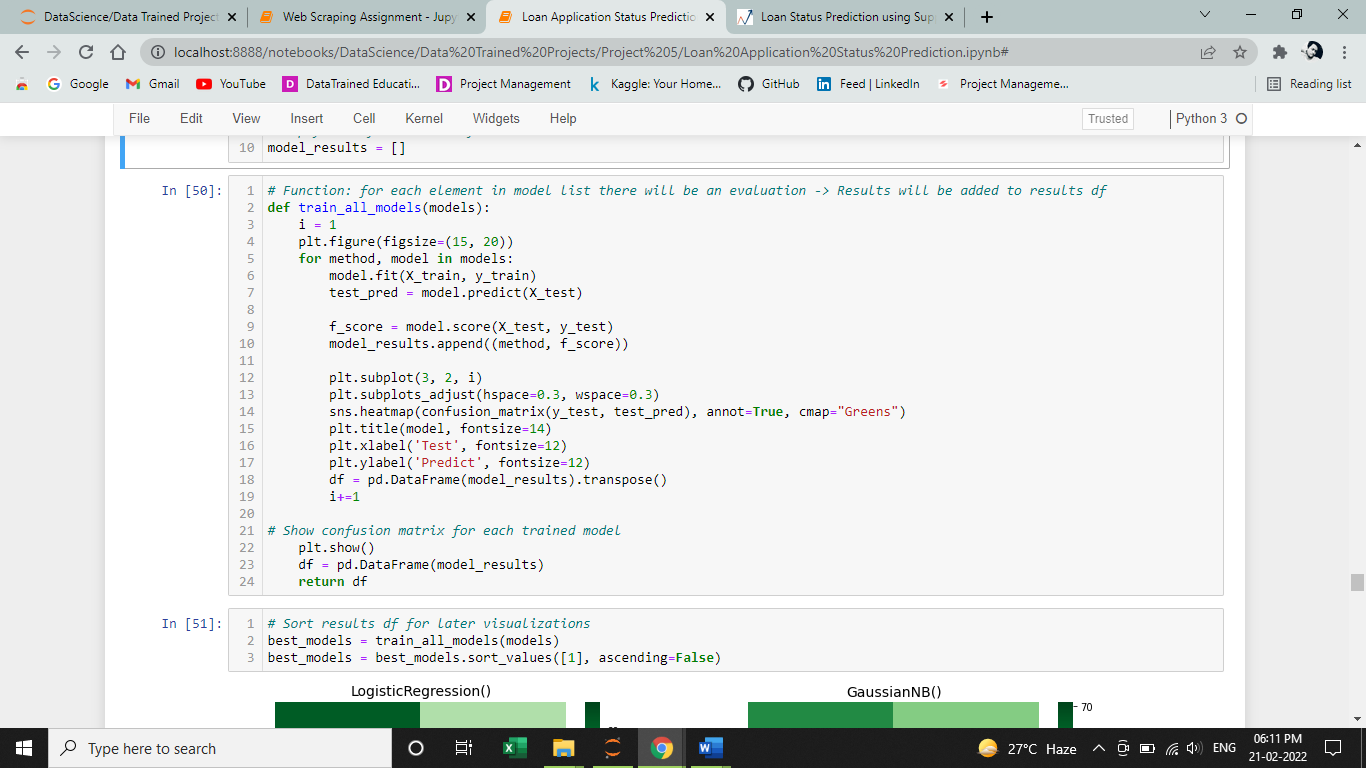
**Step 26**

****

Creating an empty list and append it with different types of algorithms.

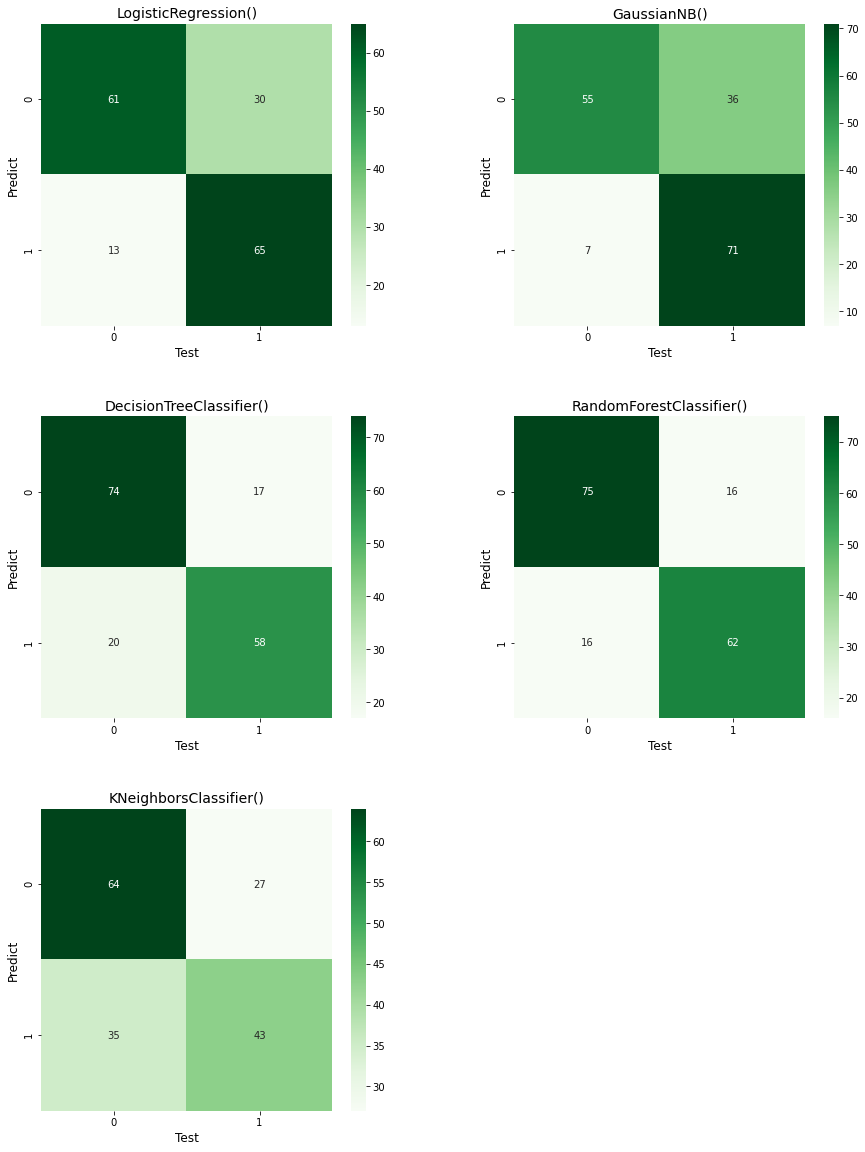
**Step 27**

**Defining a function, which will be called for fitting the model, finding f\_score & confusion\_matrix**



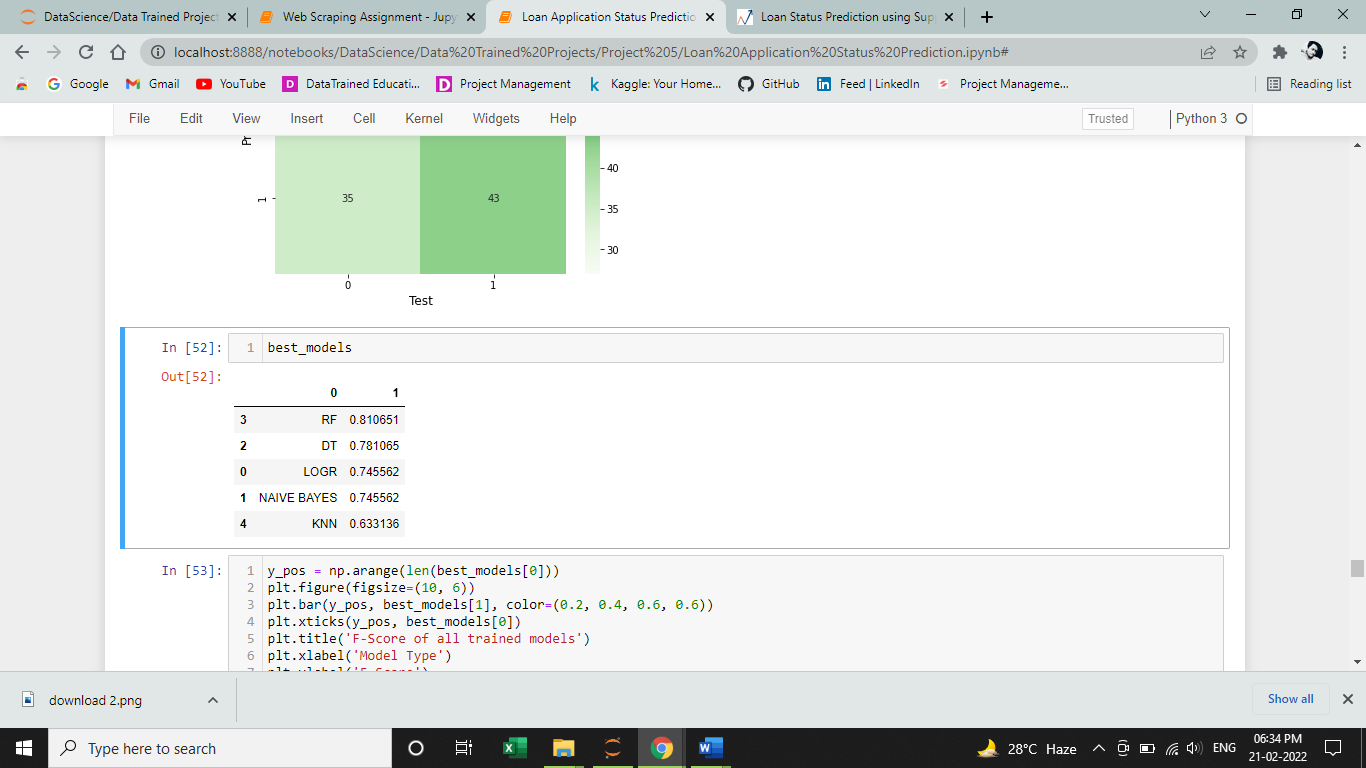
**Step 28**

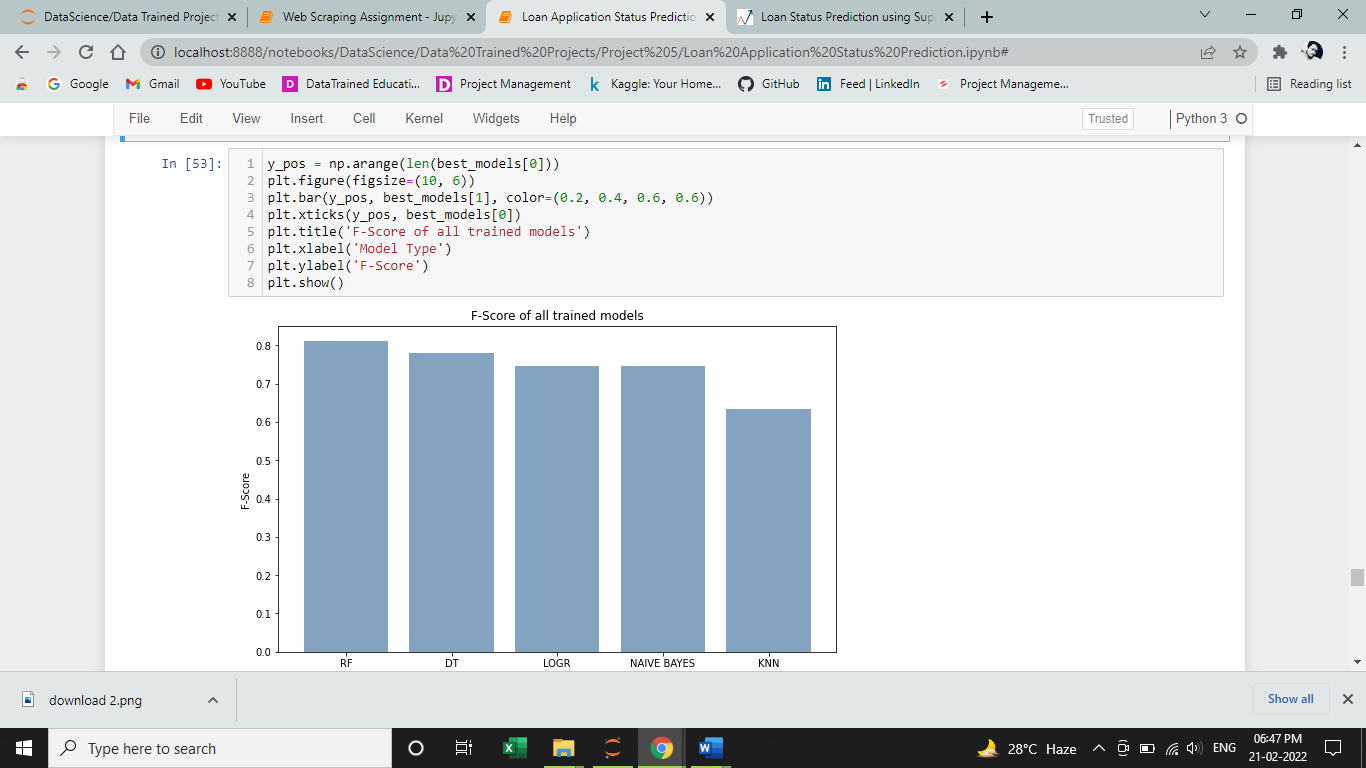
**Visualizing the confusing matrix**

****

**Step 29**

**Making a Dataframe containing scores of models and visualizing it using graph**

****



I found that Random Forest is showing the maxing f-score of the dataset while predestining. Where as KNN classifier shows the least f-score.

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

The sub-sample size is controlled with the max\_samples parameter if bootstrap=True, otherwise the whole dataset is used to build each tree.

This algorithm is widely used in E-commerce, banking, medicine, the stock market, etc.

Random forests are often considered the best off-the-shelf

black box algorithm for making accurate predictions. They

can readily accommodate missing values, nonlinear rela-

tionships, interactions, and a large numbers of covariates. A

number of studies have used random forests to predict the

onset of civil or interstate wars (Muchlinski et al., 2015).

We have argued here that the superior predictive perfor-

mance of random forests can be harnessed to examine the

same kinds of relationships in the data that political scien-

tists typically seek to uncover with conventional parametric

models, including making inferences about the marginal

effect of independent variable

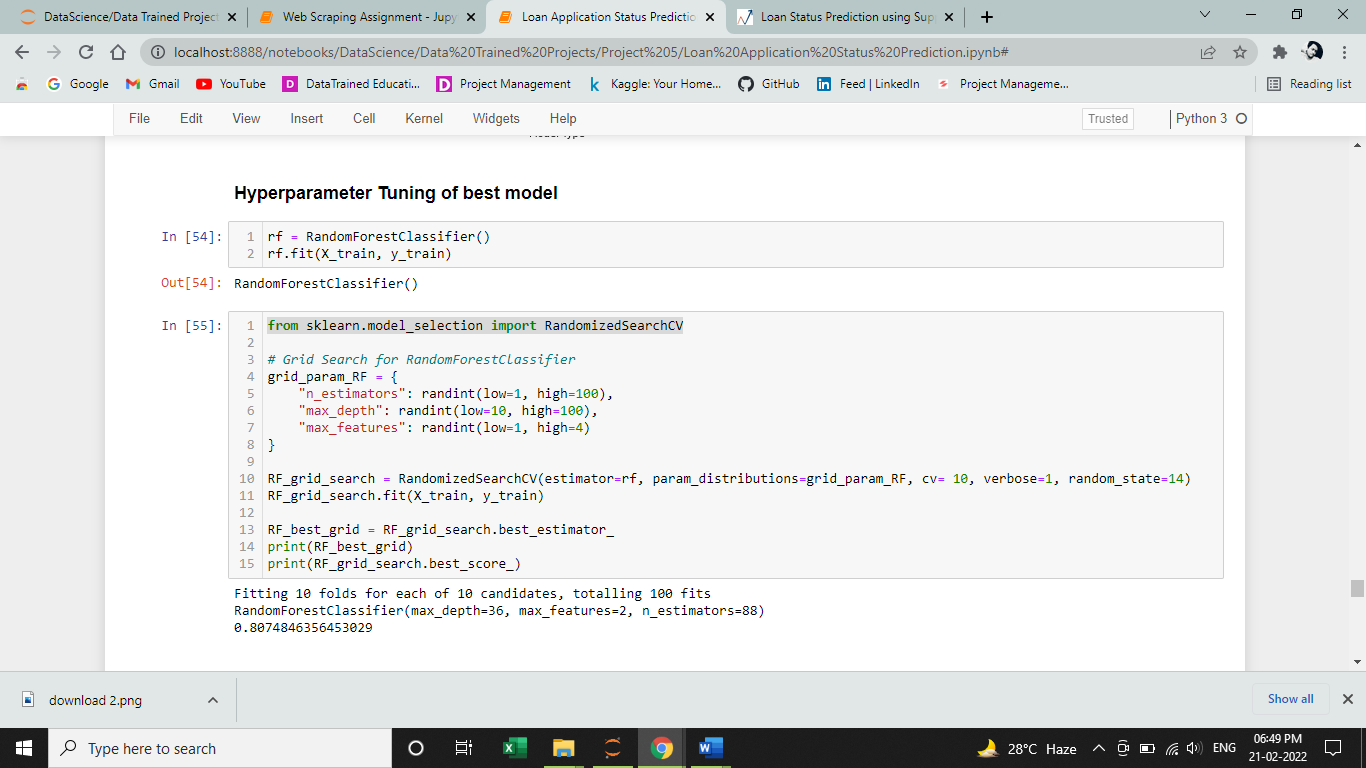
## WHAT IS RANDOM FOREST?

Random forest is a supervised learning algorithm. The "forest" it builds, is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

**Hyper Parameter Tuning of the model with maximum score**

**Step 30**

Since, RandomForestClassifier gave the maximum score, so let’s hyper parameter tune it with best parameters

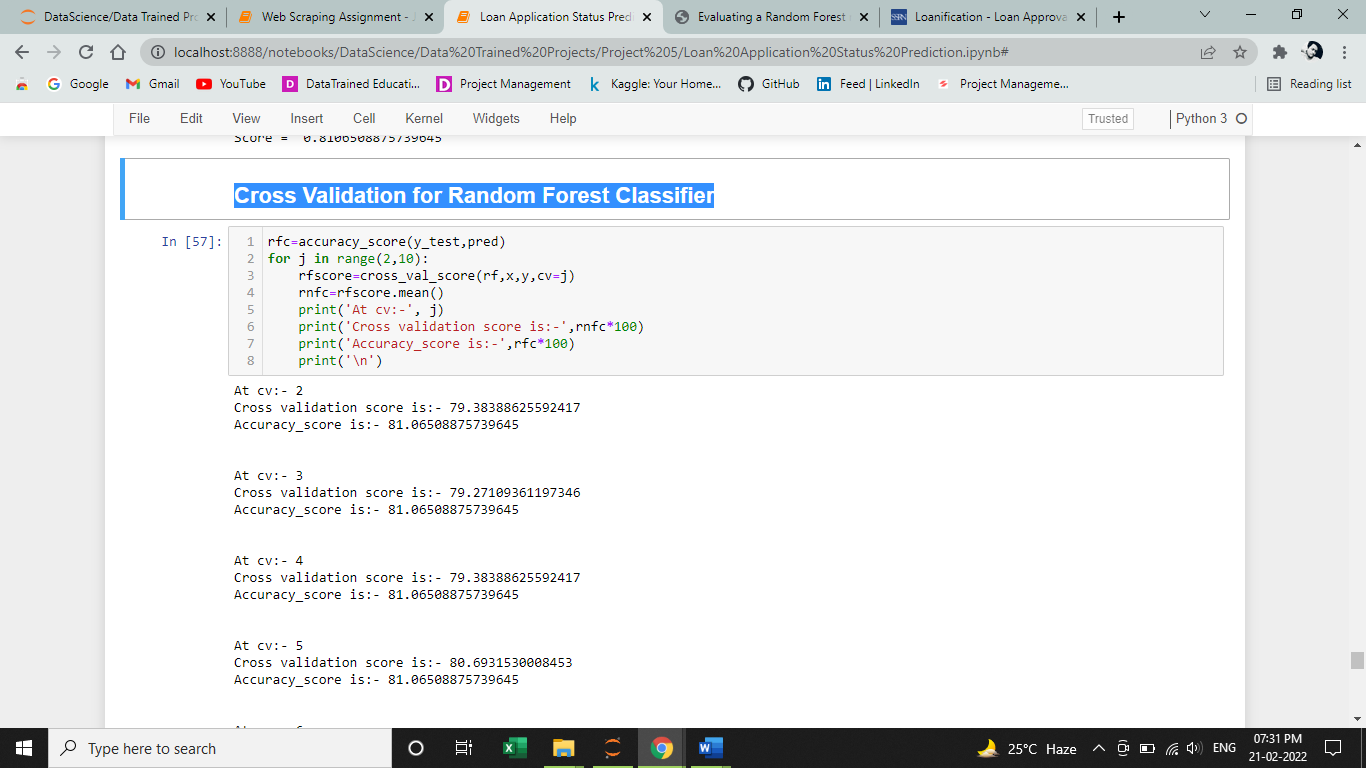
****

After Hyper-Parameter Tuning of RandomForestClassifier, got best parameter with max\_depth=36, max\_features=2 and n\_estimators=88.

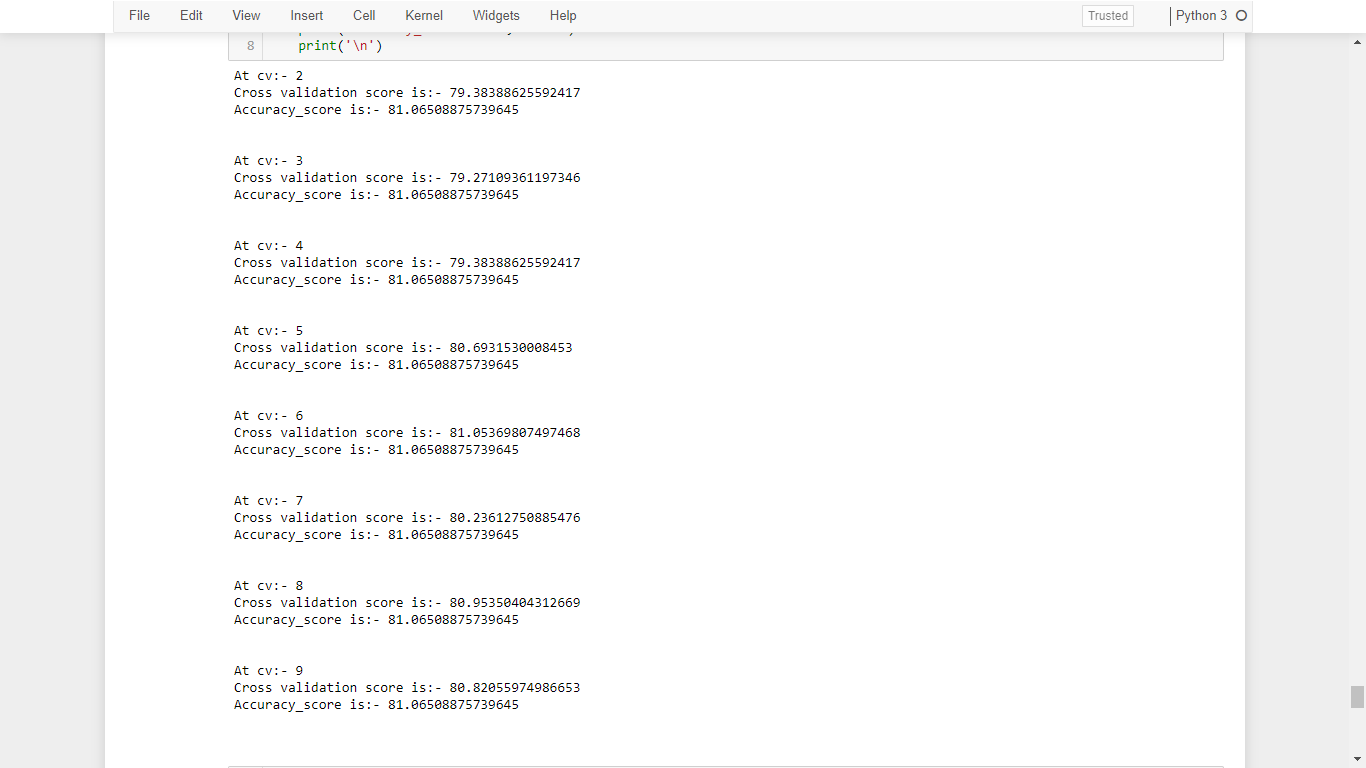
**Cross Validation for Random Forest Classifier**

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model

**Step 31**



Output of Cross Validatins – Next Page

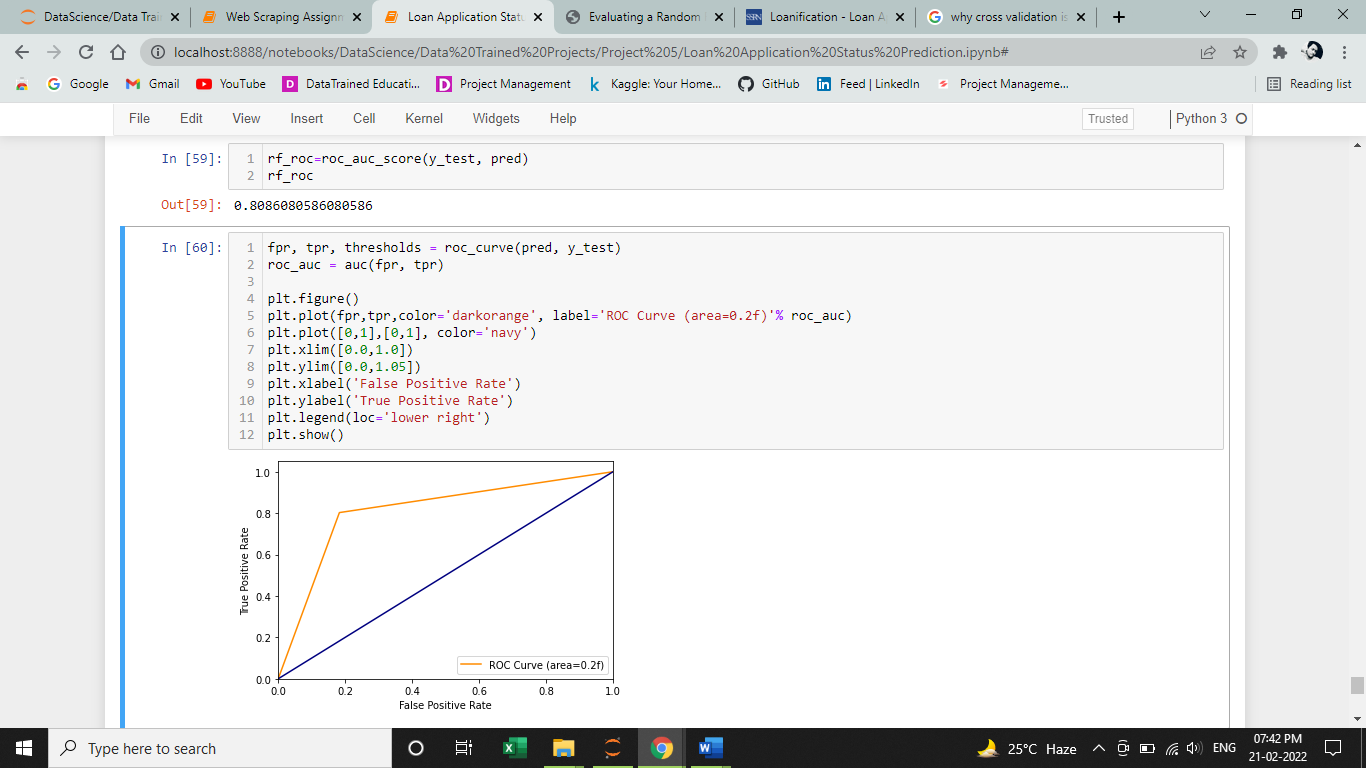


At cv = 6, it is giving the best accuracy score.

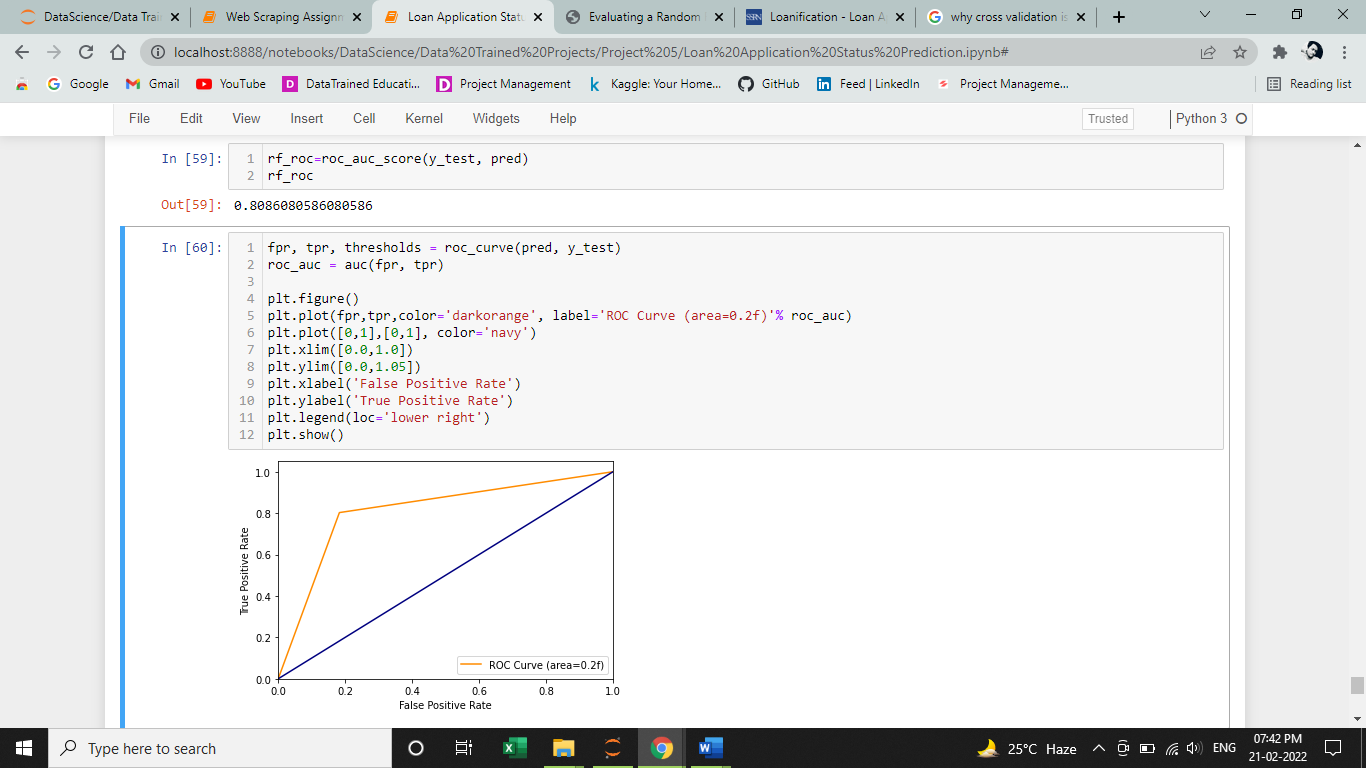
cv = 81.053, accuracy = 81.065

**ROC AUC curve for Random Forest**

**Step 32**

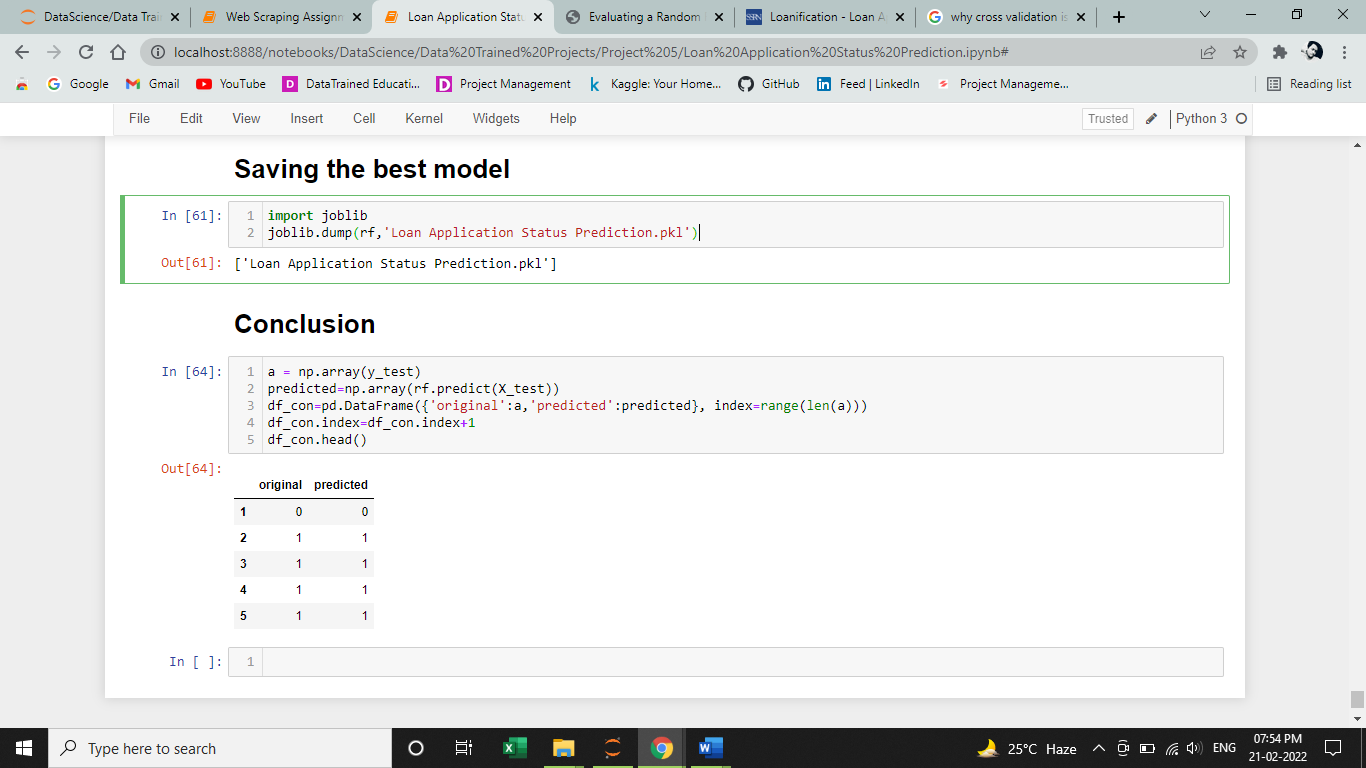
****

Hence, I got the random forest roc score of 80.86%. Hence, let plot the roc curve next.



**Saving the model**

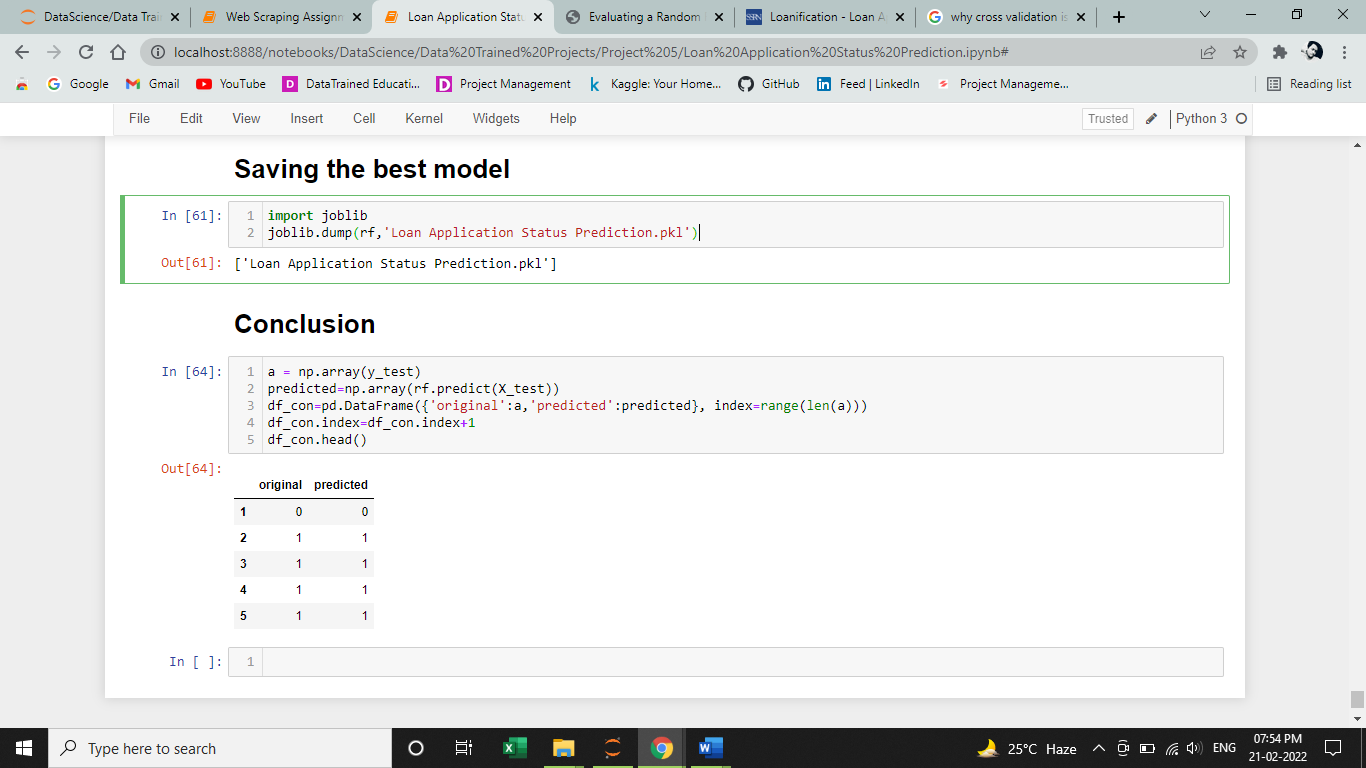
**Step 33**

****

Hence, I saved the model with the algorithm which is showing the best accuracy score in .pkl format file so that it will further use ahead for deployment.

**Let’s predict with our model**

**Step 34**

****

Hence, the model is predicting well.

**Conclusion**

Hence, by following all the steps properly, at the end, we were successfully able to train our classification model ‘Random Forest Classifier’ to predict the Loan Status of a person with an accuracy score of 81%, and have achieved the required task successfully.