**REPORT**

**Table of Contents:**

**Introduction ----------------------------------------------------------------------2**

**Data Description-----------------------------------------------------------------3**

**Approach--------------------------------------------------------------------------3**

**Visualization----------------------------------------------------------------------3-4**

**Data Pre-processing-------------------------------------------------------------5**

**Training the data----------------------------------------------------------------6**

**Evaluation------------------------------------------------------------------------7**

**Improving the Model-----------------------------------------------------------8**

**Model comparison--------------------------------------------------------------9**

**Conclusion-----------------------------------------------------------------------10**

**References-----------------------------------------------------------------------11**

**Introduction:**

The company's goal is to automate (in real time) the loan qualification process using the data that customers submit when completing an online application form. It is anticipated that ML models will be developed to help the business predict loan acceptance and speed up the decision-making process for determining whether an application is approved for a loan or not. You are asked to predict the status of a loan based on a variety of factors, including your credit history. To train and compare the data, we employ machine learning models such as decision tree classifiers, K closest neighbours, logistic regression, and support vector machines. Afterwards, we assess the model at every stage.

**Data Description:**

There are 13 variables in this data set:

* 8 categorical variables,
* 4 continuous variables, and
* 1 variable to accommodate the loan ID.

Each of the VARIABLE can be described as follows:

|  |  |
| --- | --- |
| VARIABLE | DESCRIPTION |
| Loan\_ID | Unique Loan ID |
| Gender | Male/ Female |
| Married | Applicant married (Y/N) |
| Dependents | Number of dependents |
| Education | Applicant Education [graduate/Undergraduate] |
| Self\_Employed | Self-employed (Y/N) |
| ApplicantIncome | Applicant income |
| CoapplicantIncome | Coapplicant income |
| LoanAmount | Loan amount in thousands |
| Loan\_Amount\_Term | Term of loan in months |
| Credit\_History | Credit history meets guidelines |
| Property\_Area | Urban/ Semi Urban/ Rural |
| Loan\_Status | Loan approved (Y/N) |

**Approach:**

My plan for carrying out the project is to pre-process the data after visualising the supplied data. The data was trained using models such Decision Tree Classifier, KNeighbors Classifier, SVC, and Logistic Regression. The test and train data will be hyperparameter tweaked after the models have been trained. In order to select the optimal model for the given data, I lastly examine and contrast each model, comparing accuracy scores like precision, recall, f1, log loss, and accuracy score for each model.

Steps:

* Basic visualization of data
* Data preprocessing
* Dealing Categorical and numerical data
* Training the Data
* Evaluation of the models
* Improving the model

Handling Outliers

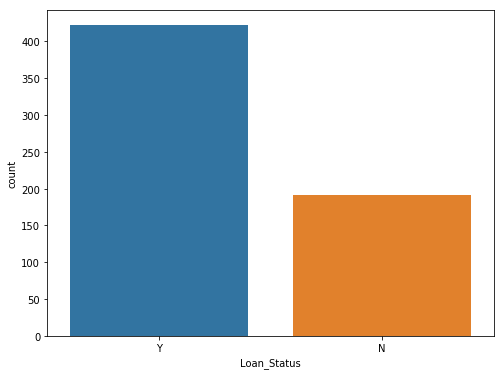
Features Selection

* Evaluating the model on test\_data and train\_data
* Comparing the accuracy scores

**Visualization:**

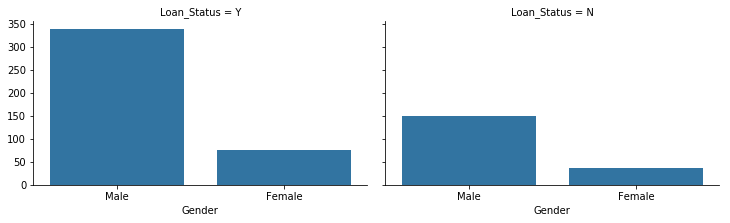
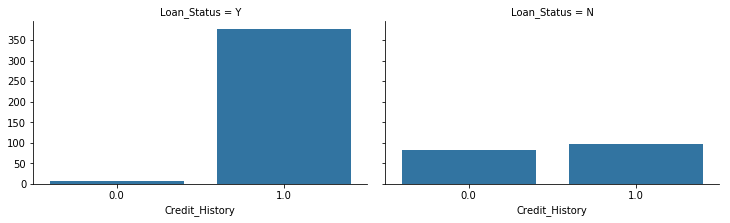
Here, we visualize each column, classify them as significant or not, and better comprehend them for the following phase.

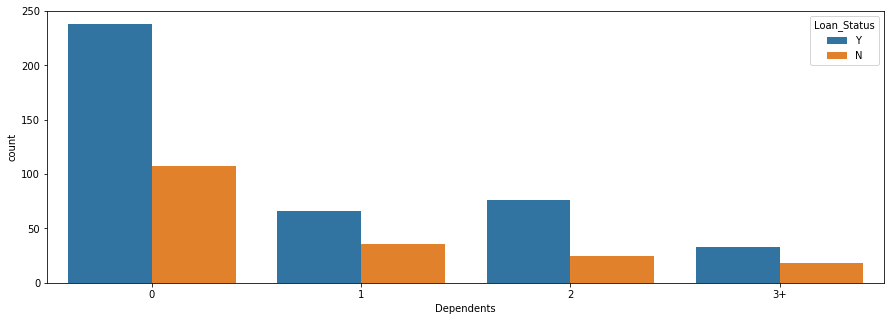
First, we look at the percentage of loan approval.

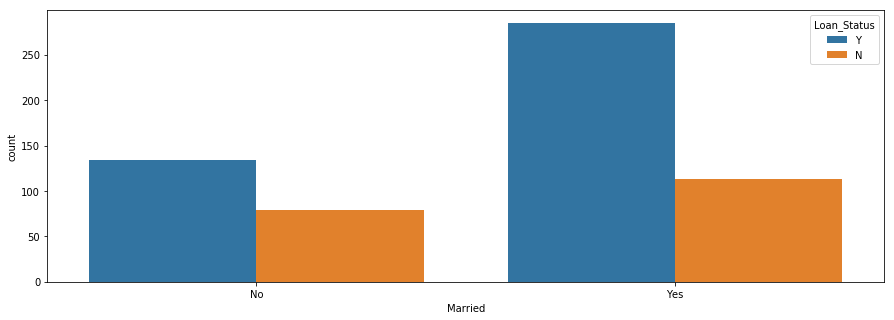


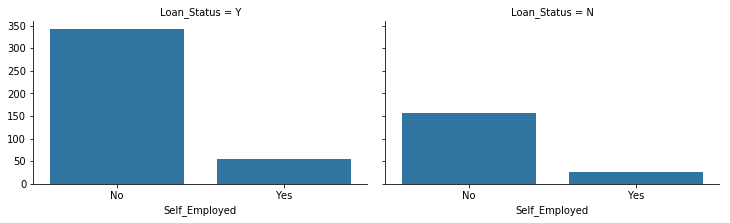
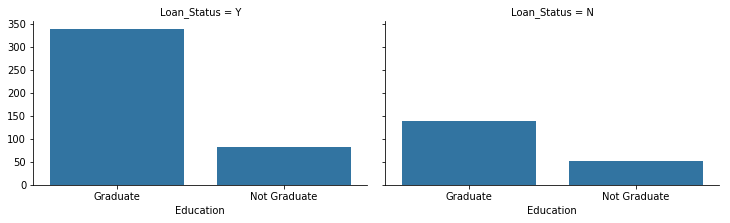
Now, let’s look at each variable as a feature and visualize it.

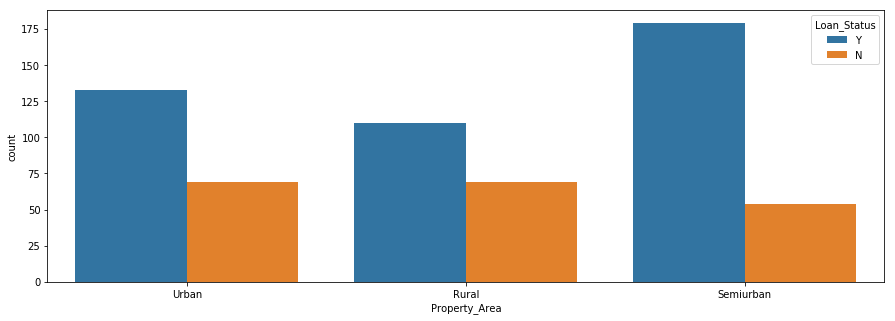
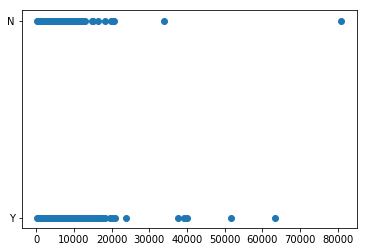
**1). Credit History 2). Gender**



**3). Married 4). Dependents**



**5). Education 6). Self-Employed.**

**7). Property Area 8). Applicant Income**

**Data pre-processing:**

Data preprocessing is an integral step in Machine Learning as the quality of data and the useful information that can be derived from it directly affects the ability of our model to learn; therefore, it is extremely important that we preprocess our data before feeding it into our model.

First, we handle Null values or the missing values. There are various ways for us to handle null values. The easiest way is by dropping the rows or columns that contain null values.

Now, we categorize the data into categorical and numerical data, and are represented as cat\_data and num\_data.

After filling the missing values, we use **LabelEncoder** for the categorical data.

**from** sklearn**.**preprocessing **import** LabelEncoder

le **=** LabelEncoder**()**

cat\_data**.**head**()**

The **LabelEncoder** encode labels with a value between 0 and n\_classes-1 where n is the number of distinct labels. If a label repeats it assigns the same value to as assigned earlier.

Then we transform the target column and other columns respectively as show below.

target\_values **=** **{**'Y'**:** 0 **,** 'N' **:** 1**}**

target **=** cat\_data**[**'Loan\_Status'**]**

cat\_data**.**drop**(**'Loan\_Status'**,** axis**=**1**,** inplace**=True)**

target **=** target**.map(**target\_values**)**

**for** i **in** cat\_data**:**

cat\_data**[**i**]** **=** le**.**fit\_transform**(**cat\_data**[**i**])**

**Train the Data:**

Here, the data is divided into train\_data and test\_data using **StratifiedShuffleSplit**.

**from** sklearn**.**model\_selection **import** StratifiedShuffleSplit

sss **=** StratifiedShuffleSplit**(**n\_splits**=**1**,** test\_size**=**0.2**,** random\_state**=**42**)**

**for** train**,** test **in** sss**.**split**(**X**,** y**):**

X\_train**,** X\_test **=** X**.**iloc**[**train**],** X**.**iloc**[**test**]**

y\_train**,** y\_test **=** y**.**iloc**[**train**],** y**.**iloc**[**test**]**

**print(**'X\_train shape'**,** X\_train**.**shape**)**

**print(**'y\_train shape'**,** y\_train**.**shape**)**

**print(**'X\_test shape'**,** X\_test**.**shape**)**

**print(**'y\_test shape'**,** y\_test**.**shape**)**

# almost same ratio

**print(**'\nratio of target in y\_train :'**,**y\_train**.**value\_counts**().**values**/** **len(**y\_train**))**

**print(**'ratio of target in y\_test :'**,**y\_test**.**value\_counts**().**values**/** **len(**y\_test**))**

**print(**'ratio of target in original\_data :'**,**df**[**'Loan\_Status'**].**value\_counts**().**values**/** **len(**df**))**

Next, we use 4 different models for training such as,

* **logistic regression**
* **KNeighborsClassifier**
* **SVC**
* **DecisionTreeClassifier**

models **=** **{**

'LogisticRegression'**:** LogisticRegression**(**random\_state**=**42**),**

'KNeighborsClassifier'**:** KNeighborsClassifier**(),**

'SVC'**:** SVC**(**random\_state**=**42**),**

'DecisionTreeClassifier'**:**DecisionTreeClassifier**(**max\_depth**=**1**,**

random\_state**=**42**)**

**}**

**Evaluation:**

After training the models, we evaluate the models against accuracy scores and compare them with each other to find the best model.

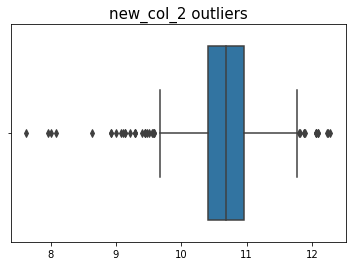
The following scores are calculated for 10 folds.

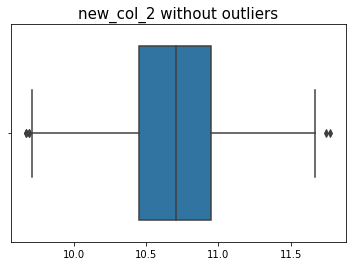
|  | **Precision** | **Recall** | **F1** | **loss** | **Accuracy** |
| --- | --- | --- | --- | --- | --- |
| 0 | 0.857143 | 0.375000 | 0.521739 | 7.598547 | 0.780000 |
| 1 | 1.000000 | 0.187500 | 0.315789 | 8.980082 | 0.740000 |
| 2 | 1.000000 | 0.312500 | 0.476190 | 7.598531 | 0.780000 |
| 3 | 1.000000 | 0.375000 | 0.545455 | 6.907755 | 0.800000 |
| 4 | 1.000000 | 0.466667 | 0.636364 | 5.638984 | 0.836735 |
| 5 | 1.000000 | 0.733333 | 0.846154 | 2.819492 | 0.918367 |
| 6 | 0.857143 | 0.400000 | 0.545455 | 7.048746 | 0.795918 |
| 7 | 0.900000 | 0.600000 | 0.720000 | 5.036922 | 0.854167 |
| 8 | 0.666667 | 0.266667 | 0.380952 | 9.354285 | 0.729167 |
| 9 | 0.875000 | 0.466667 | 0.608696 | 6.476037 | 0.812500 |

**Improving the model:**

**Handling the outliers** and **feature selection** will help us to improve our model at each step.

We will use boxplot to detect outliers and use IQR techniques to handle these outliers.





**Model Comparison:**

**LogisticRegression:**

**pre: 0.895**

**rec: 0.447**

**f1: 0.596**

**loss: 6.458**

**acc: 0.813**

**----------------------------------------**

**KNeighborsClassifier:**

**pre: 0.652**

**rec: 0.395**

**f1: 0.492**

**loss: 8.705**

**acc: 0.748**

**----------------------------------------**

**SVC:**

**pre: 0.895**

**rec: 0.447**

**f1: 0.596**

**loss: 6.458**

**acc: 0.813**

**----------------------------------------**

**DecisionTreeClassifier:**

**pre: 0.895**

**rec: 0.447**

**f1: 0.596**

**loss: 6.458**

**acc: 0.813**

**CONCLUSION:**

The recall score does not increase following training and assessment since there is insufficient data. However, other scores skyrocketed, indicating that feature selection and handling outliers came first. Ultimately, DecsiontreeClassifier and Logistic Regression fared better than SVC and KNeighborsClassifier.

**References:**

1.Scikit learn [Data preprocessing](https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing).

2. Scikit learn [Model evaluation](https://scikit-learn.org/stable/modules/cross_validation.html).

3. Kaggle (Yaheaal) Loan status prediction.