INTRODUCTION

Loan is the essential product of banks and other financial institutions. As a big number of people go to banks to borrow money for different activities, the number of customers have increased and some banks expect to earn a lot of money as a result of interest paid on loans. However, loans are associated with risk of defaulting, i.e. the possibility that some borrowers may not be able to pay back their loans. Thus, high levels of non-performing loans can be a source of instability of the banking sector and lead to bankrupt. One of the important steps for banks to decide if a loan has to be authorized is to ensure that the candidate to borrow has the capacity of paying back the loan in the proposed terms. The advancement of technology like machine learning, computer science and other science is playing an important role by supporting banks to predict the probability of defaulting for a given customer based on his past behavior.

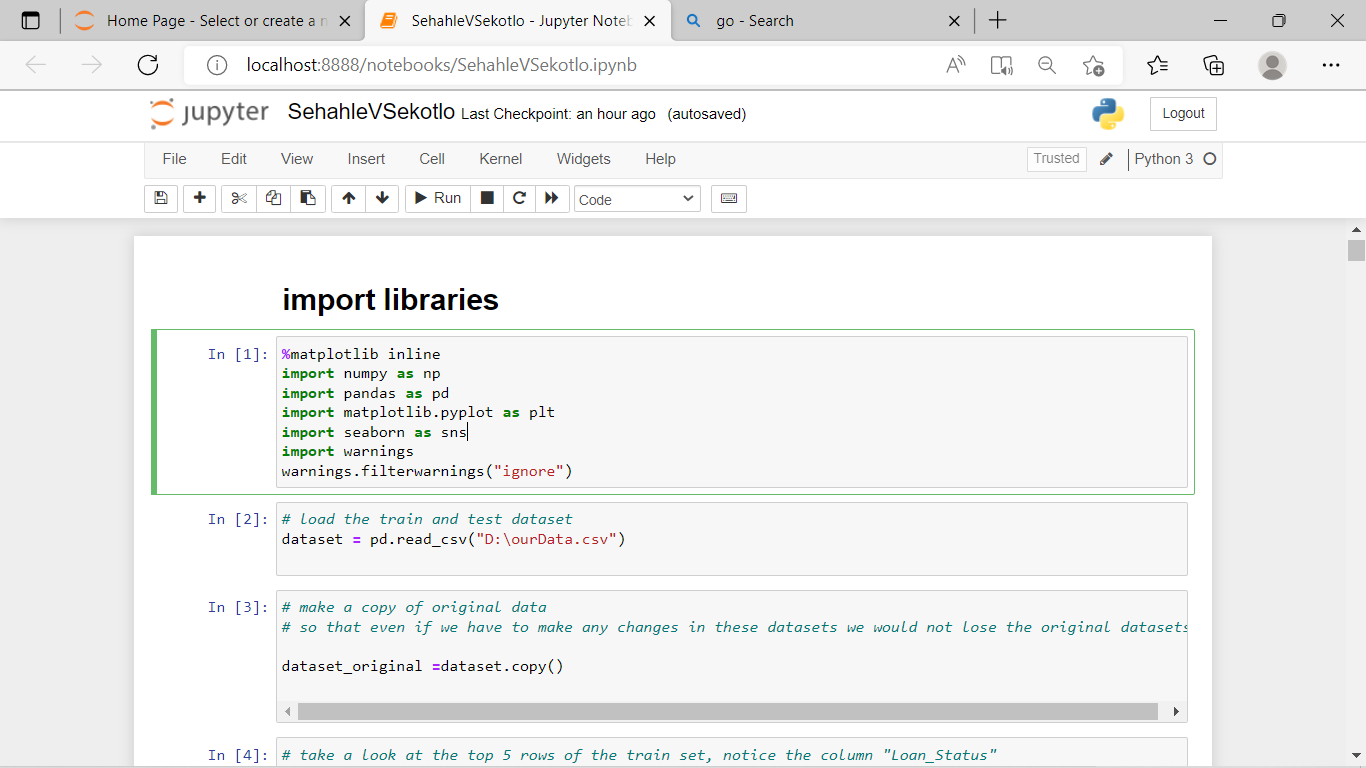
**Data Analysis**

[1]Different classification learning methods are heavily dependent on quantity and the quality of the data provided for training of the model. In this report, we will have an overview of the loan repayment dataset from Bank and perform exploratory data analysis in order to preprocess the data and improve the prediction results. The data will also be split into test sets (30%) and the training sets (70%). Training set is a data initial dataset that helps the program to understand how to learn and apply sophisticated technology. In addition, training models determine the good values for all weights and preference from the labeled dataset. For the train dataset, the Machine learning algorithm builds models that examine many dataset and attempts to find the model that minimizes the loss. The test data set is independent of the training data dataset, but it has the same probability distribution as the training dataset, and it is used to measure the performance. Training dataset will be used to fit the model, and the test dataset will be to evaluate the best model to get an estimation of generalization error.

**Data importation**

The first thing we have did is to import libraries and a given dataset into anaconda IDE product called Jupyter notebook python 3 for the purpose of data cleaning and processing in order to come up with an appropriate model to predict data accuracy.

Figurer 1 below is an illustration of the above description:



**Fig 1**

**Data description**

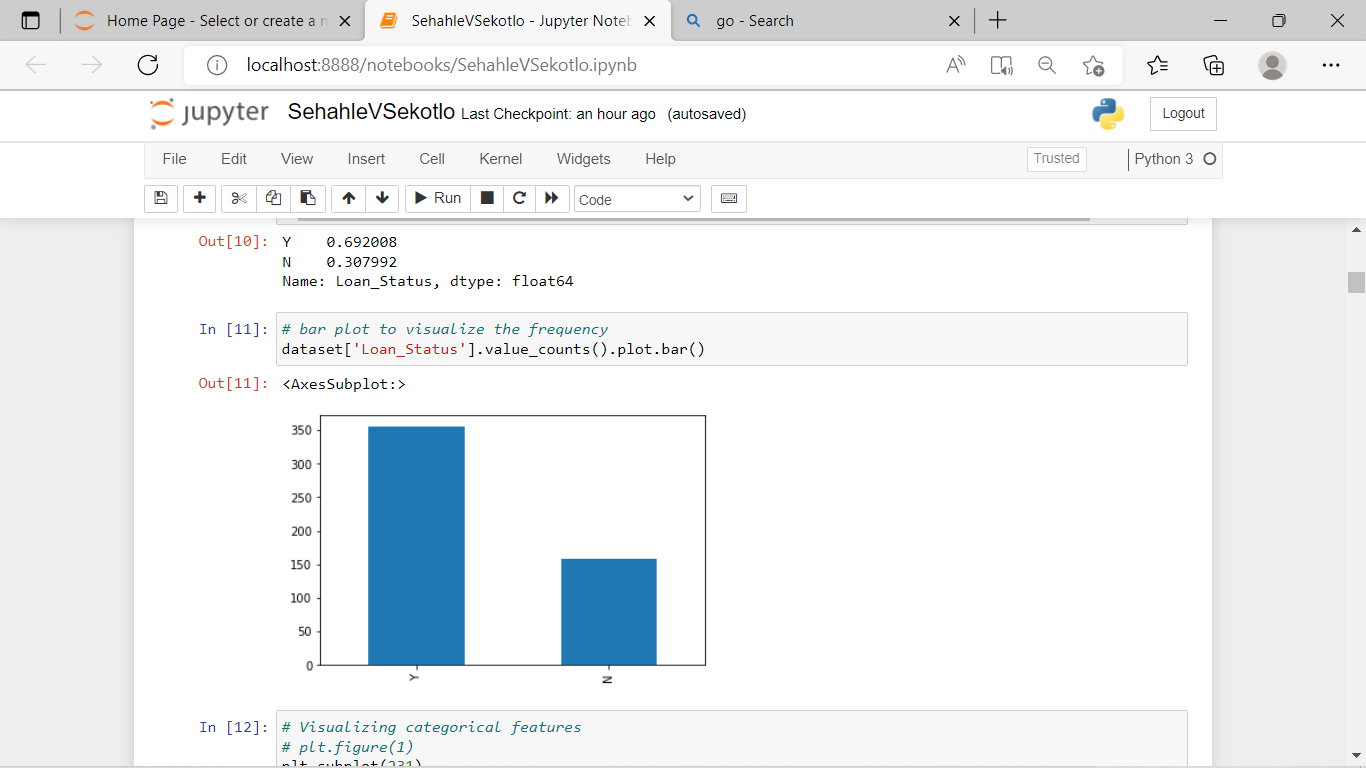
We have used Python to explore the data in order to gain a better understanding of the features and target variable. We will also analyze the data to summarize their main characteristics, using various visualization techniques, more rapidly we are going to use pie charts for visualization.

**Univariate analysis**

Univariate analysis is when we analyze each variable individually. For categorical features, we can use frequency table or bar plots, which will calculate the number of each category in a particular variable. For numerical features, a histogram or a box-plot can be used to look at the distribution of the variable. With a histogram, you can check the central tendency, variability, modality, and kurtosis of a distribution. Note that a histogram cannot show you if you have any outliers. This is why we also use box-plots.

**Target Variable (Categorical)**

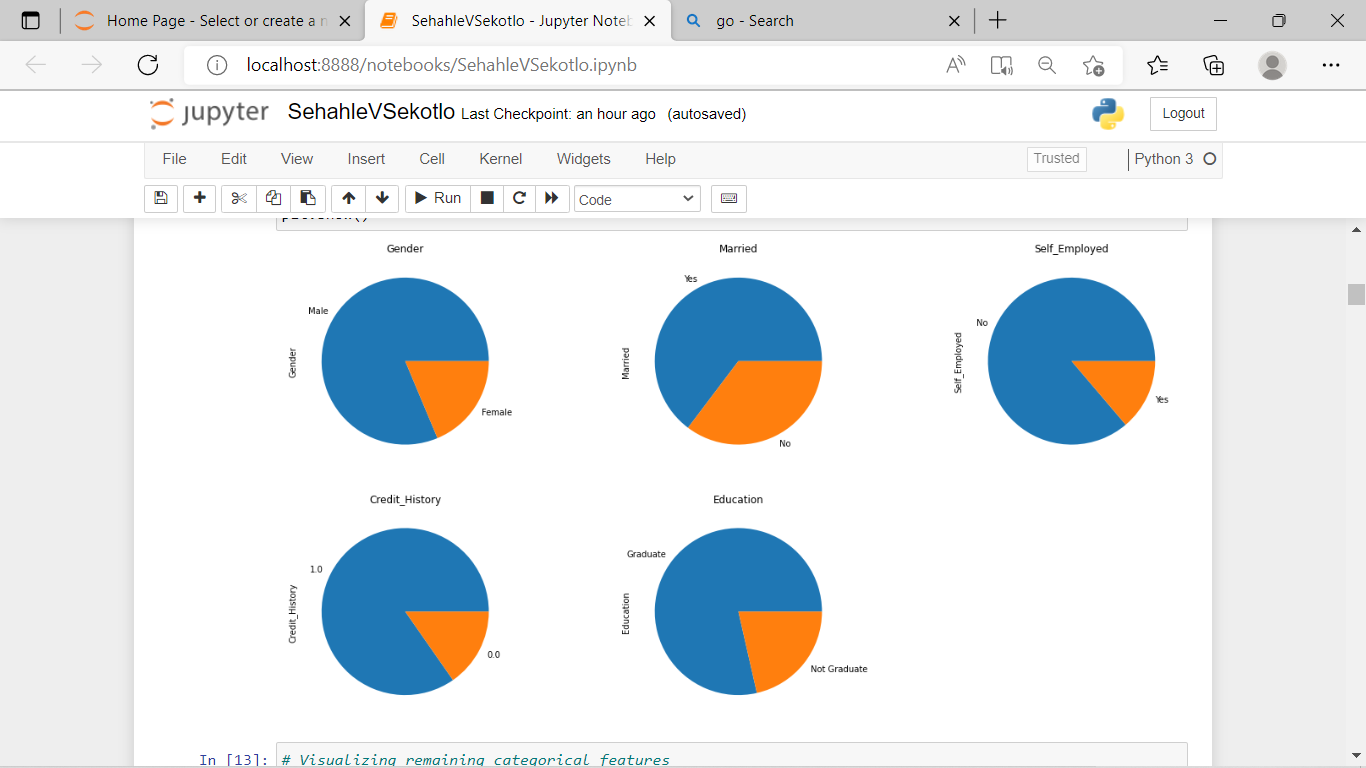
We will first look at the target variable, i.e., Loan\_Status. As it is a categorical variable, let us look at its frequency table, percentage distribution and bar plot.



**Fig 2**

From the bar chat above, we can see that most of applicants (69%) appears to be approved, that is demonstrated by Y (yes) and less applicants (31%) are disapproved and are demonstrated by N (no) as their loan Status.

**Other features**

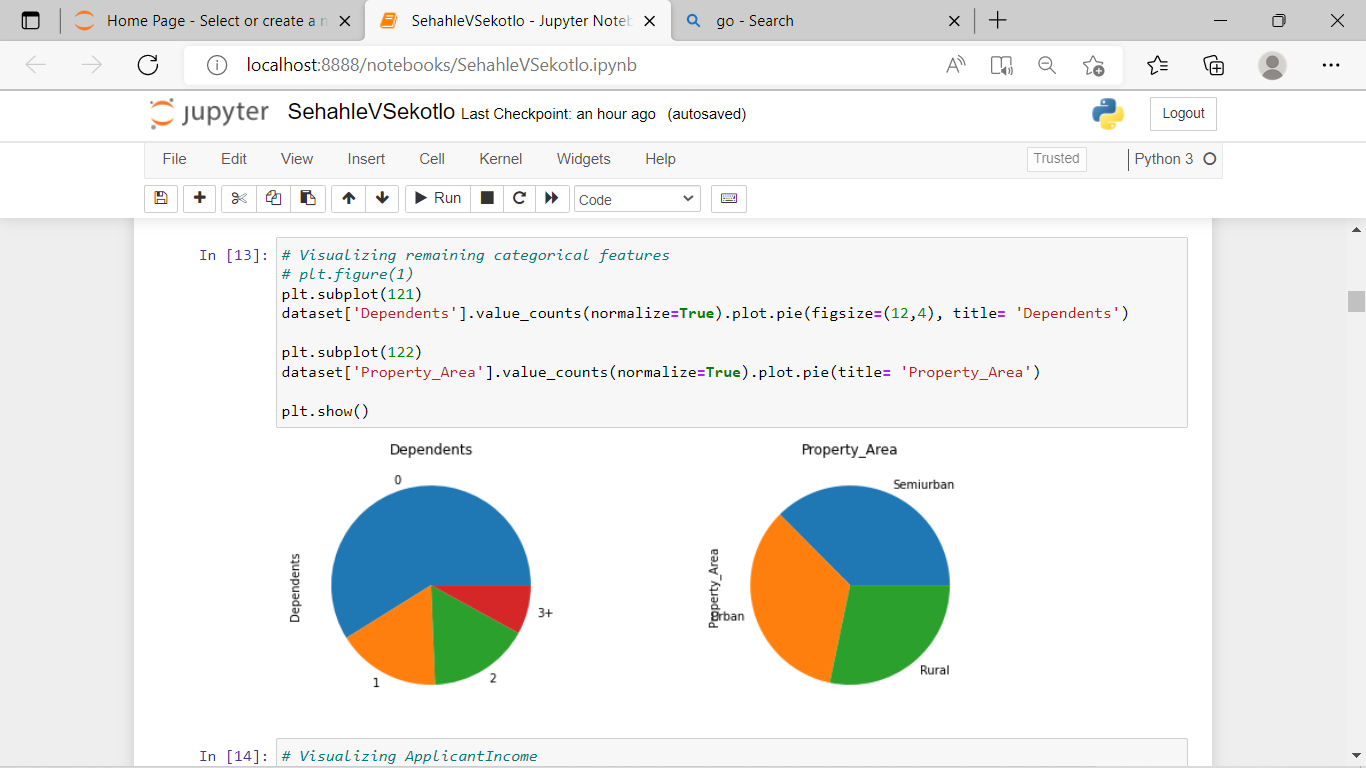
****

**Fig 3**

It can be inferred from the above pie plots fig 3 that:

* 80% applicants in the dataset are male.
* Around 65% of the applicants in the dataset are married.
* Around 15% applicants in the dataset are self-employed.
* Around 85% applicants have credit history (repaid their debts).
* Around 80% of the applicants are Graduate.

**More other features (independents)**

****

**Fig 4**

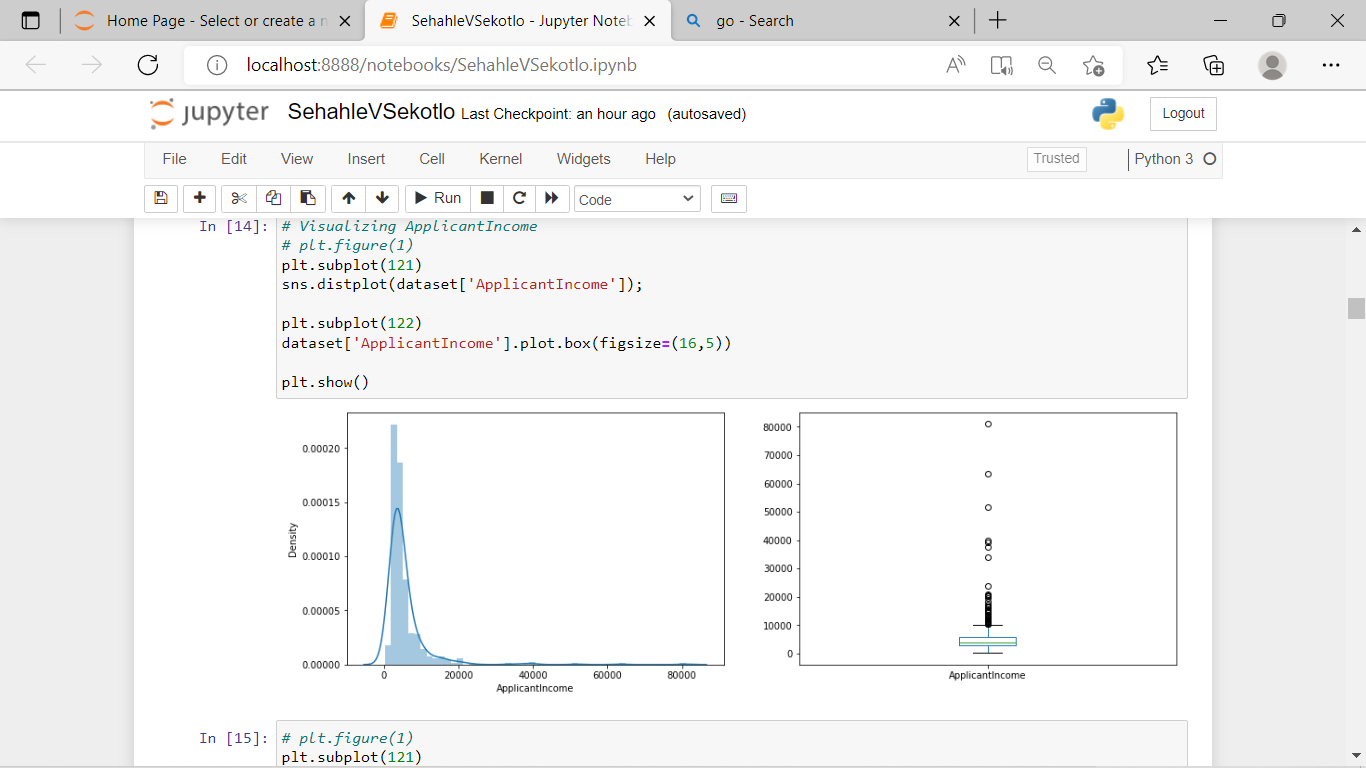
Following inferences can be made from the above pie plots:

* More than half of the applicants do not have any dependents.
* Most of the applicants are from Semiurban area.

### Independent Variable (Numerical)

There are four features that are Numerical: These features have numerical values (ApplicantIncome, CoapplicantIncome, LoanAmount, Loan\_Amount\_Term)

Firstly, let us look at the Applicant income distribution

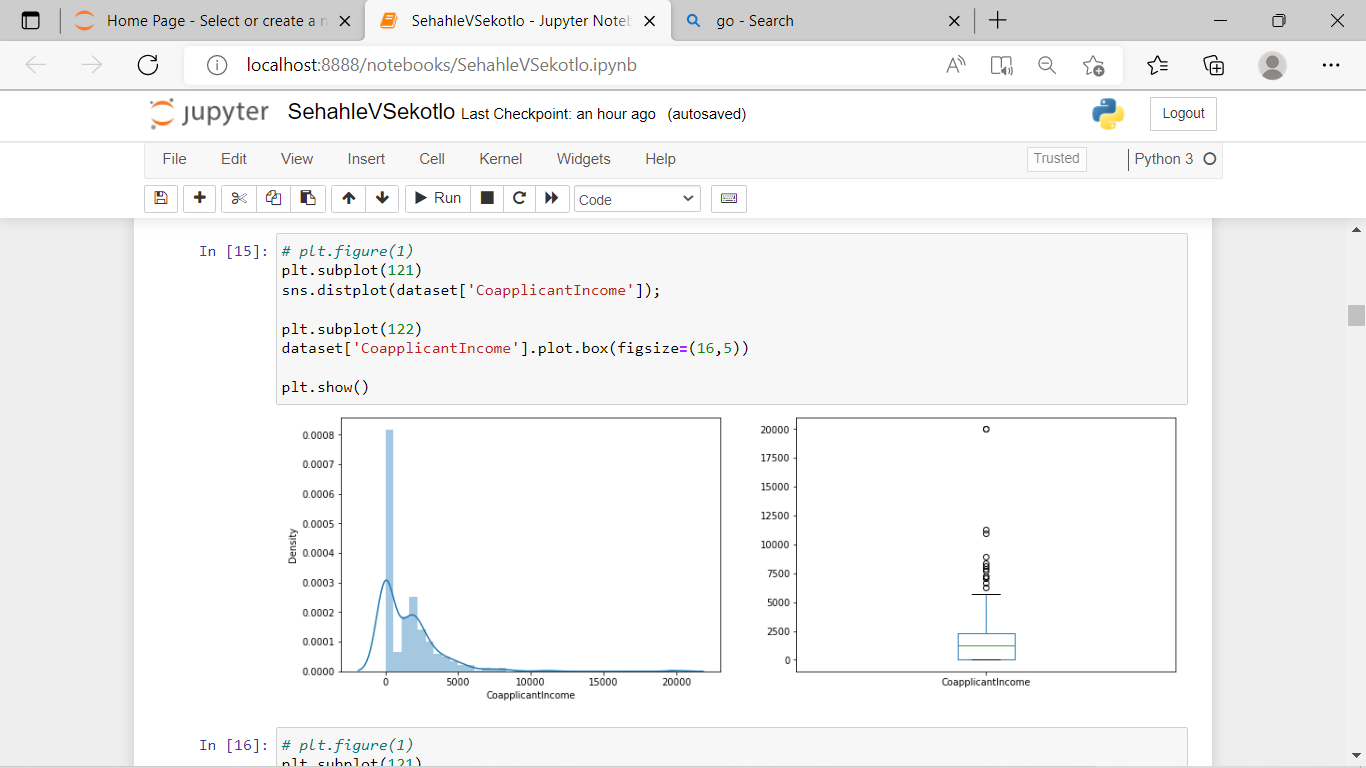


It can be inferred that most of the data in the distribution of applicant income is towards left, which means it is not normally distributed. The distribution is right-skewed (positive skewness). We will try to make it normal in later sections as algorithms works better if the data is normally distributed.

The reason for this positive skewness may be because other people are graduated and others are not graduated and this makes a huge difference in their salaries.

**Co-applicant income**

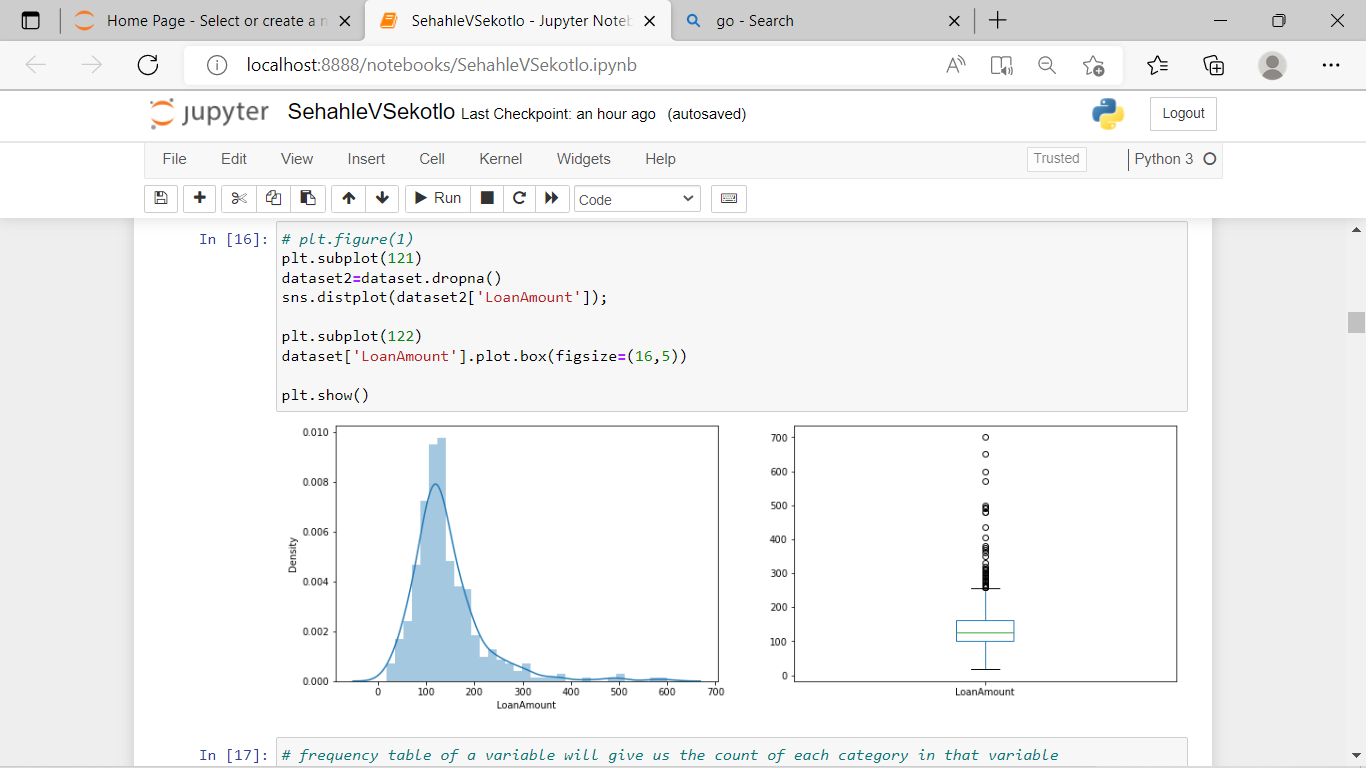
**Fig 5**

****

We see a similar distribution as that of the applicant income. Majority of co-applicant’s income ranges from 0 to 5000. We also see many outliers in the co-applicant income and it is not normally distributed.

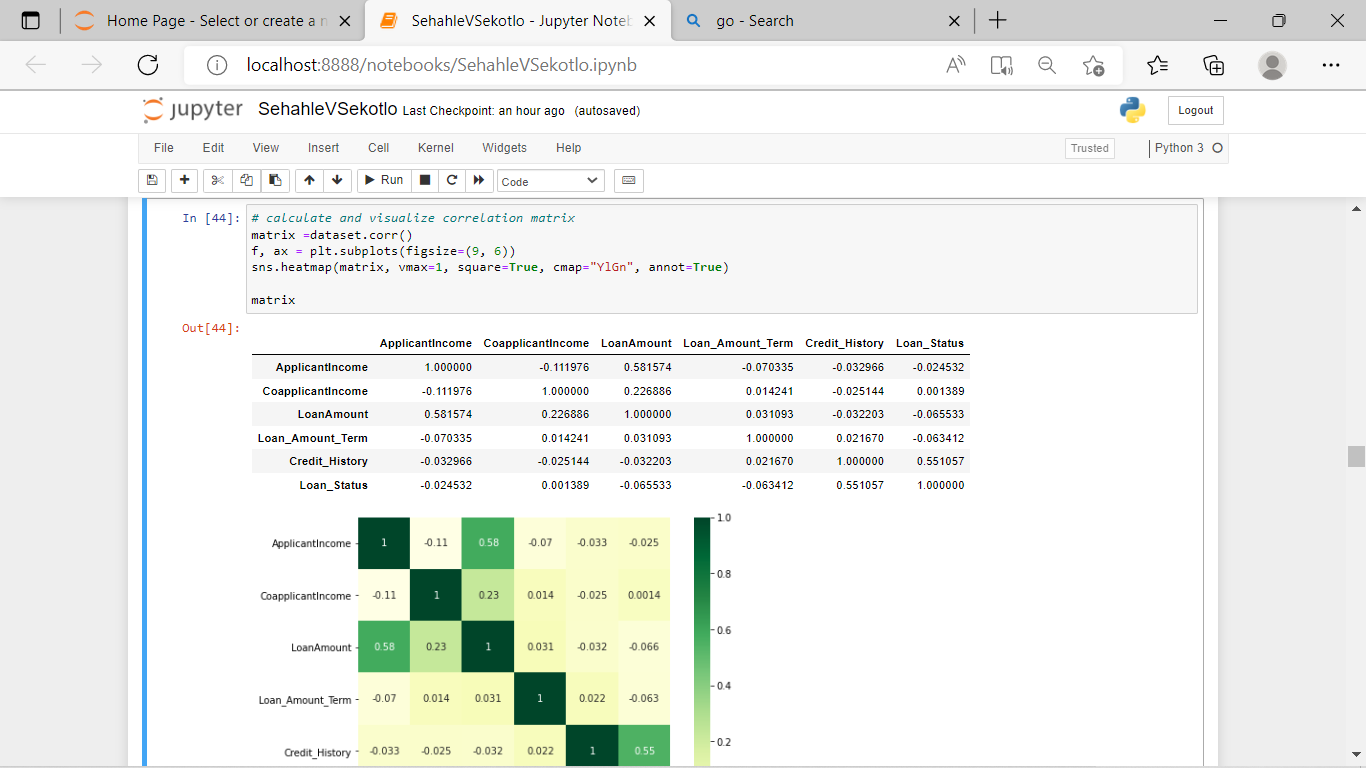
**Loan Amount**

Thirdly, let us look at the distribution of Loan Amount variable.



**Correlation**

Now lets look at the correlation between all the numerical variables. We can use the corr() to compute pairwise correlation of columns, excluding NA/null values using pearson correlation coefficient. Then we will use the heat map to visualize the correlation. Heatmaps visualize data through variations in coloring. The variables with darker color means their correlation is more.



# **Describing and correcting of Missing value and outlier treatment**

After exploring all the variables in our data, we can now impute the missing values and treat the outliers because missing data and outliers can have adverse effect on the model performance.

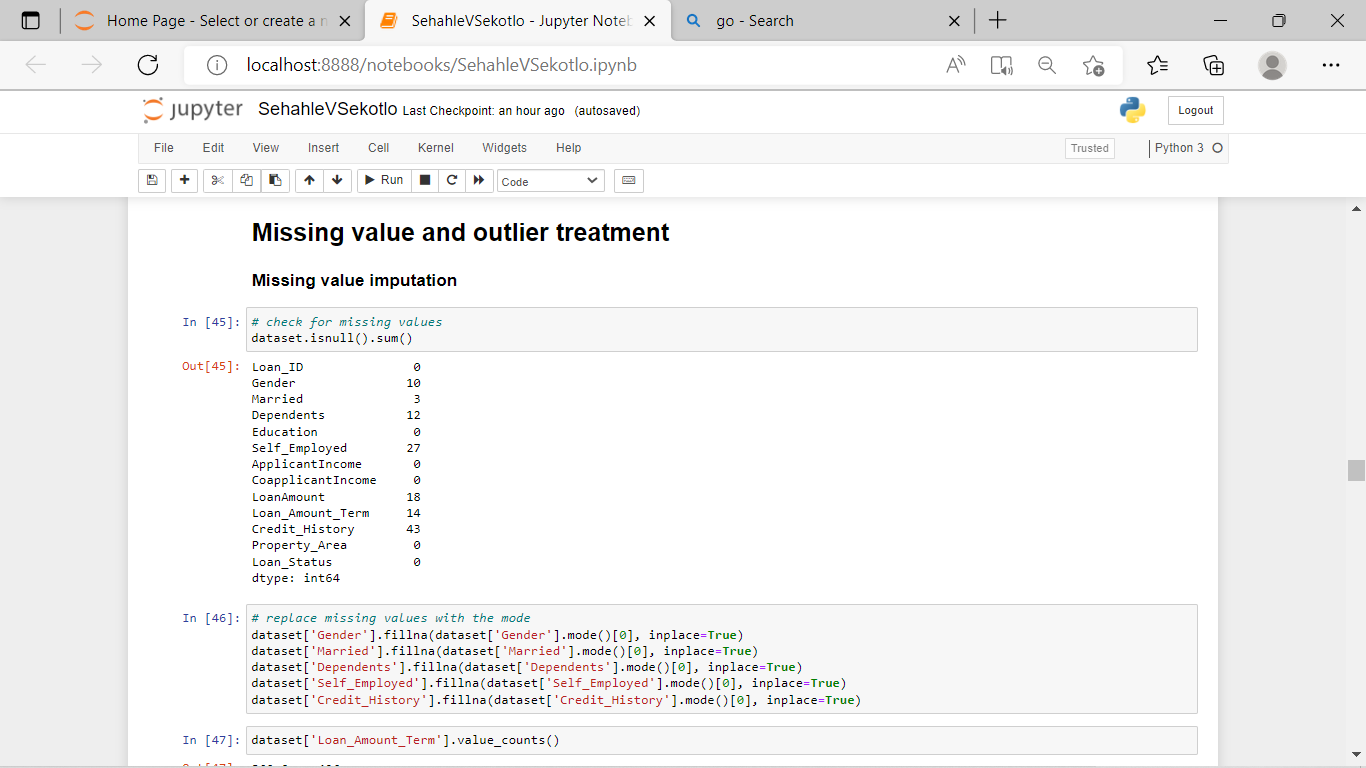
### The figure below is Missing value imputation

There are missing values in Gender, Married, Dependents, Self\_Employed, LoanAmount, Loan\_Amount\_Term and Credit\_History features. We will treat the missing values in all the features one by one.

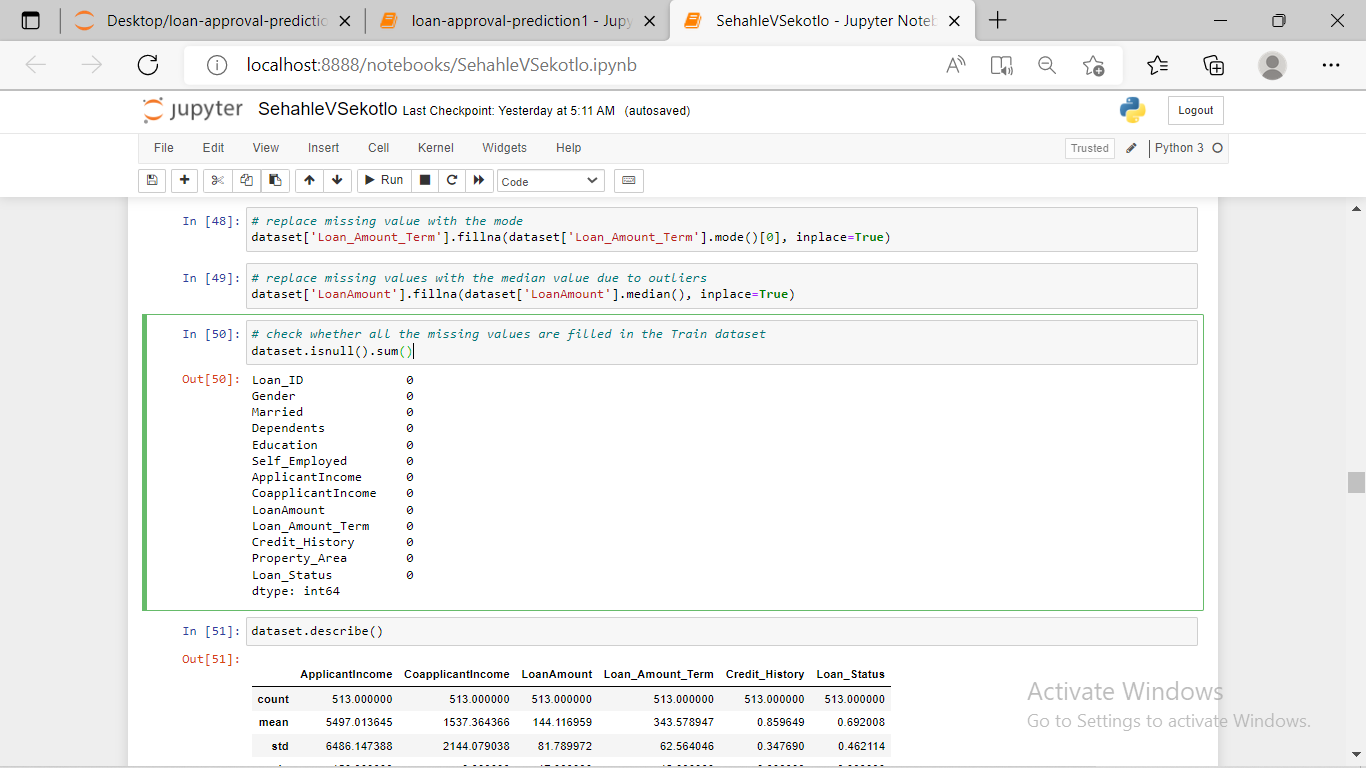
We can consider these methods to fill the missing values:

* For numerical variables: imputation using mean or median
* For categorical variables: imputation using mode

There are very less missing values in Gender, Married, Dependents, Credit\_History and Self\_Employed features so we can fill them using the mode of the features. If an independent variable in our dataset has huge amount of missing data e.g. 80% missing values in it, then we would drop the variable from the dataset

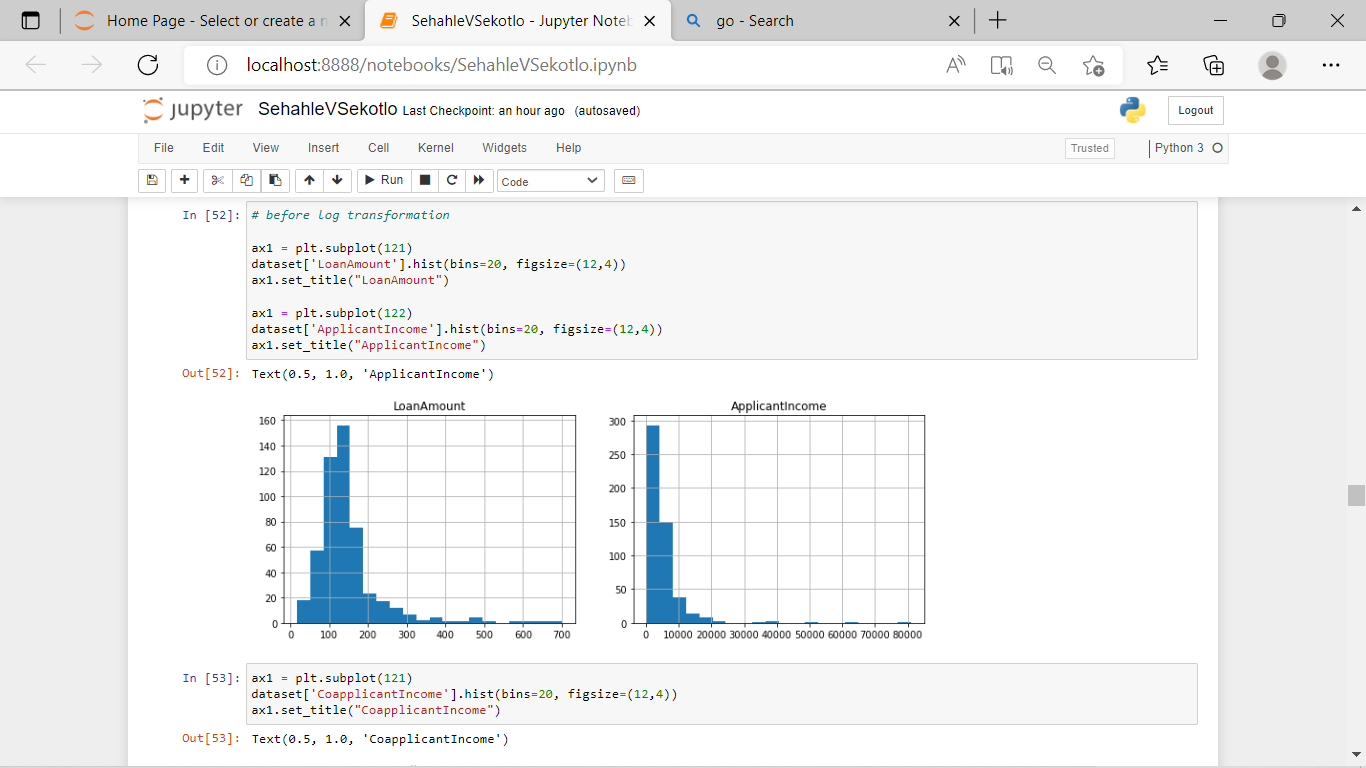


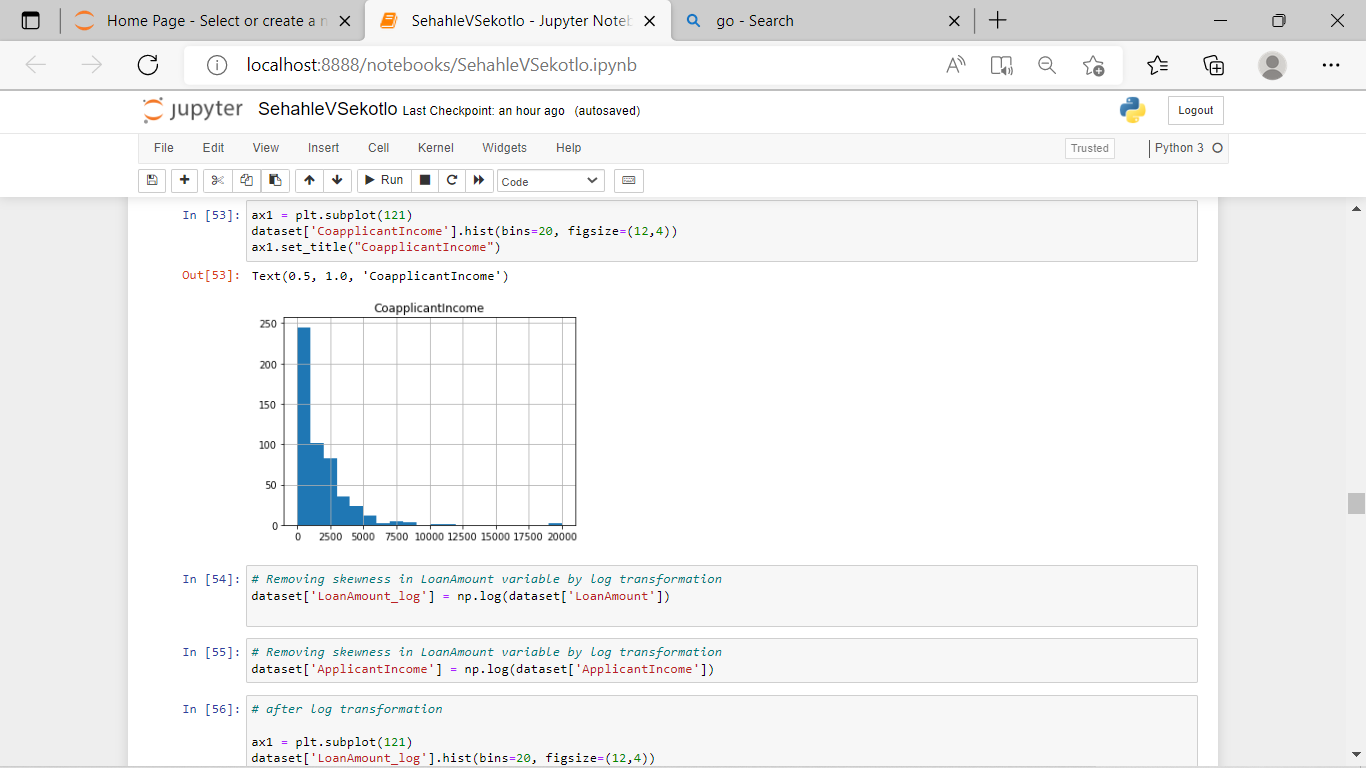
Now we will see the LoanAmount variable. As it is a numerical variable, we can use mean or median to impute the missing values. We will use median to fill the null values as earlier we saw that loan amount have outliers so the mean will not be the proper approach as it is highly affected by the presence of outliers.



### Outlier visualization

As we saw earlier in univariate analysis, LoanAmount contains outliers so we have to treat them as the presence of outliers affects the distribution of the data. Having outliers in the dataset often has a significant effect on the mean and standard deviation and hence affecting the distribution. We must take steps to remove outliers from our data sets below are transformations visualizing loan amount co-applicant income and applicant income outliers.

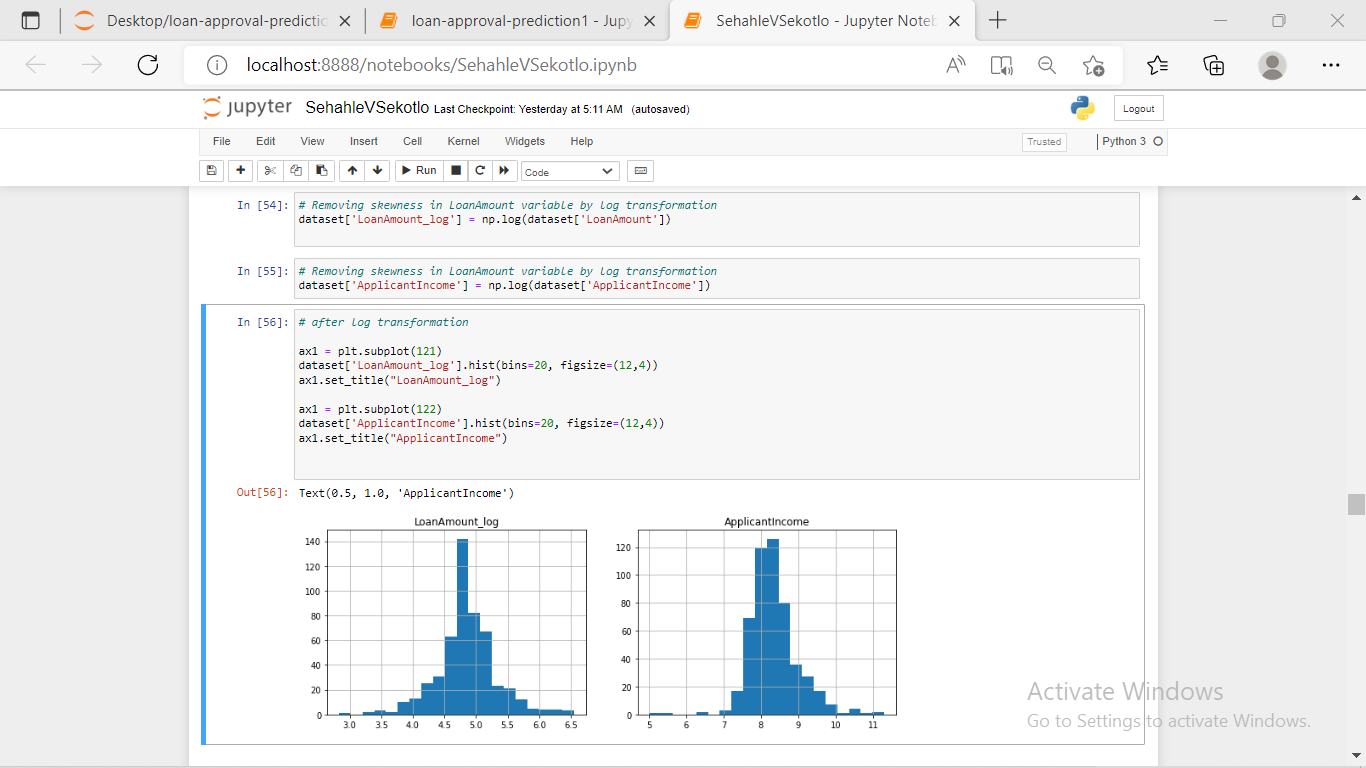




**Outliers treatment**

Due to these outliers bulk of the data in the loan amount is at the left and the right tail is longer. This is called right skewness (or positive skewness). One way to remove the skewness is by doing the log transformation. As we take the log transformation, it does not affect the smaller values much, but reduces the larger values. So, we get a distribution similar to normal distribution.

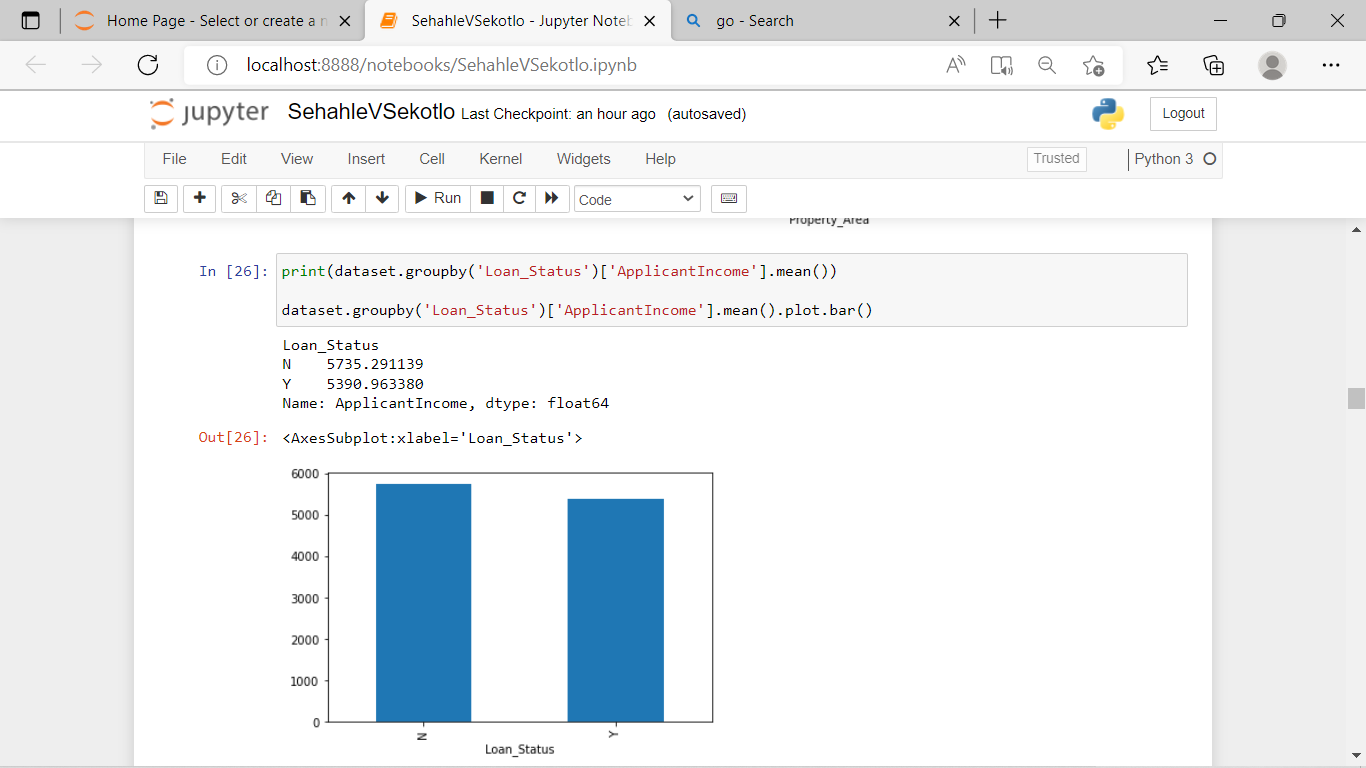
Let’s visualize the effect of log transformation. We will do the similar changes to the test file simultaneously.

**Problem**

Even though, we managed to remove the outliers in Loan amount and Applicant income, we failed to remove outliers in co-applicant income due to the fact that co-applicant income contains some zero values, which result into infinite numbers as we want to calculate mean and median values.

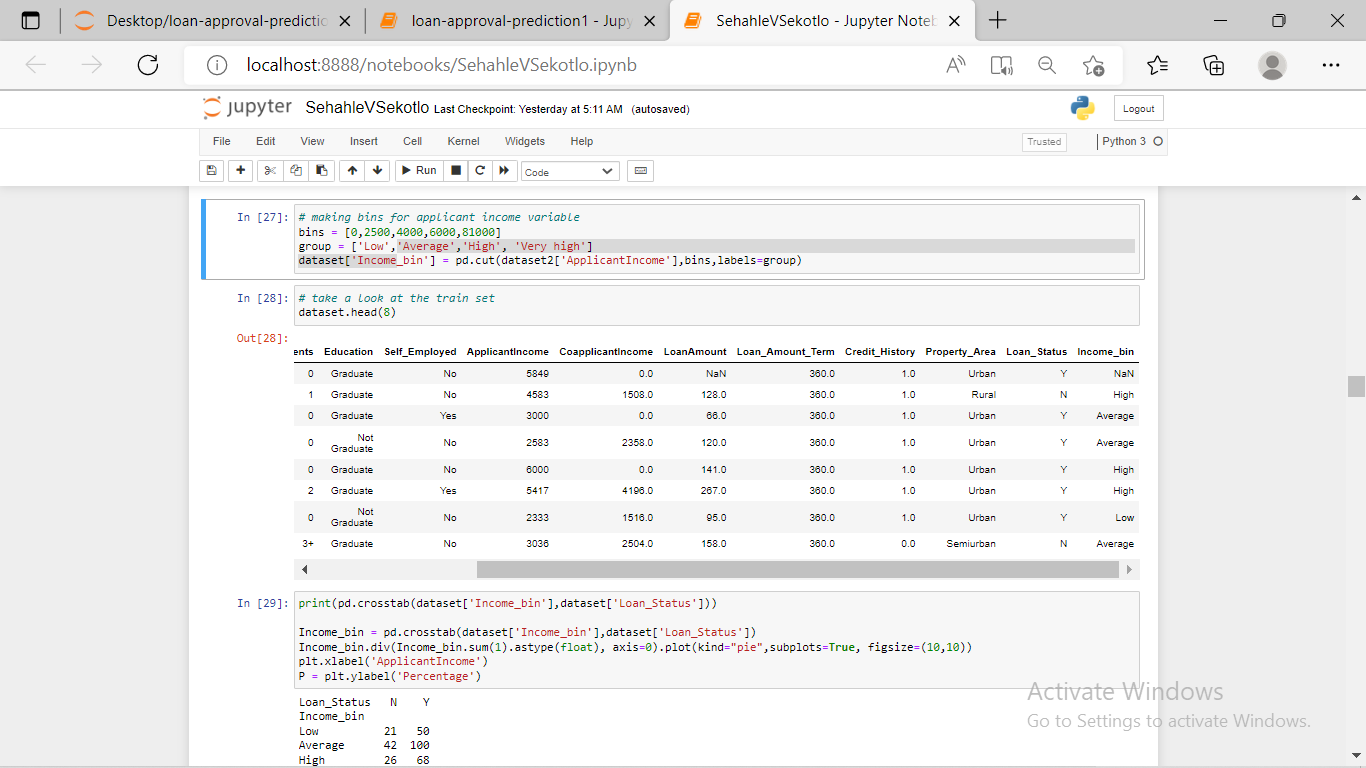
**Feature Extraction**

**Step 1**

****

Referring figure above, we do not see any significant difference in the mean income between those approval and not approved applicant (5384 vs 5446)

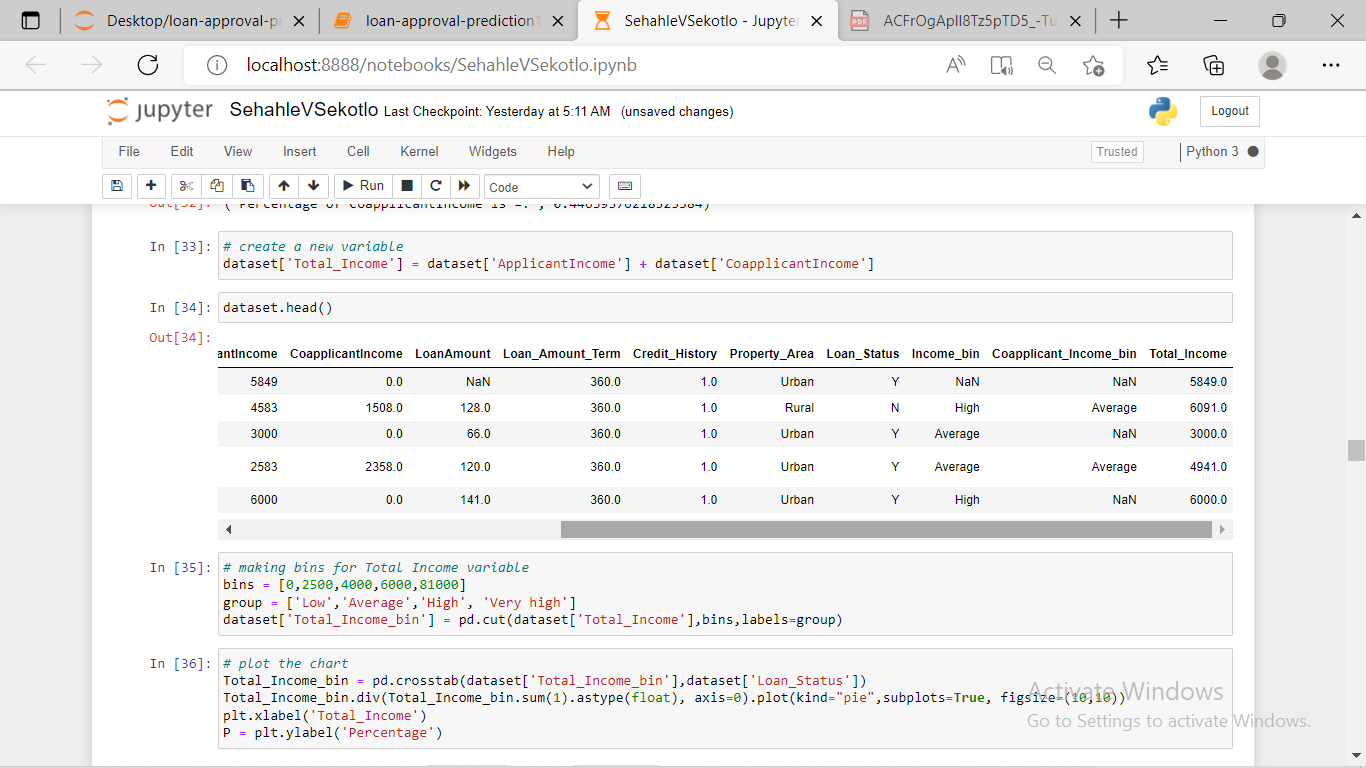
 We make bins for the applicant income variable based on the values in it and analyze the corresponding loan status for each bin.



This bin will help us see whether income level, co-applicant income level and loan amount level has an impact on our target variable.

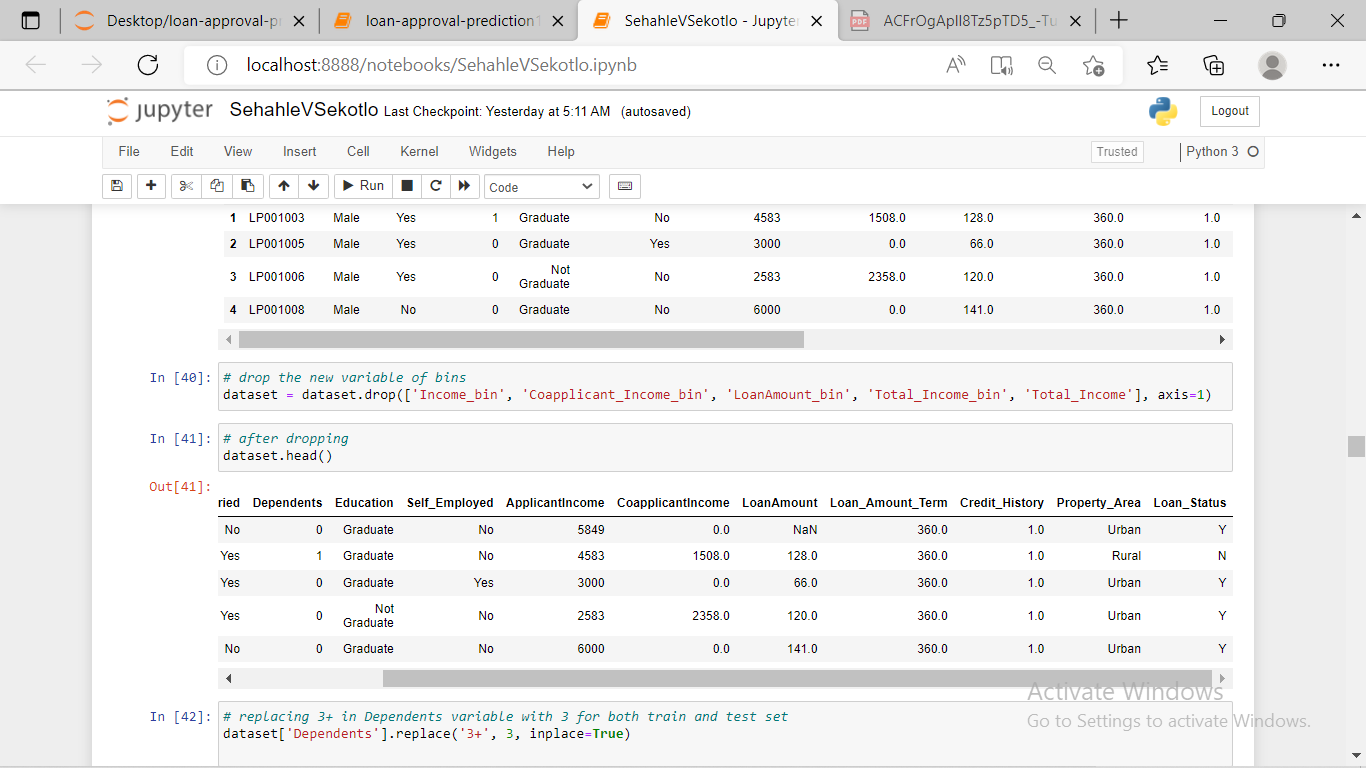
**Step 2**

Create a new variable or feature (Total income) by combining both co-applicant income and applicant income and a bin for this new feature will be created.



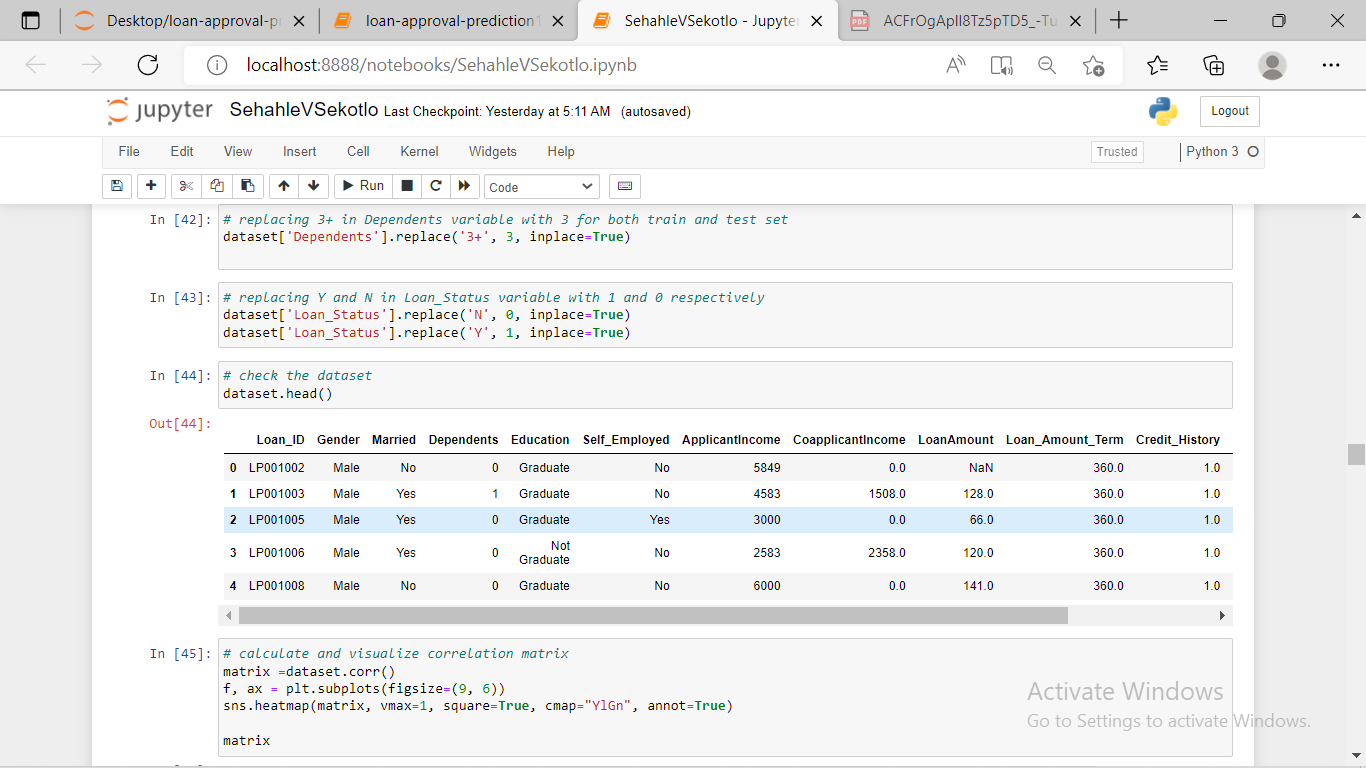
**Step 3**

After using bins, we have to drop them because we cannot use them anymore.



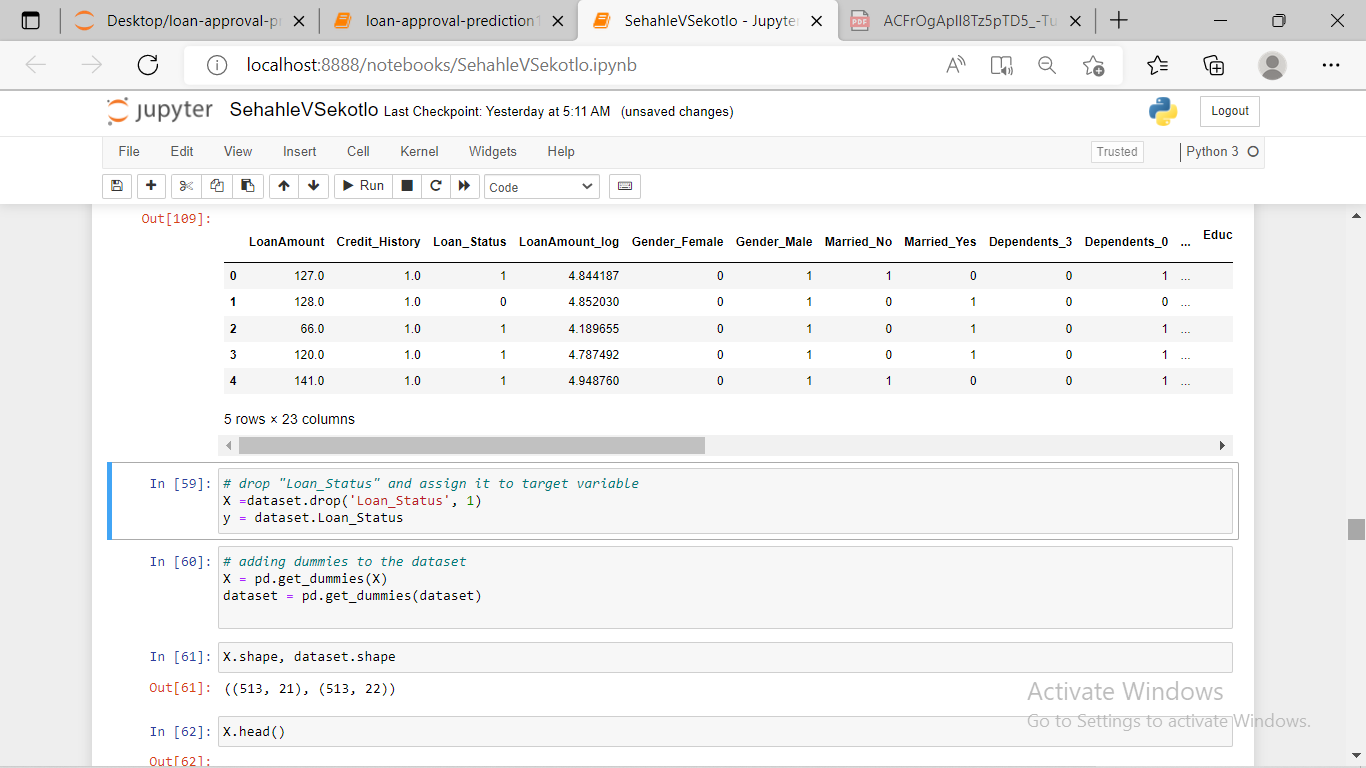
**Step 4**

We changed the 3+ in dependents variable to 3 to make it a numerical variable. We will also convert the target variable’s categories into 0 and 1 so that we can find its correlation with numerical variables. One more reason to do so is few models like logistic regression takes only numeric values as input. We replaced N with 0 and Y with 1.



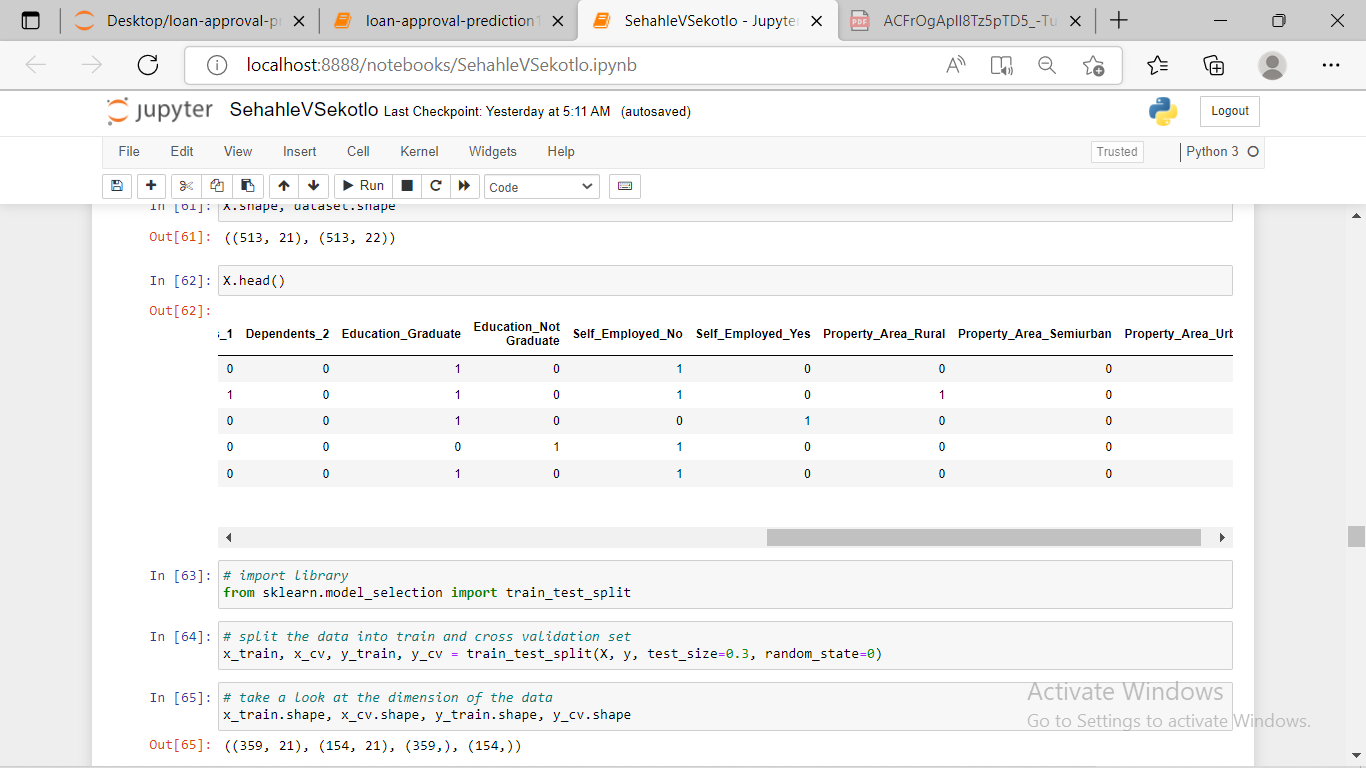
**Step 5**

We dropped the Loan ID variable, as it do not have any effect on the loan status.

****

Step 7

Now we will make dummy variables for the categorical variables. Dummy variable turns categorical variables into a series of 0 and 1, making them lot easier to quantify and compare.



**Step 7**

We divided our dataset into training and validation part. That is to make 30% of our data a testing data against 70% of our data set.

**Model determination**

We determined Logistic Regression as our model because this model is a classification algorithm that is used where the target variable is of categorical nature. Furthermore, the main objective behind this model is to determine the relationship between features and the probability of particular outcome.

On top of that, after we have created other two models/ algorithms, Random Forest and Decision Tree in order to testify that our chosen is appropriate we will see that during model evaluation.

**Model creation**

[2] We have used scikit-learn (sklearn) for making different models which is an open source library for Python. It is one of the most efficient tool, which contains many inbuilt functions that can be used for modeling in Python.

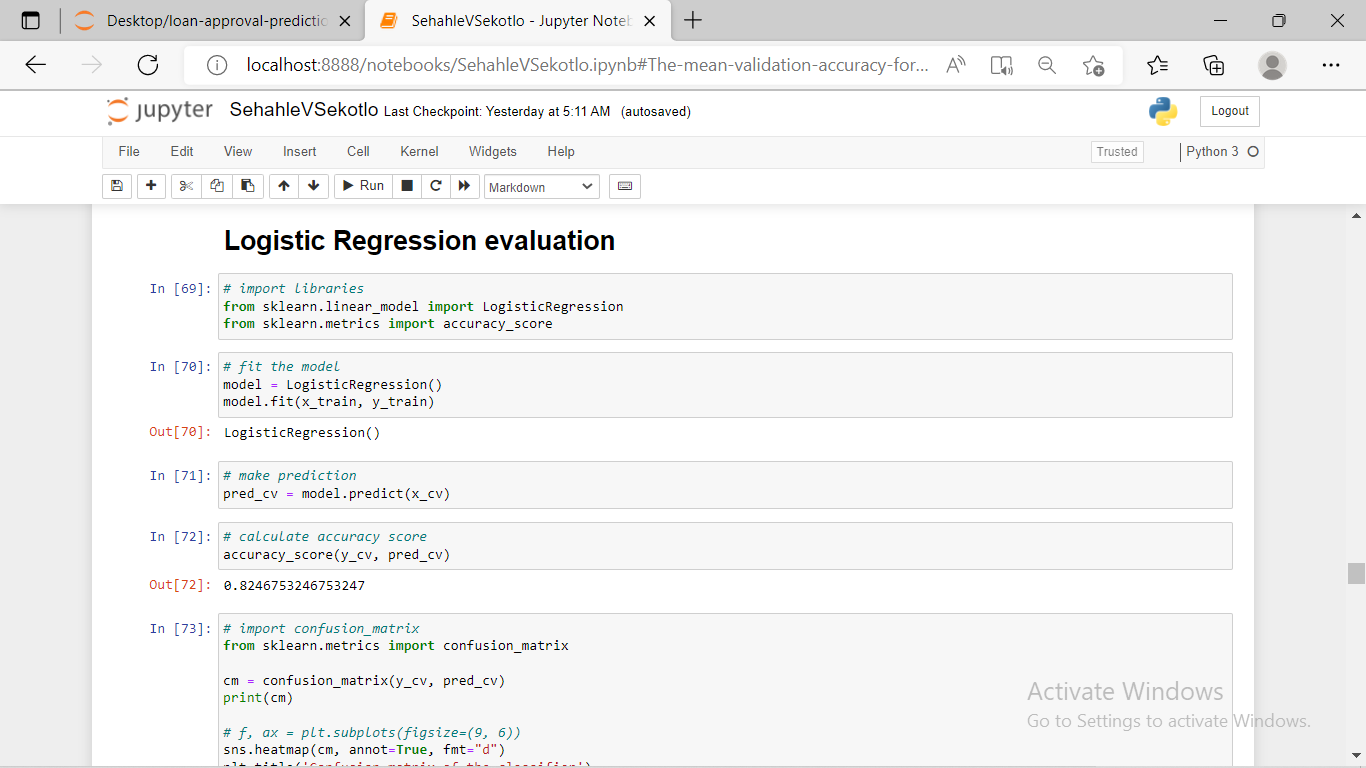
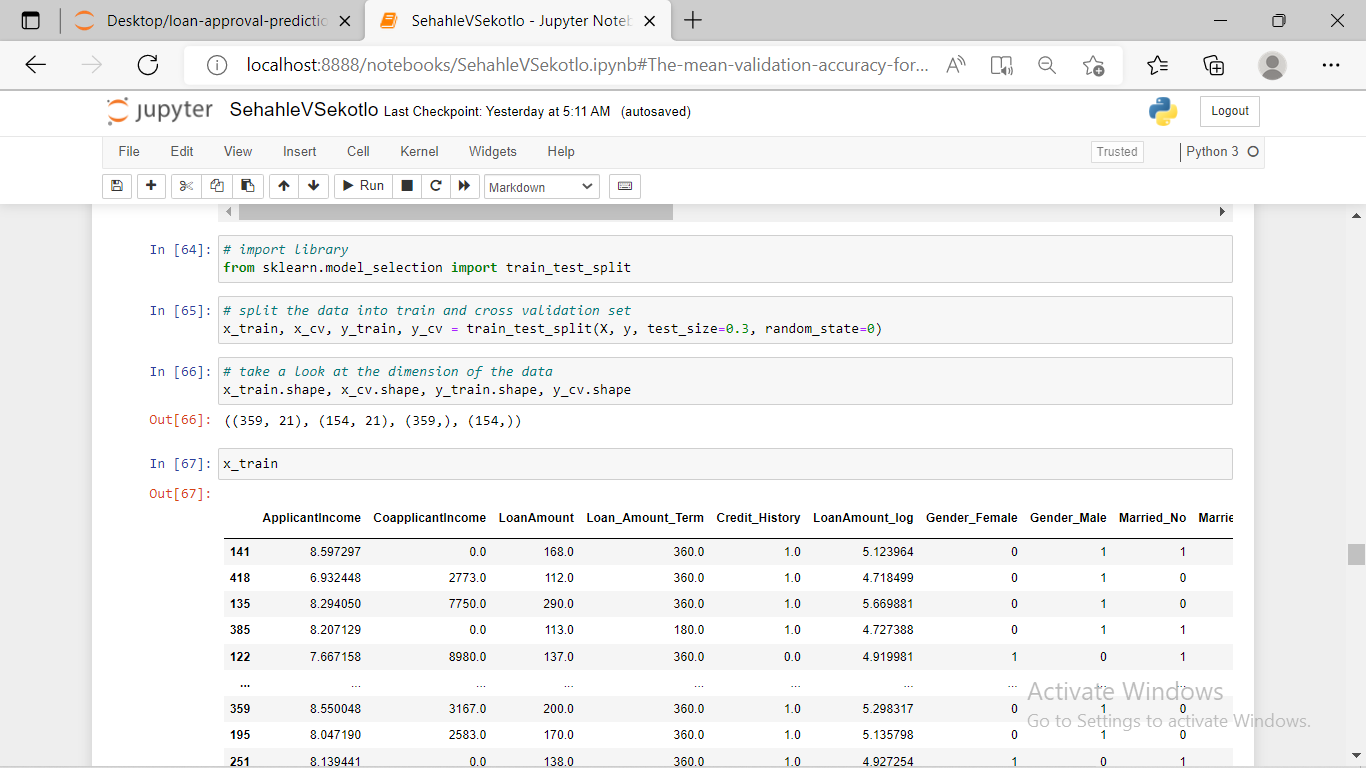
**Models created**

1. **Logistic Regression (preferred mode)**
2. **Decision Tree**
3. **Random Forest**

All steps are in feature extraction session.

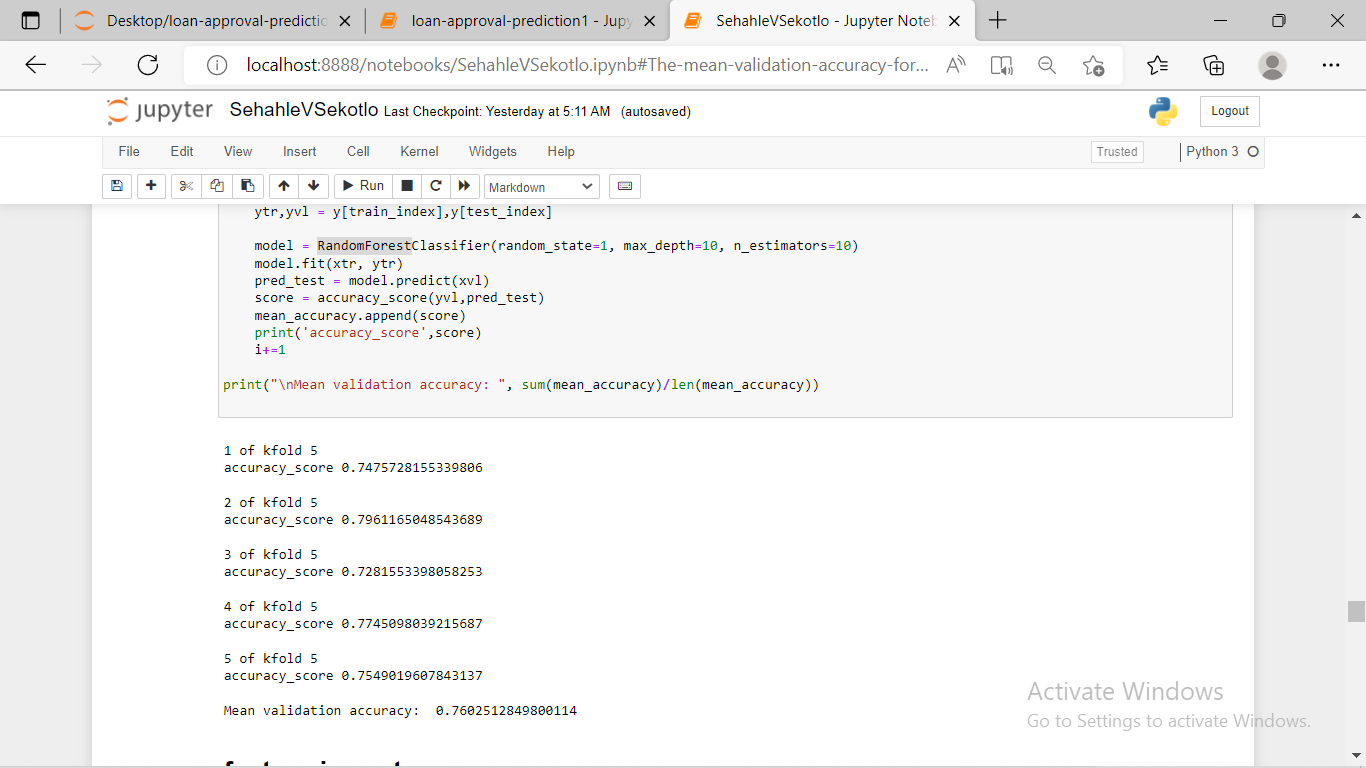
**Model Training**

We have trained our model on training dataset and make predictions for the train dataset. However, we couldn’t validate our predictions, hence we have divided our train dataset into two parts: train and validation as we have said in step 8 of feature extraction. We have trained our model on this train part and using that make predictions for the validation part. In this way, we have validated our predictions as we have the true predictions for the validation part. We have used the train\_test\_split function from sklearn to divide our train dataset.



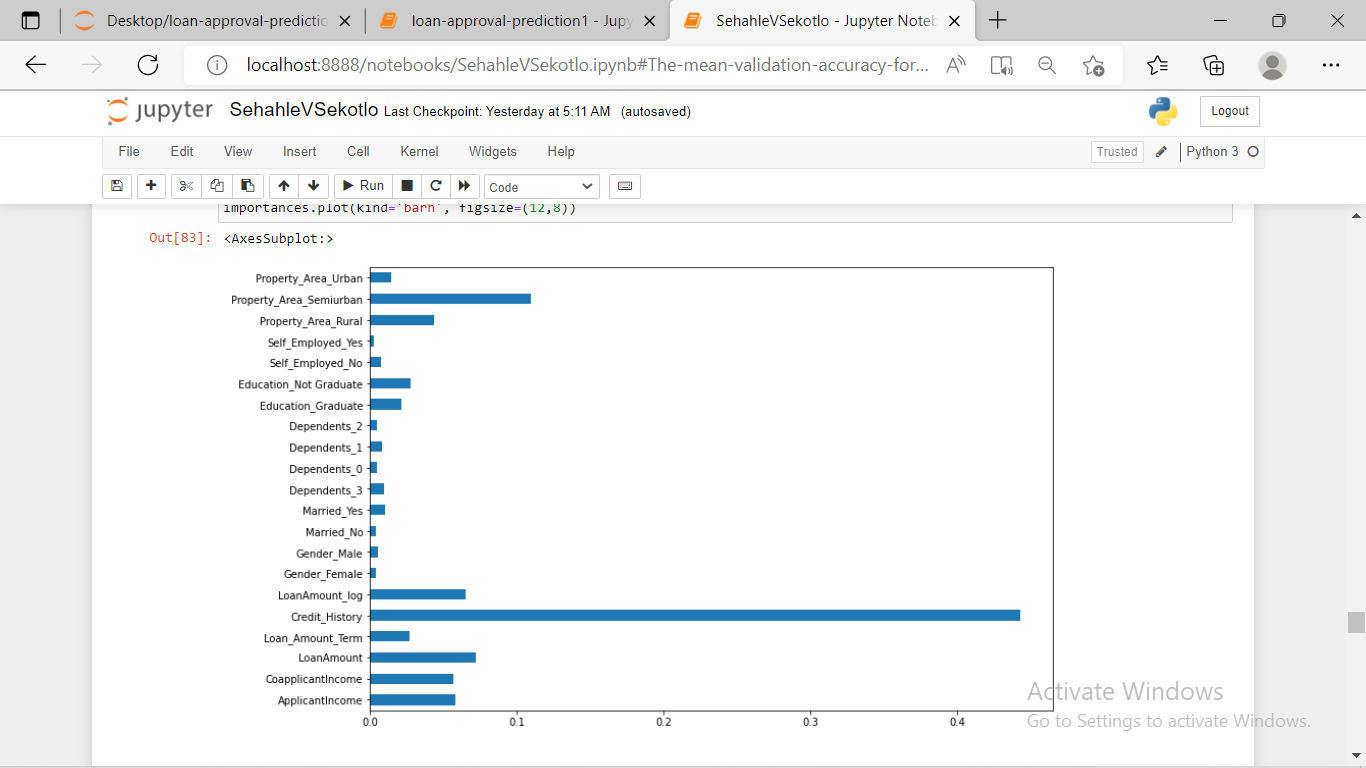
**Model Validation**

[3] We cannot make cross validation, rather we made validation logistic model with stratified 5 folds and make predictions for a dataset we used to train our model. The folds are made by preserving the percentage of samples for each class. In stratified k-fold, each fold contains roughly the same proportions of the different types of class labels. For example logistic regression.



### Feature Importance

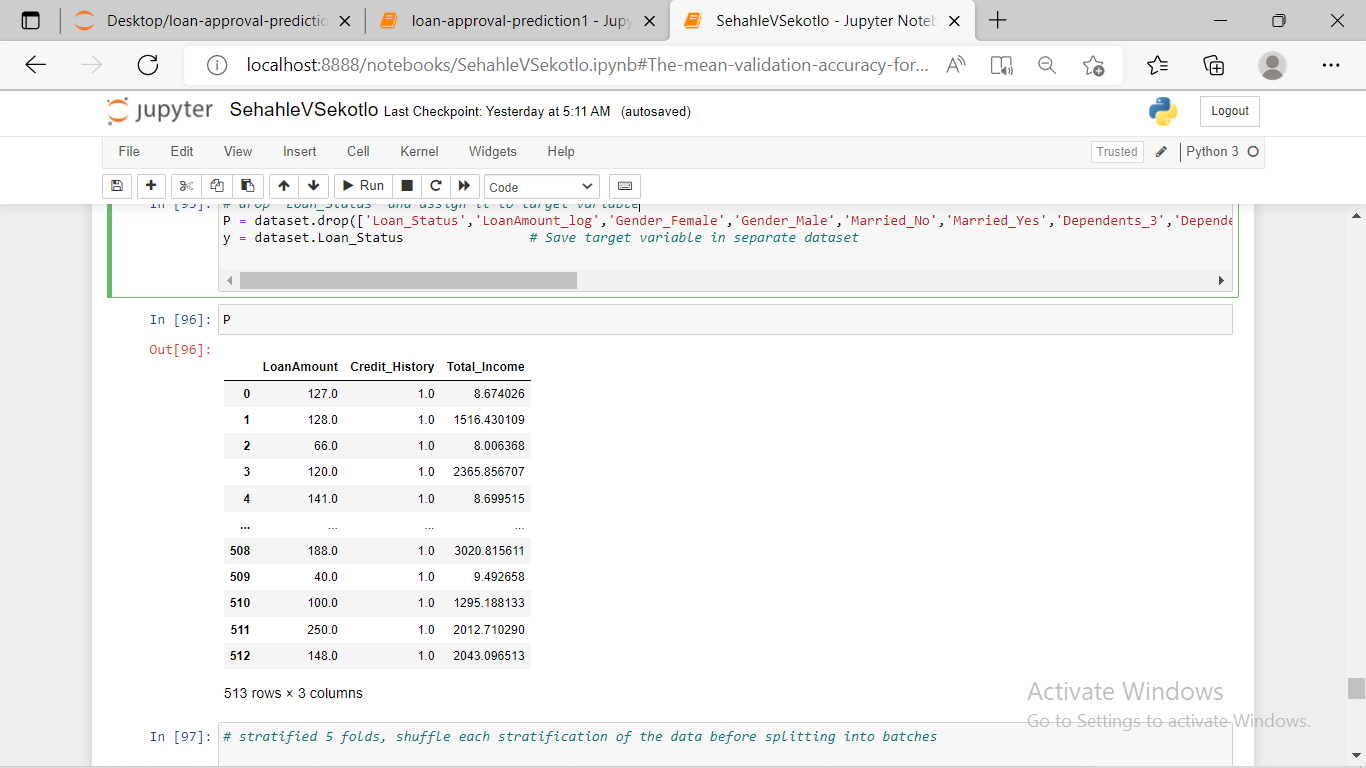
[4] We have find the feature importance, that is features which are most important for this problem. We have used feature\_importances\_ attribute of sklearn to do so. We did this because we wanted to drop unnecessary column to decrease a complexity.



**Finalizing our Model**

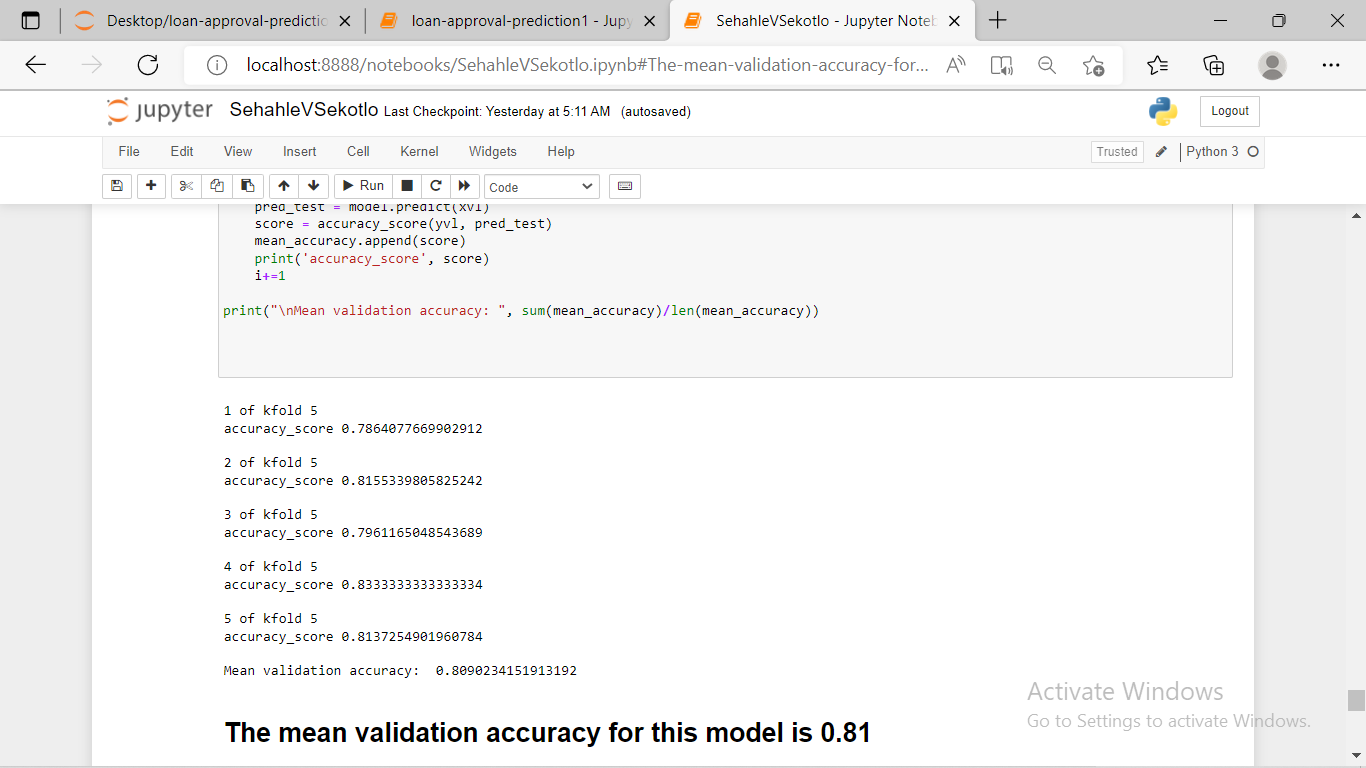
**Step 1**

We have dropped all columns, which we were not going to use on the dashboard to predict an outcome and store our dataset in storage P.



**Step 2**

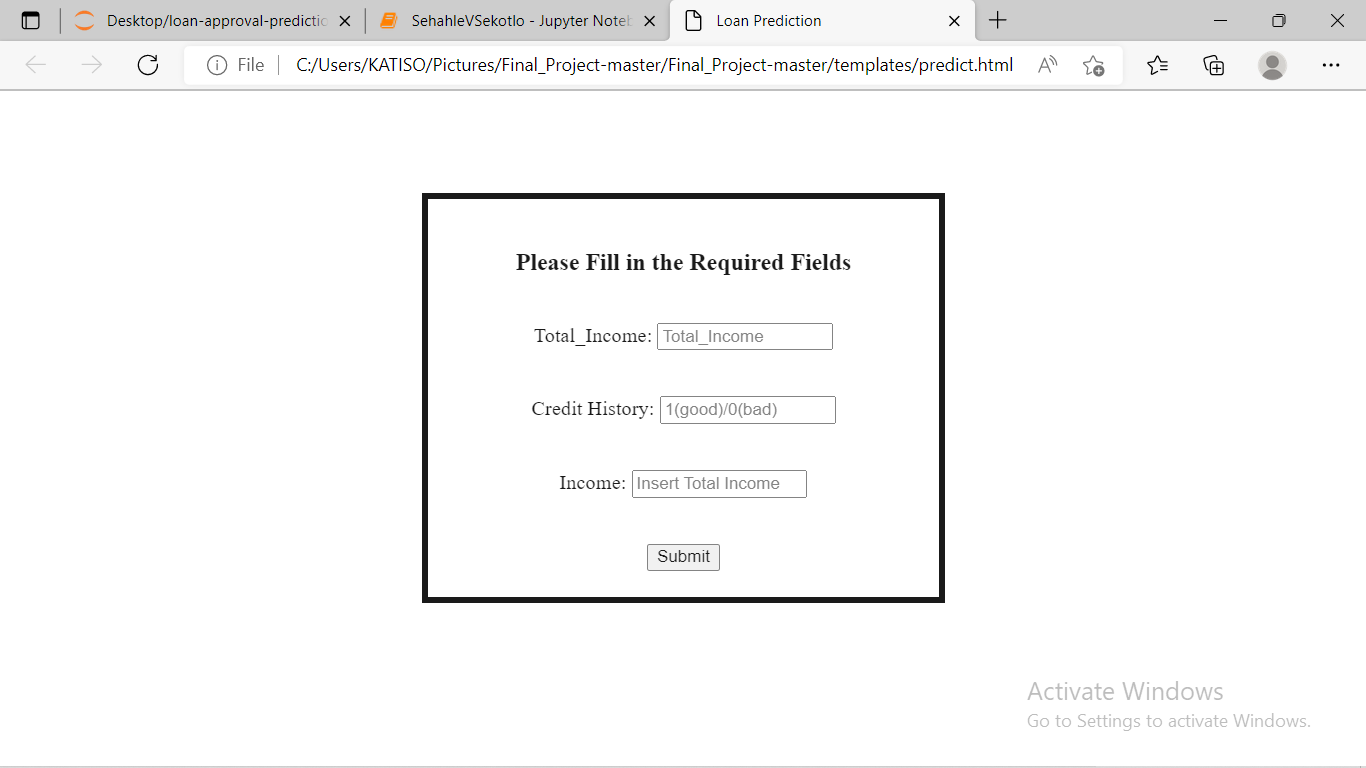
We made evaluation on the model using our new dataset and get 81% mean accuracy



**Dashboard Creation**

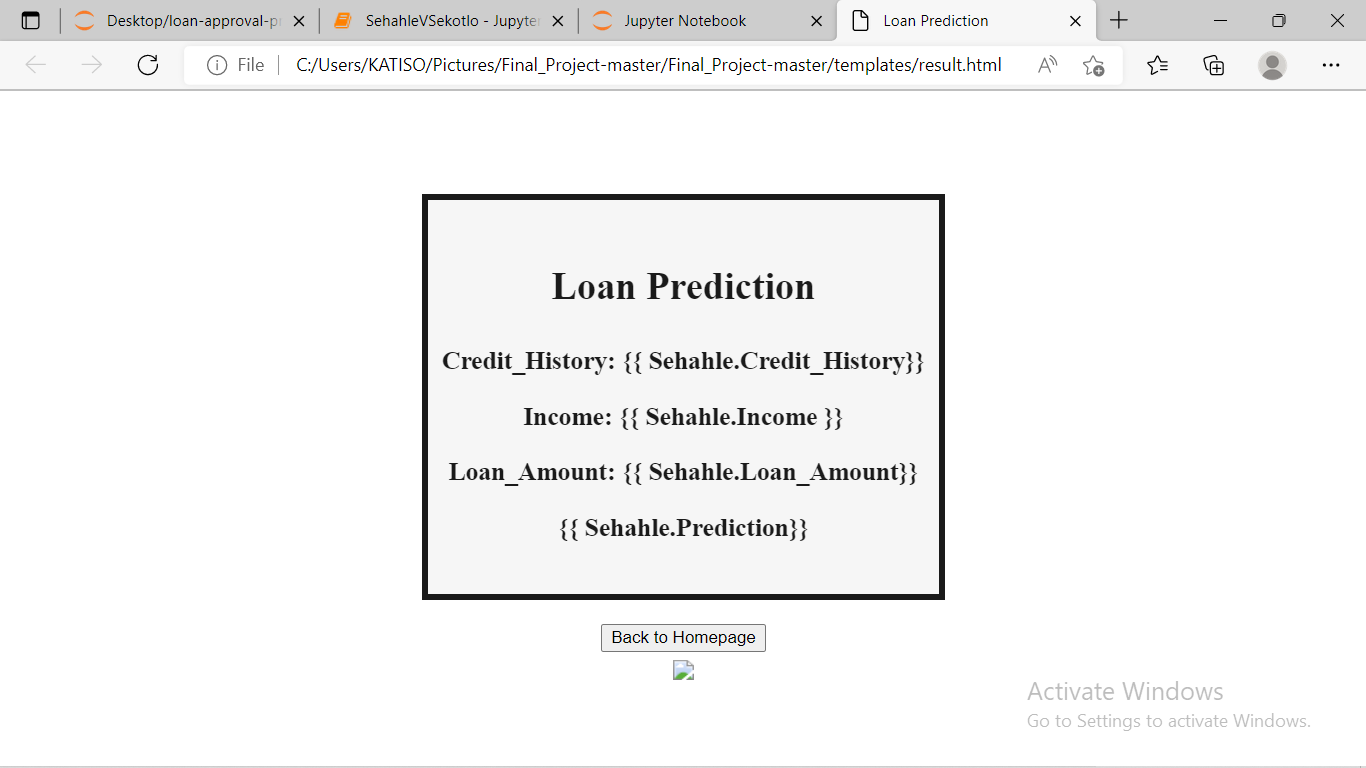
**Prediction**

We have created the dashboard using HTML, a markup language for user interface, our dashboard enable a user to input three parameter (total\_income, Credit\_history and Loan\_Amount), which we have concluded they are enough to make a prediction. As the user click the button submit, the inputted data should be delivered into Jupyter, look for specified features, and make prediction.



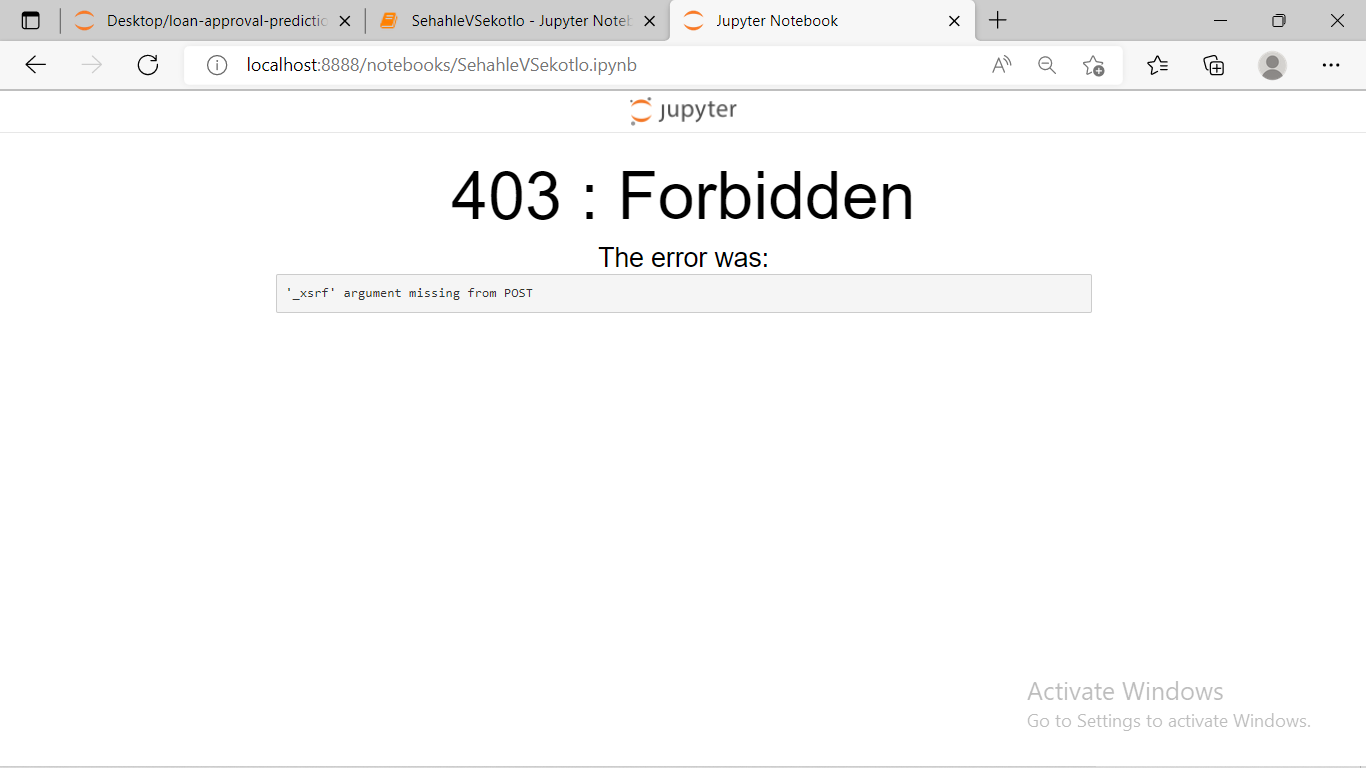
**Results**

After prediction results had to be displayed back to the user on the form format like this;



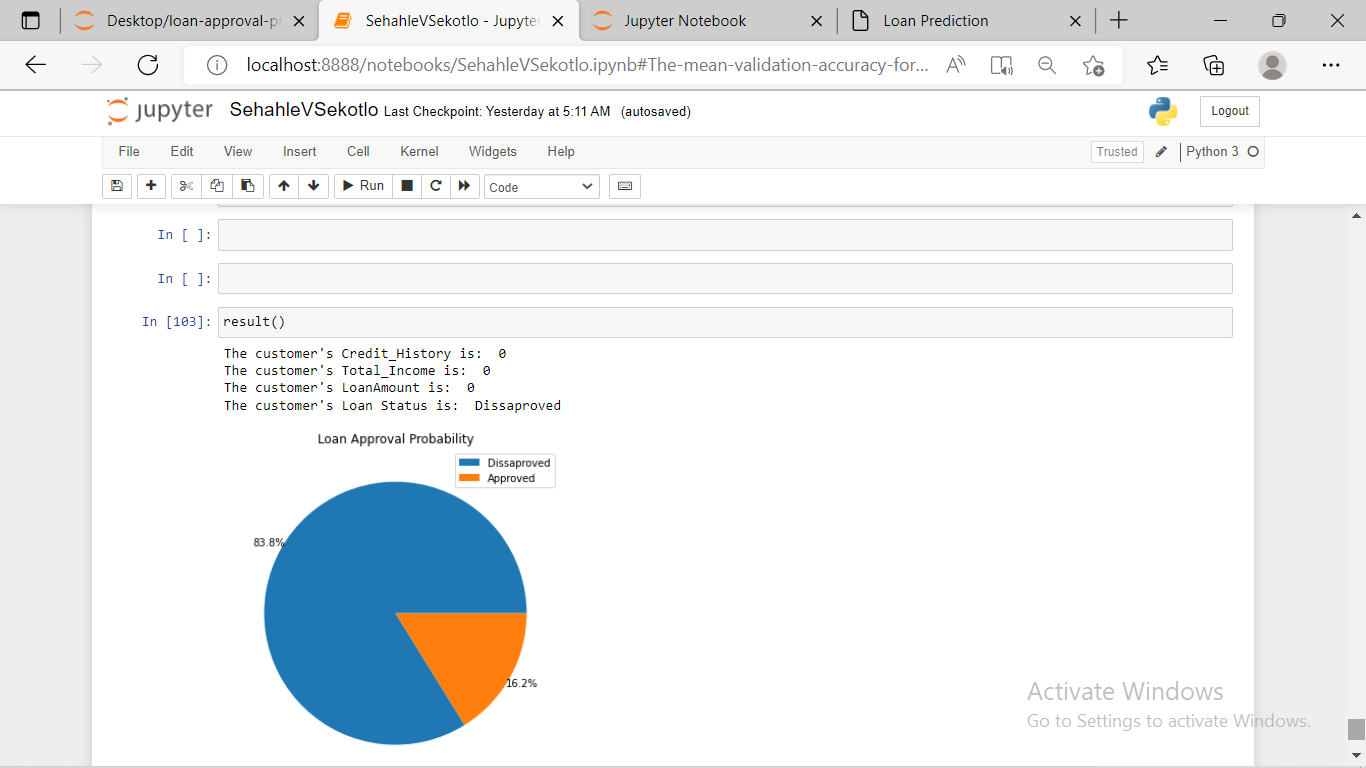
**Problem**

Unfortunately, we had a problem during the process, when trying to submit inputted data into Jupyter we had an error, which we have tried to overcome, but we failed. Below is the error we have discovered:



**Solution (Alternative)**

We decided to perform all processes with python in Jupyter by declaring varibles in Jupyter and allow user to input data within the Jupiter and get the results right in there.



**Conclusion**

After trying and testing three different algorithms, the best accuracy on the public leaderboard is achieved by Logistic Regression (0.81), followed by Random Forest (0.760) and Decision Tree performed the worst (0.705). While new features created via feature engineering helped in predicting the target variable, it did not improve the overall model accuracy much. Overall, a logistic regression classifier provides the best result in terms of accuracy for the given dataset, without any feature engineering needed. Because of its simplicity and the fact that it can be implemented relatively easy and quick, Logistic Regression is often a good baseline that data scientists can use to measure the performance of other more complex algorithms.

# References

|  |  |
| --- | --- |
| [1] | J. Achavi, "towarddatascience," 2 09 2018. [Online]. Available: https://towardsdatascience.com/data-types-in-statistics-347e152e8bee. [Accessed 26 05 2022]. |
| [2] | J. donald, "machinelearning-blog," machinelearning-blog, 8 12 2018. [Online]. Available: https://machinelearning-blog.com/2018/04/23/logistic-regression-101/. [Accessed 26 05 2022]. |
| [3] | Chen, "A,datum corparartion," analyticsvidhya, 25 11 2018. [Online]. Available: https://www.analyticsvidhya.com/blog/2015/08/introduction-ensemble-learning/. [Accessed 26 05 2022]. |
| [4] | jacky, "analyticsvidhya," analyticsvidhya, 23 10 2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2015/09/questions-ensemble-modeling/. [Accessed 26 05 2022]. |