**PREDICTING THE LOAN APPLICATION STATUS**



**Problem Definition:**

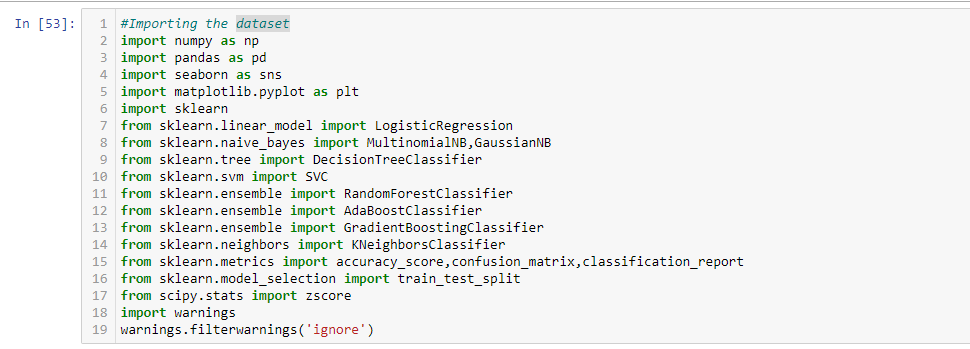
Loan lending has been an important part of daily lives for organizations and individuals alike. With the ever-increasing competition in the financial world and due to a significant amount of financial constraints, the activity of taking a loan has become inevitable. Individuals around the world depend on the activity of loan lending for reasons such as overcoming their financial constraints for them to achieve some personal goals. Similarly, banks, small to large firms depend on the activity of loan lending for the basic purpose of managing their affairs and to function smoothly in times where there are financial constraints.

Although it is quite profitable and beneficial for both the lenders and the borrowers. However, it carries a great risk, which in the domain of loan lending is referred to as Loan risk. Industry experts and researchers around the world assign individuals with numerical scores known as credit scores to measure the risk and their creditworthiness.

Throughout the years, machine learning algorithms have been used to calculate and predict credit risk by evaluating an individual’s historical data.

In this project, the dataset includes details of applicants who have applied for loan. The dataset includes details like credit history, loan amount, their income, dependents etc.  We must build a model that can predict whether the loan of the applicant will be approved or not based on the details provided in the dataset.

**Importing the libraries:**



*Exhibit 1*

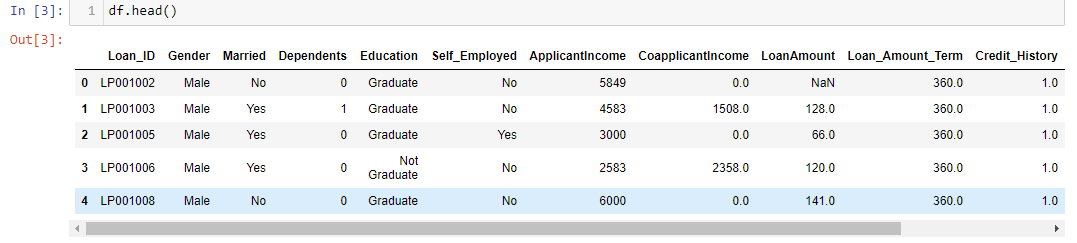
**Getting the data:**

**C:\Users\Neeti\Documents\two.png**

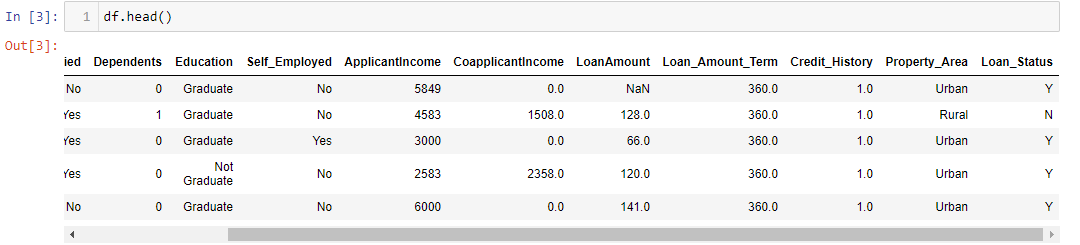
*Exhibit 2*

**Data Analysis:**

* Checking the first 5 rows of the dataset

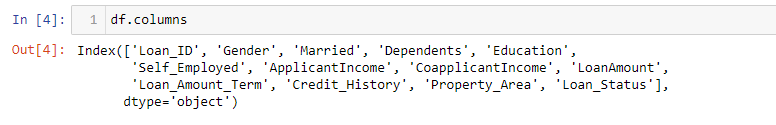


*Exhibit 3*



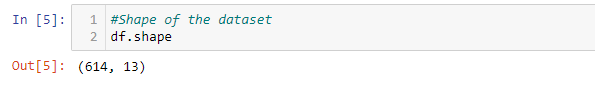
*Exhibit 4*

* Checking the columns of the dataset



*Exhibit 5*

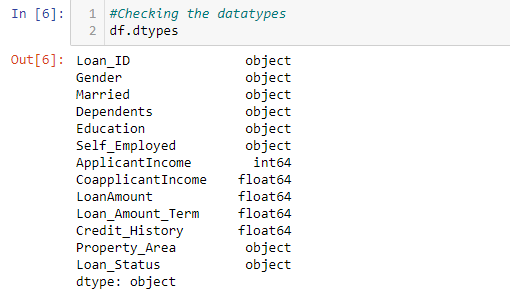
* Checking the shape of the dataset



*Exhibit 6*

The rows in the dataset is 614 and the columns are 13.There are 13 features including the target variable.

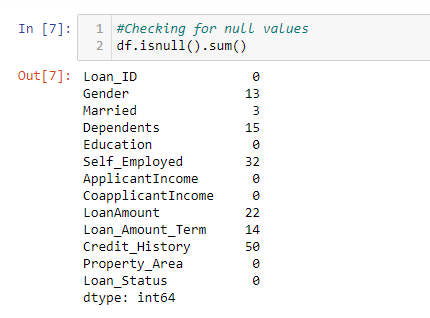
* Checking the datatypes



*Exhibit 7*

There are 8 object type features which we must encode in our future steps so that machine can understand.

* Checking for null values in the dataset



*Exhibit 8*

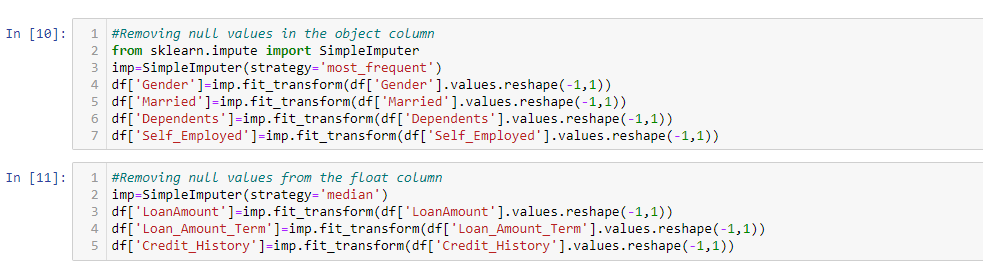
There are null values present in our dataset which we must work on in our future steps to make our data clean and ready for machine building process.

Now,we will start cleaning our data in our next steps so that we can build an efficient machine learning model.

**Data Pre-processing:**

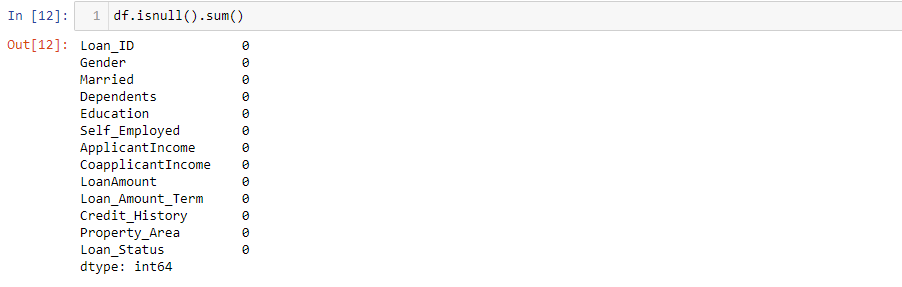
1. For cleaning our data, we have to remove the null values present in our dataset, so I will be using SimpleImputer method for removing those values.

I am attaching a screenshot of my code where I have applied SimpleImputer to remove the null values. I have applied two different strategies for object type data and float type data.



*Exhibit 9*

Now let’s check if any null values are left again to remove.

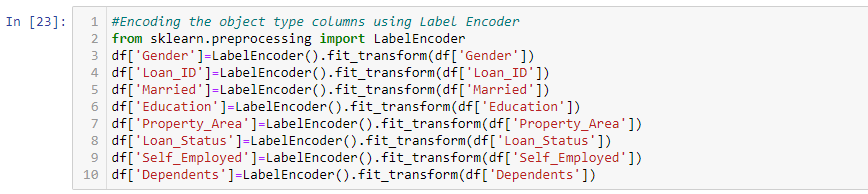


*Exhibit 10*

From the above screenshot,I can say that null values are not present in my dataset now.

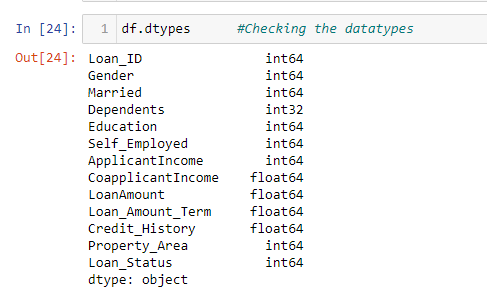
1. Now, I must encode the object type data as we know that a machine can understand only data in numeric form. So, I will use LabelEncoder to encode the object type data. The data are categorical type, so I am using LabelEncoder. But if the data in the object type column are continuous type data, then I will be using OrdinalEncoder.

Let’s encode it.



*Exhibit 11*

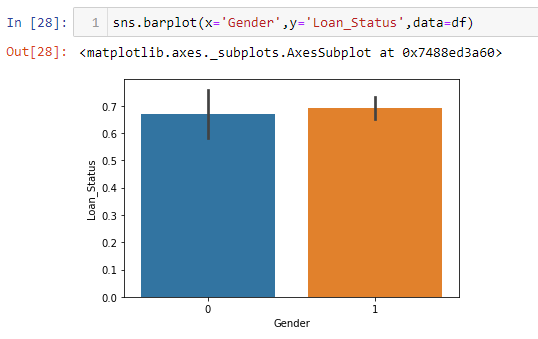
Checking the datatypes after encoding



*Exhibit 12*

No object data are available now. I have done partial cleaning of my data.

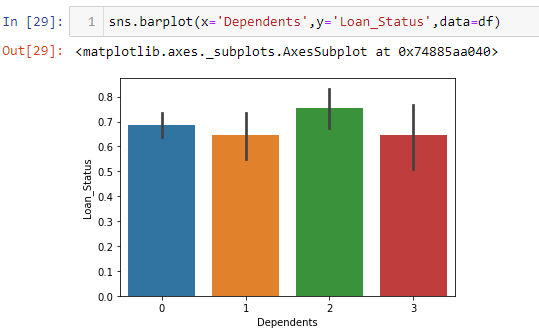
1. I have tried to visualize my data using graphs. I have taken two features including target variable and tried to draw a link in between those variables.



*Exhibit 13*

In the barplot, I can see that both male and female (encoded as 0 and 1) have almost same loan application status.

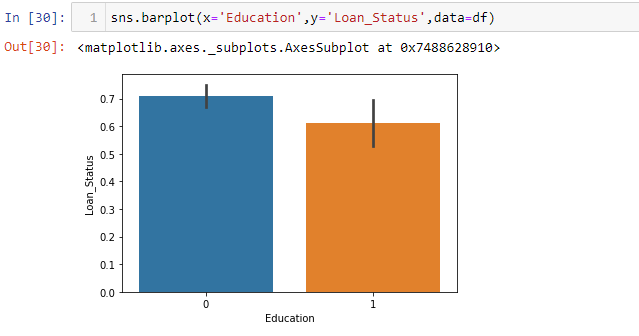
Now, I will check the relation between ‘Dependents’ and ‘Loan Status’ using barplot. Let’s see what the outcome of the plot will be.



*Exhibit 14*

There are four classes of dependents as seen in the barplot. Out of which the third one has more loan application status.

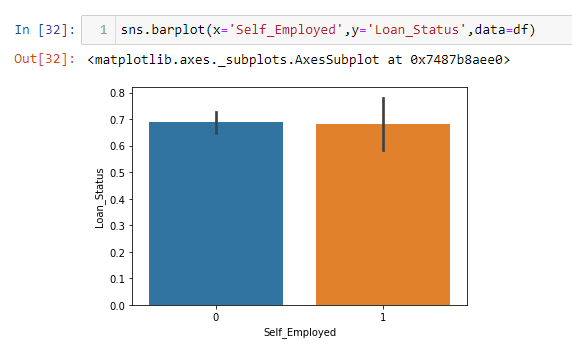
Below is the barplot between ‘Education’ and ‘Loan Status’ where graduates are encoded as 0 and not-graduates as 1.



*Exhibit 15*

It is clearly visible in the barplot that graduates have more loan application status.

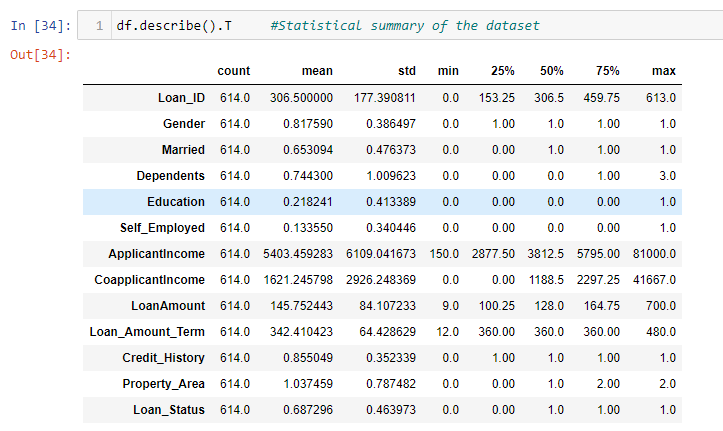
Now, let’s draw a link between ‘Self\_Employed’ and ‘Loan\_Status’ where non self-employed is encoded as 0 and self-employed as 1.



*Exhibit 16*

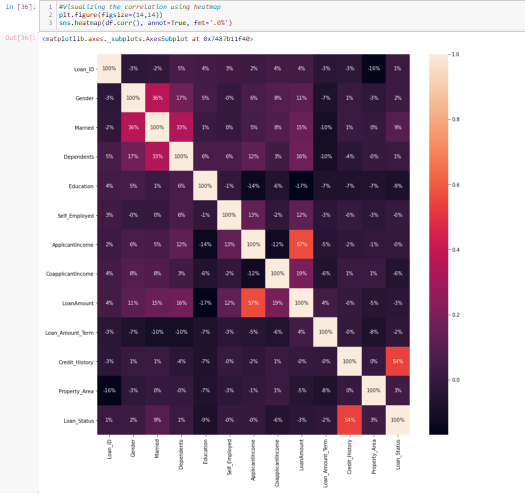
From the barplot, it is seen that both non self-employed and self-employed have same loan application status.

1. I will now go through the statistical summary of the dataset. I can draw some conclusions looking at the mean and standard deviation whether any outliers are present in the dataset or not. I think there are some outliers in the data and some skewed data present. I will look deep into it in my future steps.



*Exhibit 17*

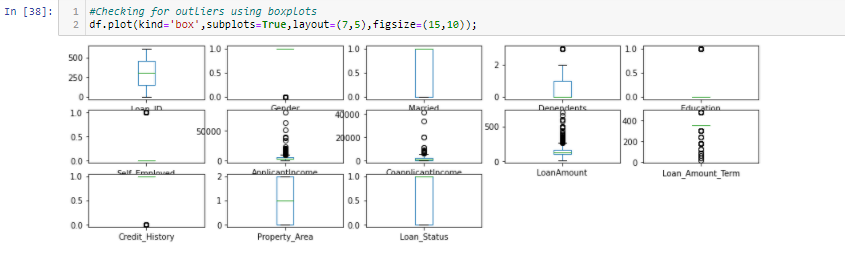
1. Let’s visualize how my data are correlated with each other and with the target variable, i.e. Loan\_Status.



*Exhibit 18*

From the heatmap, I can say that the feature ‘Credit\_History’ is highly correlated with the target variable ‘Loan\_Status’.

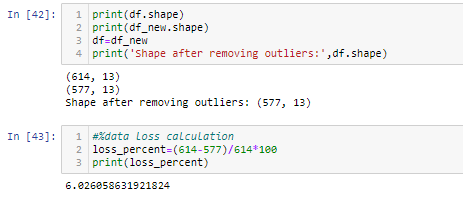
1. According to me, my data is cleaned to 60%. Now let’s check for outliers if any. I will check outliers using boxplots.



*Exhibit 19*

From above image,we can clearly see that there are number of black dots in most of thecolumn which are referring to the outliers, so it means most of the data are outsidethe distribution. Now I will remove them using zscore method.

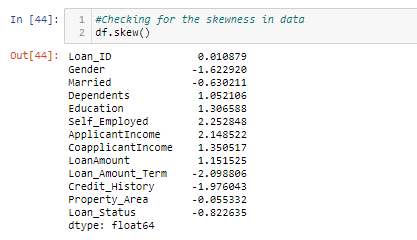
I will check what is the new shape of the dataset after outliers removal.



*Exhibit 20*

The new shape of my dataset is (577,13) means rows are 577 and columns are 13.We have lost 6% of our data during this process of outlier removal. But 6% of data loss is acceptable and now I can proceed with my new shape data.

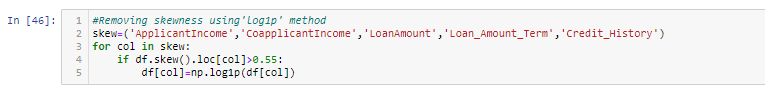
1. I will have a look on my skewed data now. I must minimise the skewness in my data if any before machine building.



*Exhibit 21*

By looking at the data, I can say skewness is present in some columns. But I will not remove skewness from categorical columns but have to remove skewness from float type columns.

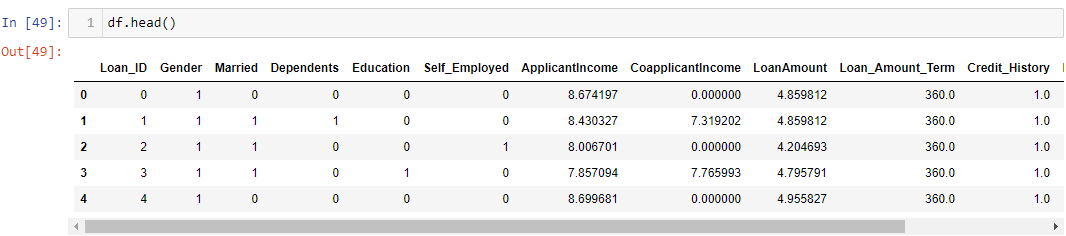
I am using ‘log1p’ method for skewness removal, we can also use power transform function for removing skewed data from the dataset.



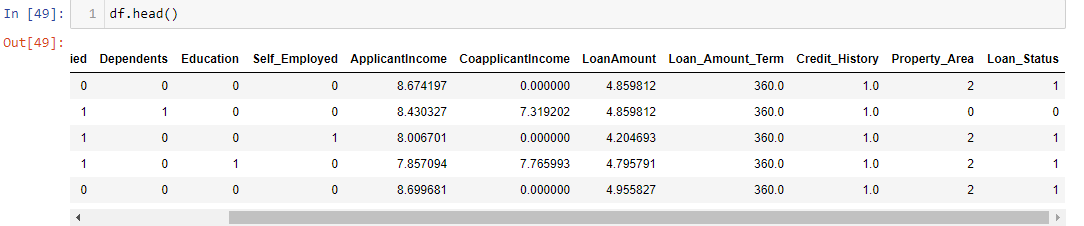
*Exhibit 22*

I have applied this method on selected columns and minimise the skewness in my data.

Now let’s see the first 5 rows of the dataset after cleaning the data.



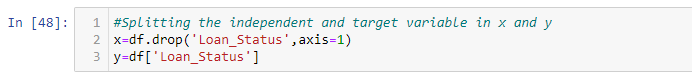
*Exhibit 23*

*Exhibit 24*

Data cleaning is completed. My data is getting ready for machine building process.

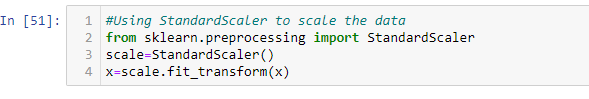
Before I start building the Machine Learning model, I will split my independent data and dependent data into x and y. Then I will scale the data to a standard form using StandardScaler.

**Splitting the data into x and y:**

.

*Exhibit 25*

**Scaling the data using StandardScaler:**



*Exhibit 26*

**EDA concluding remark:**

After going through data analysis and data processing, I can conclude that the raw data that I have received is now cleaned and is ready for Machine Learning process. Steps that I follow are listed below:

1. At the beginning, I have analysed the data by checking its shape, its datatypes and information regarding presence of null values.
2. After checking for null values, I have seen that there are some null values present. I removed them using SimpleImputer method.
3. While checking for datatypes, I have seen many object type columns are present, I have encoded them into numeric so that Machine can understand. The encoding process that I have used for categorical type data is LabelEncoder and for continuous type data I have used OrdinalEncoder.
4. I also visualized my data using barplots.
5. Then I have checked the statistical summary of the dataset and checked the correlation between the features and the target variable using heatmap.
6. I have removed the outliers present in the dataset using Zscore method and minimised the skewness in data using ‘log1p’ method.
7. When my DataFrame gets ready for Machine Learning process, I have split the independent variables and target variable into x and y.
8. Then I scaled my data into a standard form using StandardScaler.

**Building Machine Learning Models:**

Now I will train several Machine Learning models and compare their results. Later on I will use cross validation.

****

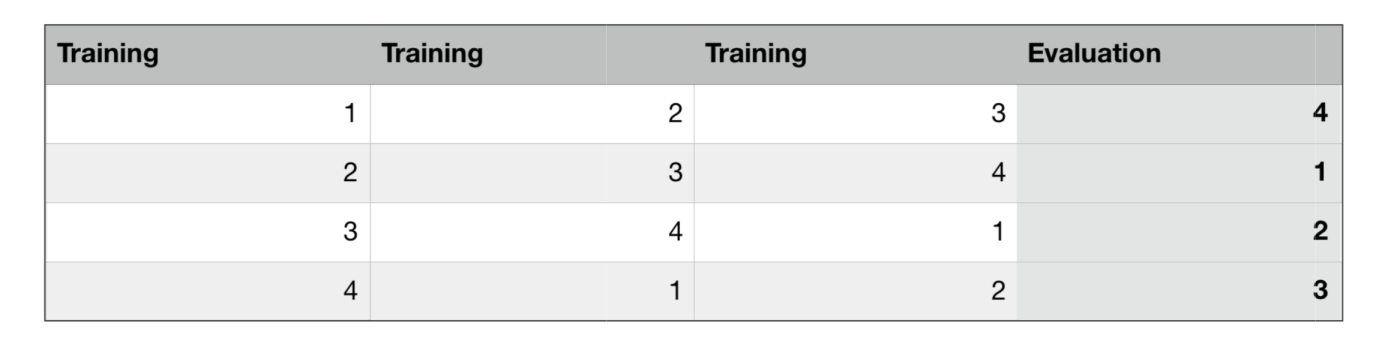
*Exhibit 27*

As we can see, LogisticRegression is giving a good accuracy of 88% among all the classification models at a random state 68. But, let us check, how LogisticRegression performs when we use cross validation.

**K Fold Cross-Validation:**

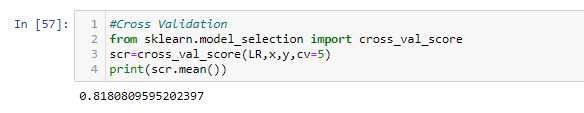
K-Fold Cross Validation randomly splits the training data into **K subsets called folds**. Let’s image we would split our data into 4 folds (K = 5). Our model would be trained and evaluated 5 times, using a different fold for evaluation every time, while it would be trained on the remaining 4 folds.

The image below shows the process, using 4 folds (K = 4). Every row represents one training + evaluation process. In the first row, the model gets trained on the first, second and third subset and evaluated on the fourth. In the second row, the model gets trained on the second, third and fourth subset and evaluated on the first. K-Fold Cross Validation repeats this process till every fold acted once as an evaluation fold.

****

*Exhibit 28*

The code below performs K-Fold Cross Validation on our LogisticRegression model, using 5 folds (K = 5 or cv=K).

****

*Exhibit 29*

After cross validation, we get the actual accuracy of the model,i.e. 82%. Before it was 88% because of over-fitting. Now, I will try to increase its performance even better in the following section.

**Hyperparameter tuning:**

A Machine Learning model is defined as a mathematical model with several parameters that need to be learned from the data. By training a model with existing data, we can fit the model parameters. However, there is another kind of parameters, known as **Hyperparameters**, that cannot be directly learned from the regular training process. They are usually fixed before the actual training process begins. These parameters express important properties of the model such as its complexity or how fast it should learn.

Models can have many hyperparameters and finding the best combination of parameters can be treated as a search problem. Two best strategies for Hyperparameter tuning are:

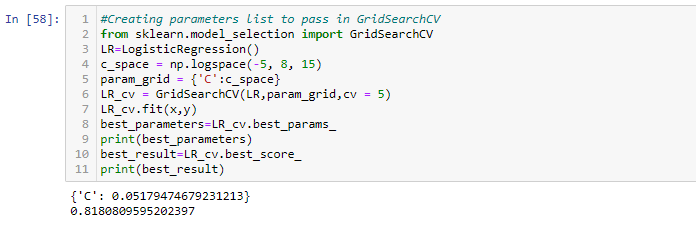
* **GridSearchCV:**

In GridSearchCV approach, machine learning model is evaluated for a range of hyperparameter values. This approach is called GridSearchCV, because it searches for best set of hyperparameters from a grid of hyperparameters values.

* **RandomizedSearchCV:**

RandomizedSearchCV solves the drawbacks of GridSearchCV, as it goes through only a fixed number of hyperparameter settings. It moves within the grid in random fashion to find the best set hyperparameters. This approach reduces unnecessary computation.

Now I will try to tune the hyperparameters and check if we can increase the model's accuracy.

****

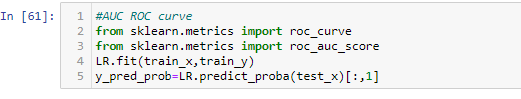
*Exhibit 30*

The accuracy after hyperparameter tuning is also 82%. The best parameters are {‘C’:0.0517}.

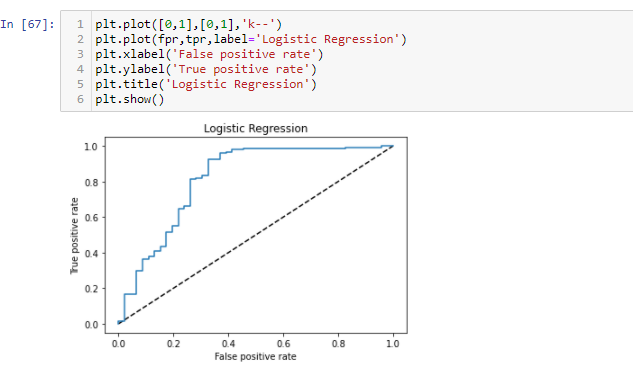
Now,I will fit the hyperparameters into the LogisticRegression model.

**ROC AUC Curve:**

Another way to evaluate and compare our binary classifier is provided by the ROC AUC Curve. This curve plots the true positive rate (also called recall) against the false positive rate (ratio of incorrectly classified negative instances).



*Exhibit 31*

****

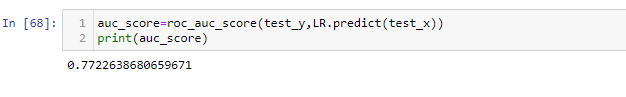
*Exhibit 32*

The dotted line in the middle represents a purely random classifier (e.g a coin flip) and therefore our classifier should be as far away from it as possible.

**ROC AUC score:**

The ROC AUC Score is the corresponding score to the ROC AUC Curve. It is simply computed by measuring the area under the curve, which is called AUC.

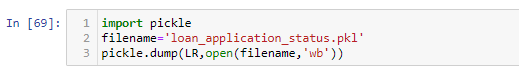
A classifierthat is 100% correct, would have a ROC AUC Score of 1 and a completely random classifier would have a score of 0.5.



*Exhibit 33*

The score is 0.7722 which is good but not so satisfactory.

Now, I am saving the model.



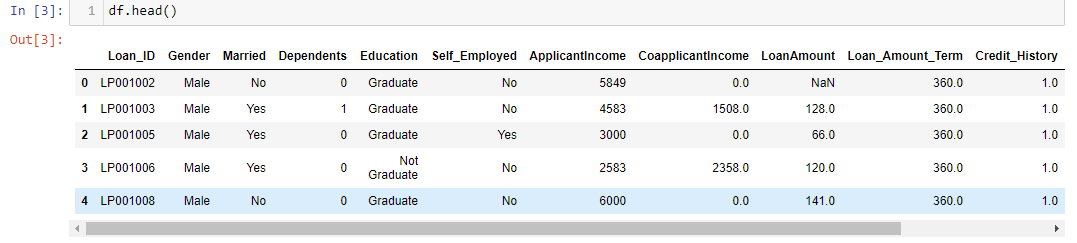
*Exhibit 34*

**Conclusion:**

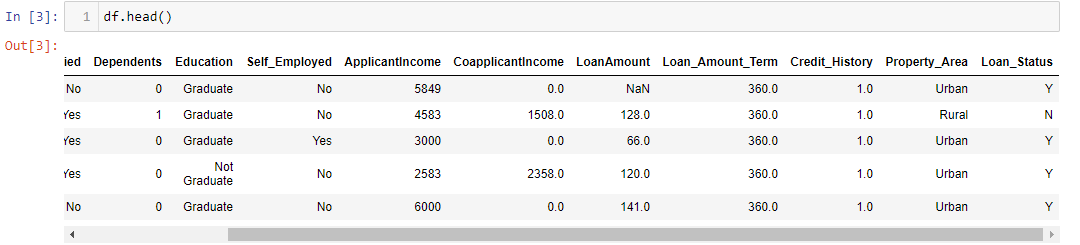
We started with the data exploration where we got a feeling for the dataset, checked about missing data, and learned which features are important. During this process we used seaborn and matplotlib to do the visualizations. During the data pre-processing part, we converted features into numeric ones,grouped values into categories and computed missing values. Afterwards we started training 5 different machine learning models, picked one of them (LogisticRegression) and applied cross validation on it. Then we discussed how LogisticRegression works, took a look at the importance it assigns to the different features and tuned its performance through optimizing its hyperparameter values.

Below I can see a before and after picture of the “df” dataframe:

**Before:**

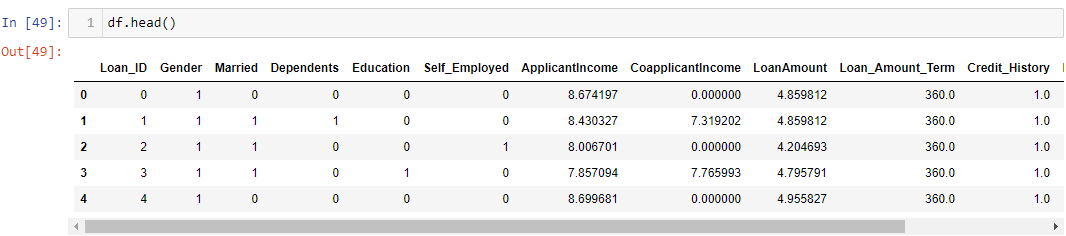
****

*Exhibit 35*

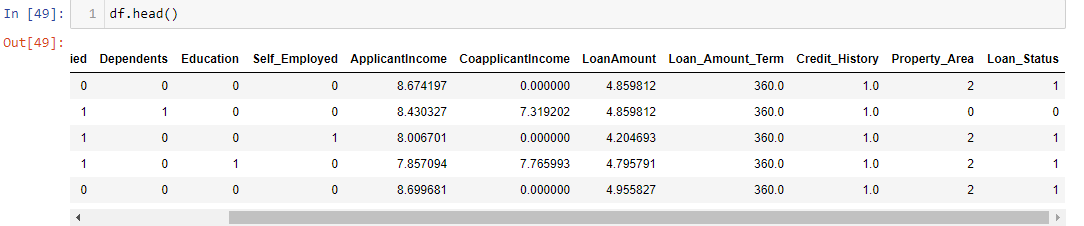
****

*Exhibit 36*

**After:**

****

*Exhibit 37*

****

*Exhibit 38*

Naturally, there is still room for improvement, like doing a more extensive feature engineering, by comparing and plotting the features against each other and identifying and removing the noisy features.