Default Rate Prediction for Lending Club Data

For details, please refer to my Github. If GitHub fails to load the jupyter notebook, nbviewer might be a great help.

Author: Pengfei Li

1. Data Description

The data is provided by the famous the peer to peer lending and alternative investing company, Lending Club. The file is in a tidy excel format containing loan data for all loans from 2007 to 2018 Q2, up to now. Besides some profile data, the most important features are the loan status (Current, Late, Fully Paid, etc.) and latest payment information.

With the data at hand, I am going to build an imbalanced binary classification model for default (Current and Fully Paid) and not default (Otherwise).

2. Default Rate Prediction

2.1 Problem Definition

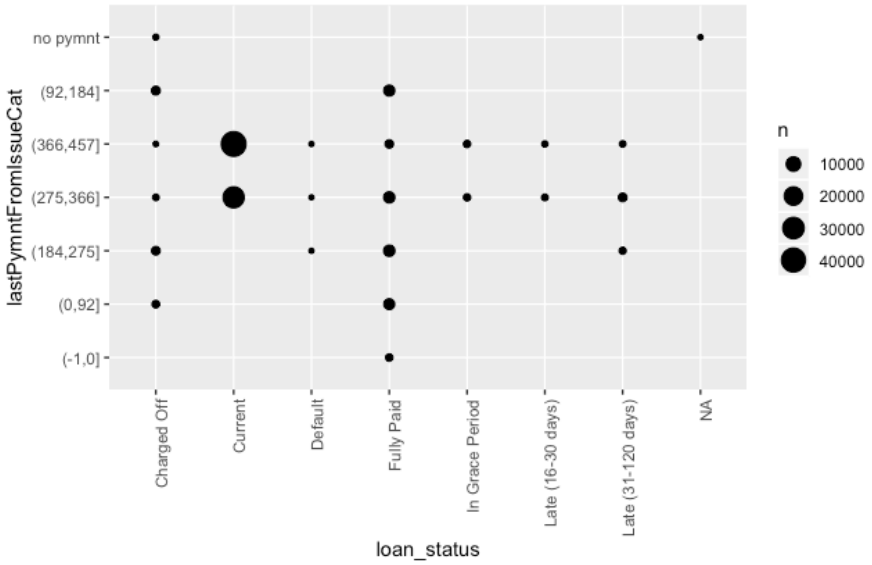
Default rate prediction is slightly analogous to the fraud detection, one of the top priorities for banks and financial institutions. It can also be addressed using imbalanced machine learning techniques. Thus, it can help the company minimize potential default risk and get a comprehensive understanding of connections of user behaviors.

2.2 Data Preprocessing ('data.table' package in R) (Section 2.1 in the R script)

Necessary string manipulations are performed to cleanse the data.

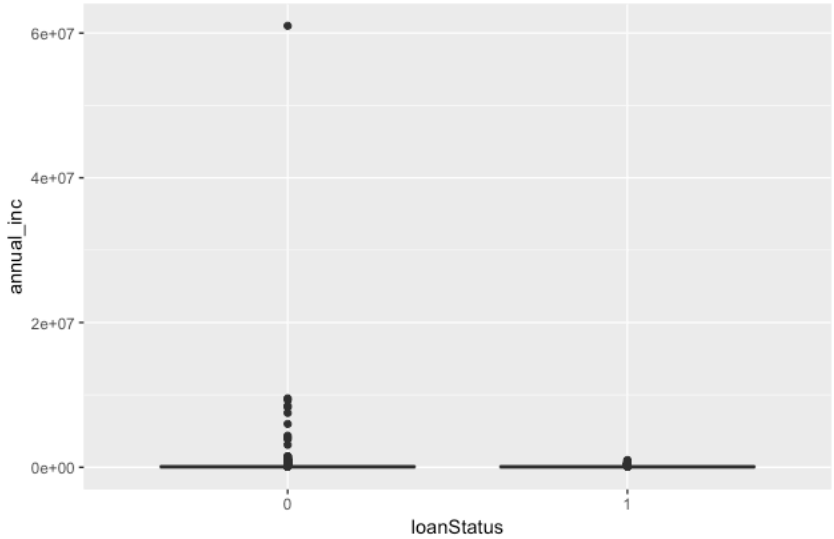
2.2.1 Data Cleansing and transformation

Though the original data is in a clean format (csv format), necessary transformations need to be done to get a dataset for modelling.



The image above helps understand the meaning of the loan status, or to 'validate' the loan status. The better the status is, the earlier the last payment is recorded.

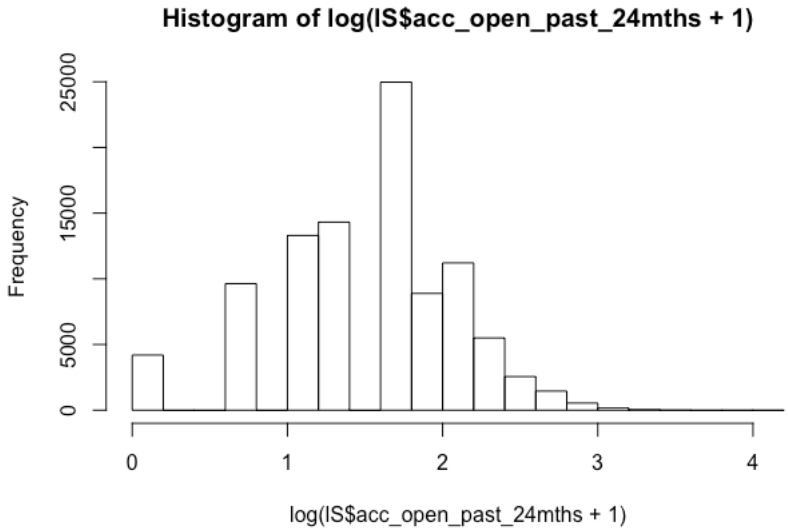
a. The date-format features need to be grouped into categorical variables (by 'zoo' package). They are first converted to number of days from the issue date and missing values are imputed by its median. Lastly, they are grouped into 3 categories by the bins: minimum, 10%, 90%, maximum.
b. The address states and zip codes are categorized by their corresponding default rates.
c. The '_joint' features are treated as the supplement of the its related features.
d. Remove catetgorical features with 1 level
e. There are also some potential outliers in some feature. For example, in the annual outcome,



We can see some exaggeratedly high annual income in the bad loan. It is not reasonable and should be removed.

f. For missing values, there are mainly 3 patterns: 'mths_since_.*', 'sec_app_.*', ' hardship' and 'settlement' features, which are handled in the r script.

g. Some features have a huge proportion of missing values (>=98%). Those features are grouped, and the missing values consists of a new level.
h. Log(x+1) transformation is done on the variables with a large range



The above is the log transformation result of the number of accounts opened in the past 24 months. It turn outs to be well bell-shaped.

2.2.2 Default Label

0	1
89496	7287

This is a loan data including around 100 thousand records. We treat the Current and Fully Paid loan as not default loan, denoted as 0, 89496 records, and everything else as default loan, denoted as 1, 7287 records, whose ratio is almost 92:1. Thus, an imbalanced binary classification model can be built to predict the default rate of every loan

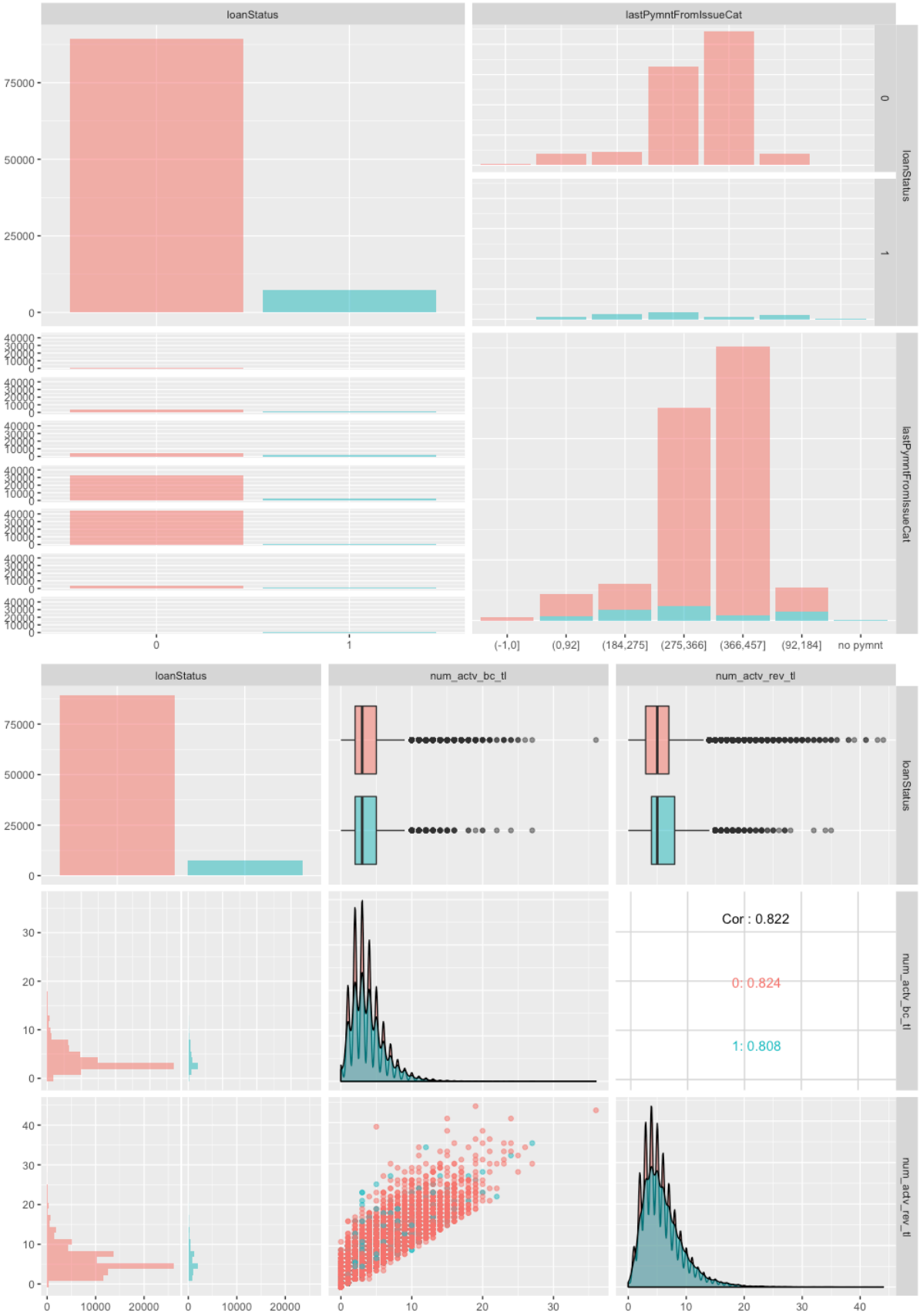
2.3 Feature Exploration ('GGally' and 'corrplot' package in R)

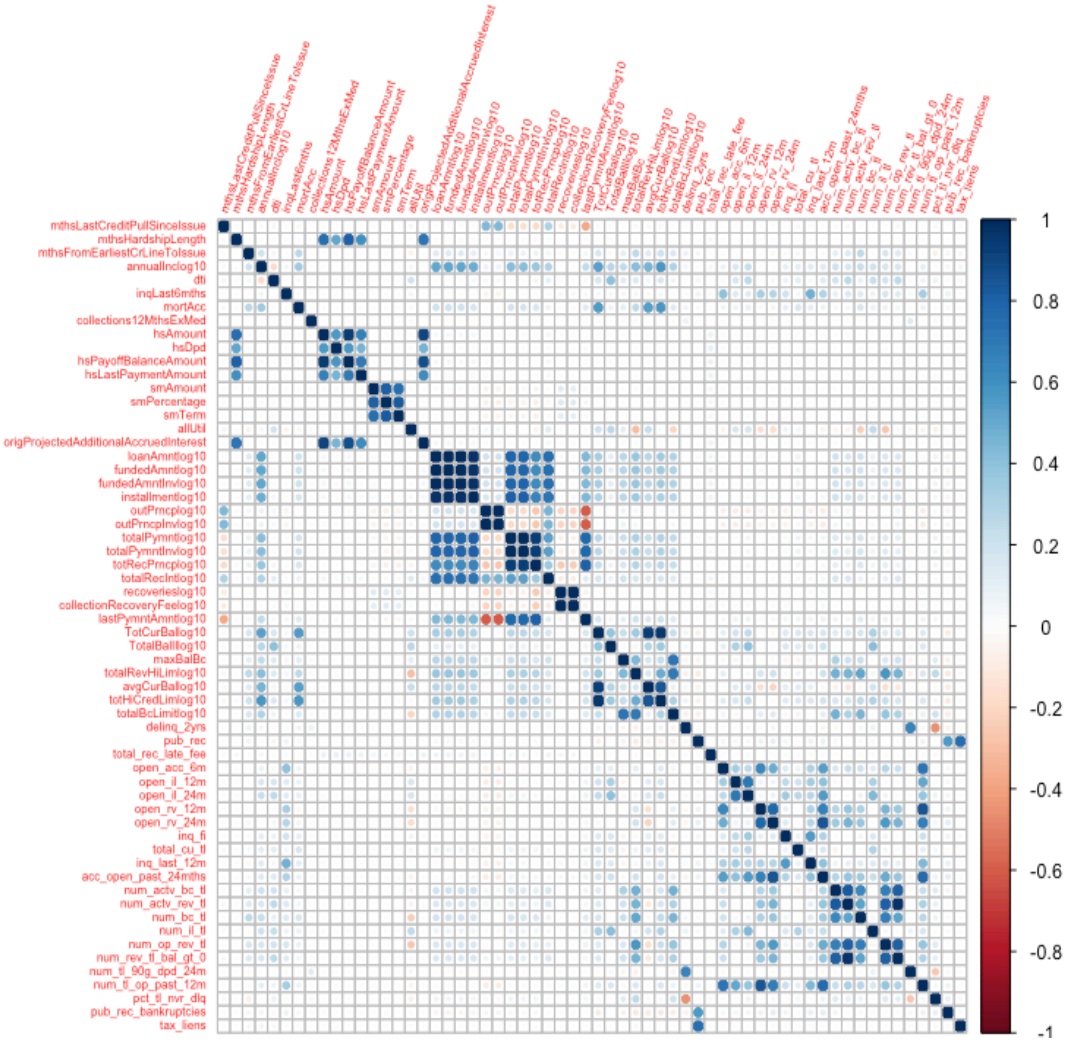
A simple features sifting is performed by the picking the variables with p-value greater than 0.05. Since our response is categorical, all our features are either continuous or discrete after 2.2.1, the statistical testing is t-test for the continuous ones and chi-sq test for discrete ones.

After all the very complicate procedures, we got a date-set with 96783 rows and 103 columns out of the original 2017Q1 data.

The following are 3 example plots for data explorations.

a. loan status VS the last payment from issue date (categorical)
b. loan status VS the number of currently active bankcard accounts
c. correlation plot of all numerical features


loanStatus
lastPymntFromIssueCat
0
1
(-1,0]
(0,92]
(184,275]
(275,366]
(366,457]
(92,184]
no pymnt
num_actv_bc_tl
num_actv_rev_tl
Cor : 0.822
0: 0.824
1: 0.808



2.4 Model (scikit-learn, imblearn package in python)

Two ways are considered to apply in this imbalanced class machine learning problem. One is called weight-penalized machine learning method, or case-sensitive training. It is realized by tuning the weights parameters in the function of the related machine learning package.

The other is using the 'imblearn' package to build a pipeline: resampling method + ML. The resampling methods consist of over-sampling, under-sampling, etc. Details can be found in the manual of 'imbalanced-learn' package. ML is just the machine learning method in 'scikit-learn' package.

Both two ways are tested on the dataset and parameters are tuned for the best performance by precision-recall curve.

2.5 Model Validation (Grid Search for pipeline model)

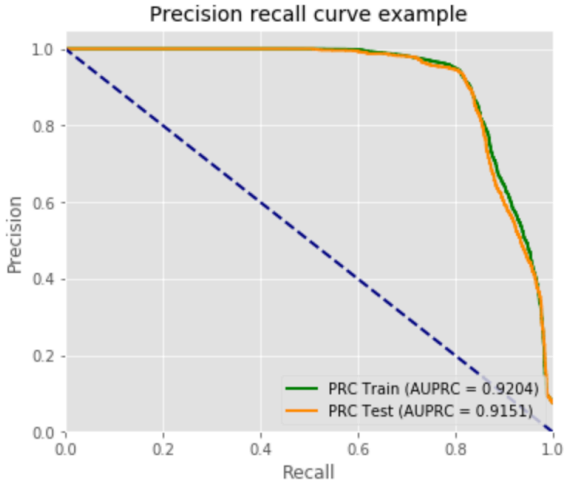
The reason using AUPRC instead of AUC is as follows.

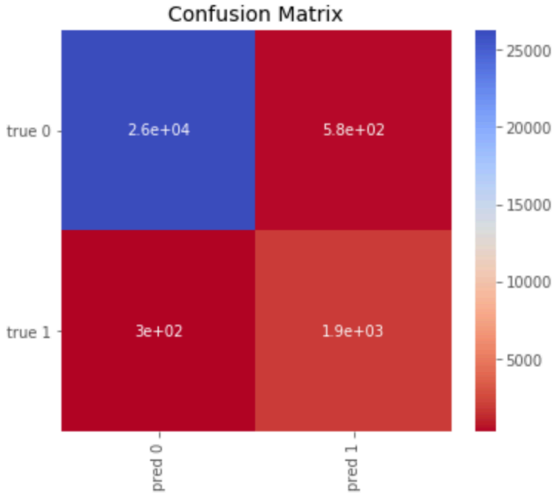
We know that ROC curve is TPR (sensitivity, or true positive rate) vs FPR (false positive rate), while precision-recall curve is precision (positive predicted value) vs recall, i.e. sensitivity, TPR.

We can see that ROC cares both true positive and true negative. And precision-recall curve is only concerned about TP over both condition positive and predicted condition positive.

Thus, two models are determined below. The first is recall-driven, SMOTE + logistics, which takes short to fit. If we do not wish to miss many bad loans, we should choose this.

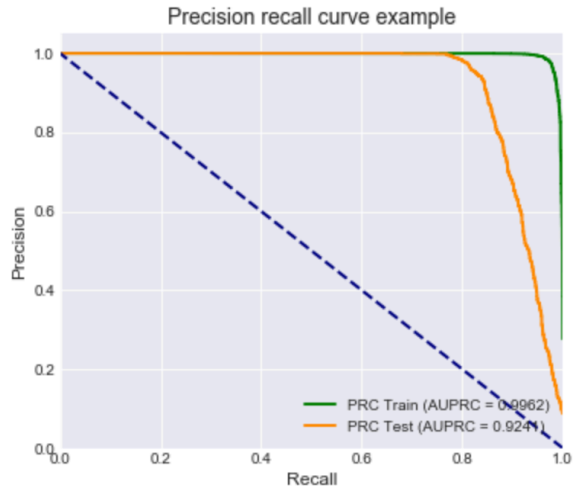
metrics	train	test
AUPRC	0.920359	0.915103
Precision	0.770877	0.765854
Recall	0.869164	0.862243
f1-score	0.817075	0.811195
AUC	0.977688	0.977499
Accuracy	0.970714	0.969794

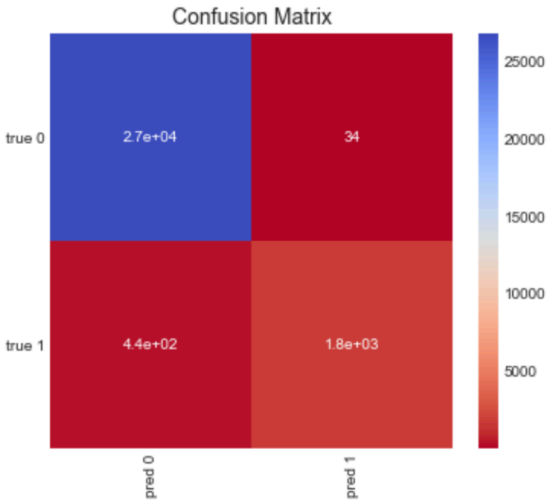




The second is weights-penalized xgboost, which is precision-driven. It is relatively time-consuming. If we just we want more what we predict as bad loans are really bad loans, i.e. a higher positively predicted rate, we ought to choose this.

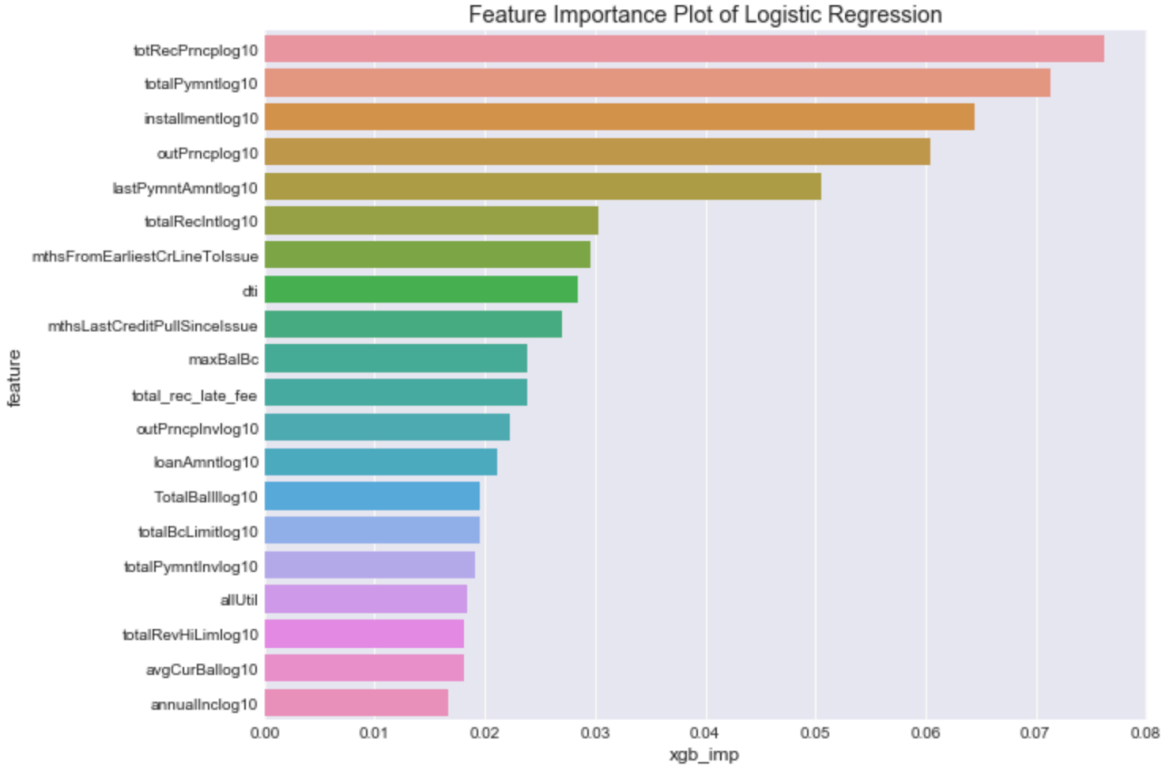
metrics	train	test
AUPRC	0.996196	0.924100
Precision	0.999555	0.980942
Recall	0.882111	0.800915
f1-score	0.937168	0.881834
AUC	0.999544	0.977925
Accuracy	0.991099	0.983847





2.6 Insight Identification

2.6.1 Feature Importance



We can generate the features importance plot in the second model. The top five importance features for default loan rate prediction are
total principles received to date,

total payment received to date,
the monthly payment owed by the borrower if the loan originates,
Remaining outstanding principal for total amount funded,
Last payment amount.

2.6.2 Default Rate Result Analysis

Machine learning is instrumental in solving default rate prediction problem.
https://www.marutitech.com/machine-learning-fraud-detection/

Pros:

a. Speed: human process is time-consuming and involves manual interaction
b. Scale: more effective with large scales of data
c. Efficiency: ML can often be more effective than humans at detecting subtle or non-intuitive patterns to help identify fraudulent transactions.

Cons:

a. Lack of inspectability

b. Cold Start: Without the appropriate data, the machines may learn the wrong inferences and make erroneous or irrelevant fraud assessments
c. Blind to connections in data.

All in all, from all analysis above, besides standard criteria, like the credit history, we could predict a much more accurate probability of default loan. So, the company can take necessary actions to prevent the risk. One might strengthen the manual review for the highly-risk lender. One can also get a real time analysis for those who are currently in a loan and especially focus on the high-risk loan.