**Loan Application Status Prediction**

This repo contains the Loan Approval Status Prediction project as part of my data science portfolio. This project is completed as part of the Data trained Evaluation Project. Evaluation metric of the project is accuracy i.e., percentage of loan approval that is correctly predicted. After trying and testing 4 different algorithms, the best accuracy on the leader board is achieved by Logistic Regression (0.8255) and Decision Tree performed the worst (0.8255). followed by SVC (0.8209) and Random Forest (0.8023).

This project covers the whole process from problem statement to model development and evaluation:

1. Problem Definition.

2. Hypothesis Generation

3. Data Analysis.

4. EDA Concluding Remark.

5. Pre-Processing Pipeline.

6. Building Machine Learning Models.

7. Concluding Remarks.

**Problem Definition**

**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.

**Problem Statement is Translated into Data Science Problem:**

This is a classification problem where we have to predict whether a loan will be approved or not. Specifically, it is a binary classification problem where we have to predict either one of the two classes given i.e., approved (Y) or not approved (N). Another way to frame the problem is to predict whether the loan will likely to default or not, if it is likely to default, then the loan would not be approved, and vice versa. The dependent variable or target variable is the Loan\_Status, while the rest are independent variable or features. We need to develop a model using the features to predict the target variable.

**Hypothesis Generation**

Hypothesis Generation is the process of listing out all the possible factors that can affect the outcome i.e., which of the features will have an impact on whether a loan will be approved or not. Some of the hypothesis are:

* Education - Applicants with higher education level i.e., graduate level should have higher chances of loan approval
* Income: Applicants with higher income should have more chances of loan approval
* Loan amount: If the loan amount is less, the chances of loan approval should be high
* Loan term: Loans with shorter time period should have higher chances of approval
* Previous credit history: Applicants who have repayed their previous debts should have higher chances of loan approval
* Monthly instalment amount: If the monthly instalment amount is low, the chances of loan approval should be high
* And so, on

Some of the hypothesis seem intuitive while others may not. We will try to validate each of these hypotheses based on the dataset.

**Data Analysis**

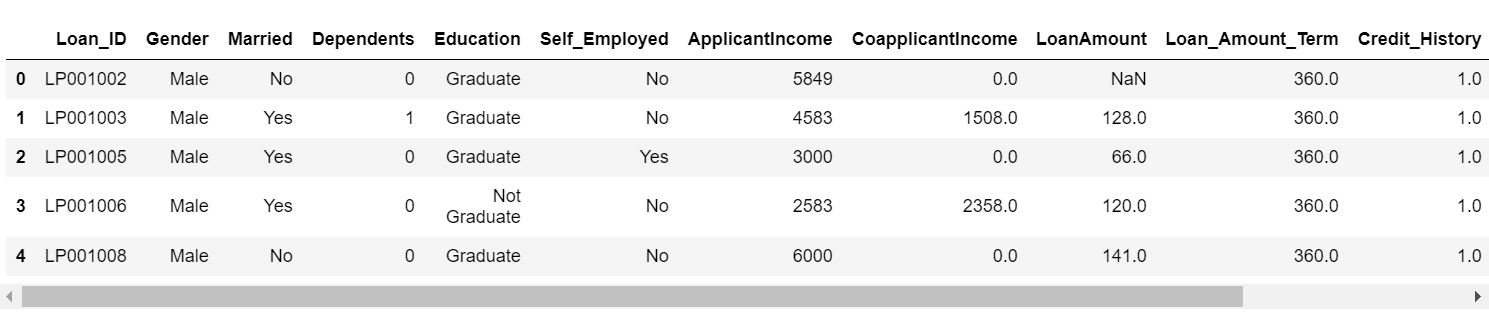
The data have already been provided by Data trained Institute. The training set will be used for training the model, i.e., our model will learn from this data. It contains all the independent variables and the target variable. The test set contains all the independent variables, but not the target variable. We will apply the model to predict the target variable for the test data. There are 13 columns of features and 614 rows of records in the training set and 12 columns of features and 367 rows of records in the test set. The dataset variables are summarized as below:

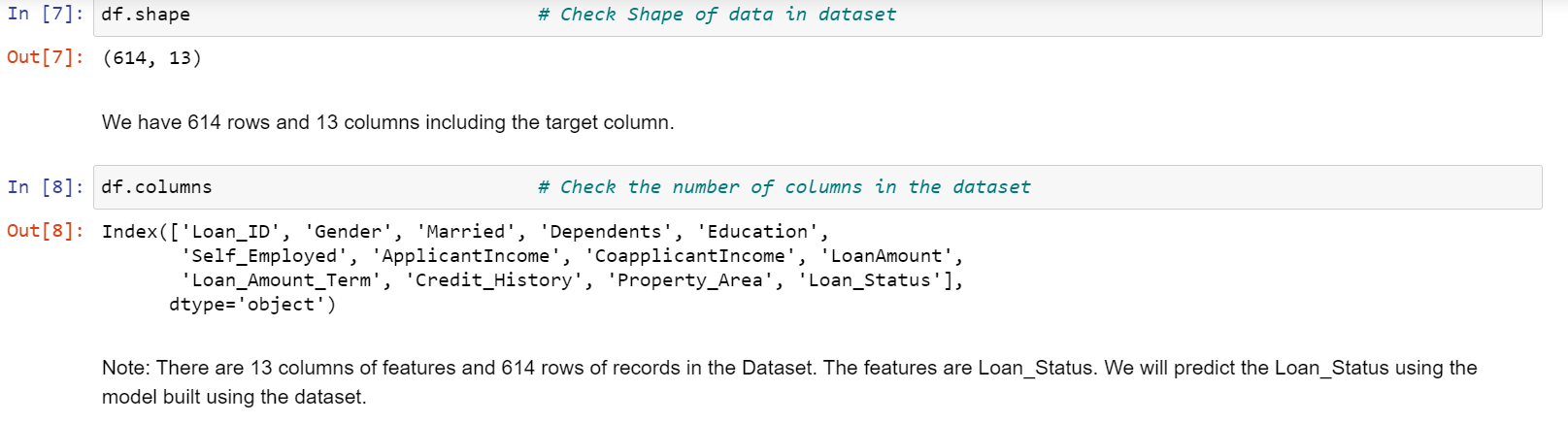
|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Variable** | **Type** | **Description** |
| 1 | Loan\_ID | Numerical – Discrete | Unique Loan ID |
| **2** | Gender | Categorical – Nominal | Male / Female |
| **3** | Married | Categorical – Nominal | Applicant married (Y/N) |
| **4** | Dependents | Categorical – Ordinal | Number of dependents (0, 1, 2, 3+) |
| **5** | Education | Categorical – Nominal | Applicant Education (Graduate / Under Graduate) |
| **6** | Self\_Employed | Categorical – Nominal | Self-employed (Y/N) |
| **7** | Application Income | Numerical – Continuous | Applicant income |
| **8** | CoapplicantIncome | Numerical – Continuous | Coapplicant income |
| **9** | Loan\_Amount | Numerical – Continuous | Loan amount in thousands |
| **10** | Loan\_ Amount\_Term | Numerical – Discrete | Term of loan in months |
| **11** | Credit\_History | Categorical - Nominal | credit history meets guidelines (0, 1) |
| **12** | Property\_Area | Categorical - Ordinal | Urban / Semi Urban / Rural |
| **13** | Loan\_Status | Categorical - Nominal | Loan approved (Y/N) |

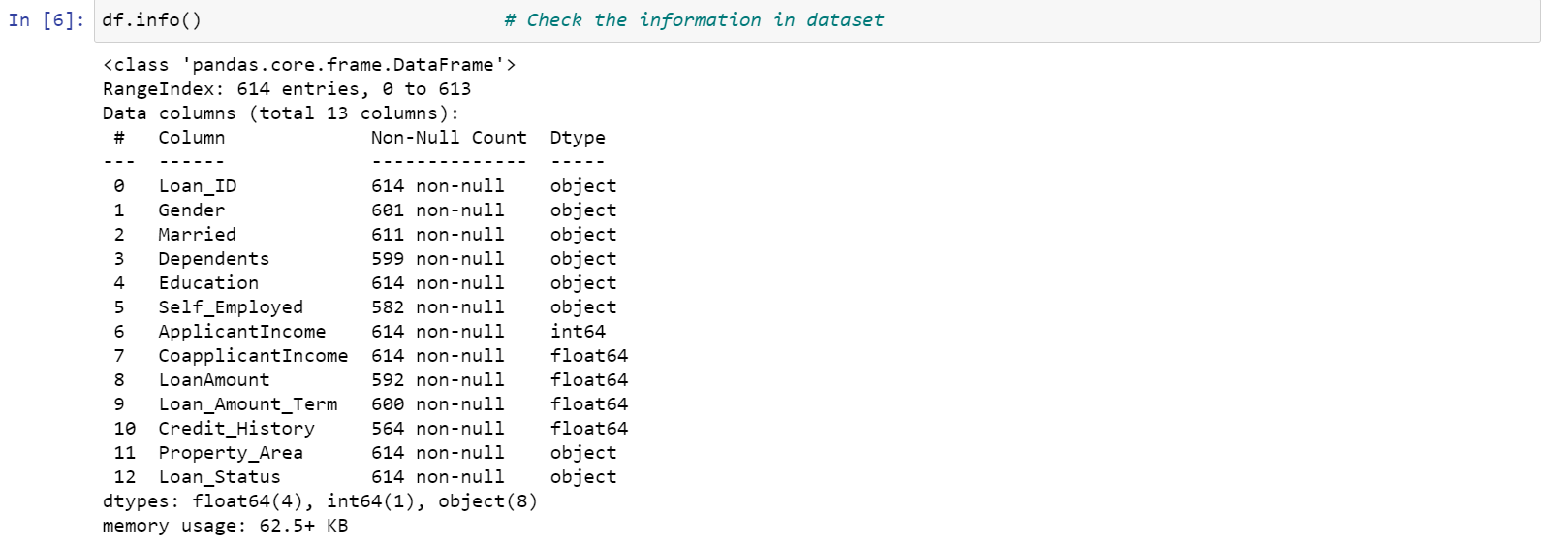
# Exploratory Data Analysis (EDA) Concluding Remark

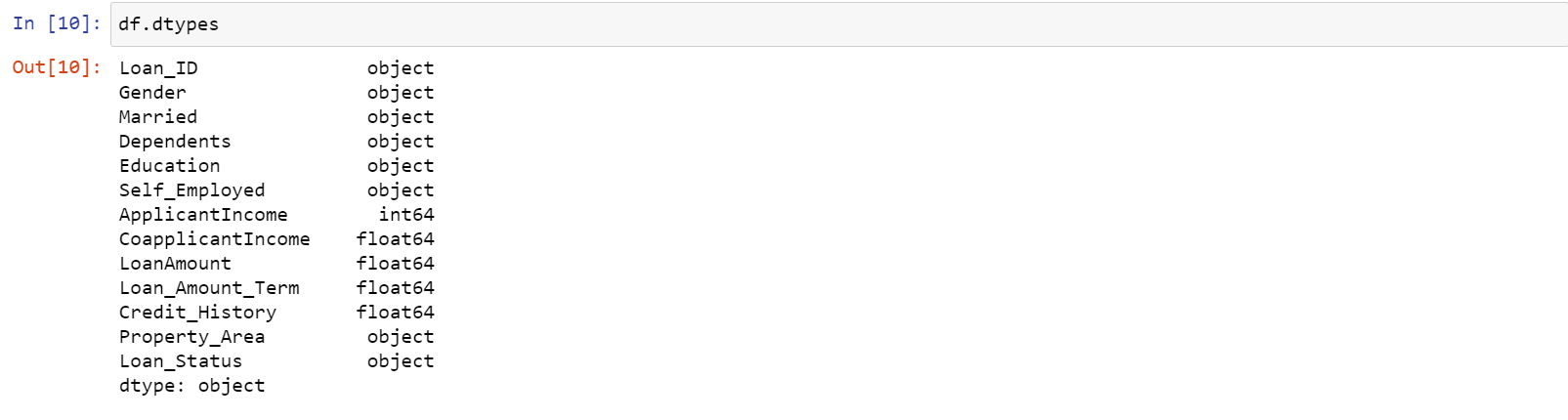
# We will use Python to explore the data in order to gain a better understanding of the features and target variable. We will also analyse the data to summarize their main characteristics, using various visualization techniques.

# 









**Note:**There are 3 data types in the data

* object: Object format means variables are categorical. Categorical variables in our dataset are: Loan\_ID, Gender, Married, Dependents, Education, Self\_Employed, Property\_Area, Loan\_Status
* int64: It represents the integer variables. ApplicantIncome is of this format.
* float64: It represents the variable which have some decimal values involved. They are also numerical variables. Numerical variables in our dataset are: CoapplicantIncome, LoanAmount, Loan\_Amount\_Term, and Credit\_History.

**Why Data Types are important?**

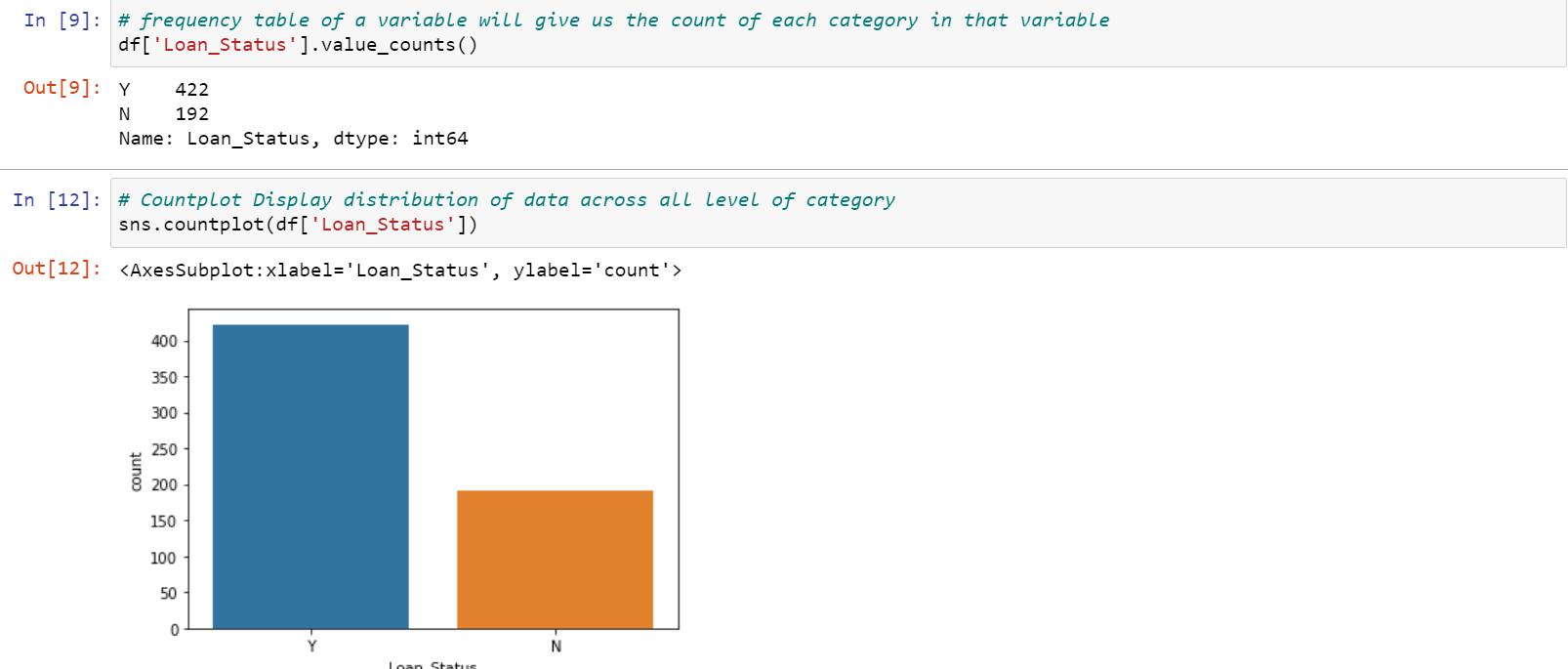
Datatypes are an important concept because statistical methods can only be used with certain data types. You have to analyse continuous data differently than categorical data otherwise it would result in a wrong analysis. Therefore, knowing the types of data, you are dealing with, enables you to choose the correct method of analysis.

## **Univariate analysis**

Univariate analysis is when we analyse 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 can’t show you if you have any outliers. This is why we also use box-plots.

### **Target Variable (Categorical)**

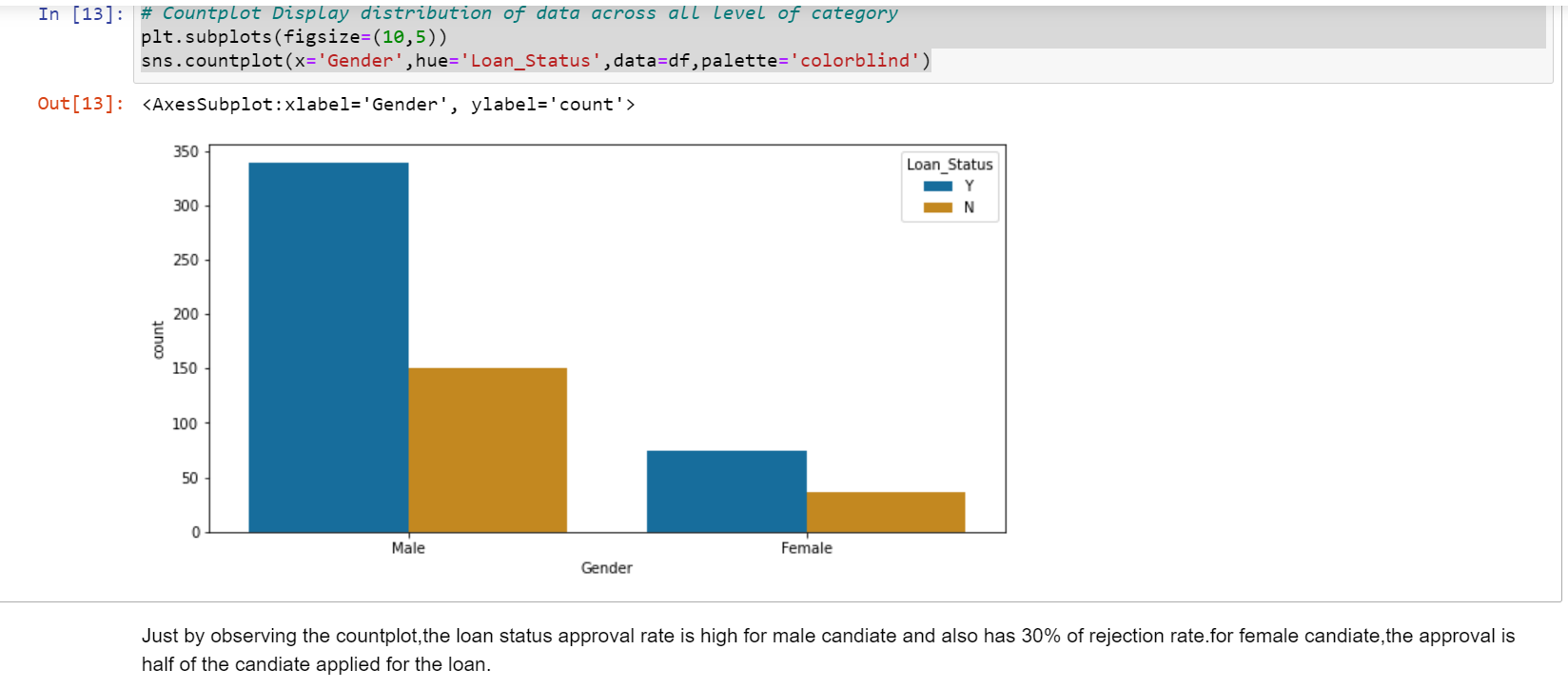
We will first look at the target variable, i.e., Loan\_Status. As it is a categorical variable, let us look at its frequency table, percentage distribution and bar plot.

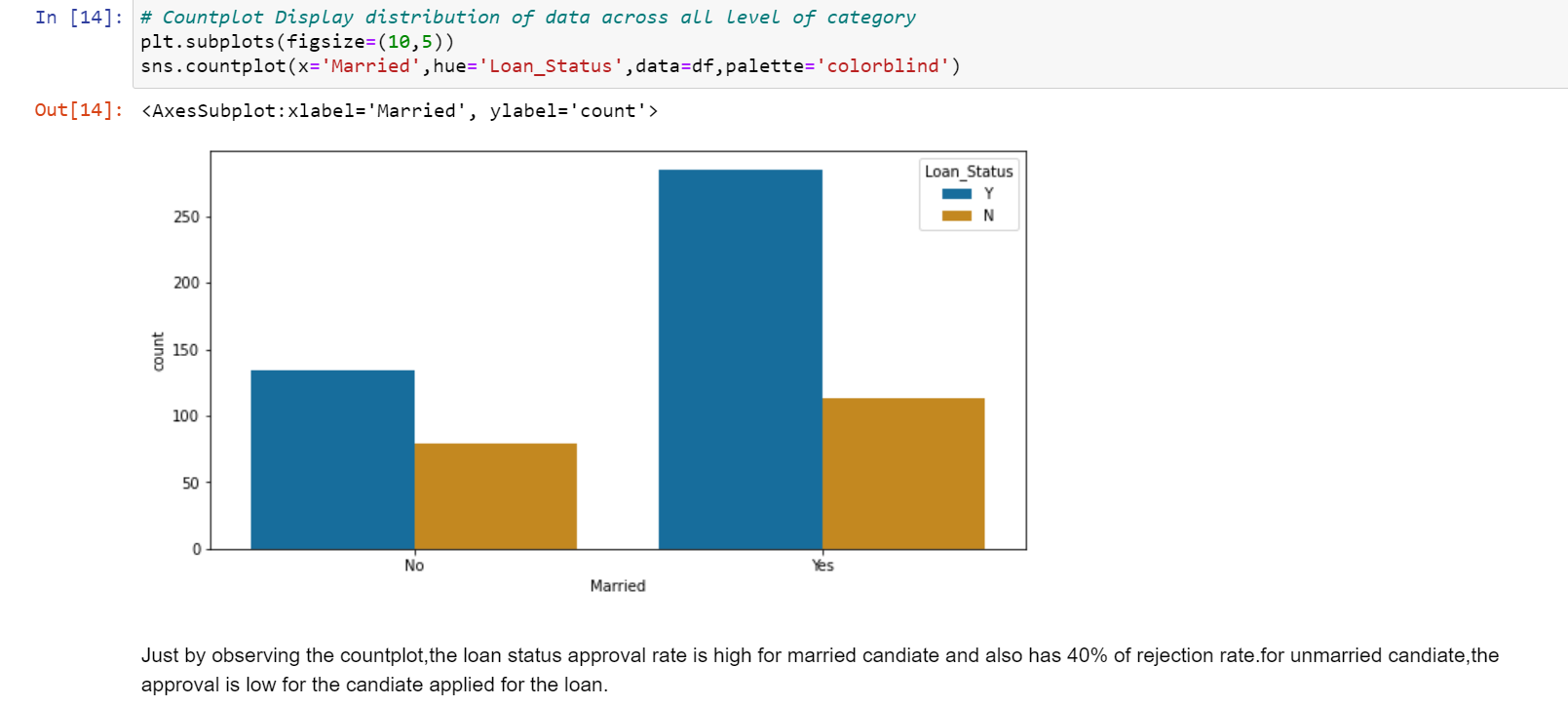


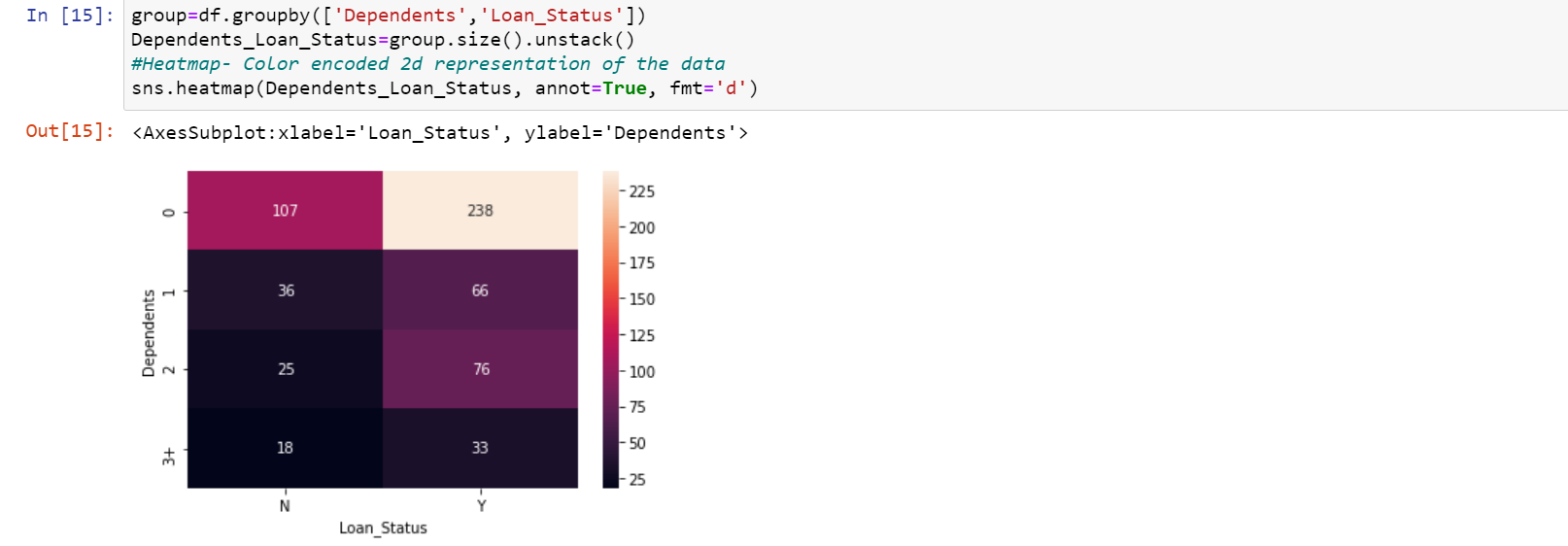
**Note:**The loan of 422 (around 69%) people out of 614 was approved. There is no imbalanced classes issue in this dataset, thus accuracy as an evaluation metric should be appropriate. On the other hand, if there are imbalanced or skewed classes, then we might need to use precision and recall as evaluation metrics.

### **Independent Variable (Categorical)**

There are 5 features that are categorical or binary (Gender, Married, Self\_Employed, Credit\_History, Education)



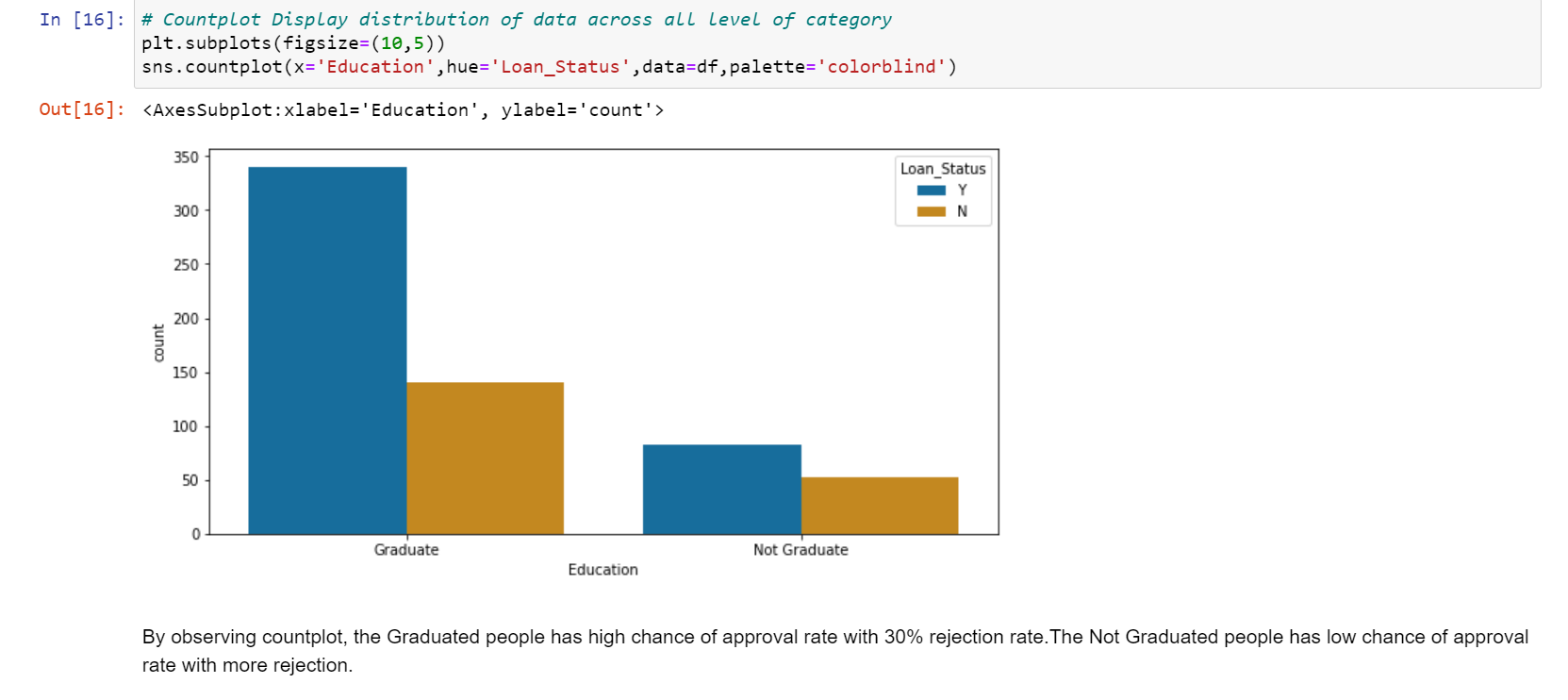


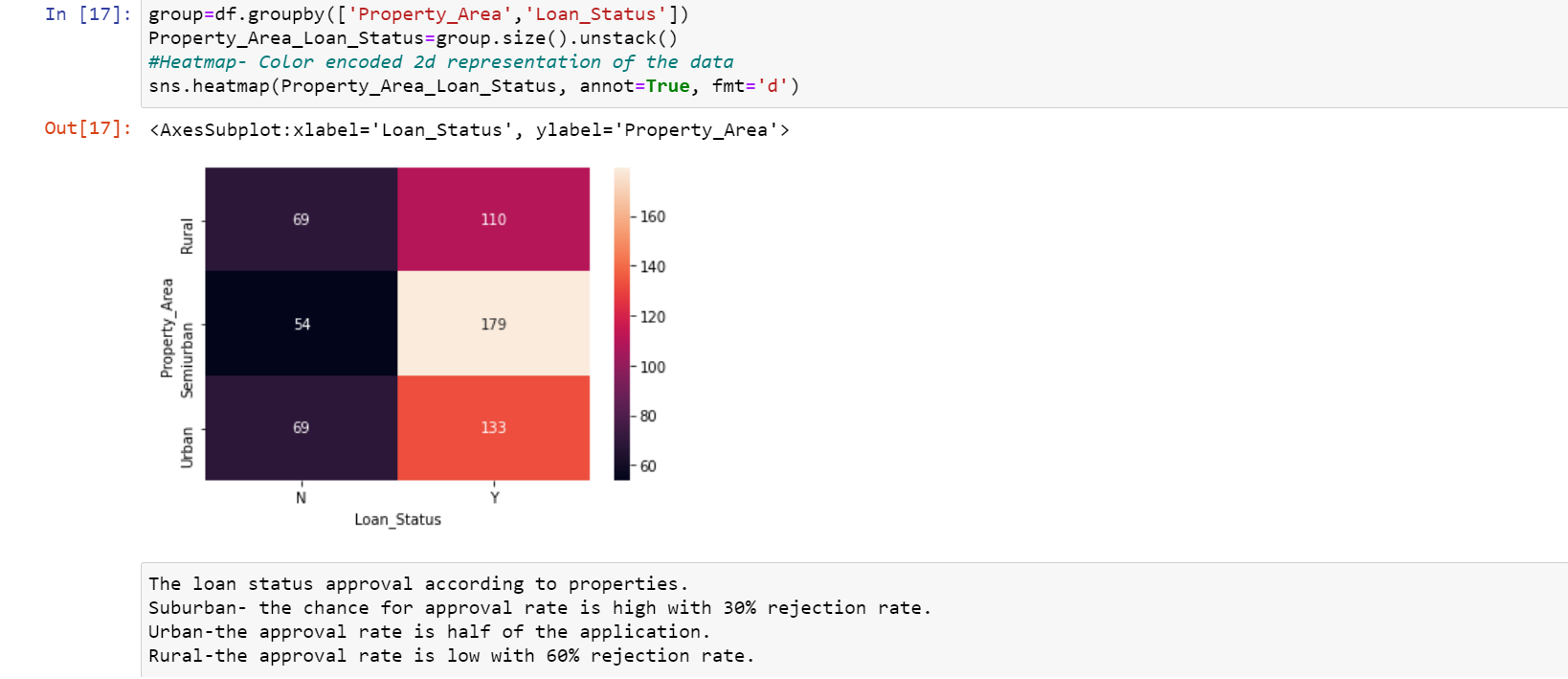


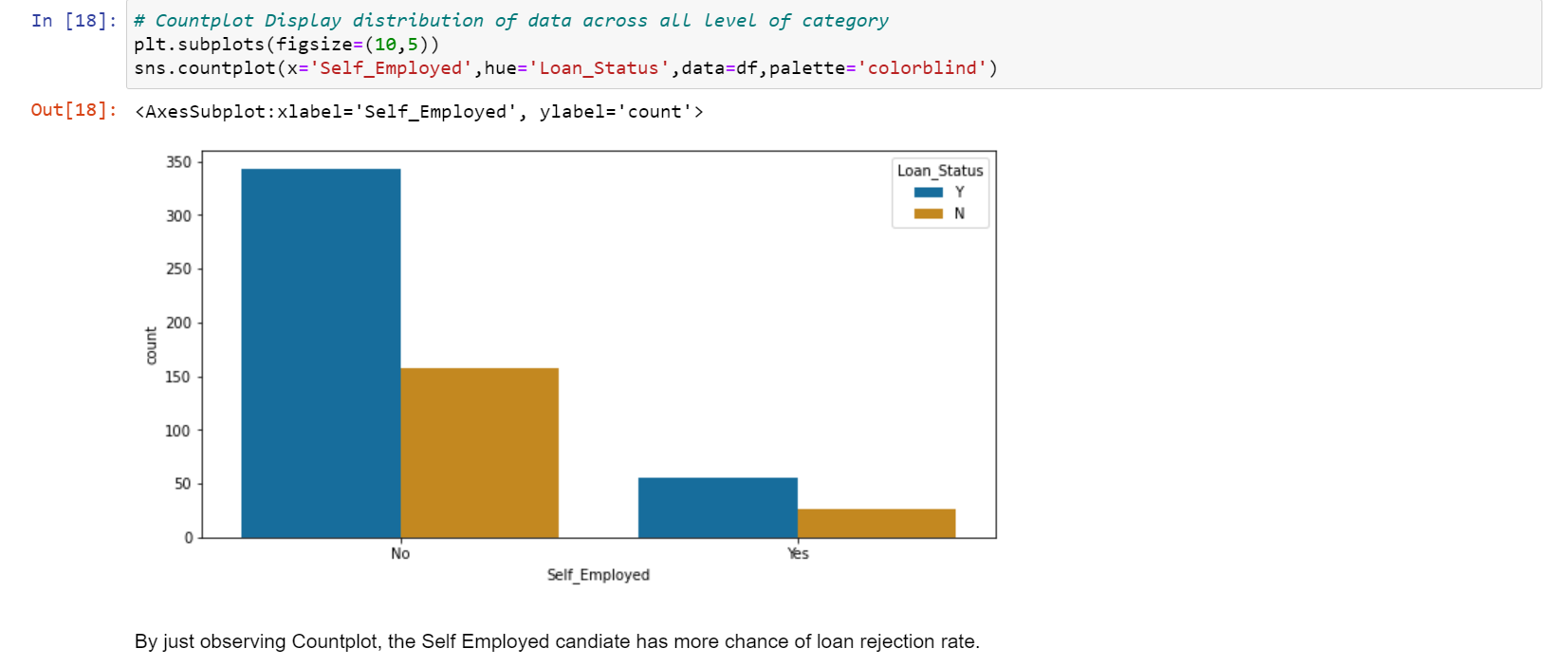
The loan status approval rate for dependent people.

No dependent- the chance for approval rate is high.

1,2,3+ dependent Peron- the chance of approval is low when compared to No dependent people.











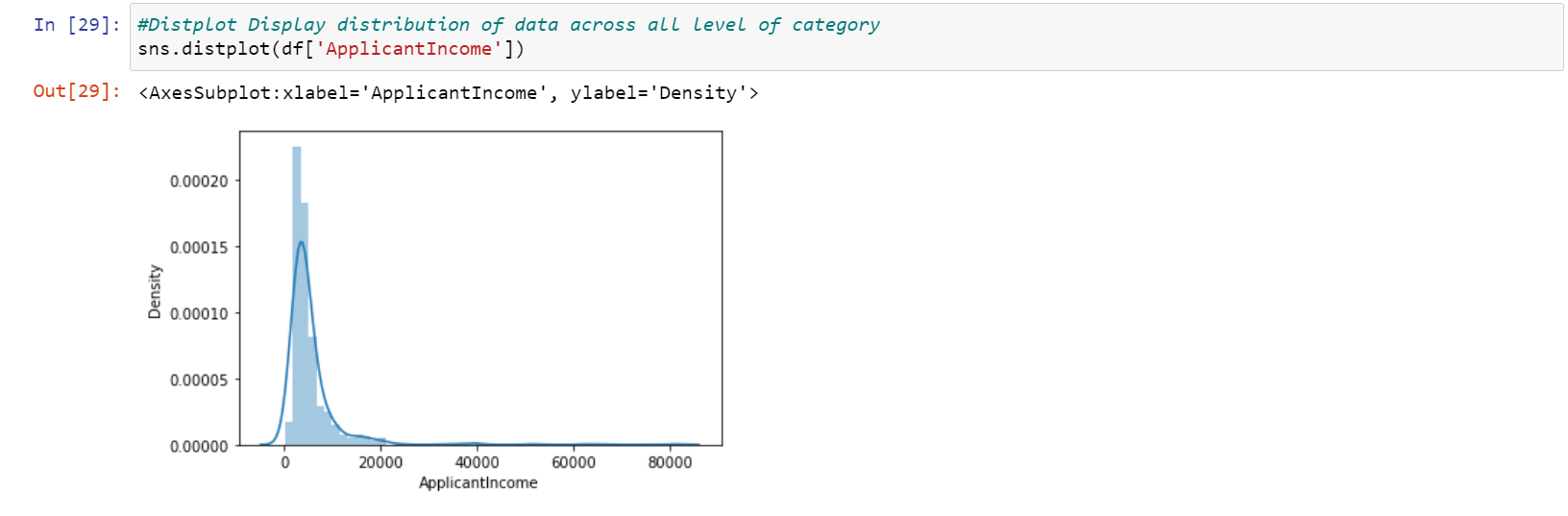
**Graphical Representation Conclusion**

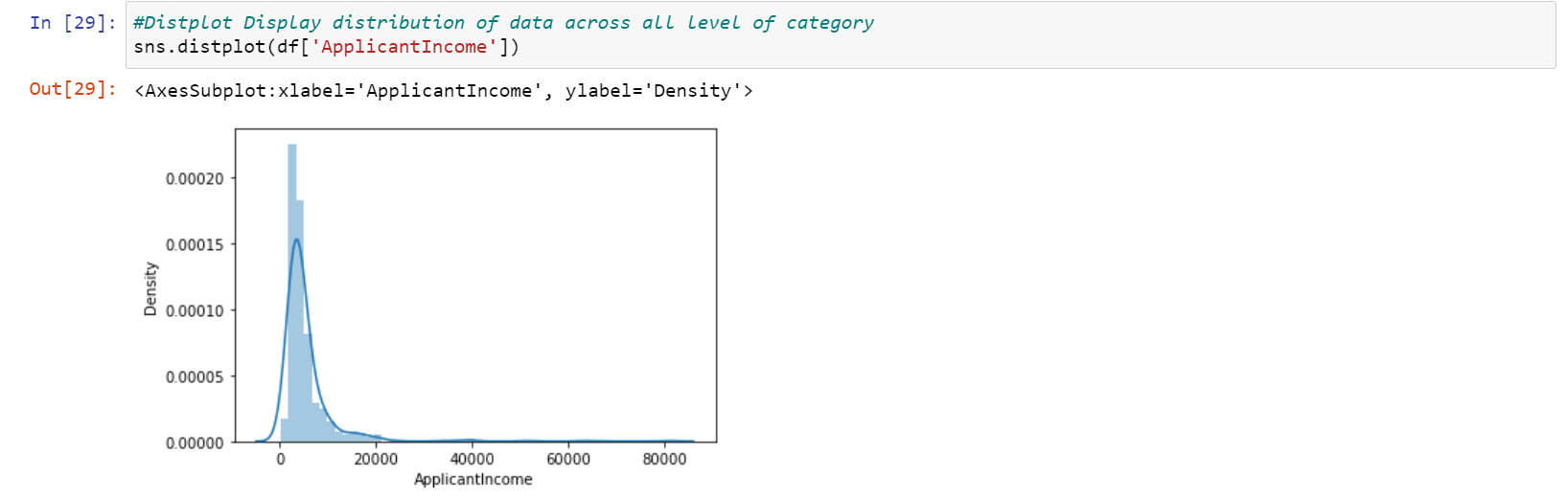
Finally, loan status will be concluded from credit history. Whether the loan is to be approved or rejection.

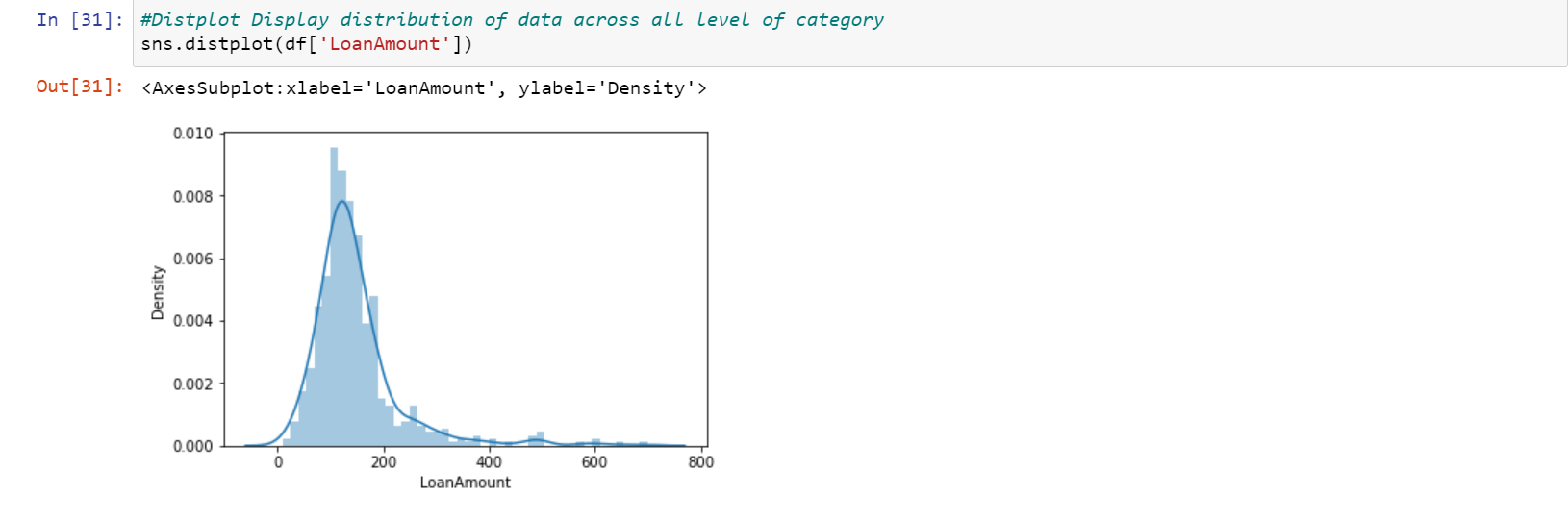
### **Independent Variable (Numerical)**

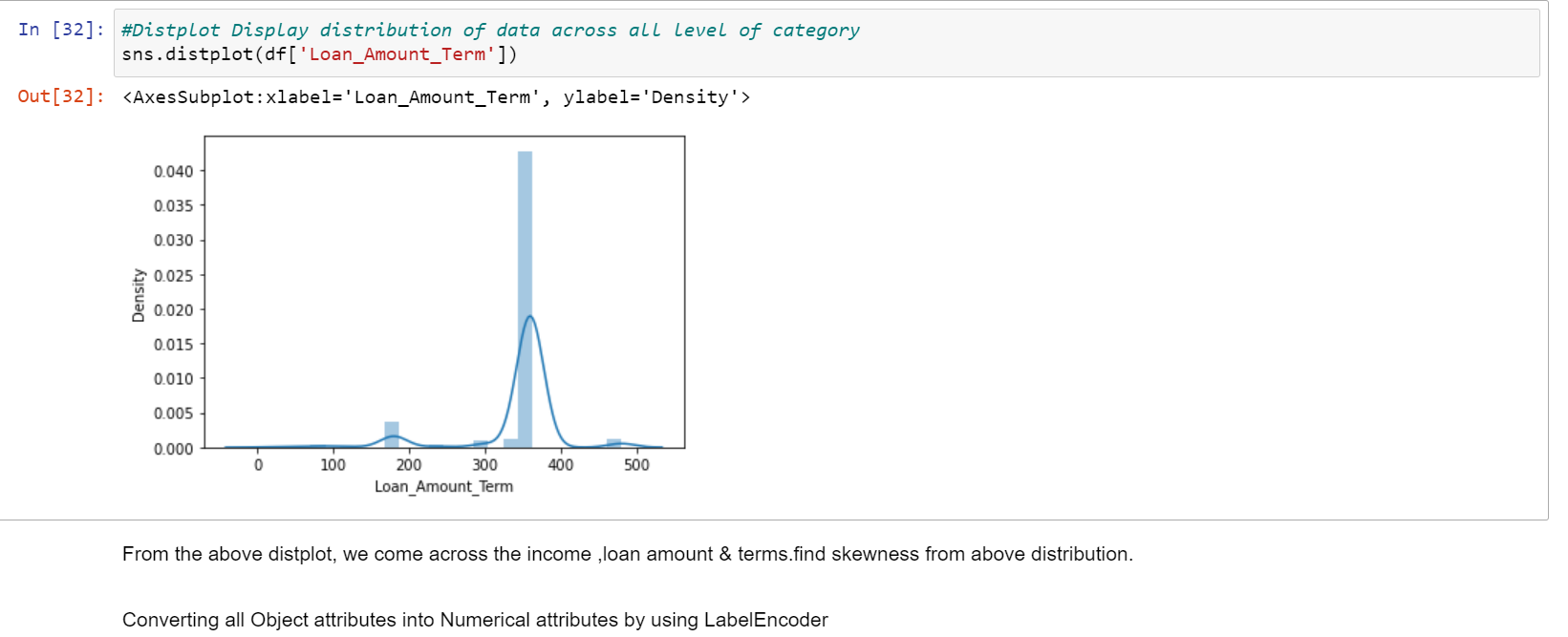
There are 4 features that are Numerical (ApplicantIncome, CoapplicantIncome, Loan\_Amount,

Loan\_Amount\_Term)

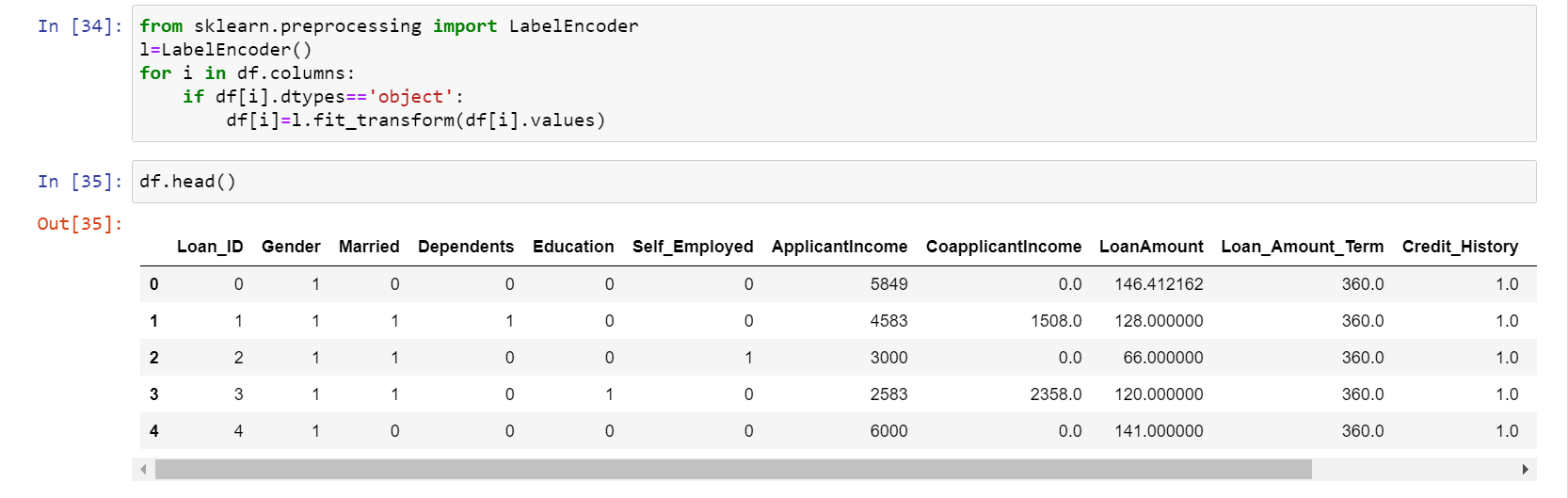






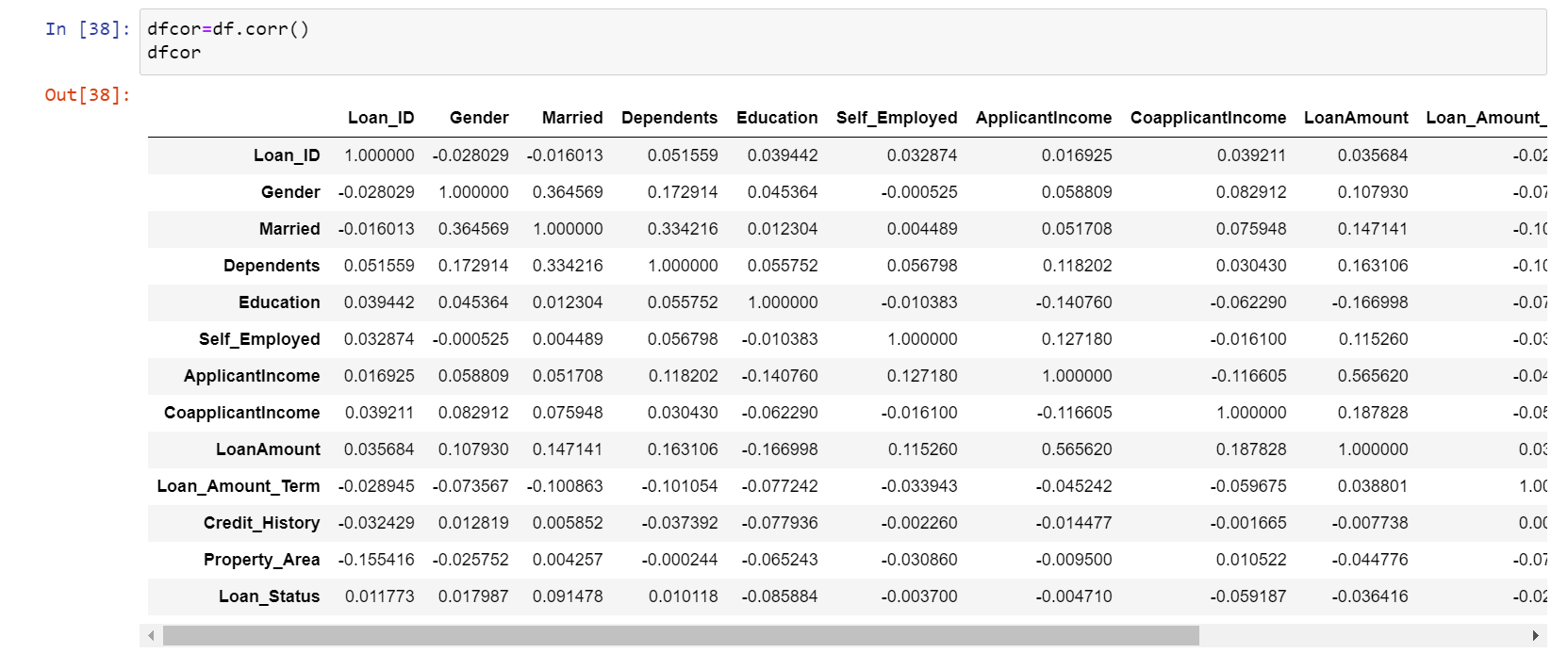


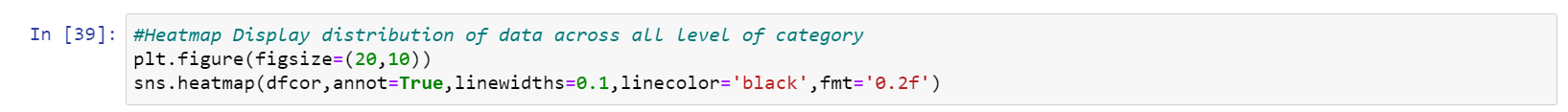
Converting all object Attributes into Numerical Attributes by using Label Encoder



# Correlation Matrix

# Now let’s 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 colouring. The variables with darker colour mean their correlation is more.







**Note:**We see that the most correlated variables are

* (ApplicantIncome - LoanAmount) with correlation coefficient of 0.57
* (Credit\_History - Loan\_Status) with correlation coefficient of 0.56
* LoanAmount is also correlated with CoapplicantIncome with correlation coefficient of 0.19.

**Pre-Processing Pipeline.**

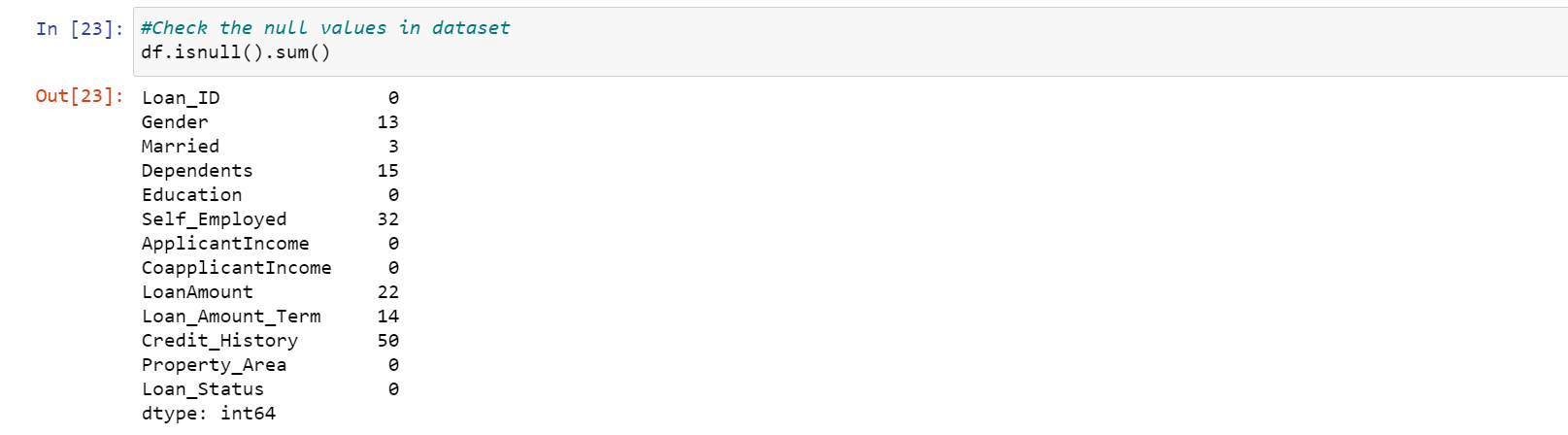
Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviours or trends, and is likely to contain many errors. Data pre-processing is a method of resolving such issues.

## 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.

### Missing value imputation

Let’s list out feature-wise count of missing values.

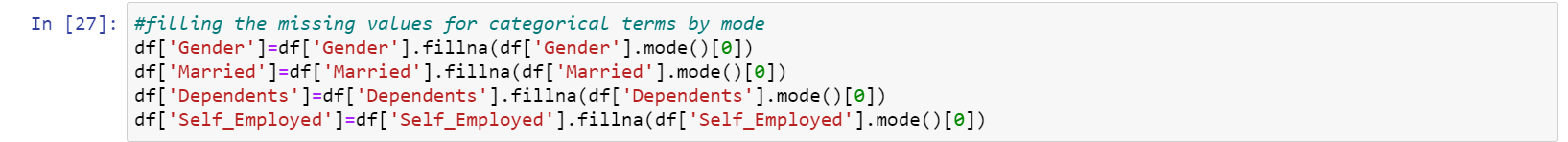
****

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.

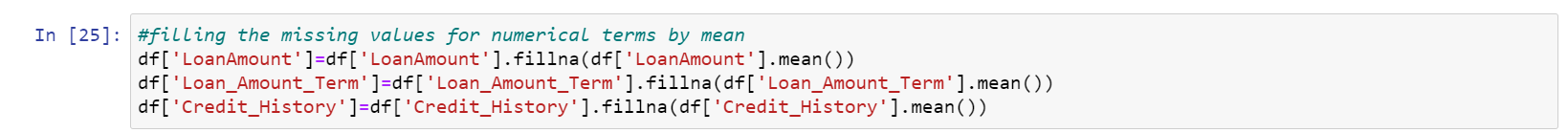
We can consider these methods to fill the missing values:

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

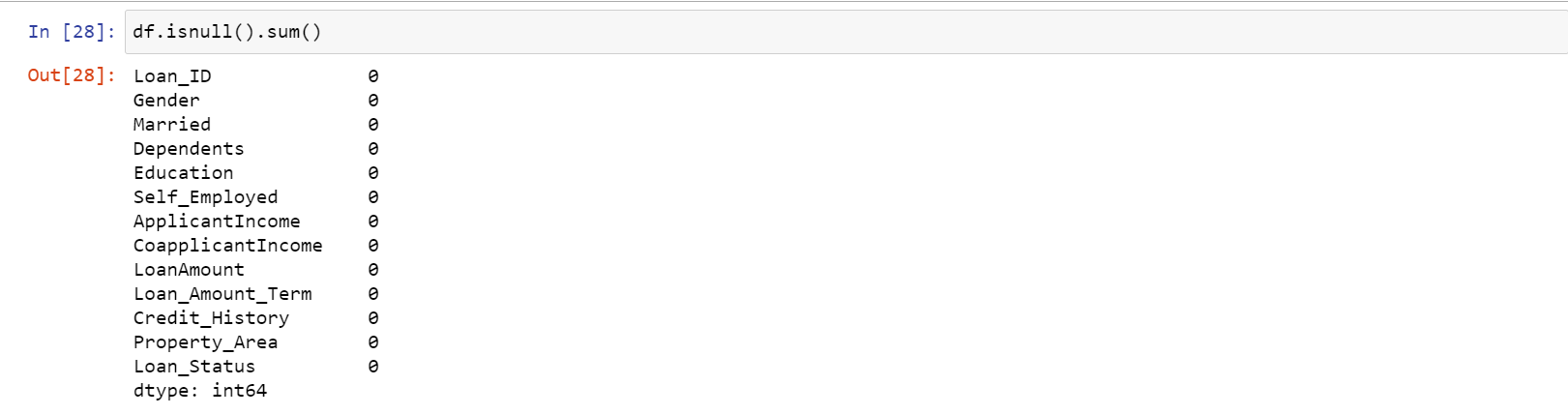
There are very less missing values in Gender, Married, Dependents and Self\_Employed features so we can fill them using the mode of the features. If an independent variable in our dataset has huge amount of missing data e.g. 80% missing values in it, then we would drop the variable from the dataset.

****

Now Filling the numerical variables like Loan Amount,Loan\_Amount\_Term and Credit\_History.

****

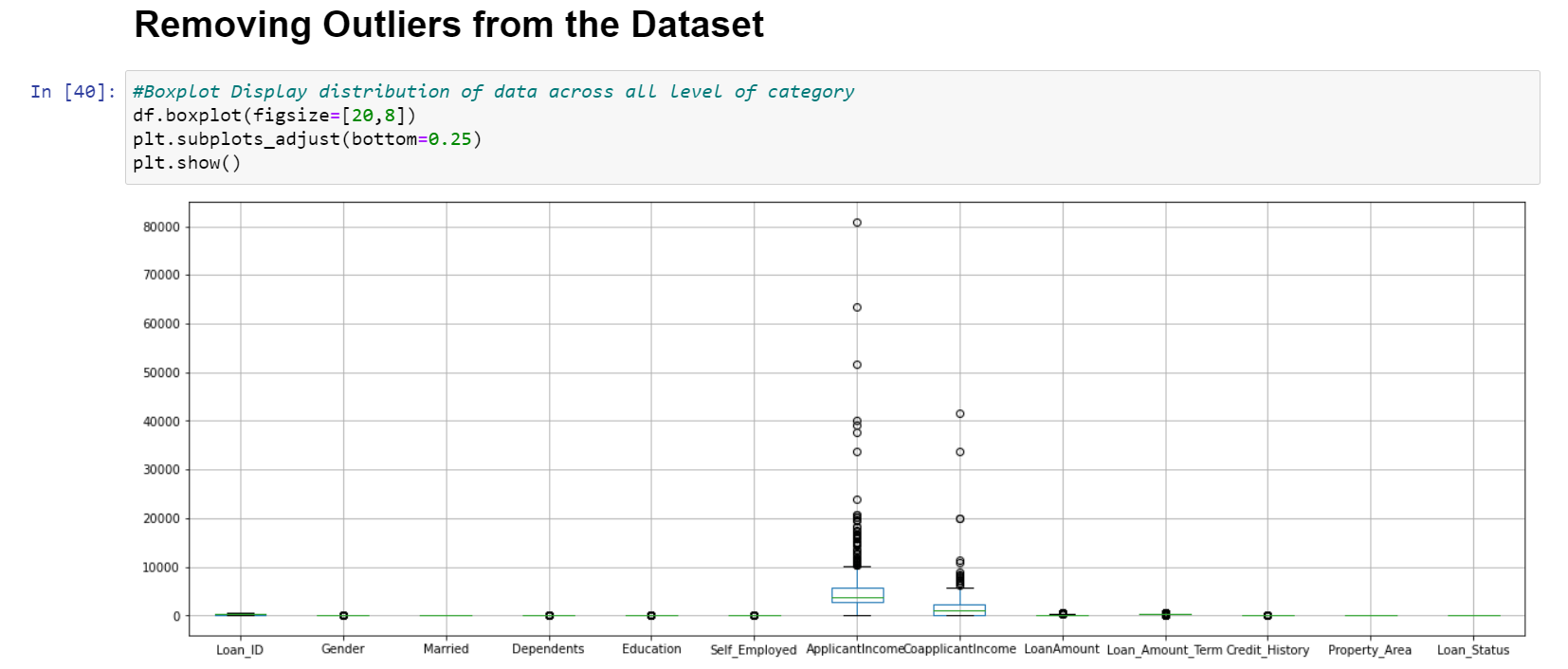
After filling the missing values in the Dataset.



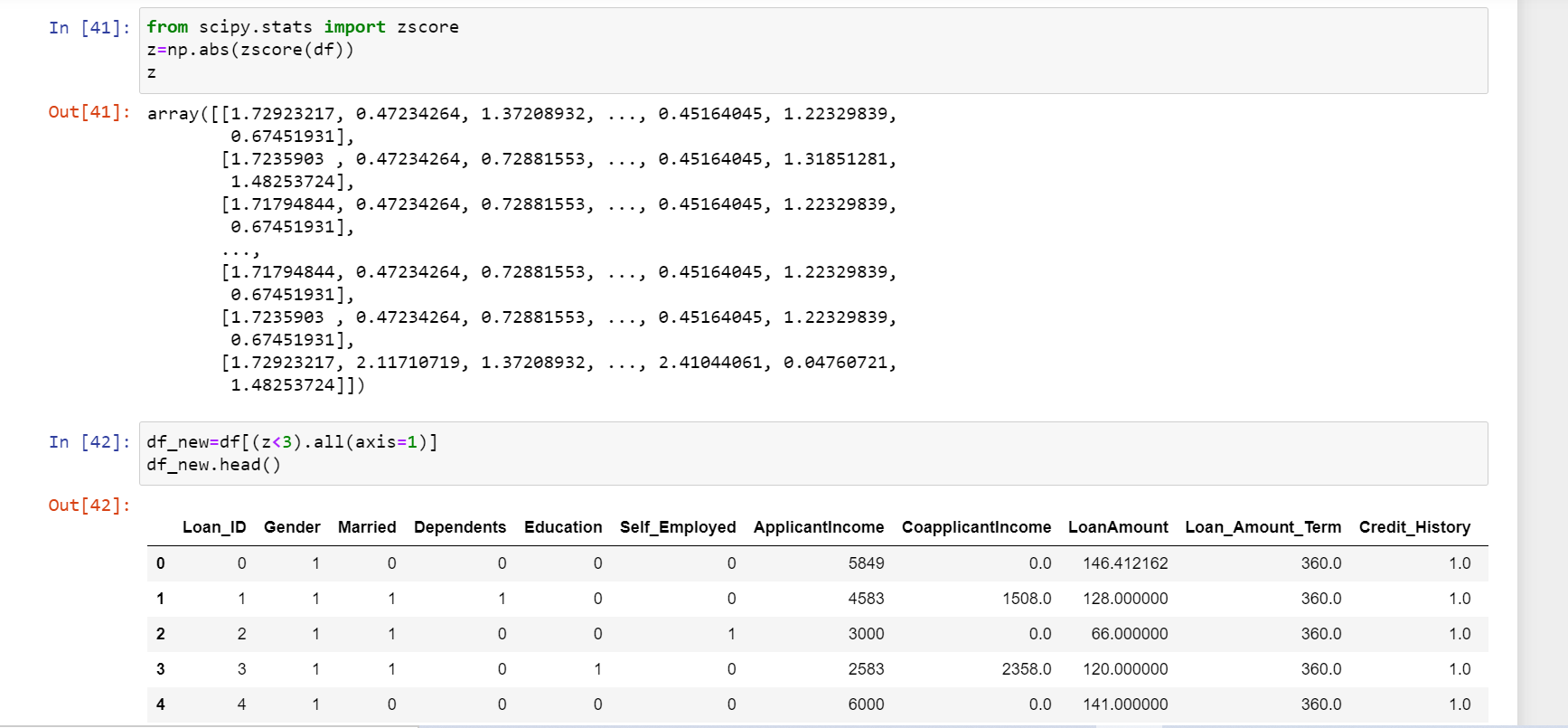
### Outlier Treatment

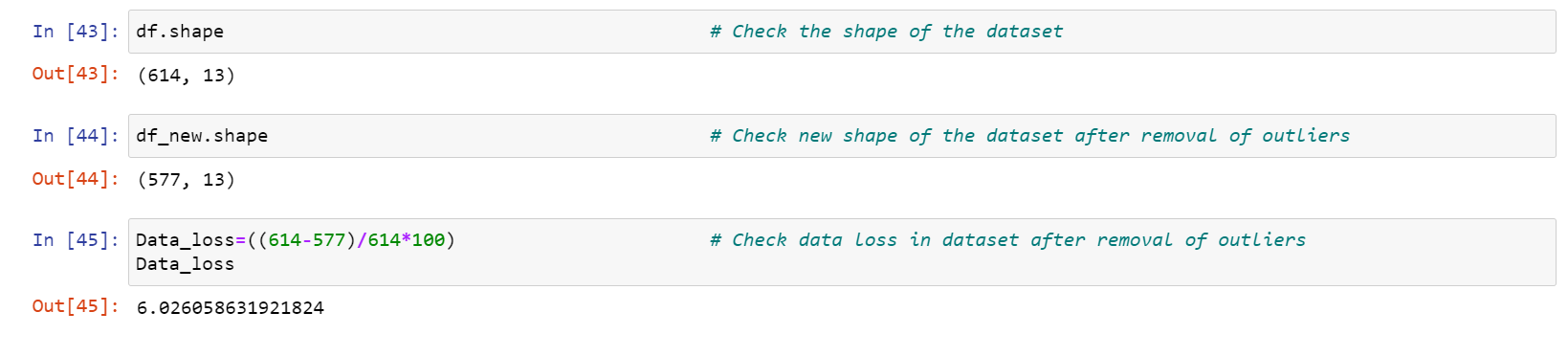
As we saw huge difference in the mean and standard deviation in the dataset, LoanAmount contains outliers so we have to treat them as the presence of outliers affects the distribution of the data. Having outliers in the dataset often has a significant effect on the mean and standard deviation and hence affecting the distribution. We must take steps to remove outliers from our data sets.

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 power transformation. As we take the power transformation, it does not affect the smaller values much, but reduces the larger values. So, we get a distribution similar to normal distribution.



Now by seeing the above visualization we came across the outliers are in few attributes. So, we are removing the outlier’s by using **Z-Score method**.





Above we can see the Data Loss of the Dataset its around 6.02%. Hence outliers are removed from the Dataset.

# To check distribution of Skewness

# 

# From above visualization we can see skewness of the dataset were few attributes found with skewness. Hence, removing skewness by using Power Transform Method.

# 

# Building Machine Learning Models

## 

## Evaluation Metrics for Classification Problems

The process of model building is not complete without evaluation of model’s performance. Suppose we have the predictions from the model, how can we decide whether the predictions are accurate? We can plot the results and compare them with the actual values, i.e., calculate the distance between the predictions and actual values. Lesser this distance more accurate will be the predictions. Since this is a classification problem, we can evaluate our models using any one of the following evaluation metrics:

**Accuracy**: Let us understand it using the confusion matrix which is a tabular representation of Actual vs Predicted values. This is how a confusion matrix looks like:

|  |
| --- |
|  |

True Positive - Targets which are actually true(Y) and we have predicted them true(Y)

True Negative - Targets which are actually false(N) and we have predicted them false(N)

False Positive - Targets which are actually false(N) but we have predicted them true(T)

False Negative - Targets which are actually true(T) but we have predicted them false(N)

Using these values, we can calculate the accuracy of the model. The accuracy is given by:

*Accuracy = (TP+TN) / (TP+TN+FP+FN)*

**Precision**: It is a measure of correctness achieved in true prediction i.e., of observations labelled as true, how many are actually labelled true.

*Precision = TP / (TP + FP)*

**Recall (Sensitivity)** - It is a measure of actual observations which are predicted correctly i.e., how many observations of true class are labelled correctly. It is also known as ‘Sensitivity’. E.g., Proportion of patients with a disease who test positive.

*Recall = TP / (TP + FN)*

**Specificity** - It is a measure of how many observations of false class are labelled correctly. E.g., Proportion of patients without the disease who test negative.

*Specificity = TN / (TN + FP)*

Specificity and Sensitivity plays a crucial role in deriving ROC curve.

**ROC curve**

* Receiver Operating Characteristic (ROC) summarizes the model’s performance by evaluating the trade-offs between true positive rate (Sensitivity) and false positive rate (1- Specificity).
* The area under curve (AUC), referred to as index of accuracy(A) or concordance index, is a perfect performance metric for ROC curve. Higher the area under curve, better the prediction power of the model.
* The area of this curve measures the ability of the model to correctly classify true positives and true negatives. We want our model to predict the true classes as true and false classes as false.
* So it can be said that we want the true positive rate to be 1. But we are not concerned with the true positive rate only but the false positive rate too. For example in our problem, we are not only concerned about predicting the Y classes as Y but we also want N classes to be predicted as N.
* We want to increase the area of the curve which will be maximum for class 2,3,4 and 5 in the above example.
* For class 1 when the false positive rate is 0.2, the true positive rate is around 0.6. But for class 2 the true positive rate is 1 at the same false positive rate. So, the AUC for class 2 will be much more as compared to the AUC for class 1. So, the model for class 2 will be better.
* The class 2,3,4 and 5 model will predict more accurately as compared to the class 0 and 1 model as the AUC is more for those classes.

This is how a ROC curve looks like:

|  |
| --- |
|  |

# At the competition’s page, it has been mentioned that our submission data would be evaluated based on the accuracy. Hence, we will use accuracy as our evaluation metric.

# Classification Model

## **Model Building: Logistic Regression**

Let us make our first model to predict the target variable. We will start with Logistic Regression which is used for predicting binary outcome.

* Logistic Regression is a classification algorithm. It is used to predict a binary outcome (1 / 0, Yes / No, True / False) given a set of independent variables.
* Logistic regression is an estimation of Logit function. Logit function is simply a log of odds in favor of the event.
* This function creates a s-shaped curve with the probability estimate, which is very similar to the required step wise function

# 

### **Decision Tree**

Decision tree is a type of supervised learning algorithm (having a pre-defined target variable) that is mostly used in classification problems. In this technique, we split the population or sample into two or more homogeneous sets (or sub-populations) based on most significant splitter / differentiator in input variables.

Decision trees use multiple algorithms to decide to split a node in two or more sub-nodes. The creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, we can say that purity of the node increases with respect to the target variable.

# 

### **Random Forest**

* Random Forest is a tree based bootstrapping algorithm wherein a certain no. of weak learners (decision trees) is combined to make a powerful prediction model.
* For every individual learner, a random sample of rows and a few randomly chosen variables are used to build a decision tree model.
* Final prediction can be a function of all the predictions made by the individual learners.
* In case of regression problem, the final prediction can be mean of all the predictions.

There are some parameters worth exploring with the sklearn Random Forest Classifier:

* n\_estimators
* max\_features

n\_estimators = As usually bigger the forest the better, there is small chance of overfitting here. The more estimators you give it, the better it will do. We will use the default value of 10.

max depth of each tree (default none, leading to full tree) - reduction of the maximum depth helps fighting with overfitting. We will limit at 10.

# 

# SVC Model

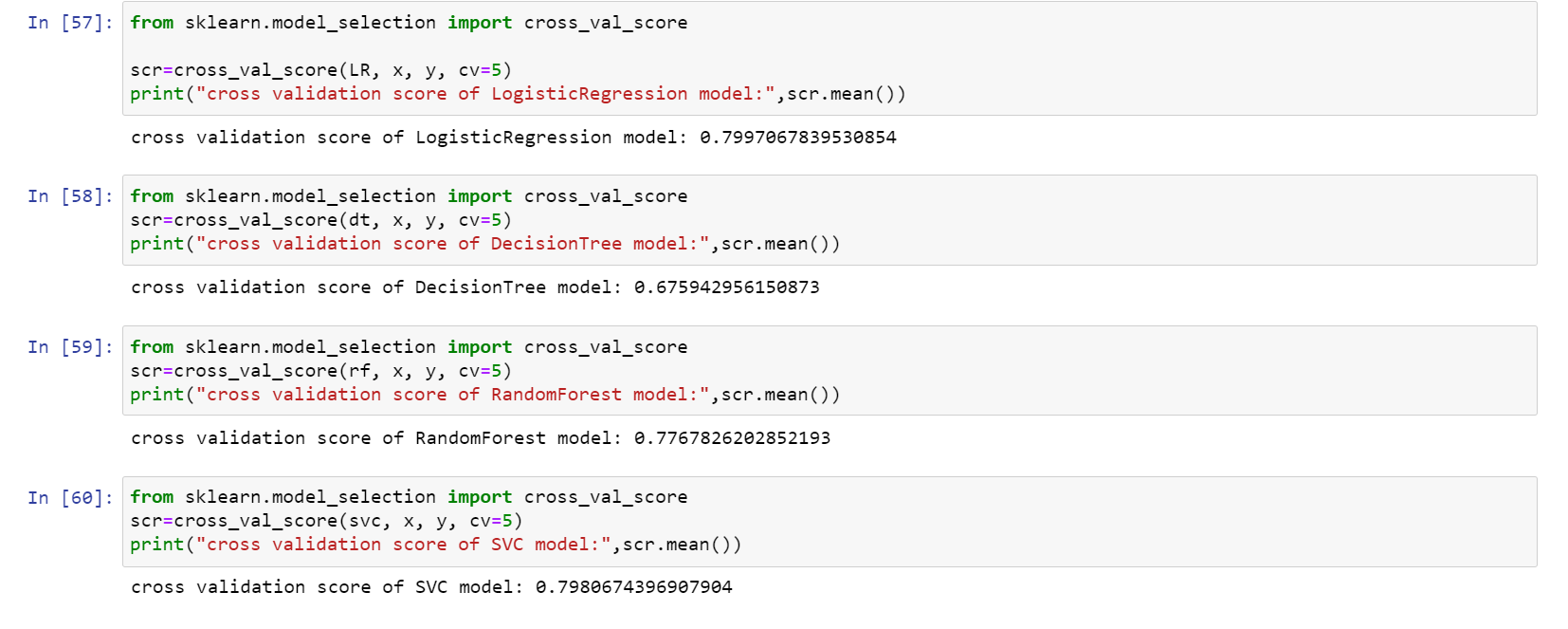
# The objective of a Linear SVC (Support Vector Classifier) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is.

# 

We are getting highest accuracy with Logistic Regression (82%). but it can be due to overhitting also so we well check cross validation scores.

# Cross Validation Scores

# 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.



Min difference in accuracy and cross validation score is for Decision Tree Classifier model.so this is our best model.

### **Grid Search CV**

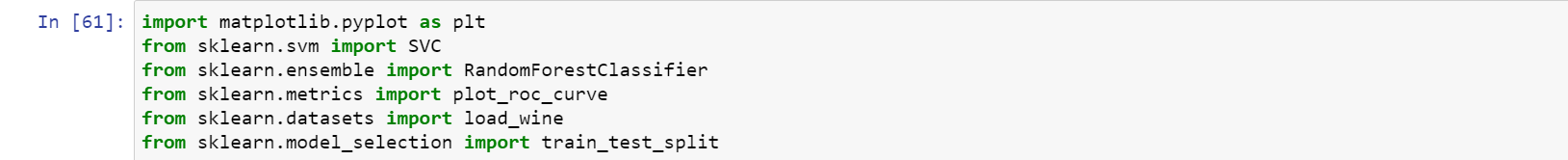
We will try to improve the accuracy by tuning the hyperparameters for this model. We will use grid search to get the optimized values of hyper parameters. Grid Search is a way to select the best of a family of hyper parameters, parametrized by a grid of parameters.

We will use Grid Search CV in sklearn.model\_selection for an exhaustive search over specified parameter values for an estimator. Grid Search CV implements a “fit” and a “score” method. It also implements “predict”, “predict\_proba”, “decision\_function”, “transform” and “inverse\_transform” if they are implemented in the estimator used.

The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid, hence Grid Search CV. More info at <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>

We will tune the max\_depth and n\_estimators parameters. max\_depth decides the maximum depth of the tree and n\_estimators decides the number of trees that will be used in random forest model.

# Hyper parameter Tuning

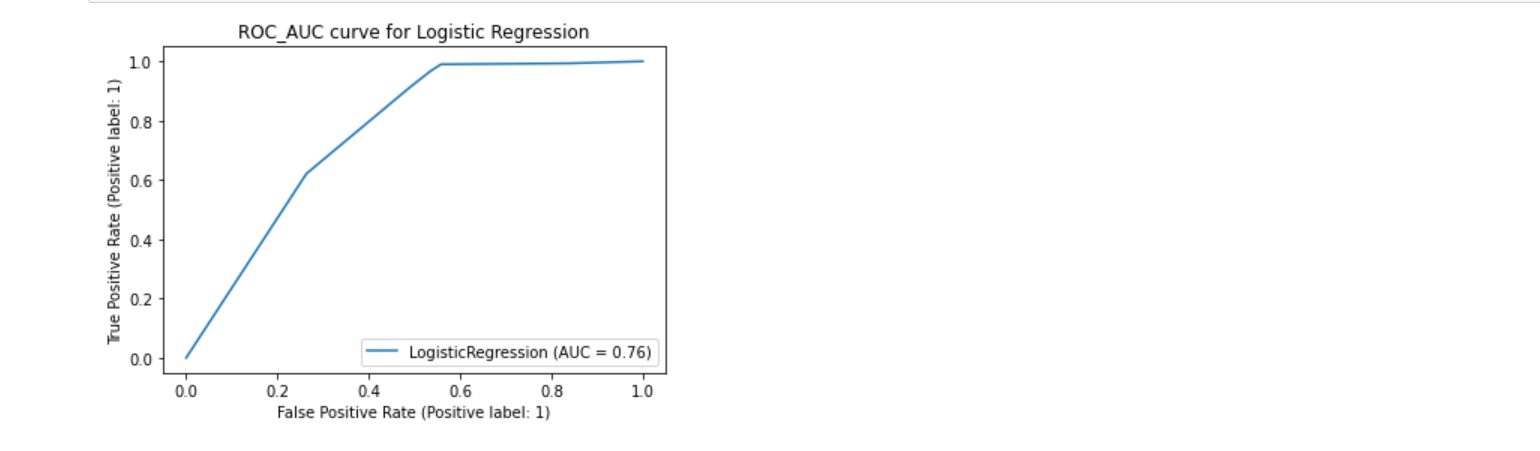


**Hyperparameter tuning for Logistic Regression Model**

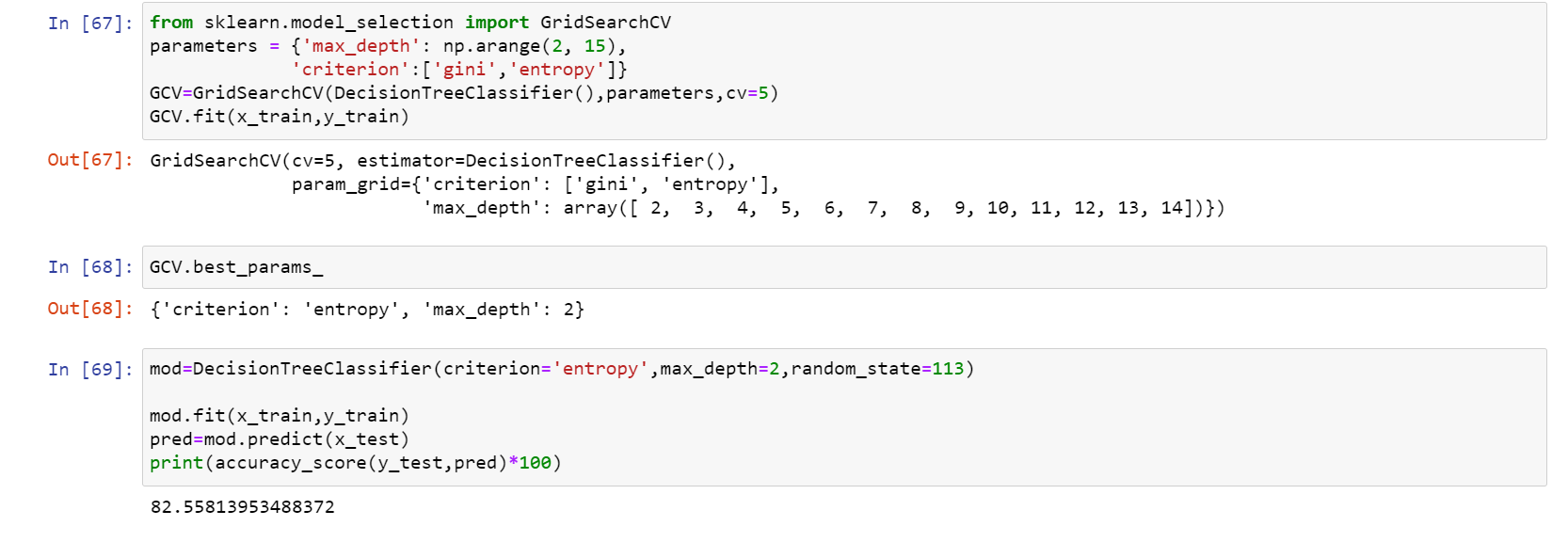


**ROC\_AUC Score for Logistic Regressor Model**

****

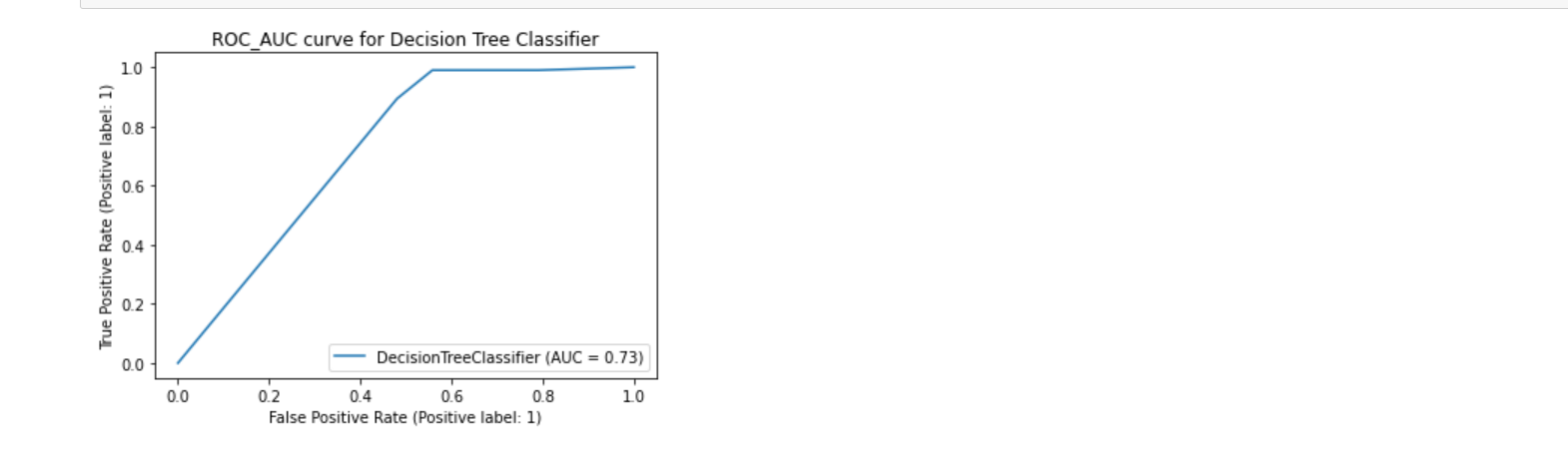
****

**Hyperparameter tuning for Decision Tree Classifier Model**

****

**ROC\_AUC Score for Decision Tree Classifier Model**

****

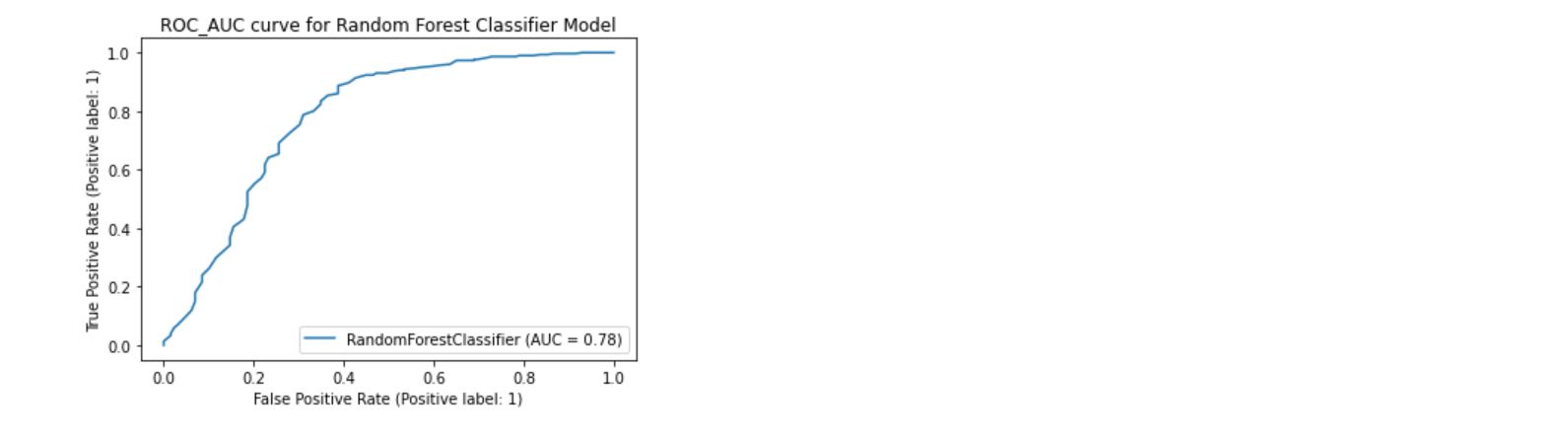
****

**Hyperparameter tuning for Random Forest Classifier Model**

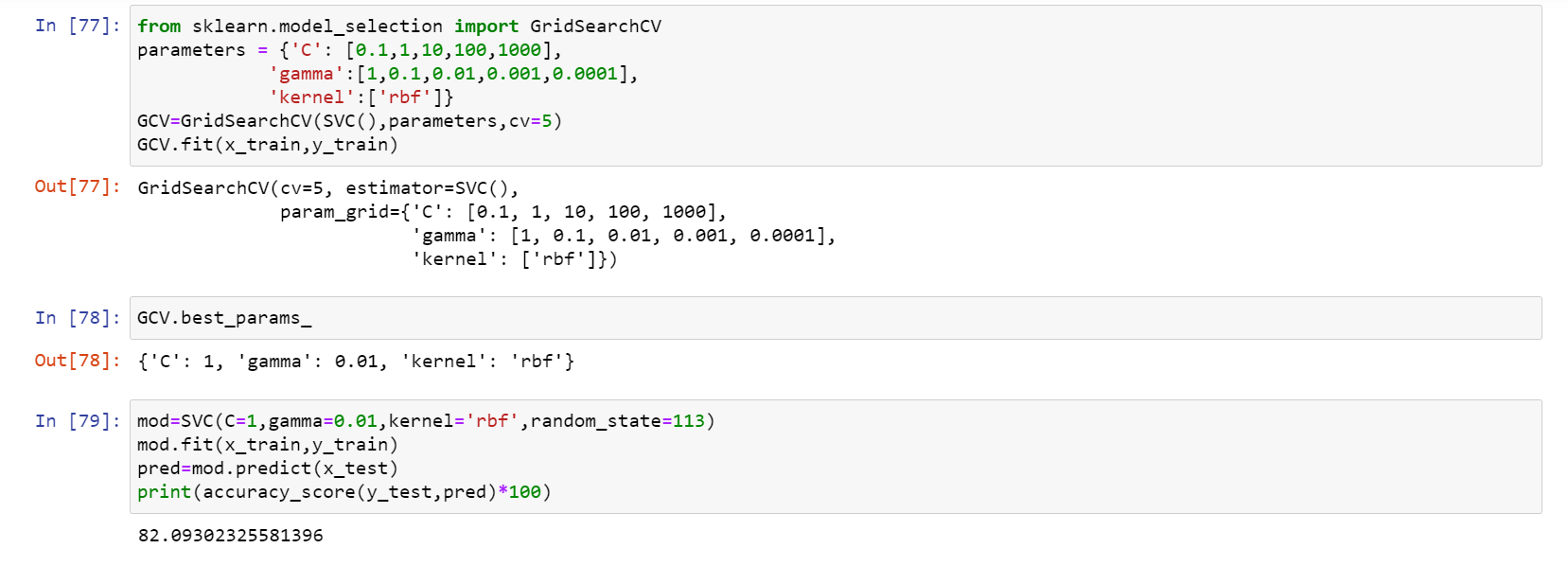


**ROC\_AUC Score for Random Forest Classifier Model**

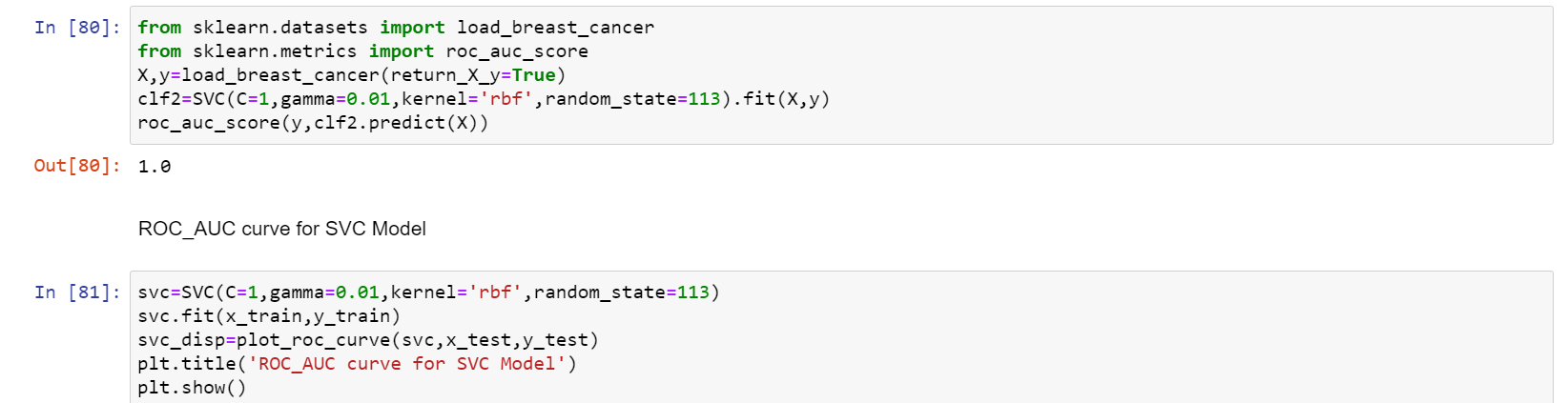
****

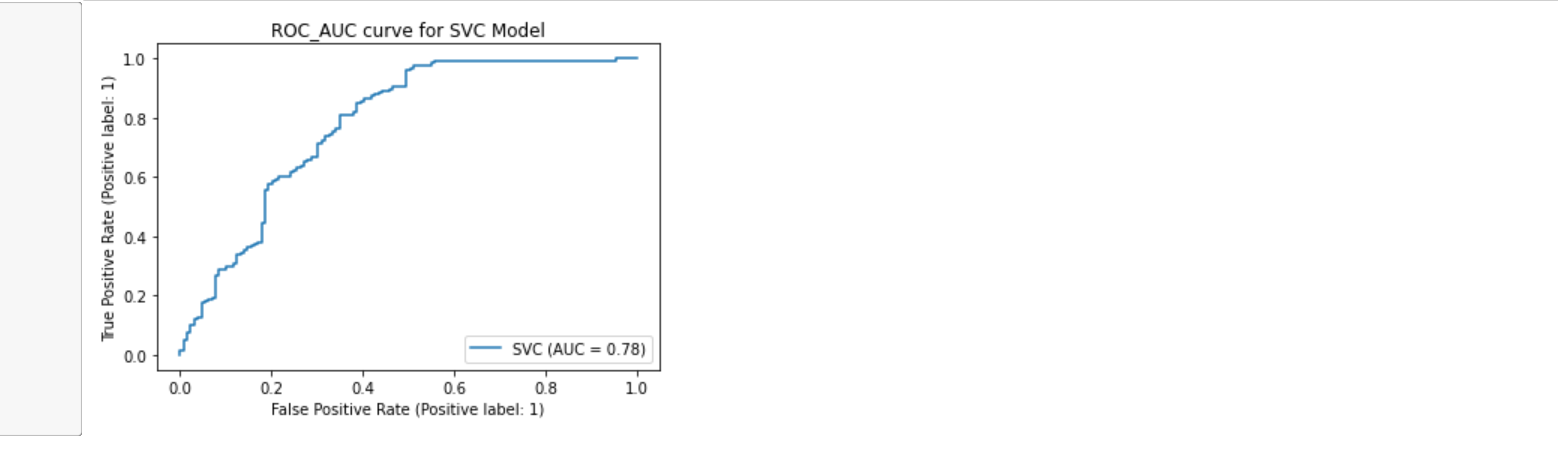
****

**Hyperparameter tuning for SVC model**

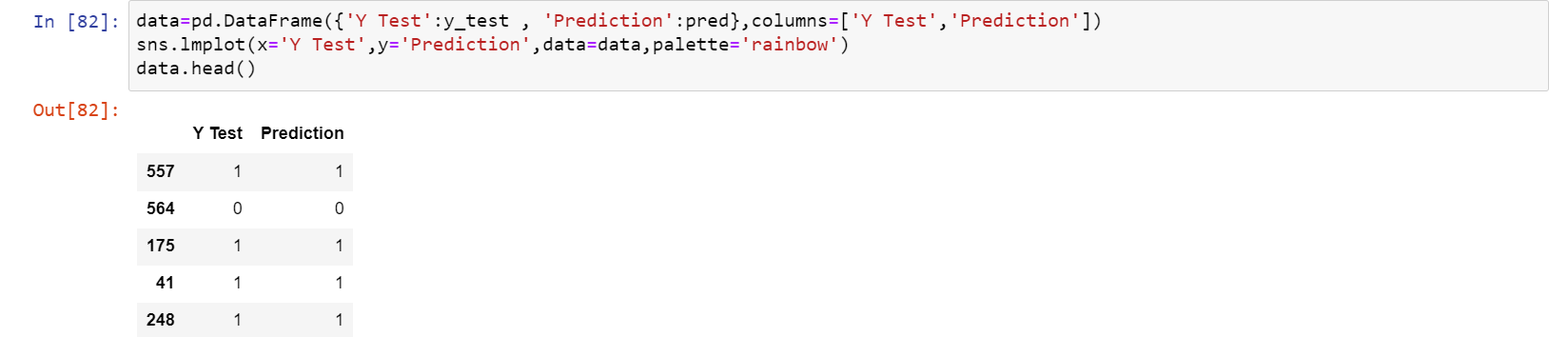
****

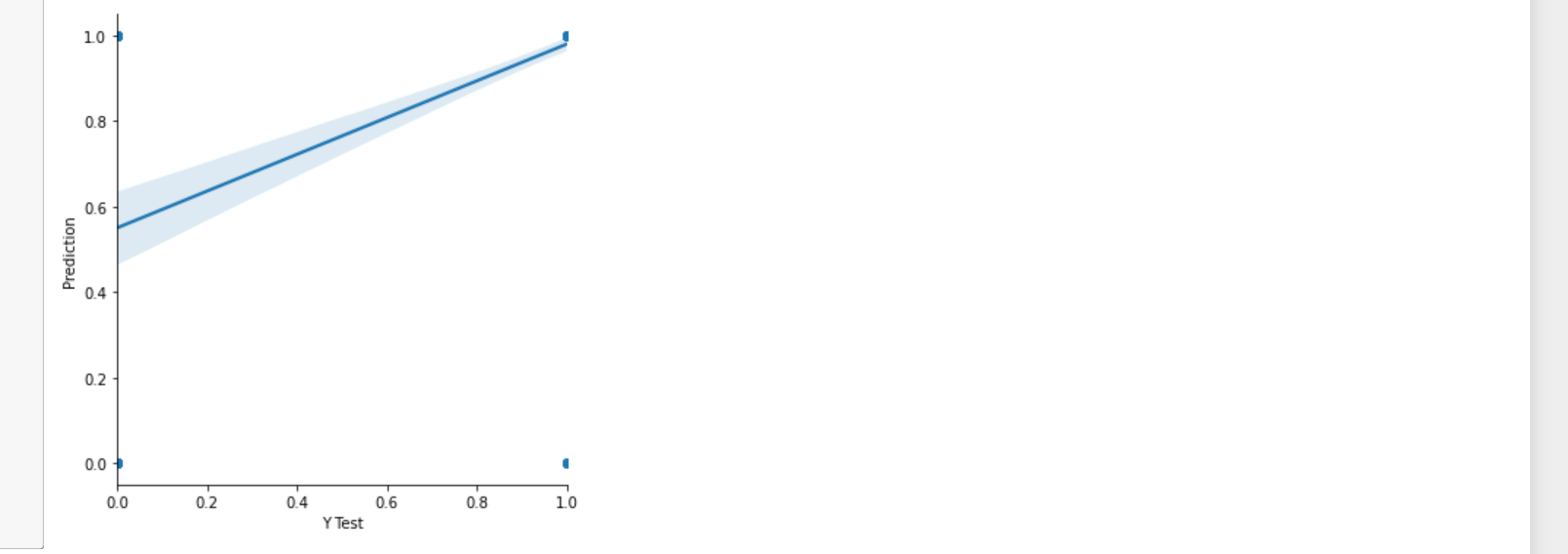
**ROC\_AUC Score for SVC Model**

****

****

**Compare with Actual Final Vs Sample Prediction**

****

****

# Model Saving

****

# Conclusion

After trying and testing 4 different algorithms, the best accuracy on the leader board is achieved by Logistic Regression (0.8255) and Decision Tree (0.8255), Followed by SVC (0.8209) and Random Forest performed the worst (0.8023). While new features created via feature engineering helped in predicting the target variable, it did not improve the overall model accuracy much. Compared to using default parameter values, Grid Search CV helped improved the model's mean validation accuracy by providing the optimized values for the model's hyperparameters. On the whole, 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. In this case, however, a basic Logistic Regression has already outperformed other more complex algorithms like Random Forest and SVC, for the given dataset.

**Suggestions for Improvement**. There are many things that can be tried to improve the models’ predictions. We can create and add more variables, try different models with different subset of features and/or rows, etc. Some of the ideas are listed below:

* Combine the applicants with 1,2,3 or more dependents and make a new feature as discussed in the EDA part.
* Make independent vs independent variable visualizations to discover some more patterns.
* Arrive at the EMI using a better formula which may include interest rates as well.
* Try ensemble modelling (combination of different models). More about ensemble techniques can be found at the references.
* Try neural network using TensorFlow or PyTorch.

****

\*\*\*\*\*\*