**Loan Eligibility Prediction using Machine Learning Techniques**



**1. Introduction**

Various financial and banking institutions do a huge amount of business by extending loans to individuals and thus help in cash flow in the market and society. Before the loan is sanctioned the institutions predicts the probability that an individual would repay their loan in time or not. Machine learning models are effective in prediction of credit risk for customers who have applied for loans. In this post we will talk about how to use machine learning to predict the loan approval status of applicant. Various data processing techniques and exploratory data analysis have been covered in this article. The performance of models were compared based of different parameters.

**2. Problem Definition**

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):**

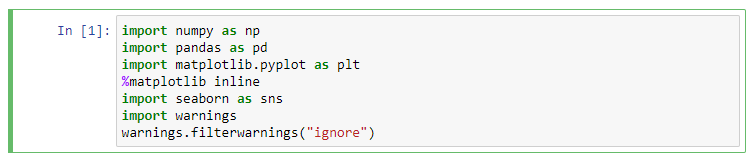
- Loan\_Status

We will 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.

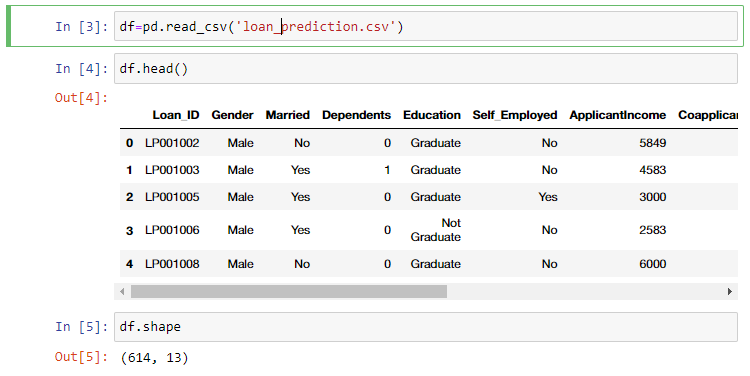
**3. Data Preprocessing and EDA**

The first step in building a model is to understand the dataset and prepare it for efficient prediction. In this step we use various cleaning techniques to produce a normally distributed dataset.

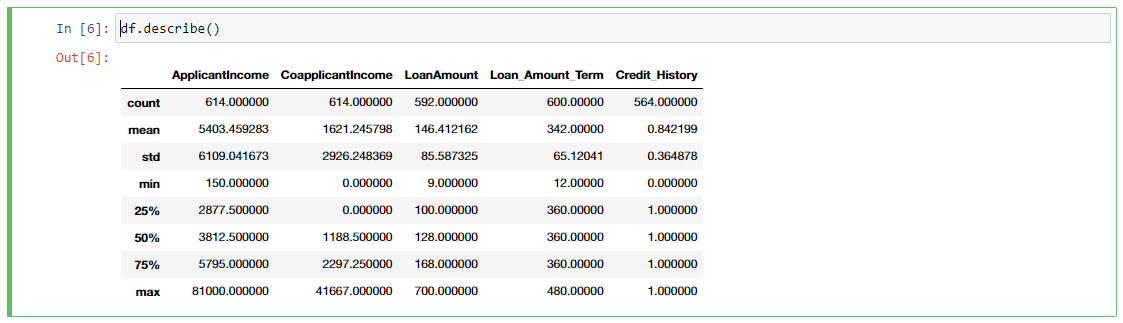
Initially we will import all the necessary libraries which we will use for our analysis



The next step is to load the csv file which contains our dataset. After loading the first thing which we do is check the dataset and understand the features available in the dataset. We will also check the size of the dataset, number of entries and number of columns.

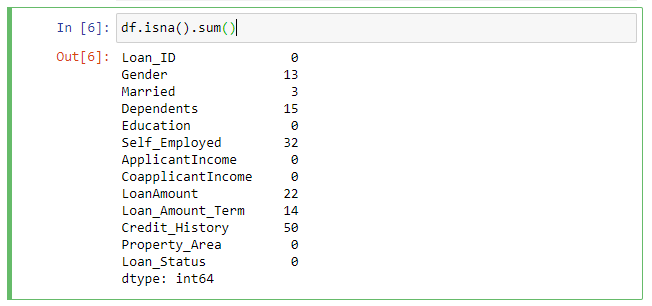


We see that there 614 entries and each of the entries has 12 features and 1 label. Our features are a mixture of categorical features and numerical features. We will identify the categorical features and numerical features and analyze them separately. Let us look at the data description to have an initial look at the variation of data.



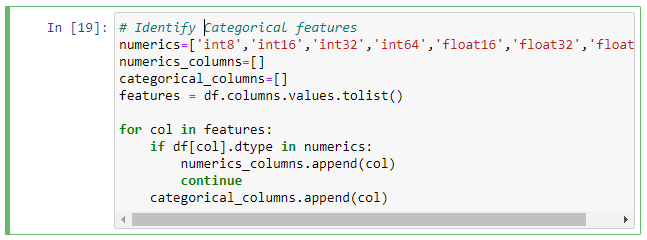
Only 5 columns popped up using df.describe() method which means that only 5 features have numerical values while the rest 7 features are categorical features. The count in all the features must be 614 but it is less than that for 4 features. Thus, we can notice that there are some missing values in the dataset. We will use appropriate imputation technique to fill the missing values.

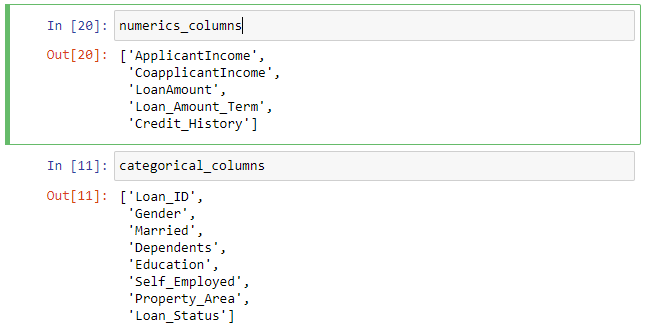
Lets find out how many missing values are there in the dataset.



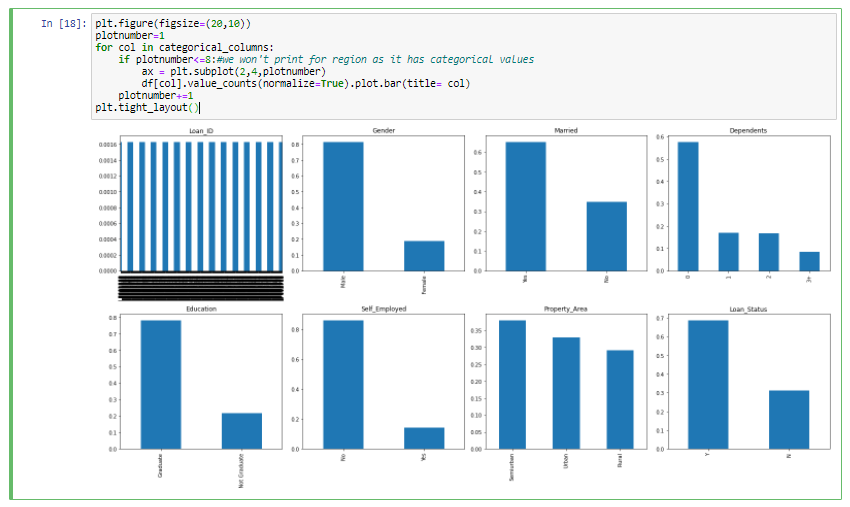
The features which have missing values are Gender, Married, Dependents, Self\_Employed, LoanAmount, Loan\_Amount\_Term and Credit\_History.

We will use some imputing technique at a later stage to impute the missing values in the dataset. The categorical features and numerical features have to handles in different manner. This calls for a need for identification of categorical and numerical features. We will first identify the categorical features and numerical features so that we can use proper encoder to encode categorical features.





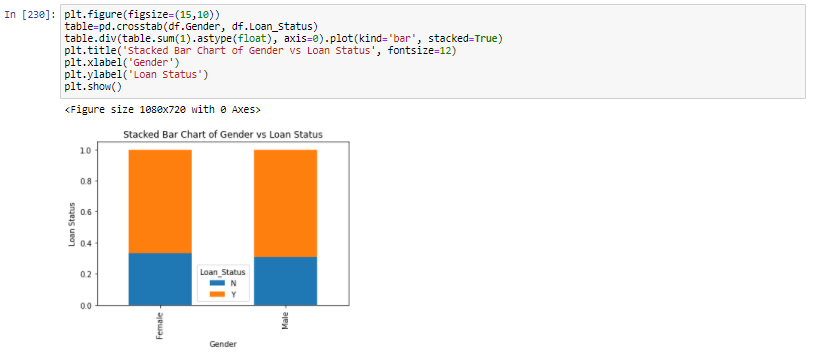
Bar plot can be very helpful in visualizing the categorical features. We will count the occurrence of a category for each feature and plot a bar chart for all the features. This will help us to see the variation in the dataset.



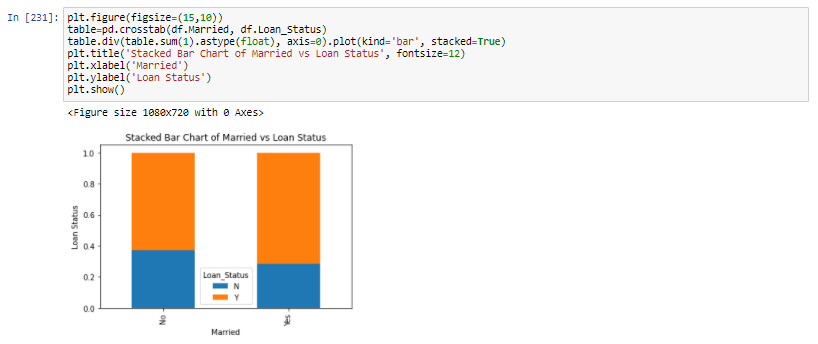
The following conclusion can be drawn from the box plot of categorical features:

* We can drop the feature 'Loan\_ID' as it has nominal values.
* The feature ‘Dependents’ has around 10 % value categorized as 3 +. The rest of the categories have numerical and this category has string value. So, there is a need to address this as all the values must be of same datatype for better analysis.
* 81% of applicants are male.
* 65% of applicants are married
* 58% of applicants don't have depedents, 17% have 1 depndenet nad 17 % have 2 dependents, while around 8% have more than 3 dependents

The loan approval can be visualized with respect to different categories in a feature using stacked bar chart. This is going to help us checking bias in the approval system. Now let us analyze if the loan approval is biased for a particular category of the applicants. The stacked bar chart of Gender vs Loan Status have similar proportion for approved and rejected applicants both in male and female category. Hence, there is bias in loan approval based on gender.



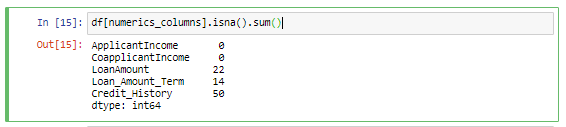
Similarly we can visualize the proportion of approval for different categories in the feature Married, Dependents, Education, Self\_Employed and Property\_Area.



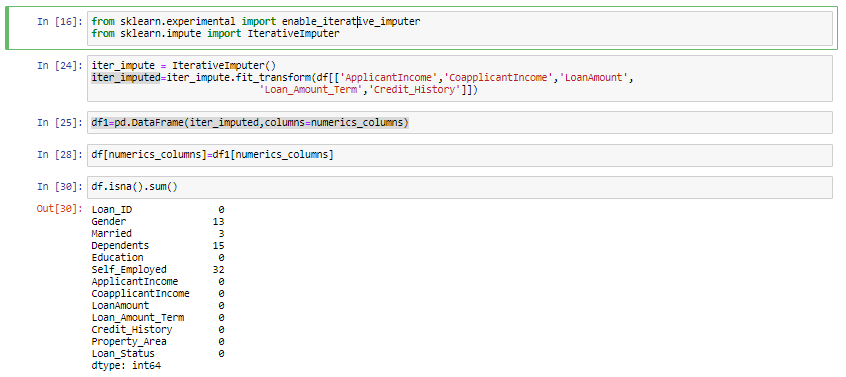
The proportion of approval and rejection in the application for married and bachelor applicants are comparable but do not have huge difference. Hence, we can say that there is no bias in loan approval based on the marital status of the applicants. Stacked bar chart of other categorical features showed that the loan approval is not biased for number of dependents, level of education and whether the applicant is self-employed or not. The individual contribution of these categorical features do not appear to be significant at in univariate analysis but their value will become significant and we will be able to visualize it in multivariate analysis.



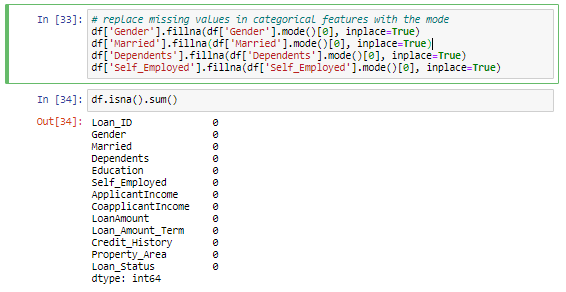
The stacked bar plot of Property\_Area shows that the rejection is more in rural, followed by urban and semi-urban. The next step is to make a list of features having numerical values. This is going to help us examine them in a better way. At first we are going to check the numerical features for missing values. It is a good practice to identify and replace missing values for each feature prior to model fitting and prediction.



The numeric features have missing values. We can either drop the entries which have missing values or impute using imputing techniques in such a way that the data distribution remains intact. Iterative imputer is one such technique for data imputation. This method predicts values of each missing feature as a function of other features in a round robin fashion. This multivariate approach to imputation is robust and provides accurate representation of missing values.



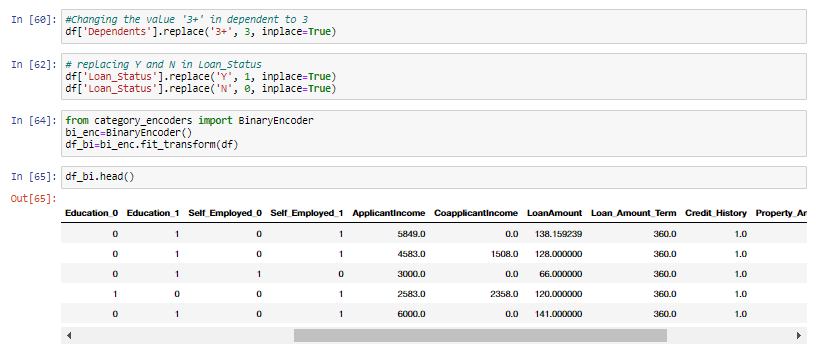
Further we need to identify and replace the missing categorical values. One approach to get rid of missing values in categorical features is to replace them with the most common class using mode of the feature.



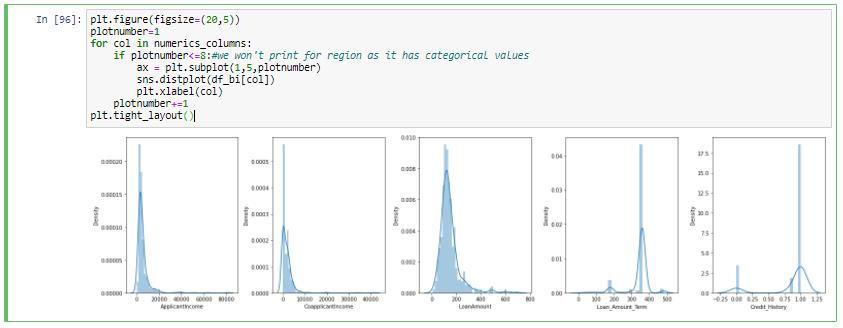
Now that we have imputed all the missing values we are done with the data cleaning. We need to do

feature engineering for the feature ‘Dependents’. In this feature there is a category ‘3+’. Conversion of this category to ‘3’ can prove to be very much meaningful during prediction as all the categories will become integer. Let us replace all the values ‘3+’ in ‘Dependents’ feature and then we will encode all the categorical columns.

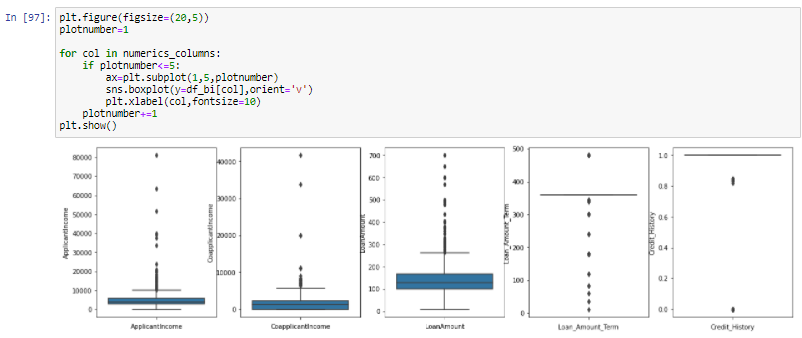
The label ‘Loan\_Status’ has two categories ‘Y’ for the applicants whose loan is approved and ‘N’ for the applicants whose loan is not approved. The model requires all input and output variables to be numeric, so we need to convert all ‘Y’ to 1 and all ‘N’ to 0. The rest of the categories can be encoded using binary encoder as this encodes the given information into a compact form. Binary encoder converts text attributes into numerical values for further processing.

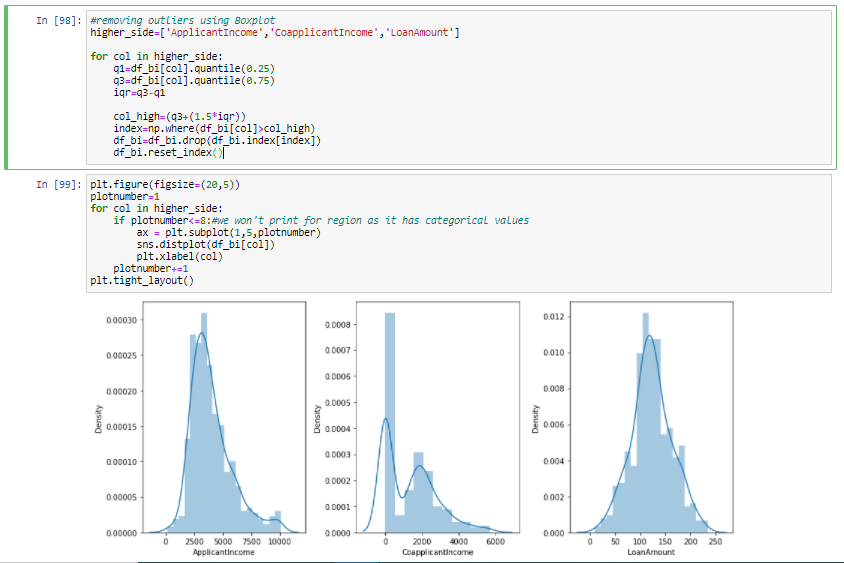


Let us look at the distribution of numeric features. The features must be normally distributed.

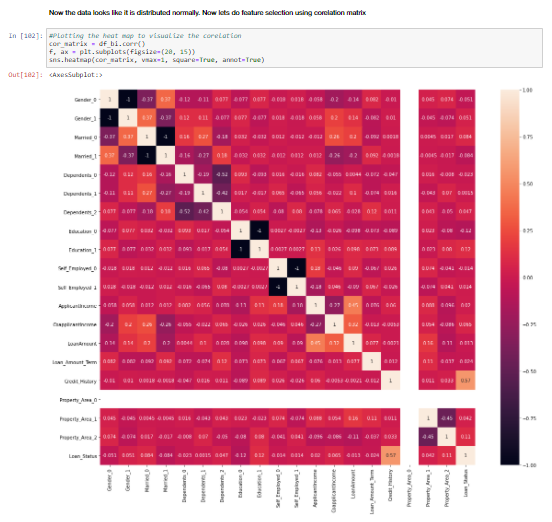


The distribution plot clearly shows that there are skewness in the features. We will use the Box plot method to remove the outliers in the dataset. Removal of outliers creates a normal distribution in some of the variables and makes transformations for other variables more effective. Outliers increase the variability in the data and removal of outliers may make the data significant.

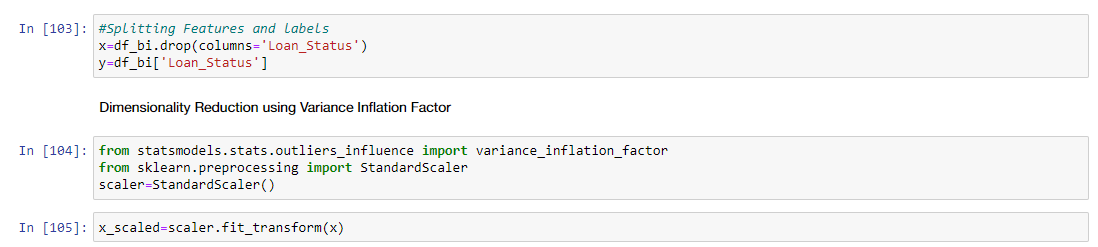


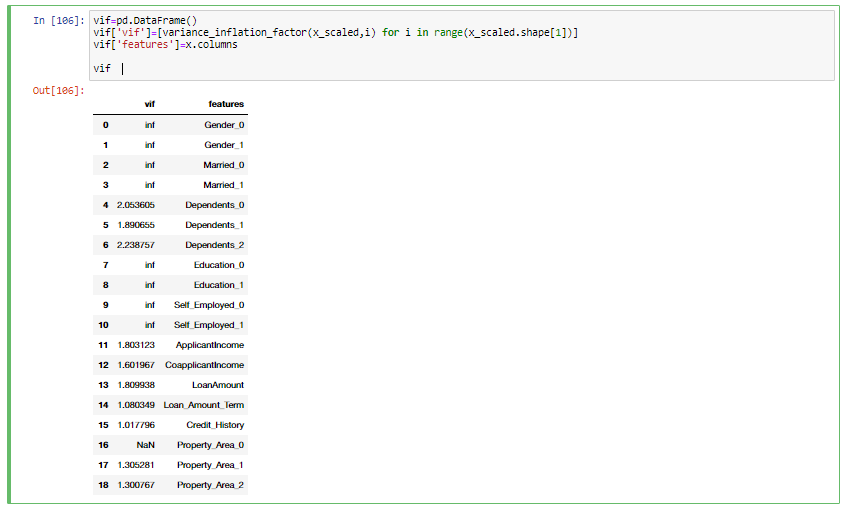


After outlier removal we have achieved normal distribution of the data. Let us find out correlation matrix and plot heatmap to understand the relation between feature vs feature. This is going to help in feature selection and we will drop the features which are highly correlated.

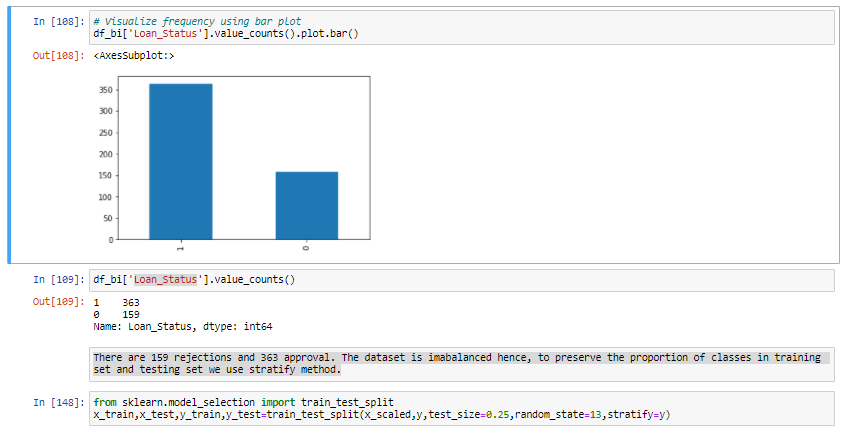


The heat map shows that the correlation between the features is not that significant. Hence, we will keep all the features for model prediction. The next step is to determine if there is multicollinearity. Multicollinearity can be a problem as we won’t be able to distinguish between the individual effects of independent variable and dependent variable. We will split the features and label and use variance inflation factor (VIF) to reduce dimensionality. VIF determines the strength of correlation between features.





The vif for all the features is less than 5. Hence, the dataset doesn’t have multicollinearity. Let us check if the labels are balanced or not as balanced dataset are preferred to efficiently predict and verify model performance.



There are 159 rejections and 363 approval. The dataset is imbalanced hence, to preserve the proportion of classes in training set and testing set we use stratify method.

**EDA Concluding Remarks**

* The dataset has 614 entries each entry has 12 features and 1 label
* The null values in the dataset are imputed with mode for categorical features and iterative imputer for numerical features
* The feature Loan\_ID is dropped as it has nominal values.
* Feature Engineering was done for ‘Dependents’.
* 81% of applicants are male.
* 65% of applicants are married
* 58% of applicants don't have dependents, 17% have 1 dependent and 17 % have 2 dependents, while around 8% have more than 3 dependents
* Loan approval is not biased for gender, marital status, number of dependents, level of education and whether the applicant is self-employed or not.
* The stacked bar plot of Property Area shows that the rejection is more in rural, followed by urban and semi-urban.
* The categorical features were encoded using Binary Encoder
* The outliers in the dataset were handled using Boxplot
* Feature Selection was carried out using Heat Map
* Multidimensional collinearity was checked using Variance Inflation Factor
* The label was imbalanced and the proportion of classes was maintained using stratify during train test split

**4. Building Machine Learning Models**

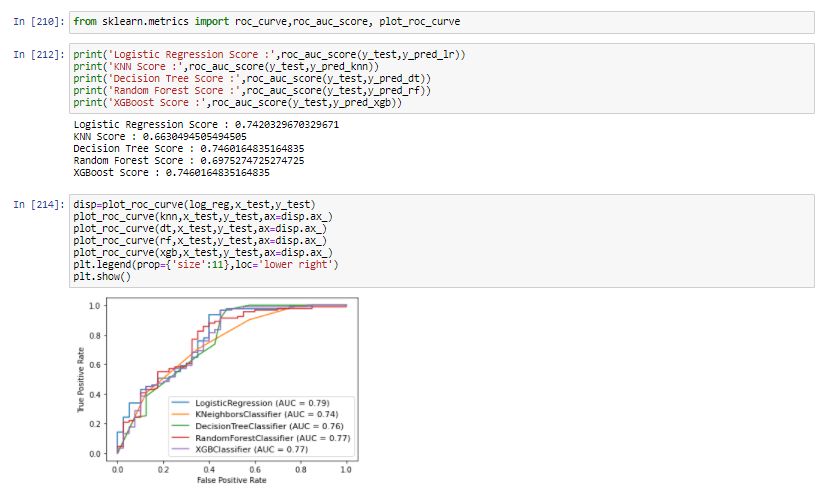
The following models were tested to predict the approval of loan:

1. Logistic Regression
2. KNN Classifier
3. Decision Tree Classifier
4. Random Forest Classifier
5. XG Boost Classifier

The hyper parameters for the models were tuned using Grid Search CV technique wherever applicable. The following table summarizes the performance of model:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S. No.** | **Model Name** | **Accuracy** | **Precision** | **F1 Score** | **Accuracy after tuning** | **Precision after tuning** | **F1 Score after tuning** |
| 1. | Logistic Regression | 0.82 | 0.81 | 0.76 | - | - | - |
| 2. | KNN | 0.76 | 0.72 | 0.68 | 0.76 | 0.72 | 0.68 |
| 3. | Decision Tree | 0.74 | 0.71 | 0.71 | 0.83 | 0.85 | 0.77 |
| 4. | Random Forest | - | - | - | 0.79 | 0.79 | 0.72 |
| 5. | XG Boost | - | - | - | 0.83 | 0.85 | 0.77 |

The models were also compared from ROC Curve and AUC. The model which has the highest AUC curve is the best considered as the best fit.



The AUC is highest for the Logistic Regression. However, the accuracy and precision is highest for Decision Tree Classifier. When it comes to classification model it is standard practice to use precision and recall. Higher precision means that the model is accurately predicting the positive instances into positive class. We select Decision Tree as our final model for prediction of loan approval. The hyper tuned parameters for the Decision Tree Classifier are:

{'criterion': 'gini', 'max\_depth': 5, 'min\_samples\_leaf': 20, 'min\_samples\_split': 3}

The classification report for the finalized model is shown below:

precision recall f1-score support

0 0.88 0.53 0.66 40

1 0.82 0.97 0.89 91

accuracy 0.83 131

macro avg 0.85 0.75 0.77 131

weighted avg 0.84 0.83 0.82 131

**5. Concluding Remarks**

We cleaned the data, performed EDA and successfully trained a model and tuned the hyper parameters to predict the response of a loan application based on the features and dataset available with us. The Decision Tree Classifier proved to be the best model in prediction due to its high precision and accuracy. I hope this has proven to be an informative topic.

You may refer to my repository for detailed code

<https://github.com/dewangan-ashishk/Evaluation-Project/blob/d3eb320df22fe136ccf21c7131904cc3feeefc60/Loan%20Prediction%20Final.ipynb>