dun & bradstreet

# Building Predictive Model for Loan Default

Date: 10-Aug-2022
For job role: Data Scientist
Submitted by: Dileep Sathyan

## Objective:

Build a predictive model for calculating the probability for a borrowe to default in the future. The definition of "Default" for this analysis being a loan installment being unpaid for more than 3 months (Days_Past_Due >= 90).

## About Dataset:

The dataset includes the 6000 records of payment history of users reported on dates ranging from '2018-01-02' through '2018-12-10' along with the respective loan account id, the amounts sanctioned, outstanding balance, over due, instalment amount, date of latest payment received, expected loan closure date etc…

## Data Cleaning:

The dataset includes a great amount of missing data and improper datatypes for the modeling. Hence the below steps are carried out on the data fields to make it suitable for final modeling.

- ➔ Dropped certain fields from the raw dataset with completely NULL values.
- ➔ Dropped all the duplicated records in the dataset.
- ➔ Renamed all the columns uniformly.
- ➔ Converted all the date fields to the right datetime formats.
- ➔ Dropped the records of same 'ACCOUNT_ID' which were reported on multiple 'DATE_REPORTED', to make it unique for the dataframe.

| | SUBJECT_ID | ACCOUNT_ID | DATE_REPORTED | CURR_CODE | REPAYMENT_FREQUENCY | NUMBER_OF_INSTALLMENTS | SANCTION_AMT |
|---|---|---|---|---|---|---|---|
| 137 | 110117010002384406 | 2201170002918195 | 2018-03-08 | TZS | MonthlyInstalments30Days | 61.0 | 10200000.0 |
| 4369 | 110117010002384406 | 2201170002918195 | 2018-06-10 | TZS | MonthlyInstalments30Days | 61.0 | 10200000.0 |

**Handling Missing Data:**

Since the dataset involved a great amount of missing datapoints, the same have been handled using the below logics and methods. Firstly, counts the missing data points in each field.

```
SUBJECT_ID                 0
ACCOUNT_ID                 0
DATE_REPORTED              0
CURR_CODE                  0
REPAYMENT_FREQUENCY       12
NUMBER_OF_INSTALLMENTS    70
SANCTION_AMT               0
TOT_OUTSTD_BAL            13
OVER_DUE_AMT               7
INSTALLMENT_AMT           22
DATE_LATEST_PAYMENT     1345
DATE_ACC_OPEN              0
DATE_ACC_CLOSE          3046
LOAN_TYPE                 80
LOAN_STATUS                0
DPD                       71
DATE_EXP_LOAN_CLOSURE      0
```

| Attribute Name | Contents in the field. | Methodology adopted. |
|---|---|---|
| 'REPAYMENT_FREQUENCY' | MonthlyInstalments30Days<br>AtTheFinalDayOfThePeriodOfContract<br>IrregularInstalments<br>FortnightlyInstalments15Days<br>AnnualInstalments360Days<br>SixMonthInstalments180Days | Noticed that all the missing loan account are with Overdrafts in the 'LOAN_TYPE'<br><br>All overdraft loans are to be paid in 1 instalment at the end of period of contract. |
| 'NUMBER_OF_INSTALLMENTS' | Numerical variable | Replaced the missing data points by diving the 'REPAYMENT_FREQUENCY' in days from the difference between the 'DATE_OPEN' and 'DATE_EXP_LOAN_CLOSURE' |
| 'INSTALLMENT_AMT' | Numerical variable | Filled by dividing 'NUMBER_OF_INSTALLMENTS' from 'SANCTION_AMT' |
| 'OVER_DUE_AMT' | Numerical variable | For the loans which are Terminated properly, there should not be any Over-Due Amount<br><br>The loans with DPD ==0, there should not be Over_Due_Amount<br><br># For the rest of the loans with status "Existing", since they all have Repayment Frequency of 30 days &<br># since the DPDs are less than 30, their OVER_DUE should be equal to 'INSTALLMENT_AMT' |

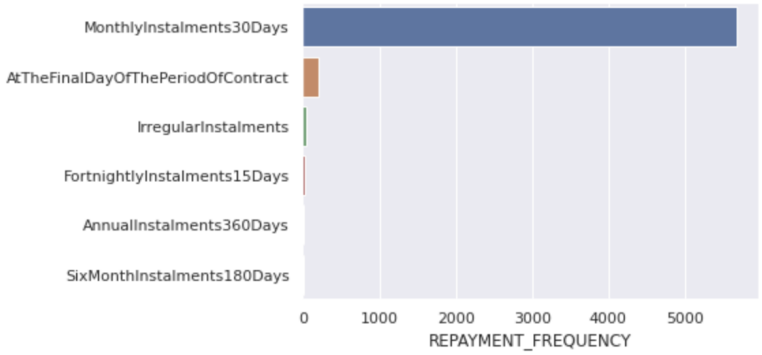
| | | |
|---|---|---|
| ‘TOT_OUTSTD_BAL’ | Numerical variable | For DPD >0 customers, their Oust_Bal must be greater than Over_Due. However, we shall consider the same amount here as we dont have the Total_Paid on date.<br><br>For DPD ==0 customers, their oustanding is at the least their next INSTALLMENT_AMT |
| ‘LOAN_TYPE’ | ConsumerLoan<br>BusinessLoan<br>OtherInstalmentOperation<br>CreditCard<br>MortgageLoan<br>Overdraft<br>LeasingFinancial | The missing values in LOAN_TYPE are filled with the mode value.<br>Because, find the the correct Loan_type using the Number of Installments seems difficult as even the loans from Credit Cards are allowed to pay in longer installments |
| ‘LOAN_STATUS’ | Existing<br>TerminatedAccordingTheContract<br>TerminatedInAdvanceCorrectly<br>TerminatedInAdvanceIncorrectly | The loans which have valid LATEST_PAYMENT_DATE, if the OVER_DUE exists, find the days_diff from reporting date.<br><br>Loans which are to be paid at the end of the contract, DPD should be 0<br><br>Notice that for the rest records with NULL DPD, REPAYMENY_FREQ is every month.<br># So, if the Overdue_Amount is greater than 3 times of the installment, it has DPD > 90<br><br>Else, it has lesser than 90 days DPD. Fill the rest NULL DPD with 0. |
| ‘LOAN_TENURE’ | Numerical variable | This field has been freshly calculated as the difference in days between the ‘DATE_EXP_LOAN_CLOSURE’ and ‘DATE_ACCT_OPEN’ |

**Insights:**

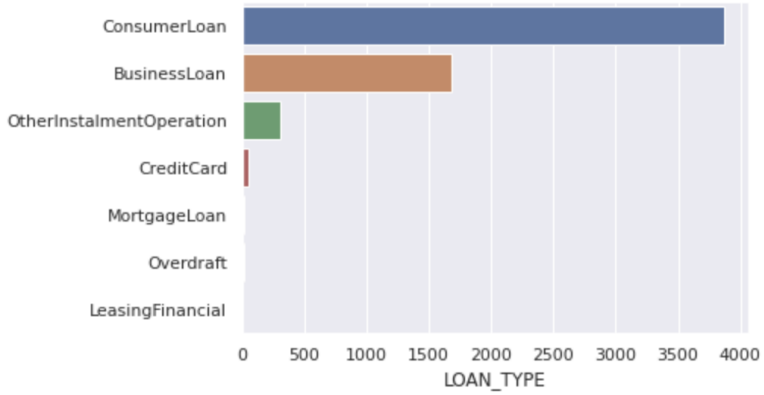
- Having 5923 users but 5960 ACCOUNT_IDs points out that there are some users who have availed more than 1 loan.
- Out of 36 users who availed more than 1 loans accounts, only 1 SUBJECT_ID has defaulted his 2nd loan ACCOUNT_ID by 286 days.

| SUBJECT_ID | ACCOUNT_ID | SANCTION_AMT | LOAN_TYPE | LOAN_STATUS | DPD |
|---|---|---|---|---|---|
| 110113010000508823 | 2201150001612310 | 14700000.0 | ConsumerLoan | Existing | 286.0 |

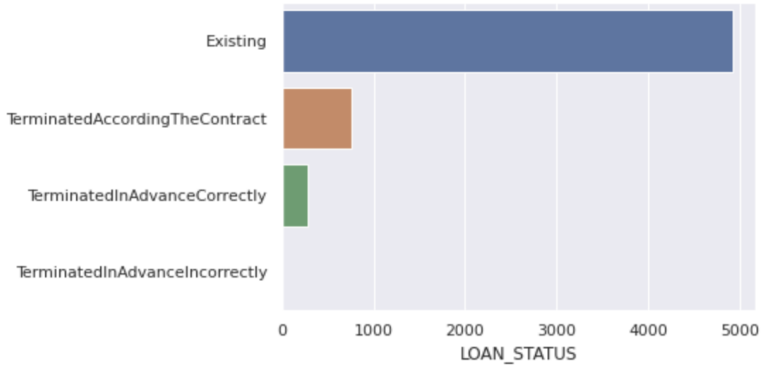
- Majority of the loans were issued with Repayment Frequency as 30 days




- The count of ConsumerLoans are more than twice of Business Loans issued.




- The given dataset contains most of the loans with Existing status. Although there are some loan accounts which were already terminated due to multiple reasons.




- Out of 5960 loans, only 5 loans were issued in the currency USD whereas all the rest were issued in TZS (Tanzanian Shillings).

## Data Manipulations:

- Since we have only 5 loan accounts in USD, its preferable to convert them into TZS to have uniformity in the Amounts. Hence the Amounts of loans issued in USD were converted to TZS using the current exchange rate of 2335 (USD to TZS) & thus getting the below TZS values for the 5 loans.

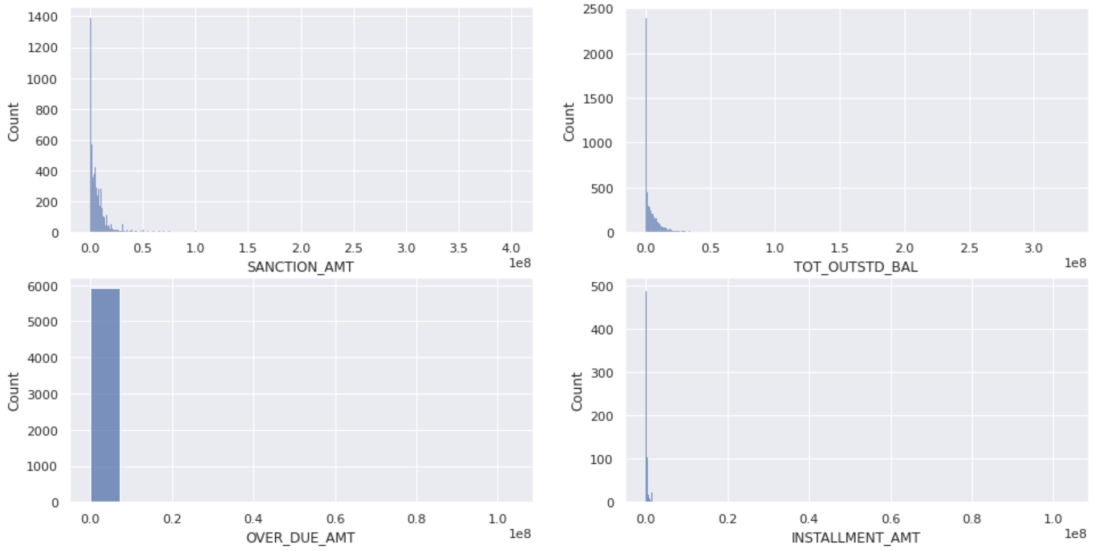
USD

| | SANCTION_AMT | TOT_OUTSTD_BAL | OVER_DUE_AMT | INSTALLMENT_AMT |
|---|---|---|---|---|
| 599 | 44500.0 | 37510.66 | 0.00 | 967.90 |
| 649 | 100000.0 | 0.00 | 0.00 | 775.00 |
| 902 | 60000.0 | 1484.13 | 1484.13 | 1484.13 |
| 4010 | 40000.0 | 15815.56 | 0.00 | 1309.55 |
| 4338 | 35000.0 | 33958.13 | 0.00 | 657.26 |

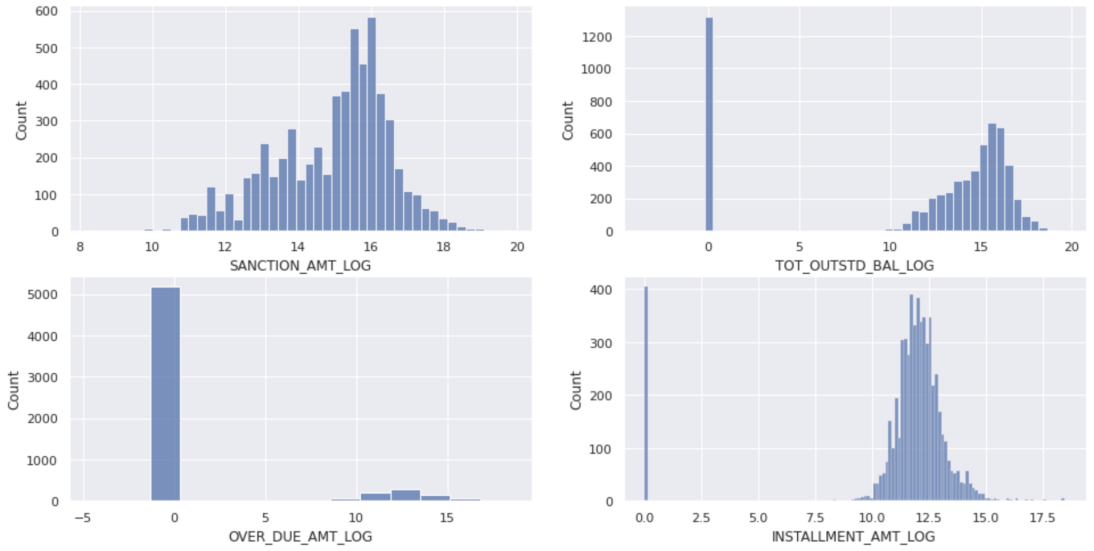
TZS

| | SANCTION_AMT | TOT_OUTSTD_BAL | OVER_DUE_AMT | INSTALLMENT_AMT |
|---|---|---|---|---|
| 599 | 103907500.0 | 87587391.10 | 0.00 | 2260046.50 |
| 649 | 233500000.0 | 0.00 | 0.00 | 1809625.00 |
| 902 | 140100000.0 | 3465443.55 | 3465443.55 | 3465443.55 |
| 4010 | 93400000.0 | 36929332.60 | 0.00 | 3057799.25 |
| 4338 | 81725000.0 | 79292233.55 | 0.00 | 1534702.10 |

- Further notice that the 'SANCTION_AMT', 'TOT_OUTSTD_BAL', 'OVER_DUE_AMT' and 'INSTALLMENT_AMT' are not uniformly distributed which may affect the model performance

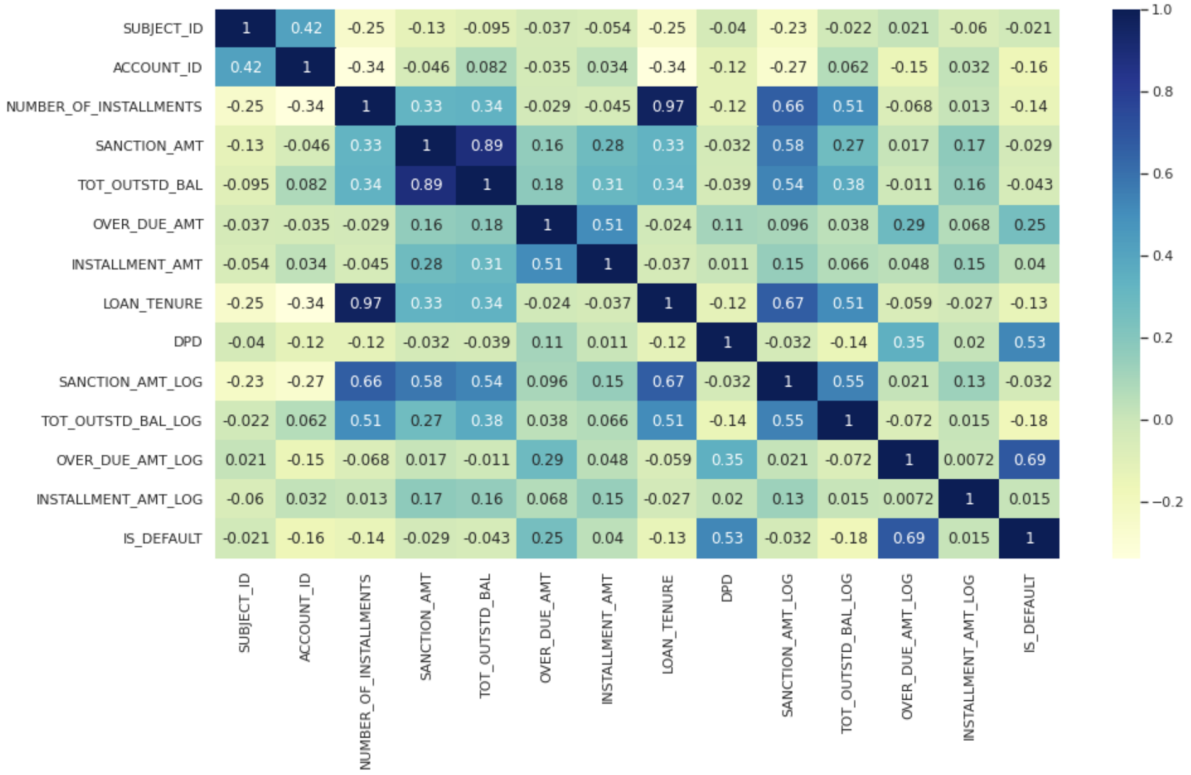



Hence, all the 4 above fields were normalised using their log values, which changes the values as in the below histograms.



## Correlation Matrix:

Below heatmap shows the relevant variables and how each of them are correlated with each other.




## Label Encoding:

All the categorical variables such as 'REPAYMENT_FREQUENCY', 'LOAN_TYPE','LOAN_STATUS' are encoded using LabelEncoder from sklearn_preprocessing module, to make them suitable for model building. The categorical variables in the resultant encoded dataframe appears like the below.

| | REPAYMENT_FREQUENCY | LOAN_TYPE | LOAN_STATUS |
|---|---|---|---|
| 0 | 4 | 5 | 0 |
| 1 | 4 | 1 | 0 |
| 2 | 4 | 1 | 0 |
| 3 | 4 | 0 | 0 |
| 4 | 4 | 1 | 0 |

## Splitting the data for Training & Testing the Model:

Using the train_test_split from sklearn.model_selection, split the data 75% for training and 25% for testing the model.

**Model Training & Selection:**

Try different classifier models using the dataset & get the Accuracy scores in percentages for each model.

➔ Logistic Regression Model

```
# Logistic Regression Model
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
classify_model(model, X, y)
```

```
Accuracy is:  93.8255033557047
Cross Validation Score:  94.12751677852349
```

➔ DecisionTree Classifier Model

```
# DecisionTree Classifier Model
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier()
classify_model(model, X, y)
```

```
Accuracy is:  97.71812080536913
Cross Validation Score:  97.09731543624163
```

➔ RandomForest Classifier Model

```
# RandomForest Classifier Model
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()
classify_model(model, X, y)
```

```
Accuracy is:  98.45637583892616
Cross Validation Score:  98.02013422818791
```

Since the RandomForest Classifier Model is performing well on the dataset, we shall use it for final prediction.

## Confusion Matrix for Validation of results:

We shall use confusion_matrix from sklearn to validate the predicted results.

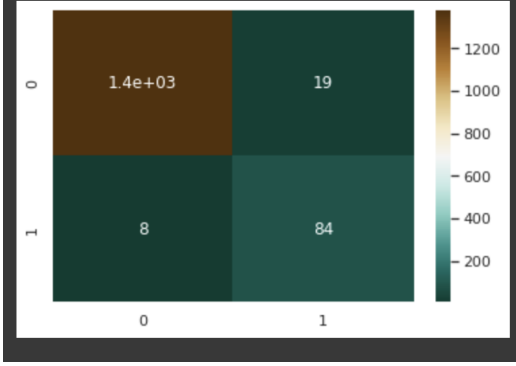
```
from sklearn.metrics import confusion_matrix

y_pred = model.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
cm
```

```
array([[1379,   19],
       [   8,   84]])
```

Use heatmap from Seaborn to visualise the predictions.

```
sns.heatmap(cm, annot=True, cmap='BrBG_r')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc289406cd0
```




Notice that the model has been doing good job at prediction with 99% Accuracy, we have built the prediction model as required by the assignment.