Loan Approval Prediction Machine Learning

Introduction

In this article, we are going to solve the Loan Approval Prediction. This is a classification problem in which we need to classify whether the loan will be approved or not. classification refers to a predictive modeling problem where a class label is predicted for a given example of input data. A few examples of classification problems are Spam Email detection, Cancer detection, Sentiment Analysis, etc.

**Table of Content**

1. Understanding the problem statement
2. About the dataset
3. Load essential Python Libraries
4. Load Training/Test datasets
5. Data Pre-processing
6. Exploratory data analysis (EDA).
7. Feature Engineering.
8. Build Machine Learning Model
9. Make predictions on the test datasets

**Understanding the Problem Statement**

Dream Housing Finance company deals in all kinds of home loans. They have a presence across all urban, semi-urban and rural areas. The customer first applies for a home loan and after that, the company validates the customer eligibility for the loan.

The company wants to automate the loan eligibility process (real-time) based on customer detail provided while filling out online application forms. These details are Gender, Marital Status, Education, number of Dependents, Income, Loan Amount, Credit History, and others.

To automate this process, they have provided a dataset to identify the customer segments that are eligible for loan amounts so that they can specifically target these customers.

## 

As mentioned above this is a Binary Classification problem in which we need to predict our Target label which is “Loan Status”.

Loan status can have two values: Yes or NO.

Yes: if the loan is approved

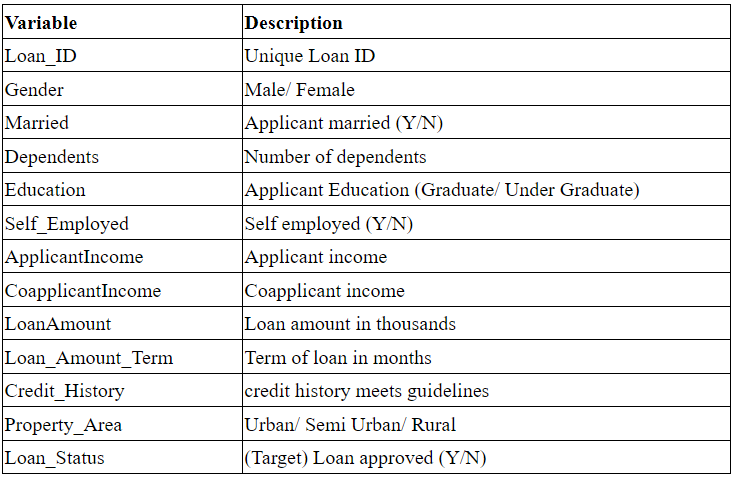
NO: if the loan is not approved

So using the training dataset we will train our model and try to predict our target column that is “Loan Status” on the test dataset.

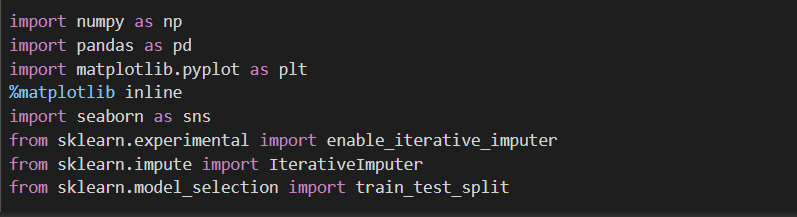
**About the dataset**

So train and test dataset would have the same columns except for the target column that is “Loan Status”.

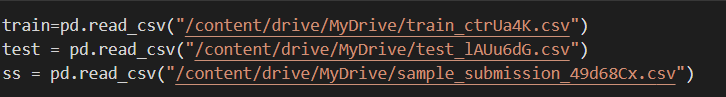
Train dataset:



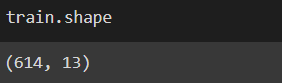
**Load Essential Python Libraries**

****

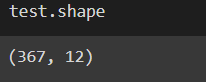
**Load Training/ Test Dataset**

****

**Size of Train/Test Data**

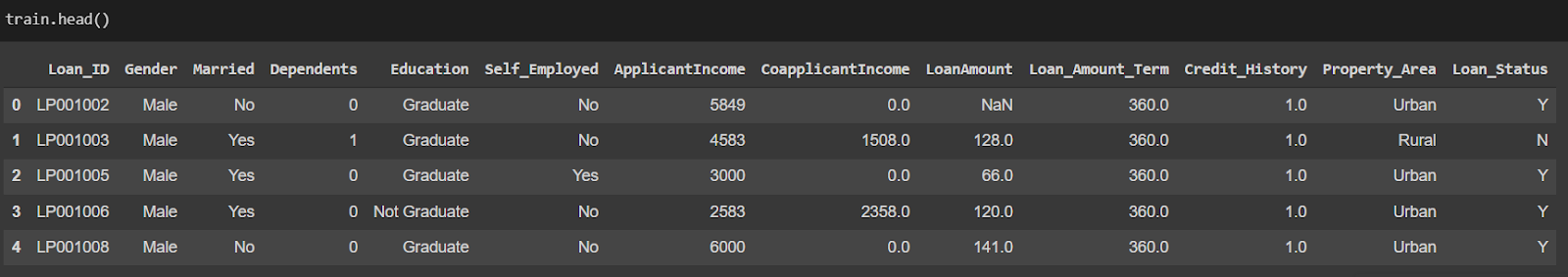
****

So we have 614 rows and 13 columns in our training dataset.

****

In test data, we have 367 rows and 12 columns because the target column is not included in the test data.

**First look at the Dataset**

****

Categorical Columns: Gender (Male/Female), Married (Yes/No), Number of dependents (Possible values:0,1,2,3+), Education (Graduate / Not Graduate), Self-Employed (No/Yes), credit history(Yes/No), Property Area (Rural/Semi-Urban/Urban) and Loan Status (Y/N)(i. e. Target variable)

Numerical Columns: Loan ID, Applicant Income, Co-applicant Income, Loan Amount, and Loan amount term

**Data Pre-processing**

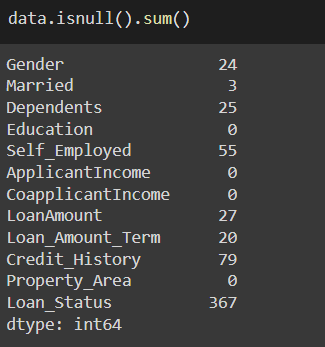
Concatenating the train and test data for data pre-processing:

https://lh4.googleusercontent.com/3AcZVHDOmprz8vvX8ccHJQYNDHp-IsRSzQm5SVvLrOS47XRfoFr3DVsXl25nuQviDggaLFLcca7ikyepJRWxz7UMVvJDzMlEPJBhsvlxGNGOGf5twrX2XiKGqbqO4_X4Rj6TyiP1

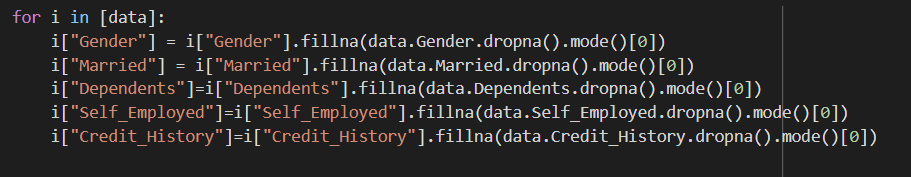
dropping the unwanted column :

https://lh4.googleusercontent.com/q2cJSC0YCe-qkgYDkGwfYYR7GHAJ3ljIe43GBHp-xAP2szVfbWMDvgdUyW4cbdOlJgJvcPJsg1wqVvnIloBYLRGO-phNvhxcQv226BbcEw-2sZaC6ASKQ40EDZ22GzUiJMiwa_wL

Identify missing values :

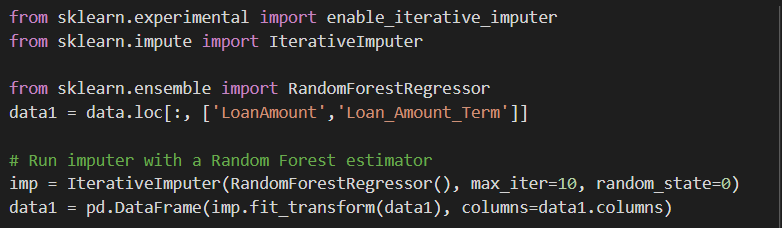


**Imputing the missing values:**

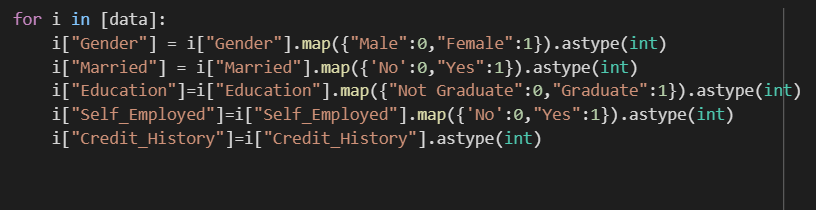
****

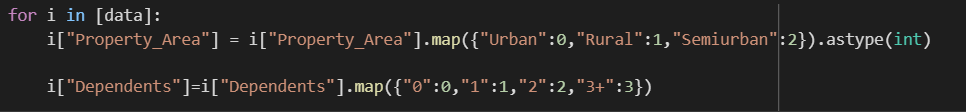
* Fill null values with mode

Next, we will be using Iterative imputer for filling missing values of LoanAmount and Loan\_Amount\_Term



So now as we have imputed all the missing values we go on to mapping the categorical variables with the integers.

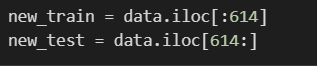




We map the values so that we can input the train data into the model as the model does not accept any string values.

**Exploratory Data Analysis (EDA)**

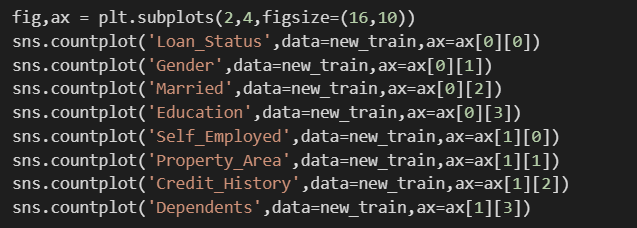
Splitting the data to new\_train and new\_test so that we can perform EDA.



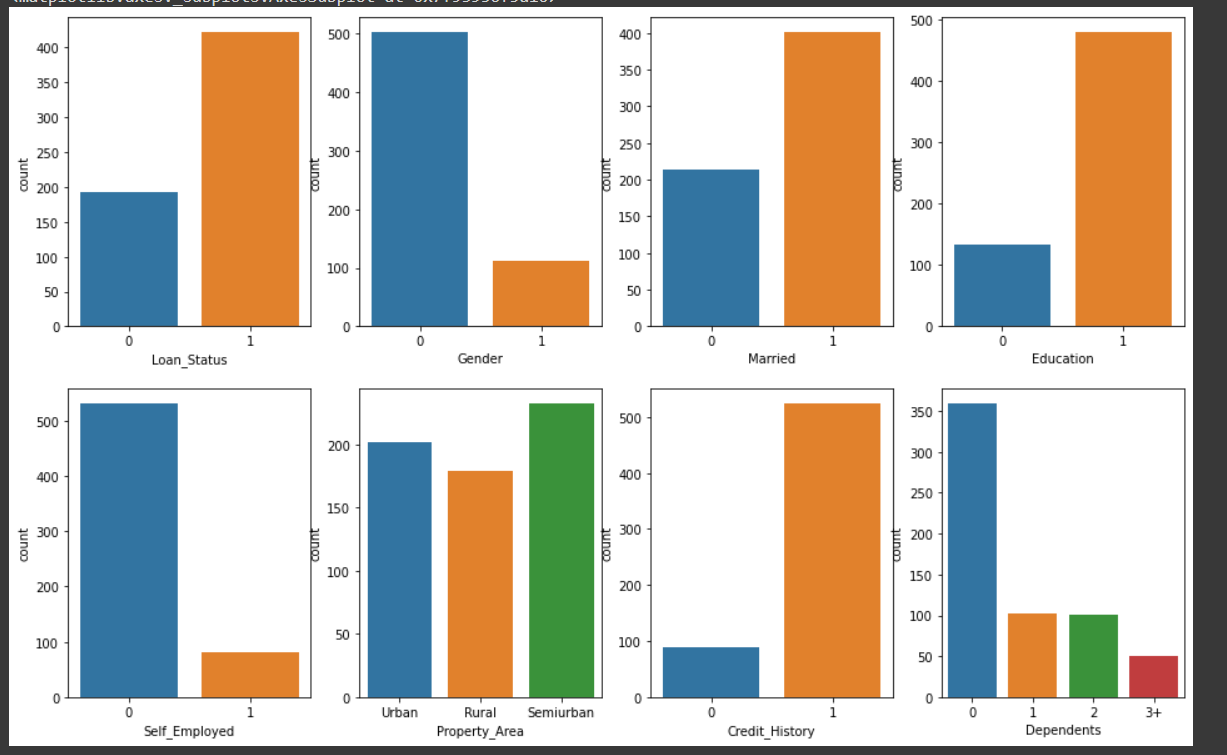
Mapping ‘N’ to 0 and ‘Y’ to 1

https://lh6.googleusercontent.com/TDT5hh7_sIWbDaNFQxdBZ48jKOuJRCZTTy7CNBgXg4cgDe_o1Poe5aZ3NeRtbHwZKPGV5rnzB69thUlRS4MBSGF46I39KWXYqnz8B9QJVhBzsvED4Os7Ogor4yQdMAsVUQ2UpF0A

Univariate Analysis:



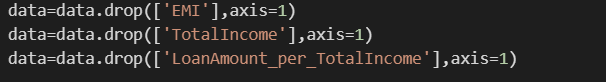
Output:



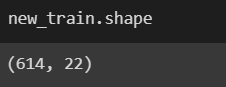
**Univariate Analysis Observations**

1. More Loans are approved Vs Rejected
2. Count of Male applicants is more than Female
3. Count of Married applicant is more than Non-married
4. Count of graduate is more than non-Graduate
5. Count of self-employed is less than that of Non-Self-employed
6. Maximum properties are located in Semiurban areas
7. Credit History is present for many applicants
8. The count of applicants with several dependents=0 is maximum.

**Drop Unwanted Column :**

****

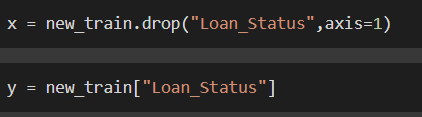
**Size after feature engineering :**

****

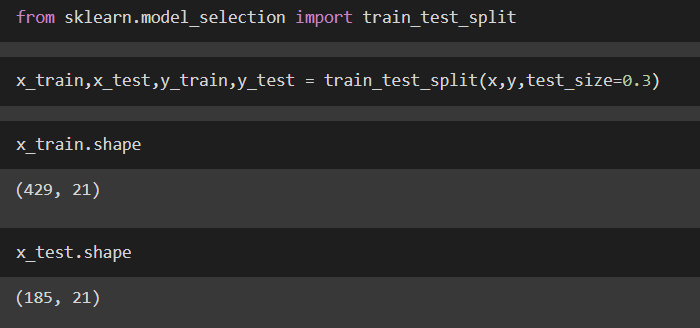
**We have added 8 new features**

**Building Machine Learning Model:**

Creating X (input variables) and Y (Target Variable) from the new\_train data.



Using train test split on the training data for validation

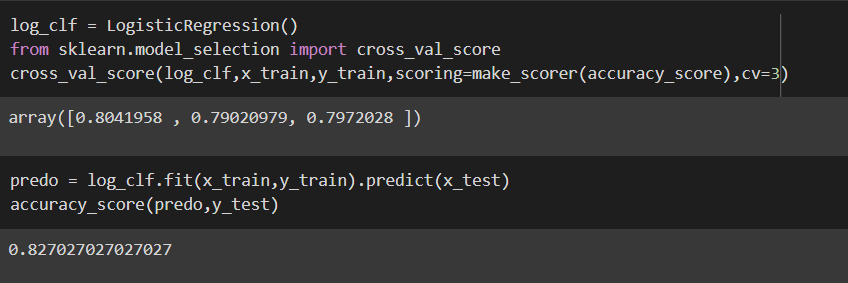


We have a (70:30) split on the training data.

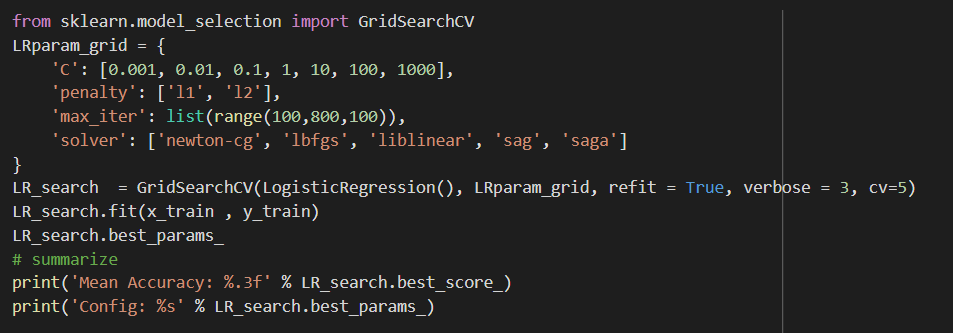
**Using ML algorithm for training**

We have used multiple algorithms for training purposes like Decision Tree, Random Forest, SVC, Logistic Regression, XGB Regressor, etc.

Among all the algorithms logistic regression performs best on the validation data with an accuracy score of **82.7%**.



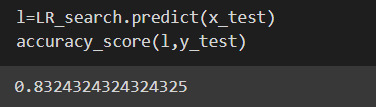
After getting an accuracy of 82.7% I tried fine-tuning it to improve my accuracy score using GridSearchCV.



The best parameters I got after fine-tuning were:

https://lh6.googleusercontent.com/78duLCH3RP2VyYRFQgDKLcCC1dwNlGGUawlDXmCy29wg41xh1ABpaasQuQfjxEyZs0N9ByjuK5NVsUimvew81fbqgNHVMmWIoVYOW-na5CkI41afhlfTR-6gBCidM4QGbKbAxer8

After fine-tuning the logistic regression model the accuracy score improved from 82.7% to 83.24%.



**Predicting on test data**

**https://lh4.googleusercontent.com/mxwWZwJQ08IWaO7SoIGr9gXtLy65Kr3GVfAjaHuYF-Mskb4FVLiwDJLVcIGZDLXnpgsRMQ35eYHsTTNsTJkfhOETzXopZ1SWz1OKH7G4Q2XFENAOJluGy7h8TUVzPrni5c8ih9Xx**

**Conclusion**

After the Final Submission of test data, my accuracy score was 78%.

Feature engineering helped me increase my accuracy.

Amazingly Logistic Regression worked better than all other Ensemble models.