# LENDING CLUB PROJECT REPORT

**Lending Club Dataset**

Lending Club (LC) is a peer to peer online lending platform. It is the world's largest marketplace connecting borrowers and investors, where consumers and small business owners lower the cost of their credit and enjoy a better experience than traditional bank lending, and investors earn attractive risk-adjusted returns.

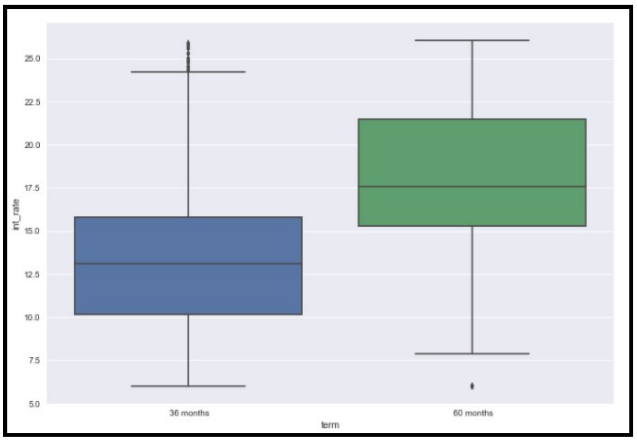
The information available for each loan consists of all the details of the loans at the time of their issuance as well as more information relative to the latest status of loan such as how much principal has been paid so far, how much interest, if the loan was fully paid or defaulted, or if the borrower is late on payments etc.

The dataset has about 115 features to be explored before predicting whether a loan will be fully paid or defaulted.

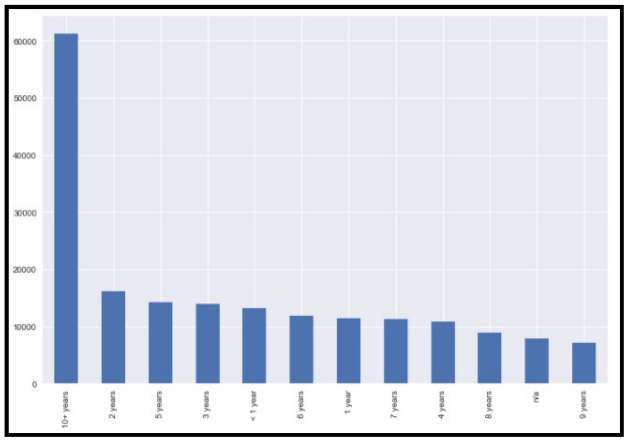
## Question 1

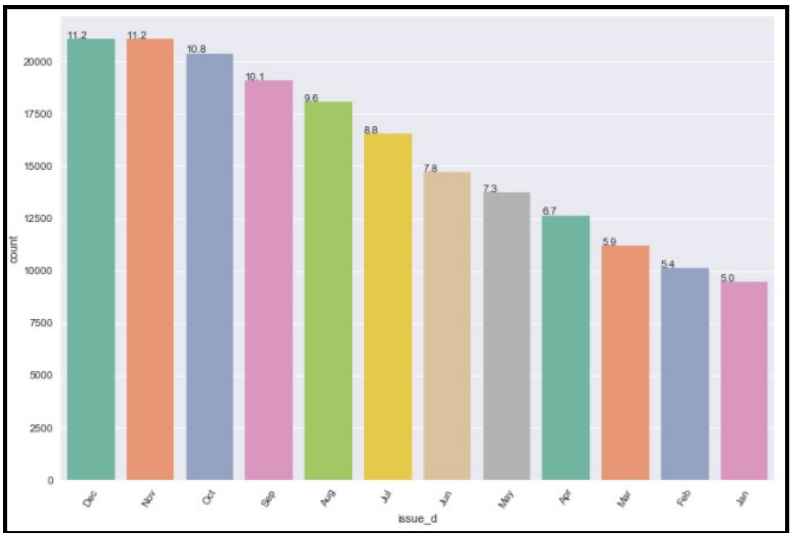
## Data Exploration

**Box plot of Interest Rate Vs Loan Term**
It is observed that interest rates are higher when the loan term is longer at 60 months.

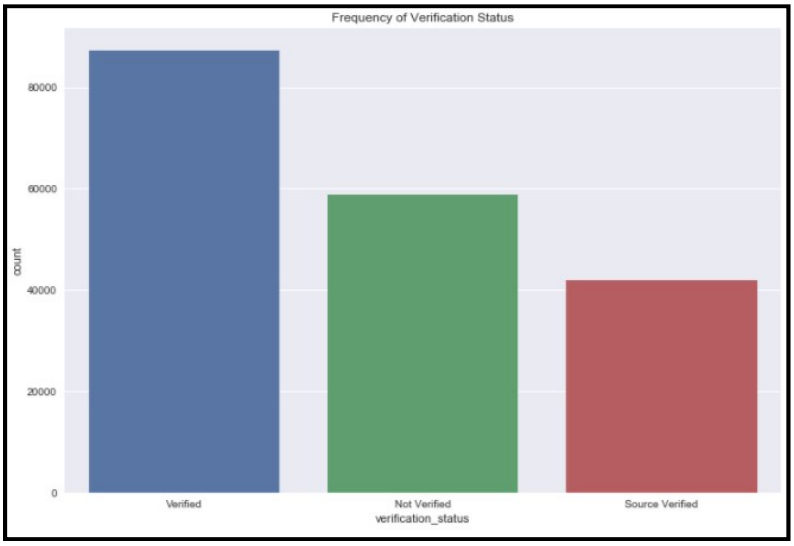



**Average employment length of people requesting loans –** Number of loan requests increases with employment length.

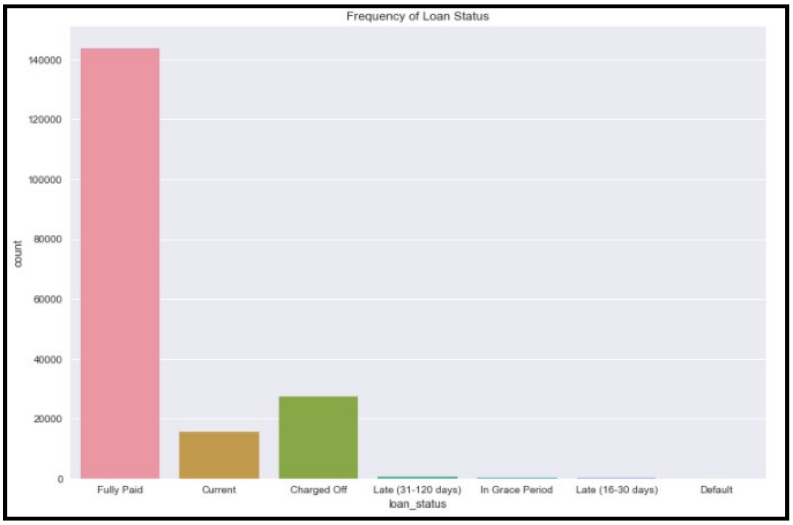




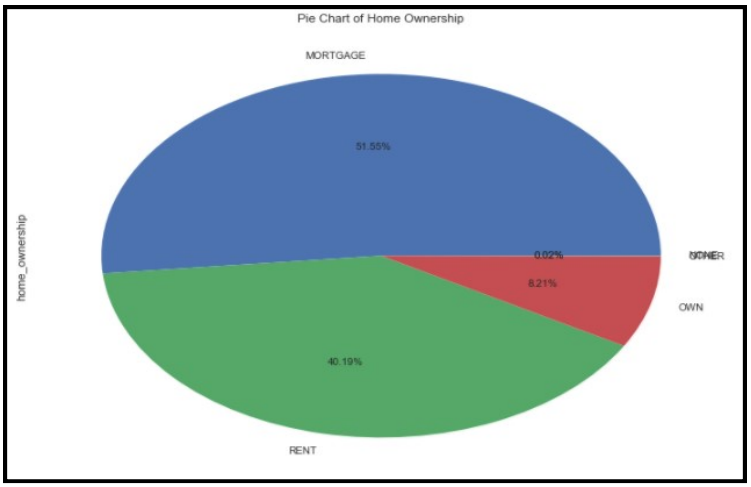

**Count of loan issues month wise**
This graph is to understand when loan requests have been high.
From the graph, it can be seen that the number of people requesting a loan gradually increases from January to December and is at its peak in November and December, probably because of holiday season.



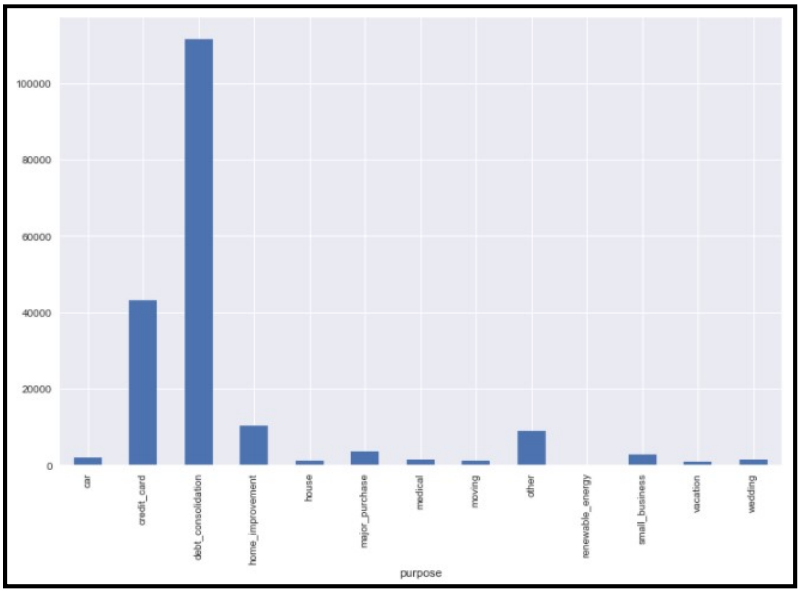

**Frequency of Verification Status**
It can be noted that most cases have been verified by LC while almost about half the number have also been source verified for income.
However, it is surprising that about 70% cases are not verified. Additional details might be required to probe further into this scenario.



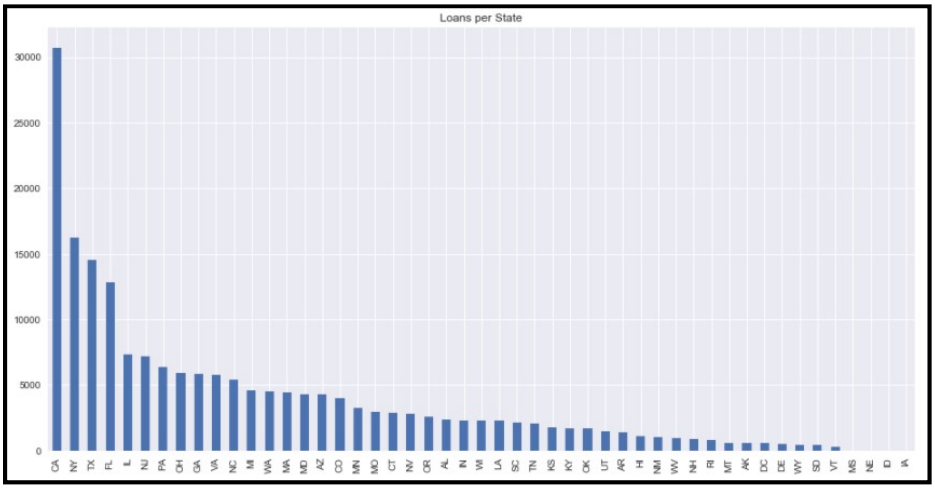

**Frequency of Loan Status**
From the graph, it looks like most customer of LC pay off their loans fully while about 20000+ were charged off and about 18000+ are current loans indicating the data could be highly skewed for analysis.



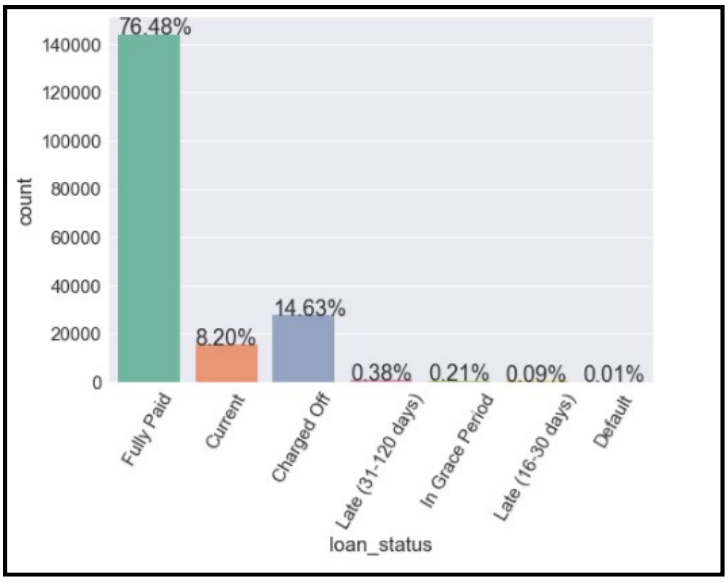

**Pie Chart of Home Ownership**
It is observed that almost 50% of loan requestors have their house mortgaged already, about 40% live in rented houses and only about 8% have their own houses. This could be a deciding factor in granting loan for some cases.



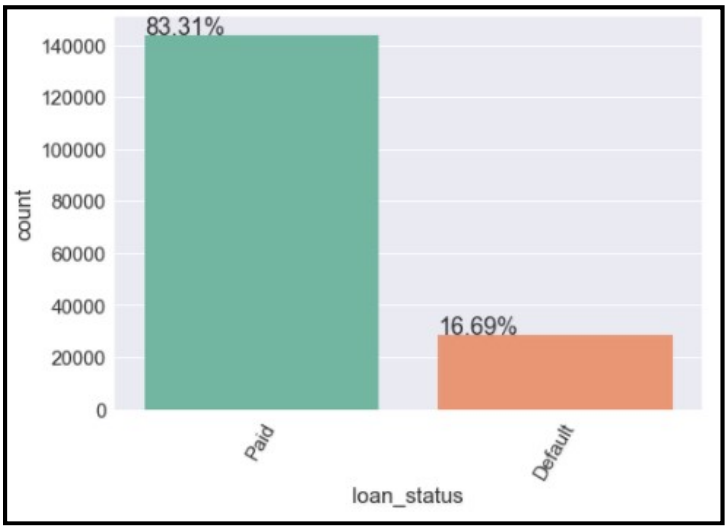

**Bar graph of Loan Purpose**
It is observed that debt consolidation is the major reason for requesting a loan with LC. Other prominent purposes include credit card, home improvement and other unexplained reasons.



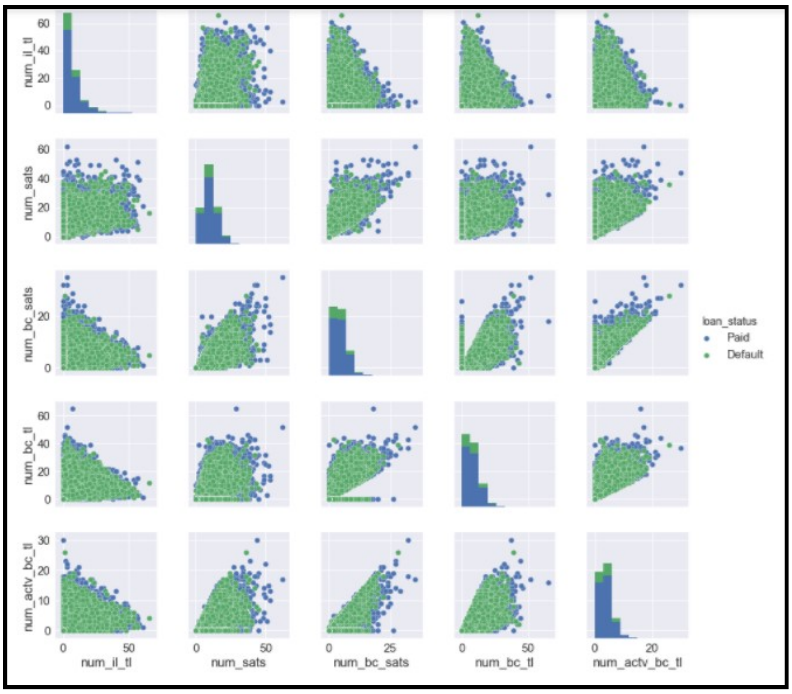

**Count of loans state-wise**
From the graph, it can be seen that California has the highest number of loans, followed by New York, Texas, Florida and Illinois. However, this feature is not included in analysis due to the large number of categories.



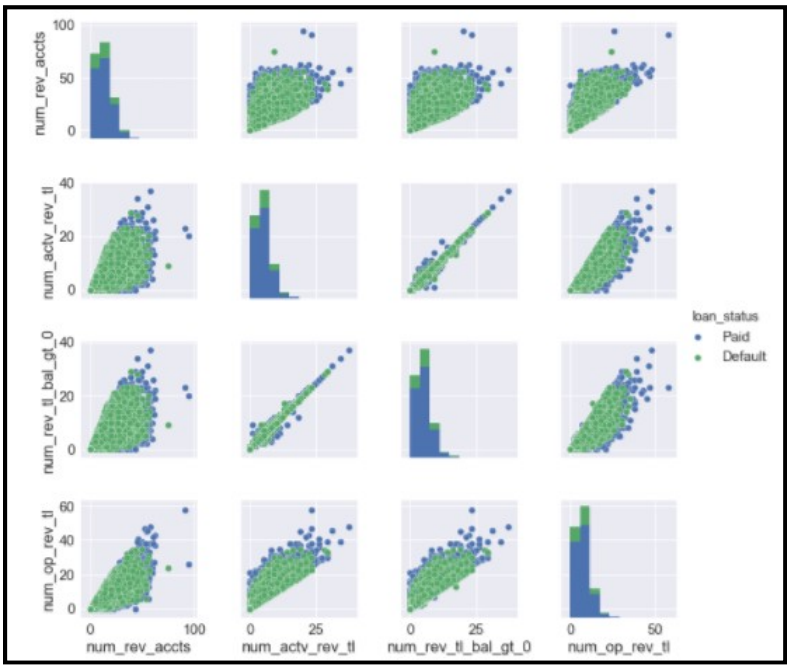

**Loan Status Percentage**
The default loans or bad loans to be considered for prediction seem to be quite less when compared to fully paid loans which are about 76%. Since this is actual data, it is quite possible that good loans are more in number than otherwise.



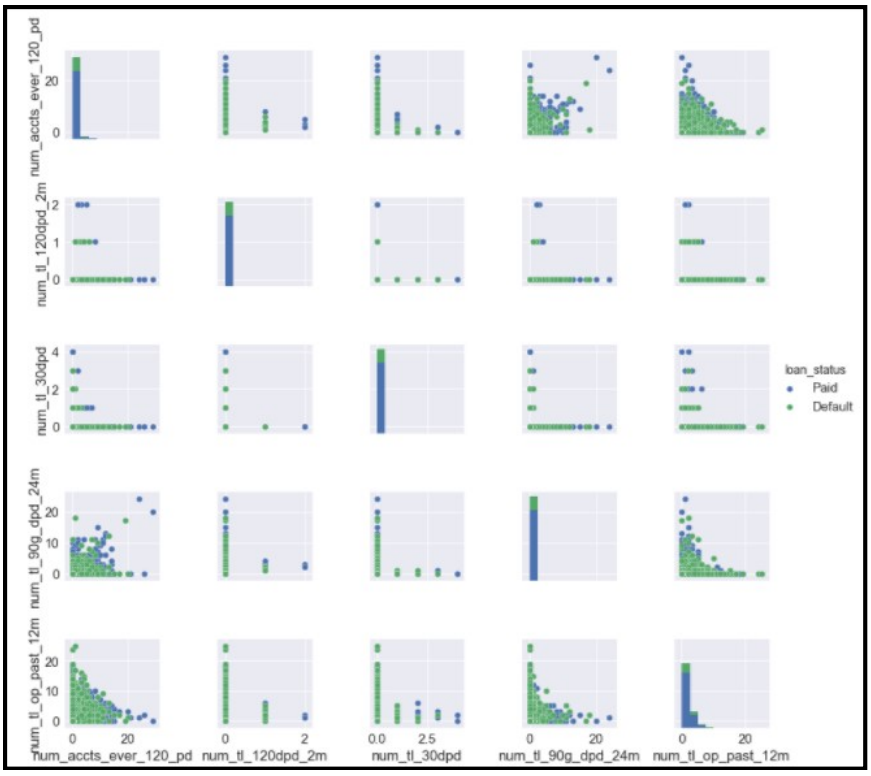

Status 'Fully Paid' has been taken as Paid and 'Charged Off', 'Default' and Late loans have been taken as Default category. Current loans have been removed from the dataset.
The new split for 100% of loans is shown in the graph here.




**Pairwise comparison of num_ fields**
This graph is the pair wise comparison of num_il_tl, num_sats, num_bc_sats, num_bc_il and num_actv_bc_tl. It can be seen that there are many outliers in the data apart from high collinearity.




**Pairwise comparison of num_ fields**
This graph is the pair wise comparison of features related to revolving accounts. It can be seen that there are many outliers in the data apart from high collinearity.




**Pairwise comparison of num_ fields**
This is the comparison of remaining num_fields with information on accounts opened, accounts past due etc. It is seen that most of them are highly skewed with minimum values for default category and also indicates the presence of outliers.

**Features considered for classification**
Features which had unary values, which had more than 70% missing values, which didn't have much information for the factor to be predicted (loan_status) and which leaked data from future (target_leakage) have been eliminated from the dataset.

After removing all the variables based on the above said conditions, I have about 47 variables left in the model for which I will be performing further analysis. The removed features and reasoning are stated as part of answer for Question 3.

**Proposed features**
Some new features that have been proposed in the model are :

**Fico_score**
Last_fico_score_high and low were data leakages as they contain the latest fico score for the user which will not be available at the time of predicting the loan. Hence they were removed from the model. Fico_high and fico_low were highly correlated and hence had to be removed and replaced by the average of both scores in a new column named fico_score.

**Num_accounts**
Most of the columns with num_ prefix are highly correlated and don't contain much useful information except for some like number of revolving accounts with balance >0, number of currently active bankcard accounts and number of satisfactory bankcard accounts. Since all of these are numeric columns denoting values, these have been combined for ease of prediction

and to avoid multicollinearity. Some values which had minimal data or were not useful, have been removed.

**Credit Age**

A new column called credit age can also be a useful indicator (last_credit_pull_d – earliest_credit_pull_d) will give the credit age value for the applicant and this could be an useful indicator in predicting if he will default or not.

*Note:* This feature has not been included in my current model though and is a proposed suggestion for enhancements on top of this.

**Question 3**

**Feature Selection and Optimal Combination of Features**

List of attributes which were used

```
loan_amnt, int_rate, annual_inc, dti, delinq_2yrs, inq_last_6mths,
open_acc, pub_rec, revol_bal, revol_util, total_acc, total_rev_hi_lim,
acc_open_past_24mths, avg_cur_bal, bc_open_to_buy, bc_util,
chargeoff_within_12_mths,delinq_amnt, mort_acc', mths_since_recent_bc,
mths_since_recent_inq, num_bc_tl, num_il_tl, num_tl_90g_dpd_24m,
num_tl_op_past_12m, pct_tl_nvr_dlq, percent_bc_gt_75,pub_rec_bankruptcies,
tax_liens, tot_hi_cred_lim, total_bal_ex_mort, total_bc_limit,
total_il_high_credit_limit, fico_score, num_accounts, term, grade,
home_ownership, verification_status, purpose, initial_list_status
```

**Sequence of how features were chosen and optimized for a combination suitable for modeling**

- Columns with more than 80% missing values were dropped. Since there were more 'not so useful' features, columns with more than 70% missing values were again dropped.
- Number of columns reduced to 91 and individual features were chosen in a batch of 10 for further analysis and exploration.
- Dropped id, member_id, sub_grade, funded_amnt, funded_amnt_inv, funded_amnt and funded_amnt_inv as some of these leak data from future and some don't have meaningful insights.
- Dropped emp_title, emp_length and url as these categorical features were not adding any value.
- Features payment_plan and issue_d leak data from future after loan approval and hence were removed.
- Employee title, zip_code, addr_state and earliest_cr_line were then dropped as none of applicant's employment or details like home address / zip code were used for prediction.
- Dropped out_prncp, out_prncp_inv, total_pymnt, total_pymnt_inv, total_rec_prncp, recoveries, collection_recovery_fee, last_pymnt_d, last_pymnt_amnt, last_fico_range_high, last_fico_range_low, last_credit_pull_d due to target leakage.

- Dropped policy_code (has value for 1.0 only), application_type(only had INDIVIDUAL) and acc_now_delinq(has mainly 0 and about 437 1.0) as features are not good to use in analysis.
- Features like mo_sin_old_il_acct, mo_sin_old_rev_tl_op, mo_sin_rcnt_rev_tl_op and mo_sin_rcnt_tl didn't have substantiable data.
- Loan_status was categorized as Paid and Default based on the criteria explained in exploration graph and assigned value of 1 for Paid and 0 for default.
- The below columns are highly correlated and had a lot of null values which were replaced and highly correlated or not useful features were removed.

| | | | |
|---|---|---|---|
| num_accts_ever_120_pd | num_bc_tl | num_rev_tl_bal_gt_0 | num_tl_90g_dpd_24m |
| num_actv_bc_tl | num_il_tl | num_sats | num_tl_op_past_12m |
| num_actv_rev_tl | num_op_rev_tl | num_tl_120dpd_2m | |
| num_bc_sats | num_rev_accts | num_tl_30dpd | |

- Also tot_cur_bal and installment have correlation values of more than 95% with avg_cur_bal and loan_amount respectively and hence have been removed.
- Then performed imputation of integer values by replacing with median and to categorical values by replacing with appropriate values.
- Remove null values from all integer as well as categorical variables.
- Divide the dataset into Dependent variable column (loan_status) and corresponding Independent variables.
- Create dummy variables for the categorical variables like 'term','grade', 'home_ownership', 'verification_status', 'purpose' and 'initial_list_status'.
- Perform MinMax standardization of the variables using Scaler.
- Split the data set into training and testing with 75% and 25% respectively.
- Import the required packages and perform cross validation for techniques like KNN, Naïve Bayes, Logistic Regression, Decision Tree and Random Forest.
- Details on accuracy scores are stated in the following section.

I also understand that methods like GridSearch can be used to get optimal number of parameters and best combinations for a given classifier which will give higher accuracy. I have not implemented it in the current model and will try this as part of additional enhancements.

**Question 4**
**Classification Accuracy for Classifiers – By doing 10 fold cross validation on training data**

Gaussian Naïve Bayes Cross Validation Scores

```
Model: Gaussian Naive Bayes
Precision = 0.882689605131
Recall    = 0.820508244059
F1 Score  = 0.85041044012
Accuracy  = 0.759622663204
```

Logistic Regression Cross Validation Scores

```
Model: Logistic Regression
Precision = 0.84653645399
Recall    = 0.981978569423
F1 Score  = 0.909240752893
Accuracy  = 0.836697237557
```

Decision Tree Cross Validation Scores

```
Model: Decision Trees
Precision = 0.864705176018
Recall    = 0.854605421986
F1 Score  = 0.859308036611
Accuracy  = 0.767171204765
```

K Nearest Neighbor Cross Validation Scores

```
Model: K Nearest Neighbor
Precision = 0.84610327928
Recall    = 0.94764056931
F1 Score  = 0.8939968216
Accuracy  = 0.81280152302
```

Random Forest Cross Validation Scores

```
Model: Random Forest
Precision = 0.85664990145
Recall    = 0.945768900024
F1 Score  = 0.898694568815
Accuracy  = 0.823398765853
```

Logistic regression seems to perform the best compared to other models in terms of accuracy, precision, recall and F1 scores. It has the highest accuracy of 83% and an F1 score of about 90%. Since both precision and recall also have reasonably high values, the model can predict defaulters better compared to other methods.
Also, since the data is highly biased and skewed with lesser values for Default category, it is difficult to exactly finalize on these models without performing additional parameter

optimization techniques to remove category imbalance, which is out of the scope for this project.

**Question 5**

**Accuracy Scores of Classifier after testing on the hold out data**

Gaussian Naïve Bayes Confusion Matrix and Classification Report

```
[[ 3185  4011]
 [ 6463 29529]]
             precision    recall  f1-score   support

          0       0.33      0.44      0.38      7196
          1       0.88      0.82      0.85     35992

avg / total       0.79      0.76      0.77     43188

('Training score is ', 0.76199223543758632)
('Testing score is ', 0.75747892933222194)
```

Logistic Regression Confusion Matrix and Classification Report

```
[[  722  6474]
 [  677 35315]]
             precision    recall  f1-score   support

          0       0.52      0.10      0.17      7196
          1       0.85      0.98      0.91     35992

avg / total       0.79      0.83      0.78     43188

('Training score is ', 0.83684385202565548)
('Testing score is ', 0.83442159859220155)
```

Decision Tree Confusion Matrix and Classification Report

```
[[ 2379  4817]
 [ 5171 30821]]
             precision    recall  f1-score   support

          0       0.32      0.33      0.32      7196
          1       0.86      0.86      0.86     35992

avg / total       0.77      0.77      0.77     43188

('Training score is ', 1.0)
('Testing score is ', 0.76873205520051868)
```

K Nearest Neighbor Confusion Matrix and Classification Report

```
[[  998  6198]
 [ 1900 34092]]
             precision    recall  f1-score   support

          0       0.34      0.14      0.20      7196
          1       0.85      0.95      0.89     35992

avg / total       0.76      0.81      0.78     43188

('Training score is ', 0.85495859157321152)
('Testing score is ', 0.81249421135500599)
```

Random Forest Confusion Matrix and Classification Report

```
[[ 1463  5733]
 [ 1957 34035]]
             precision    recall  f1-score   support

          0       0.43      0.20      0.28      7196
          1       0.86      0.95      0.90     35992

avg / total       0.78      0.82      0.79     43188

('Training score is ', 0.99256732246088775)
('Testing score is ', 0.82194127998518107)
```

On the testing data, it looks like Logistic regression again performs better as it has better precision for category 0 – Default, which means it is quite accurate in predicting the people who are likely to default.

**Question 2**

**Classifier with best results**

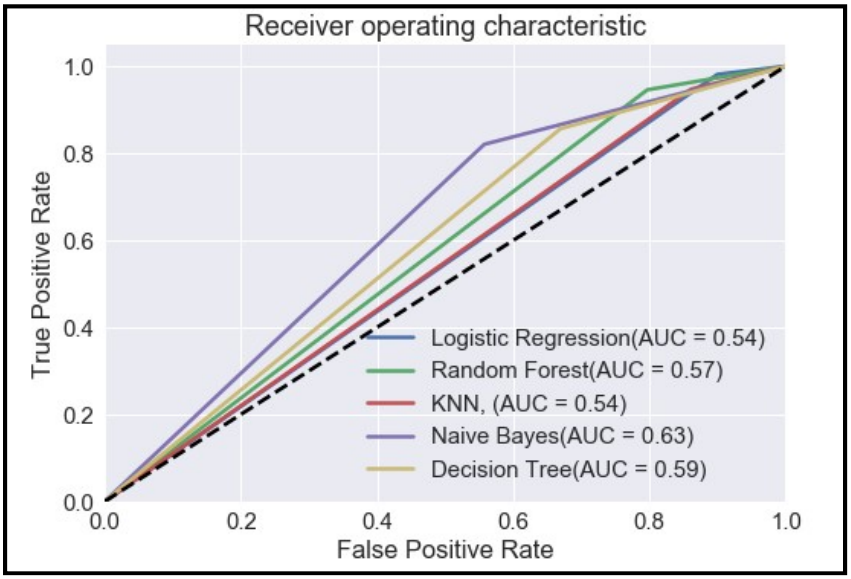
Based on the classifier scores and confusion matrix values for all the classifiers – Logistic Regression, Naïve Bayes, Decision Trees, Random Forest and K Nearest Neighbor, it is seen that Logistic Regression performs well on both training and test datasets. This is followed by Naïve Bayes classifier.

**Analysis based on AUC values in ROC curve**

Area under the curve values for all classifiers

```
Area under the ROC curve for Logistic Regression : 0.540762
Area under the ROC curve for Random Forest : 0.574467
Area under the ROC curve for KNN classification : 0.542949
Area under the ROC curve for Naive Bayes : 0.631520
Area under the ROC curve for Decision Tree : 0.593465
```

ROC graph for all classifiers




When looking at the ROC curve graph above for all classifiers with Area under the curve values, it looks like Naïve Bayes performs better compared to other model as it has maximum AUC value.