# LOAN APPLICATION STATUS PREDICTION

# Introduction

The two most critical questions in the lending industry are: 1) How risky is the borrower? 2) Given the borrower’s risk, should we lend him/her? The answer to the first question determines the interest rate the borrower would have. Interest rate measures among other things (such as time value of money) the riskiness of the borrower, i.e. the riskier the borrower, the higher the interest rate. With interest rate in mind, we can then determine if the borrower is eligible for the loan.

Investors (lenders) provide loans to borrowers in exchange for the promise of repayment with interest. That means the lender only makes profit (interest) if the borrower pays off the loan. However, if he/she doesn’t repay the loan, then the lender loses money.

**In this blog-post, I will go through the entire interaction of making a machine learning model on the celebrated Loan\_Application\_dataset, which is utilized by numerous individuals everywhere on the world**.

## DATA DESCRIPTION:

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: Unique Loan ID

Gender: Male/ Female

Married: Applicant married (Y/N)

Dependents: Number of dependents

Education: Applicant Education (Graduate/ Under Graduate)

Self\_Employed: Self-employed (Y/N)

ApplicantIncome: Applicant income

CoapplicantIncome: Coapplicant income

LoanAmount: Loan amount in thousands of dollars

Loan\_Amount\_Term: Term of loan in months

Credit\_History: credit history meets guidelines yes or no

Property\_Area: Urban/ Semi Urban/ Rural

**Dependent Variable (Target Variable):**

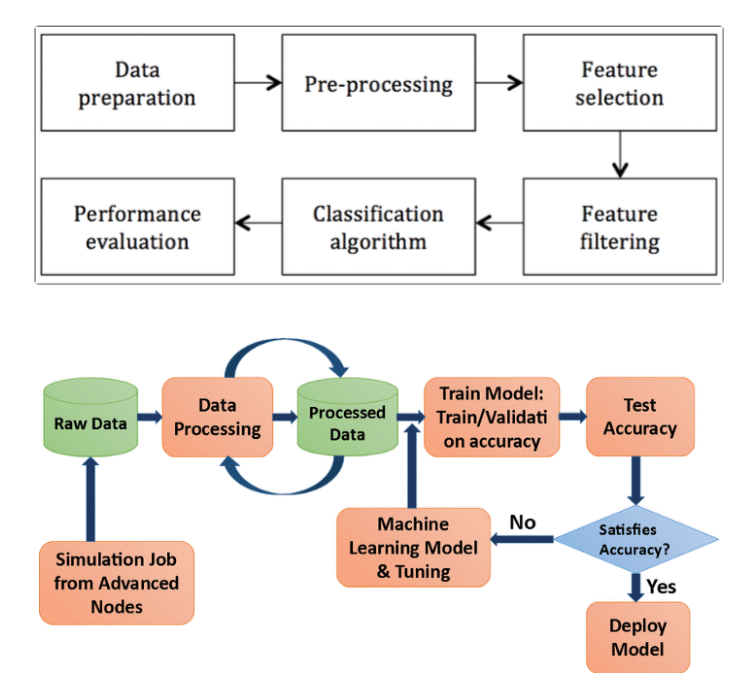
* Loan\_Status : Loan approved (Y/N) this is the target variable

PROBLEM STATEMENT:

We 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.

So, from the problem statement and the Dataset we can understand that it is a "Classification problem". So we will be using some Classification algorithms to make our model and then use GRIDSEARCHCV for hyper parameter tuning and save the predicted model using pkl.

## Work Flow:



## Exploratory data analysis:

We’ll be using seaborn for visualisation and pandas for data manipulation.

Importing necessary libraries and loading the data:

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline

import warnings

warnings.filterwarnings('ignore')

## Loading the Dataset:

In [55]:

df=pd.read\_csv ('loan\_data.csv')

In [56]:

df.columns

Out[56]:

Index (['Loan\_ID', 'Gender', 'Married', 'Dependents', 'Education',

'Self\_Employed', 'ApplicantIncome', 'CoapplicantIncome', 'LoanAmount',

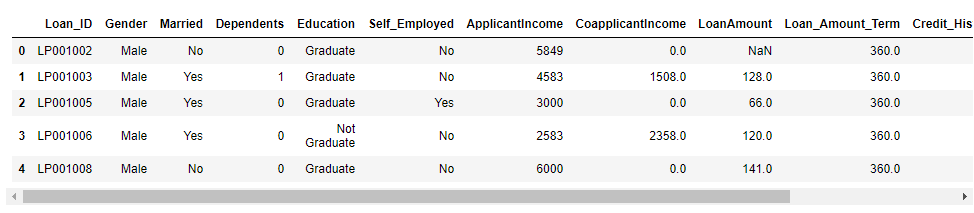
'Loan\_Amount\_Term', 'Credit\_History', 'Property\_Area', 'Loan\_Status'],

dtype='object')

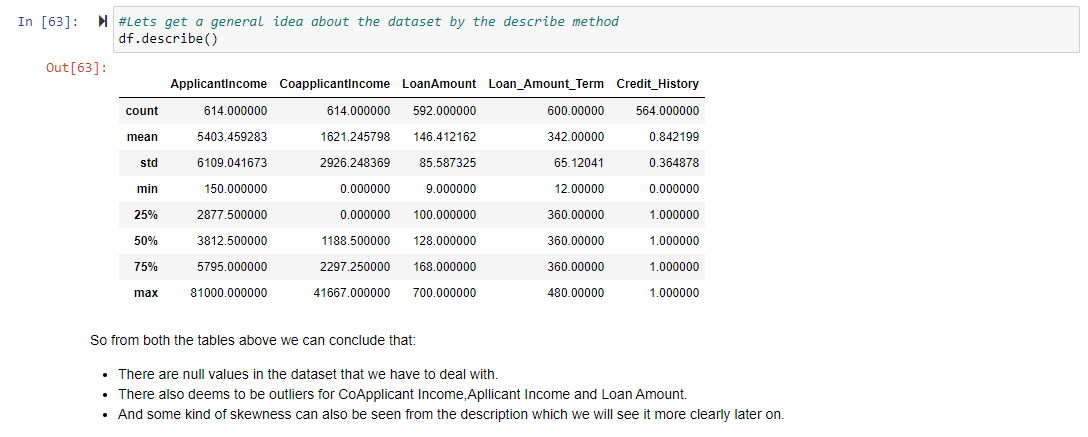
We have 12 independent variables and 1 target variable, i.e. Loan\_Status in the training dataset.

## We can look at few top rows using the head function->





We can see that there’s some missing data, we can further explore this using the pandas describe function:

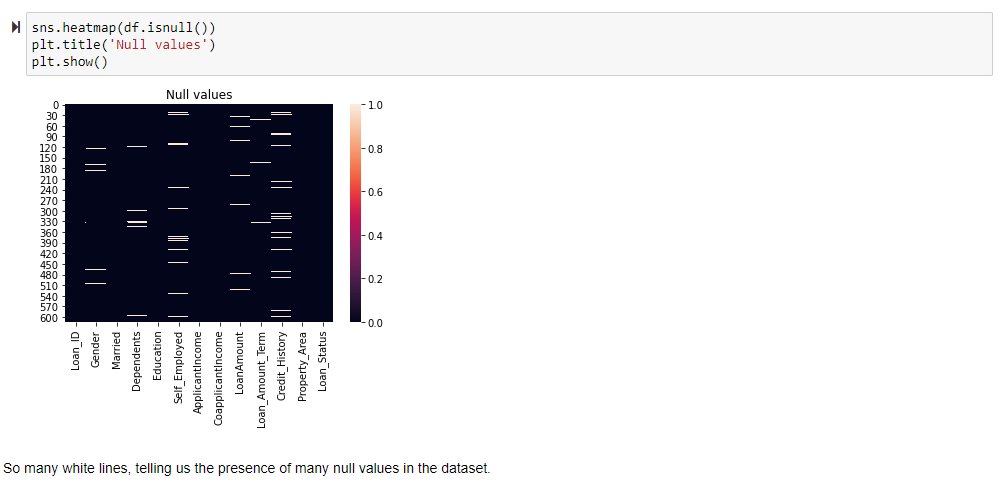
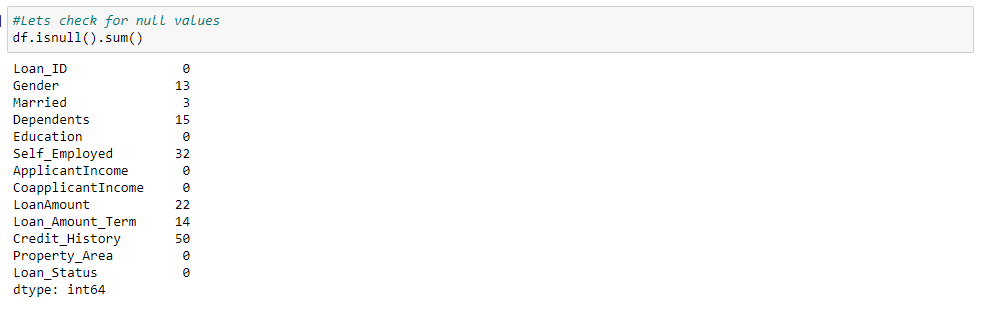


## Check the Data Dimension of the Dataset:

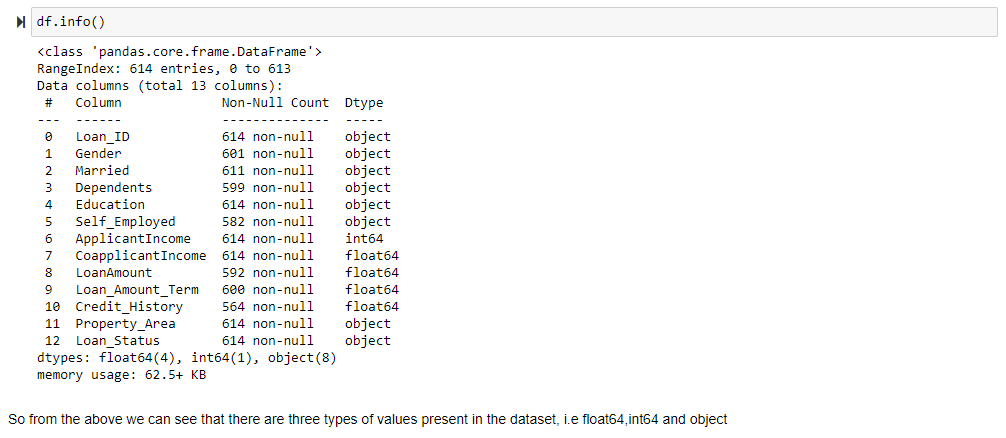
## 



## Now let’s check for Null Values:

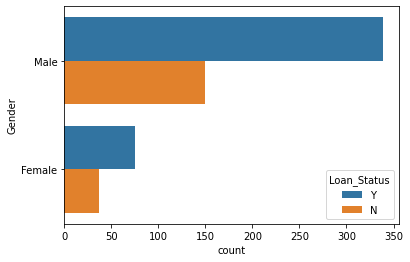


## Now we will get some general information about the Dataset using .info:



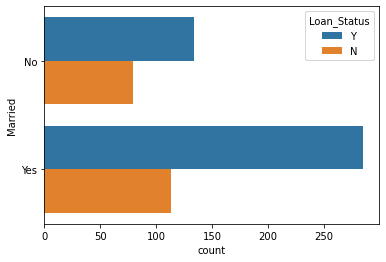
## Now let’s do some Univariate, bivariate and Multivariate on our dataset to understand our data better and find some correlation along the way:

Let’s visualise through seaborn about the relation between Gender & Loan-Status.



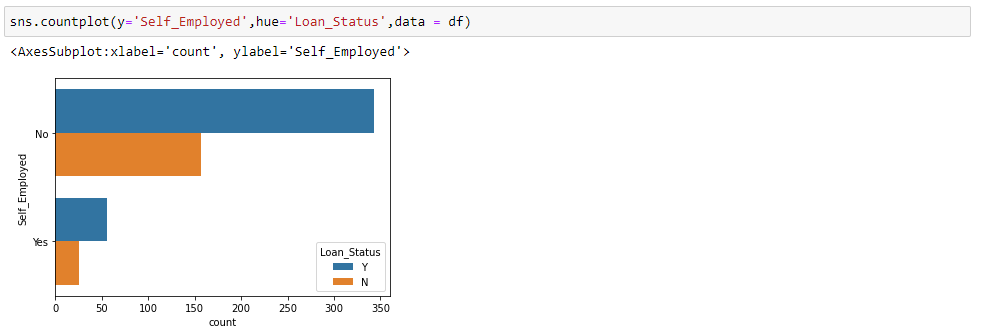
From the above we can conclude that more males are taking loans than that of females.

Let’s visualise through seaborn about the relation between Married & Loan-Status.



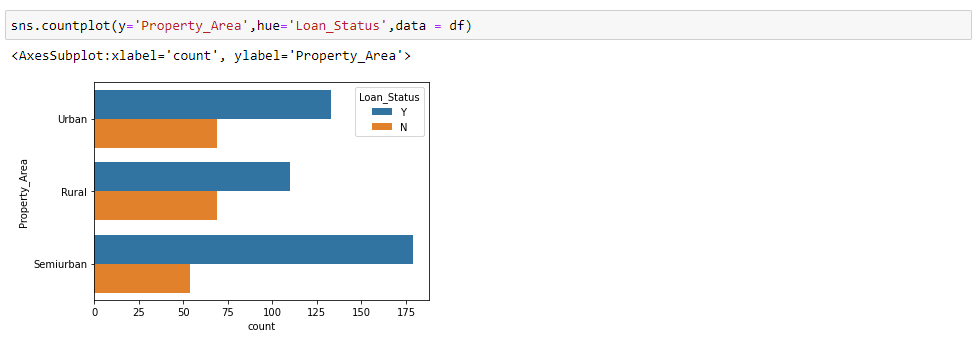
From the above we can conclude that Married People collect more loans than unmarried.

Let’s visualise through seaborn about the relation between Self-Employed & Loan-Status.



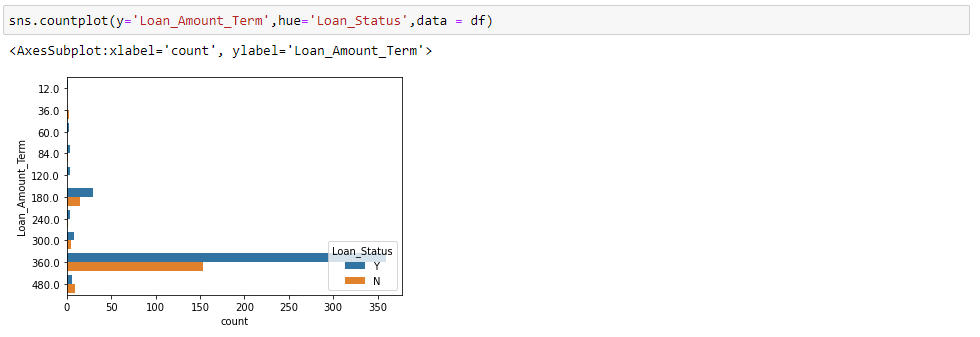
From the above we can see that self-employed people take less loans as compared to salary earners.

Let’s visualise through seaborn about the relation between Property\_Area & Loan-Status.



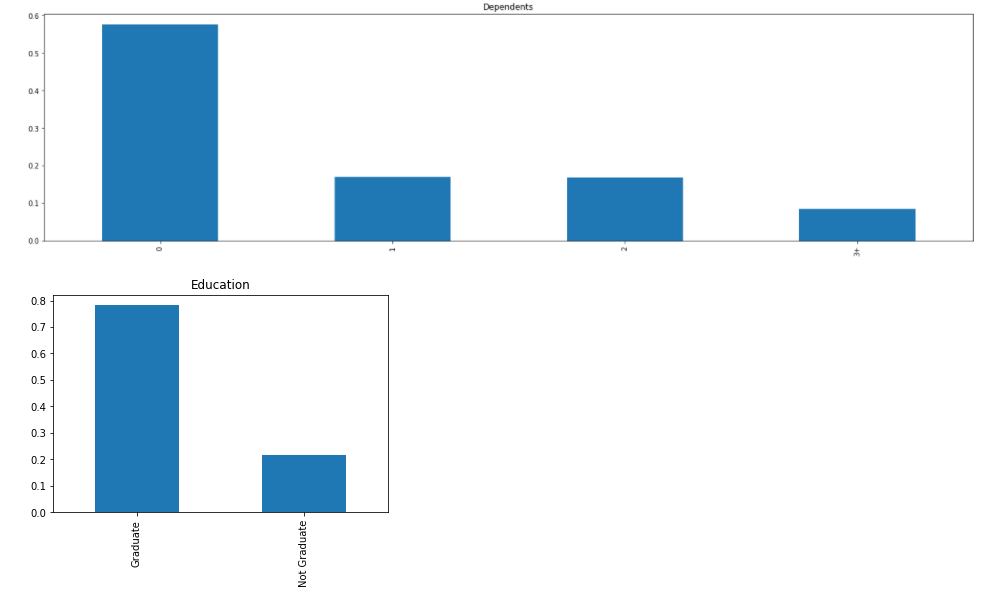
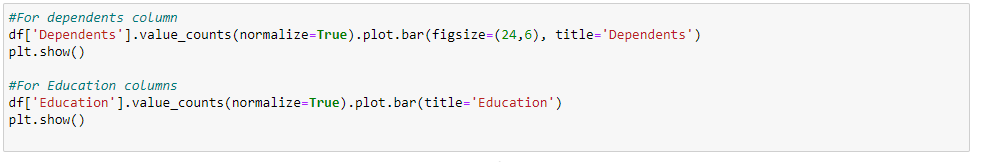
Semi-metropolitan gets more loans, trailed by Urban and afterward rural. This seems Logical.

Let’s visualise through seaborn about the relation between Loan\_Amount\_Term & Loan-Status.



An amazingly high number of them go for a 360 cyclic credit term. That is repaid inside a year.

## LET’S SEE SOME VISUALIZATION FOR OUR ORDINAL VARIABLES:

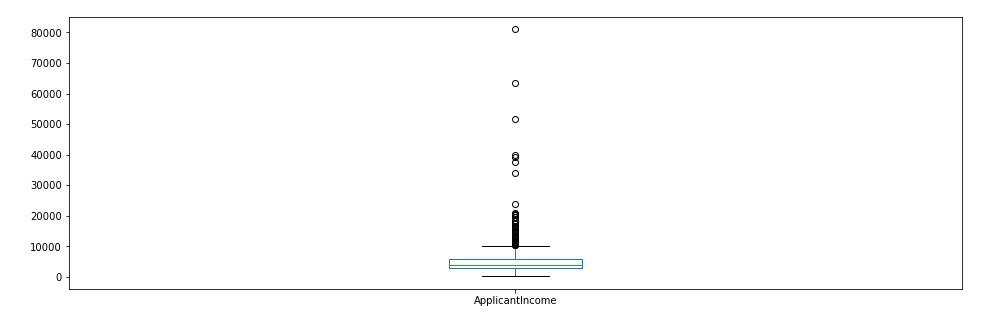
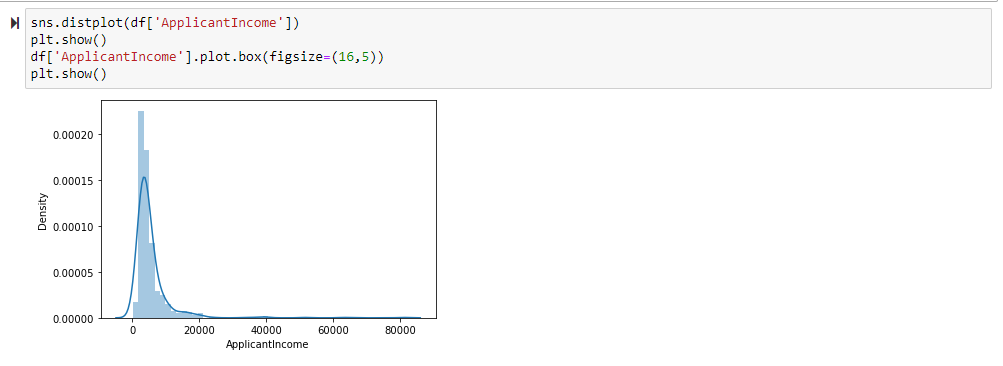


The following deductions can be made using the above bar plots:

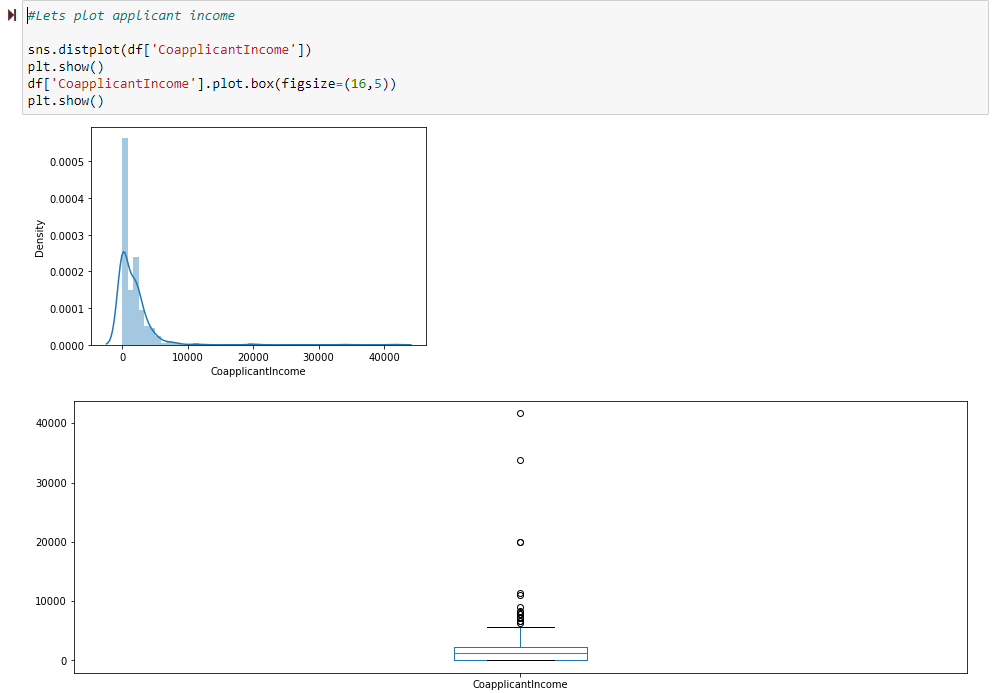
* Most of them do not have any dependents.
* Around 80% of the applicants are Graduate.

## LET’S SEE SOME VISUALISATION FOR OUR NUMERICAL VARIABLES:

It would be interesting to study the distribution of the numerical variables mainly the Applicant income and the loan amount. To do this we’ll use seaborn for visualization.

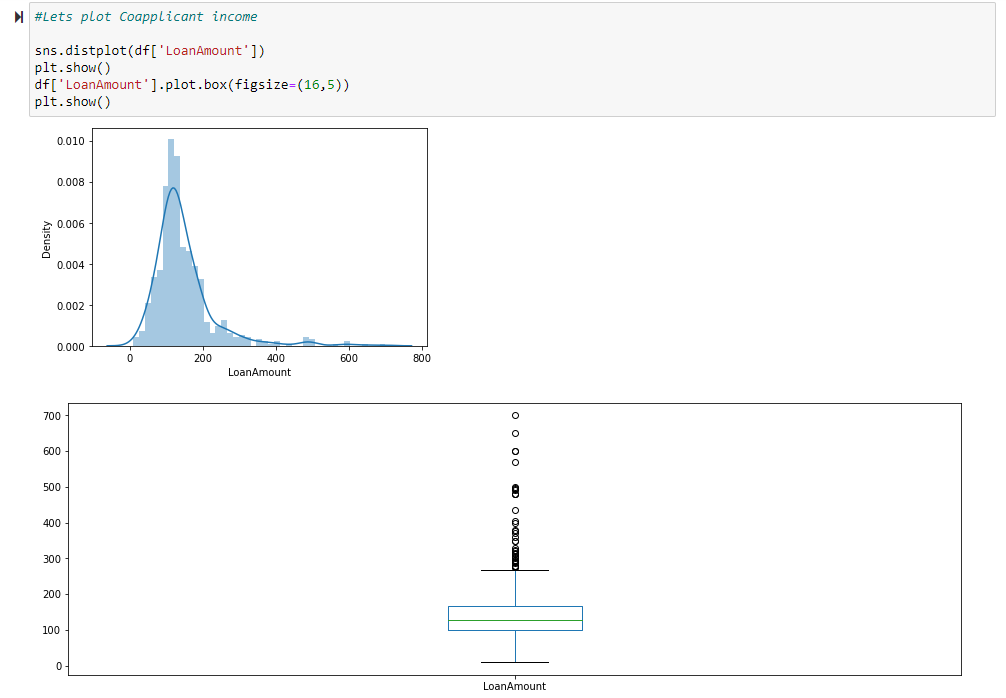


From the above graphs we can infer that most of the Data in the Distribution are left skewed which means it is not normally distributed. The boxplot confirms that there are some outliers present in the Dataset. Maybe due to so much disparity in income levels.



We see a similar distribution as that of the applicant income. There are lot of outliers in the applicants’ income as it is not normally distributed.

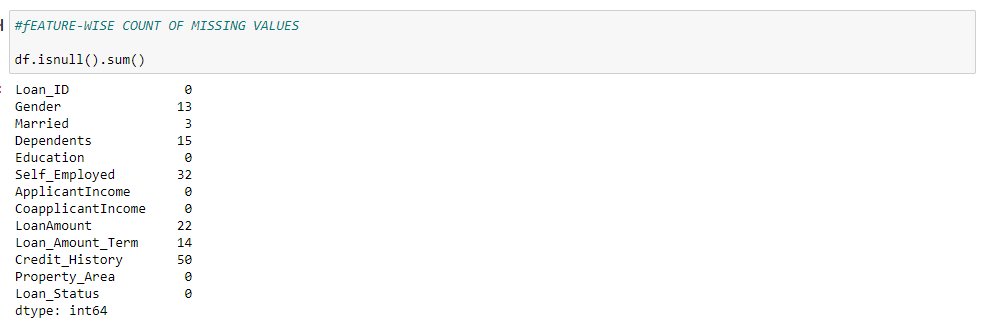
*Now let’s see the graph for LoanAmount:*



There are lot of outliers in the dataset as the loan amount highly differs from person to person. But the distribution is fairly normal.

## Data Cleaning/Pre-processing:

The first thing to do is to deal with the missing value, let’s check first how many there are for each variable.



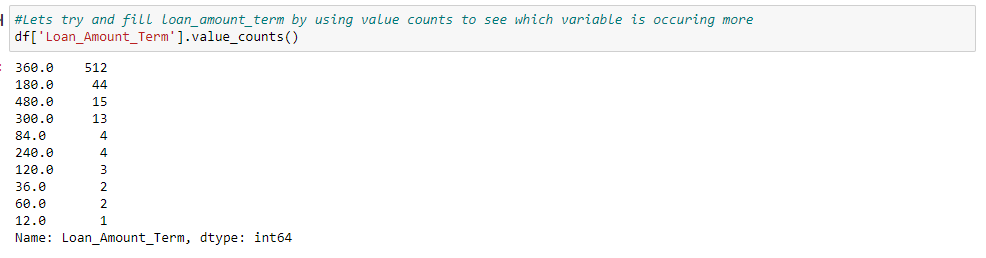
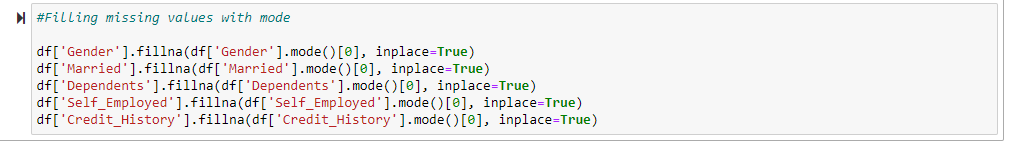
There are missing values in Gender, Married, Dependents, Self\_Employed, LoanAmount, Loan\_Amount\_Term, and Credit\_History features.

We will treat the missing values in all the features one by one. We will be using the following methods for replacing the missing values:

Numerical 🡪 using mean or median

Categorical 🡪 using mode

## Filling the Missing Value:



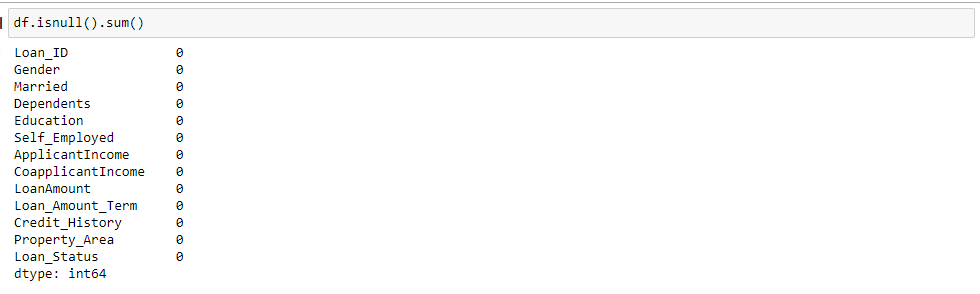
From the above we can see that '360.0' is the most occurring value, so we will replace all the missing values in Loan\_Amount\_Term with '360'.



Now let’s look into the numerical variable LoanAmount. We will use median for this purpose as we saw it has many outliers so the mean approach will not bring significant results from our model.



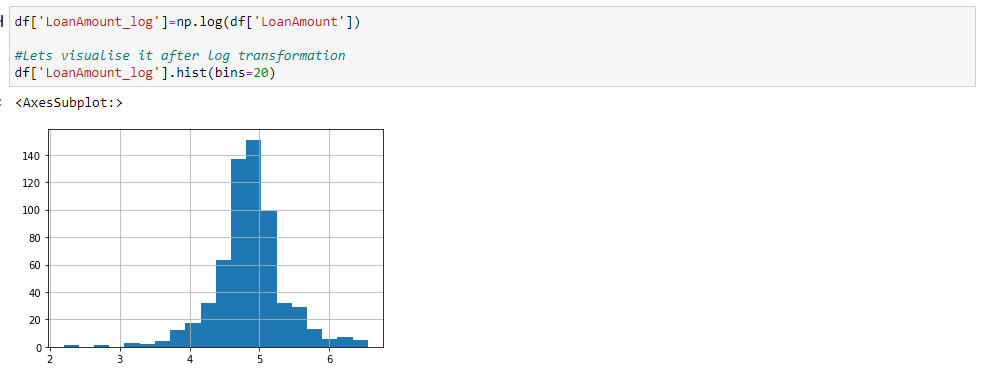
Now let’s check whether all the missing values are filled in the Dataset.



We can see from the above that all the null values have been replaced.

## NOW LETS REMOVE THE OUTLIERS:

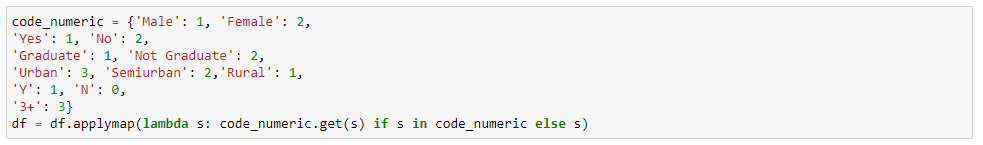
Due to outliers in the LoanAmount Data, bulk of the data is at the left and the right tail is longer. This is called right skewness. We will be using log transformation to remove the skewness as it doesn’t affect the smaller values but reduces the larger values so we get a normal distribution.



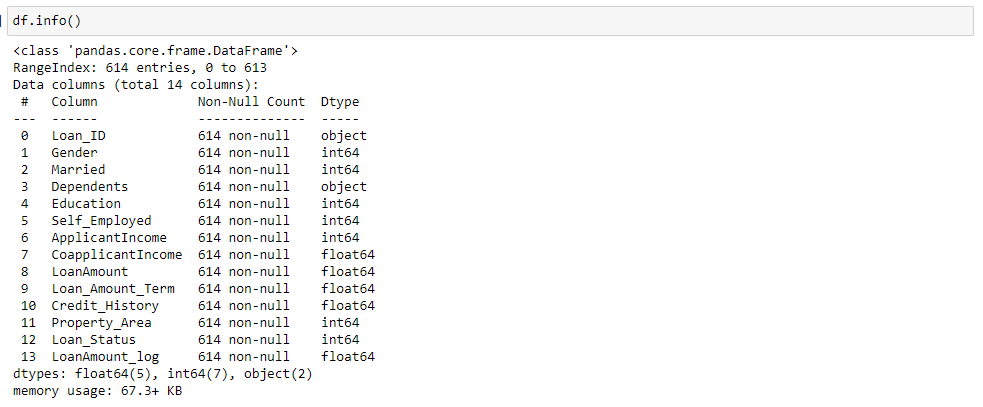
Looks like we got ourselves a graph which is much closer to normal distribution.

## Encoding Data:

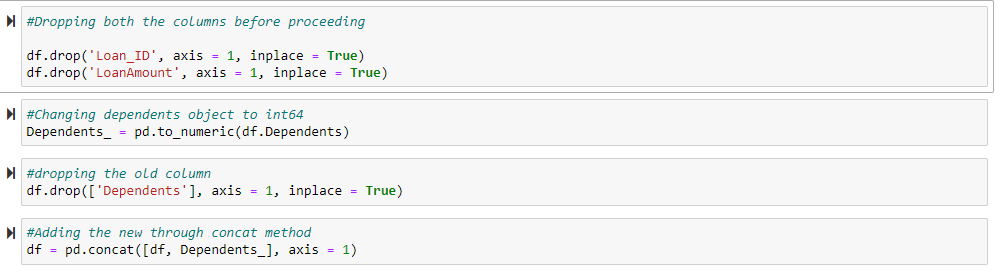
We’re going to use sklearn for our models, before doing that we need to turn all the categorical variables into numbers. We’ll do that using the Label Encoder in sklearn.



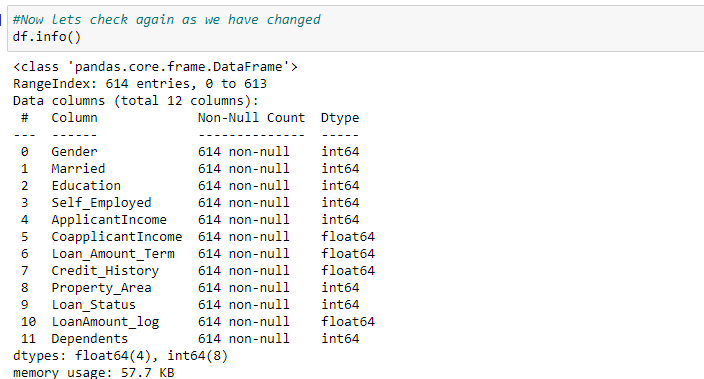
Now let’s call the .info method and see if our code worked or not:



We can see from the above that Dependents is object we need to change it to int64. We can also see that there is a duplicate column of LoanAmount as we have done a log transformation. Let’s drop the Loan\_ID column and LoanAmount as we already have a duplicate.

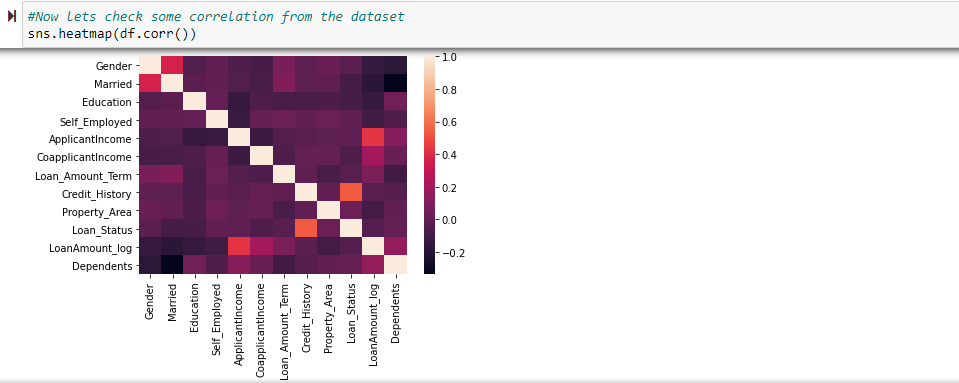


Now let’s check again for all the columns:



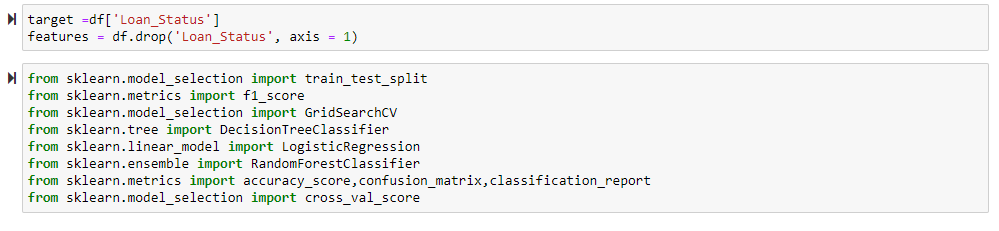
So, we have cleaned our data thoroughly so that we can get a better prediction from our Models.

## Now let’s check the correlation using the corr function:

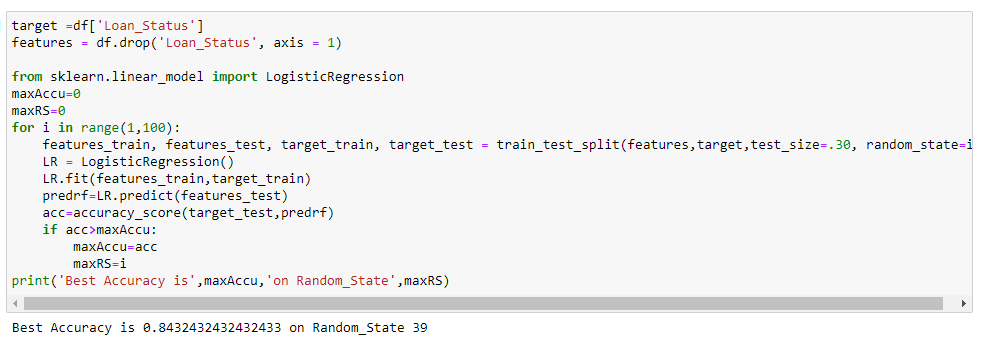


Showing the correlations of features with the target. No correlations are extremely high. The correlations between LoanAmount and ApplicantIncome are still seen.

## Separating independent variable and target variable:



## Selecting the Best Random State:

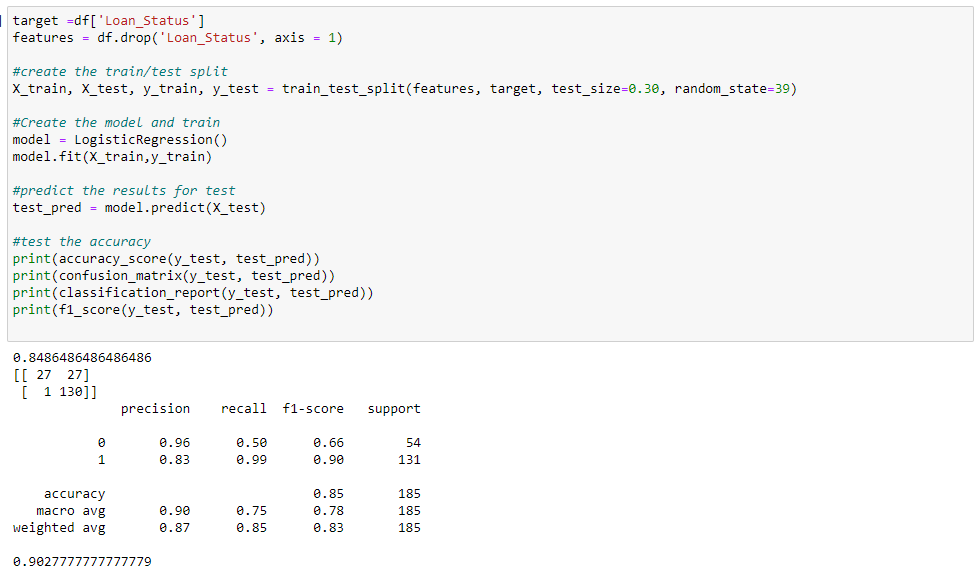


From the above we can see the best random state is 39 for test size = 0.30

## MODEL SELECTION:

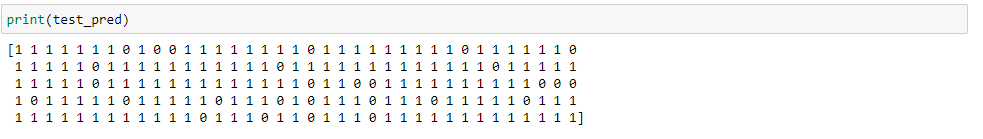
Now our data is ready! Let’s apply our model which is going to be the Classification because our Target variable 'Loan\_Status’ is not continuous. Let's now begin to train out regression model! We will need to first split up our data into an X array that contains the features to train on, and a y array with the target variable.

FIRST MODEL USING LOGISTIC REGRESSION:

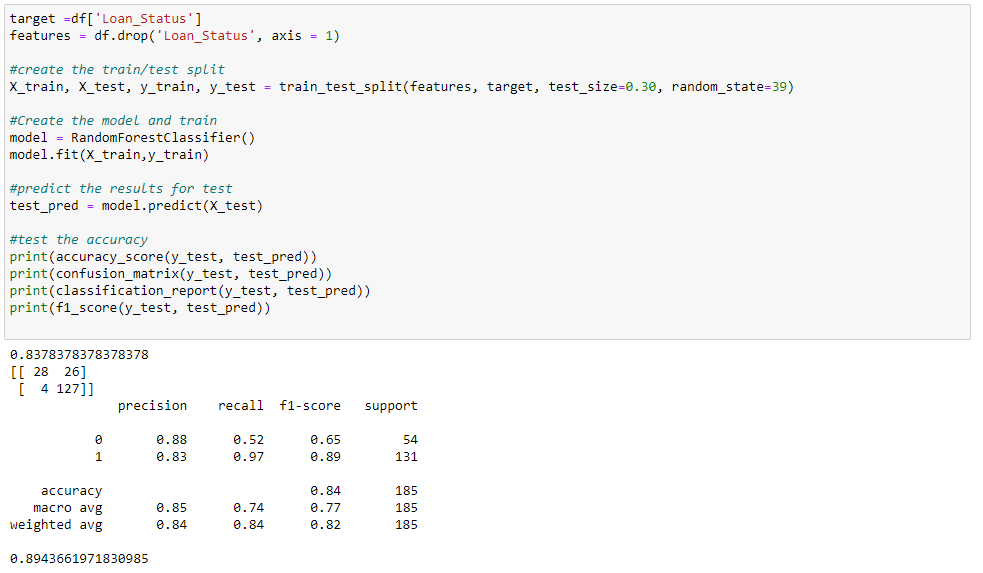


From the above we can see that the accuracy score is 0.902.

Let’s print the predicted values:

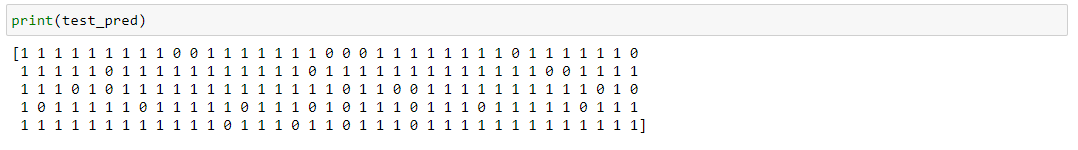


SECOND MODEL USING RANDOM FOREST CLASSIFIER:

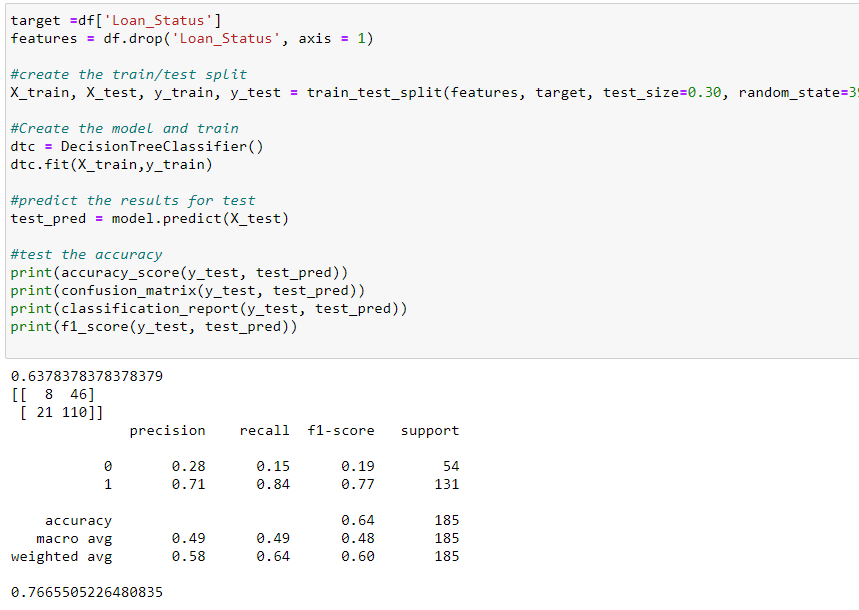


From the above we can see the accuracy score is 0.8943.

Let’s print the predicted values:

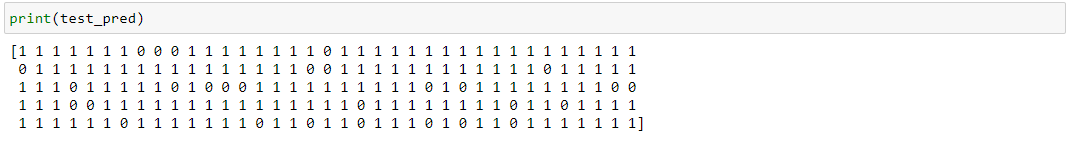


THIRD MODEL USING DECISION TREE CLASSIFIER:

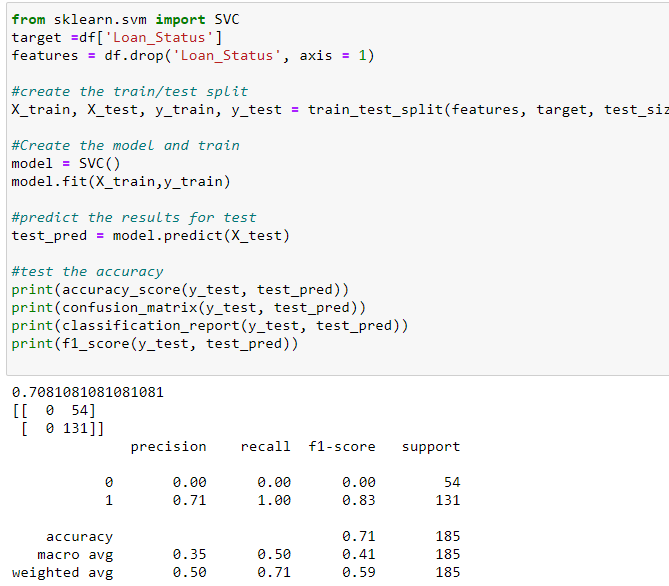


From the above we can see the accuracy score is 0.7665.

Let’s print the predicted values:

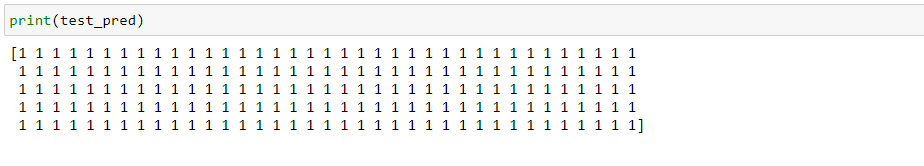


FOURTH MODEL USING SVC:

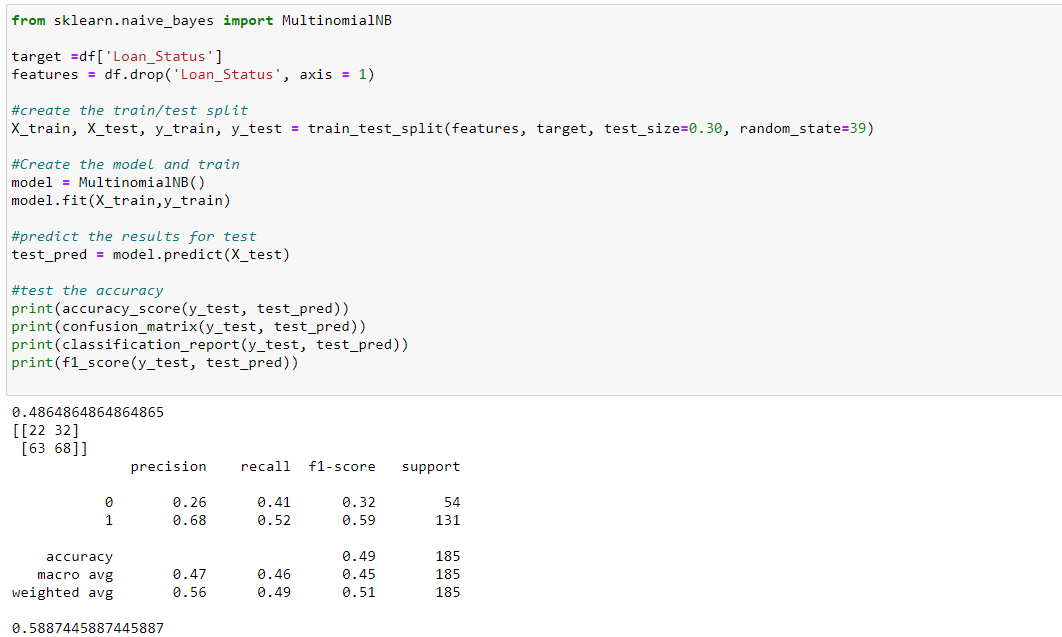


From the above we can see the accuracy score is 0.7081.

Let’s print the predicted values:

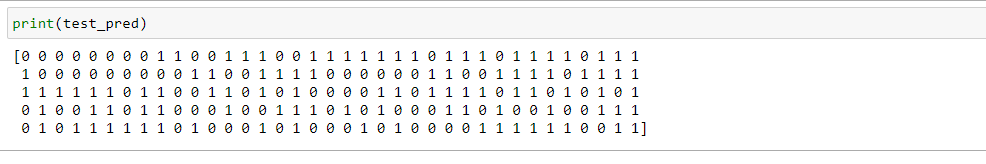


FIFTH MODEL USING MULTINOMIALNB:

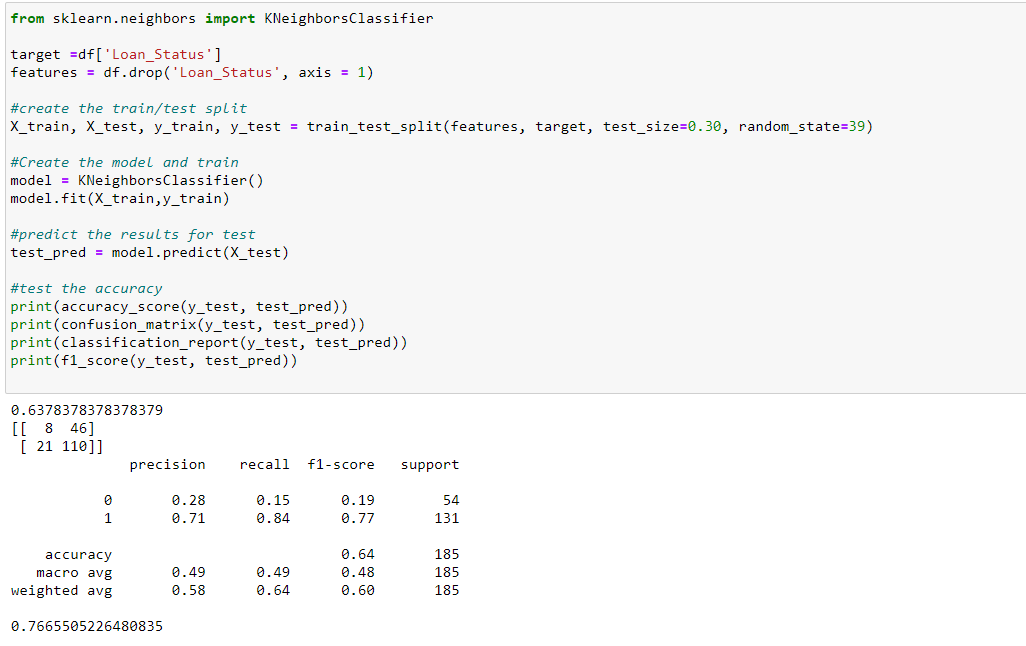


From the above we can see the accuracy score is 0.5887.

Let’s print the predicted values:

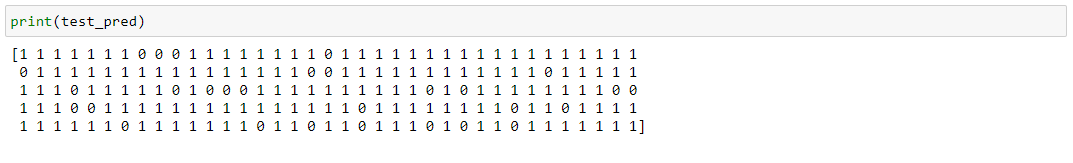


SIXTH MODEL USING KNEIGHBORSCLASSIFIER:

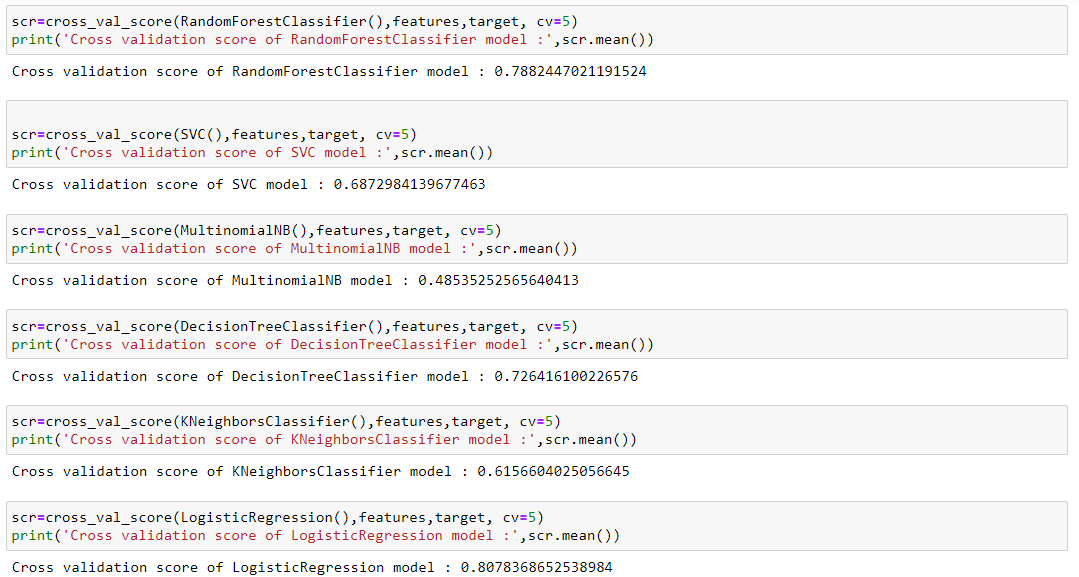


From the above we can see the accuracy score is 0.7665.

Let’s print the predicted values:

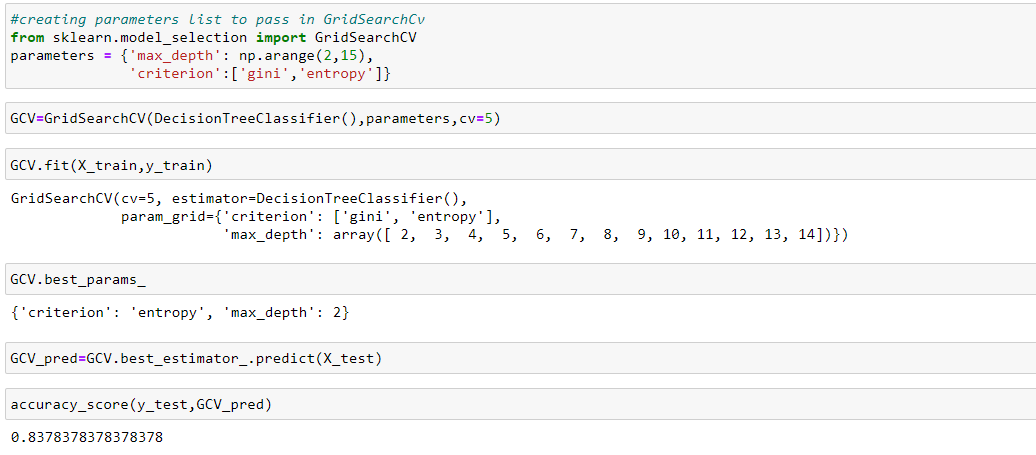


## CHECKING CROSS VALIDATION:



So, by cross checking our values with the accuracy score above, Decision Tree Classifier has the best difference in between cross validation scores and accuracy score.

## GridSearch CV Parameter Tuning:



## Saving Best Model Using Pkl:



By using pickl we saved our best model for future use.

## Conclusions:

We’ve gone through a good portion of the data science pipe line in this article, namely EDA, pre-processing and modelling and we’ve used essential classification models such as Logistic regression, Decision tree and Random forests. It would be interesting to learn more about the backbone logic behind these algorithms, and also tackle the data scraping and deployment phases.