FINAL PROJECT
MAFS6010U (L1) – Spring 2020
Home Credit Default Risk

DO Van Thuat
student ID: 20405634
Department of Mathematics, School of Science
Hong Kong University of Science and Technology
`tdovan@connect.ust.hk`

May 30, 2020

Abstract

In this project, we will data science and machine learning techniques to predict credit default risk which is very important in Fintech industry, banking and financial sector. The dataset is from "Home Credit Defaul Risk" competition on Kaggle, https://www.kaggle.com/c/home-credit-default-risk.

1 Overview and Introduction

In the world of finance, Default Risk is the probability that companies or individuals will be unable to make the required payments on their debt obligations. To rate the risk by classical methods, lenders have to evaluate every single application. This process takes a lot of time and workload if the number of borrowers is large as thousands.

Many Credit rating companies in the world strive to broaden financial inclusion for the unbanked population by providing a positive and safe borrowing experience. In order to make sure this underserved population has a positive loan experience, Credit rating institutes make use of a variety of alternative data–including telco and transactional information–to predict their clients' repayment abilities.

While their companies are currently using various statistical and machine learning methods to make these predictions, they're challenging Data Scientists to help them unlock the full potential of their data. Doing so will ensure that clients capable of repayment are not rejected and that loans are given with a principal, maturity, and repayment calendar that will empower their clients to be successful.

The breakthrough of Machine Learning (ML) can learn from an abundant dataset of previous credit applications, then gives super-fast and accurate credit evaluation which is hundreds of times more efficient than processing by the human. Thus, ML has become an indispensable tool of modern credit risk modeling.

1.1 Data Description

The data set contains 7 tables illustrating the information of clients who applied into Home Credit Default Risk company and the client's credit application history. The data is provided by Home Credit (http://www.homecredit.net/about-us.aspx), a service dedicated to providing lines of credit (loans) to the unbanked population. Predicting whether or not a client will repay a loan or have difficulty is a critical business need, and Home Credit is hosting this competition on Kaggle to see what sort of models the machine learning community can develop to help them in this task.

There are 7 different sources of data as followed.

- Application train and application test: the main training and testing data with information about each loan application at Home Credit. Every loan has its own row and is identified by the feature ID. The training application data comes with the "TARGET" indicating 0 (the loan was repaid) or 1 (the loan was not repaid).

- Bureau: All client's previous credits provided by other financial institutions that were reported to Credit Bureau (for clients who have a loan in the sample). For every loan in the sample, there are as many rows as number of credits the client had in Credit Bureau before the application date.

- Bureau balance: monthly data about the previous credits in bureau. Each row is one month of a previous credit, and a single previous credit can have multiple rows, one for each month of the credit length.

- Previous application: previous applications for loans at Home Credit of clients who have loans in the application data. Each current loan in the application data can have multiple previous loans. Each previous application has one row and is identified by the feature ID.

- POS CASH BALANCE:Monthly balance snapshots of previous POS (point of sales) and cash loans that the applicant had with Home Credit. This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in the sample.

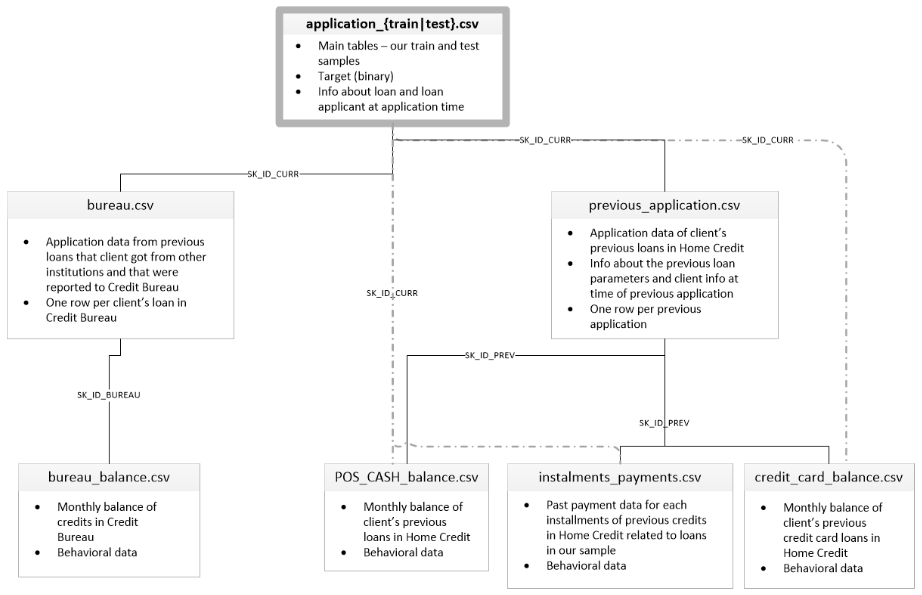
- Credit card balance: Monthly balance snapshots of previous credit cards that the applicant has with Home Credit. Each row is one month of a credit card balance, and a single credit card can have many rows.

- Installments payment: payment history for previous loans at Home Credit. There is one for every missed payment.

The details of data analyzing and exploring techniques as well as feature extraction and selection approaches will be shown in the later sections.

The general definitions of all the columns is provided in the csv file namely: "HomeCredit_columns_description.csv".

The below diagram shows how all of the data is related.


application_{train|test}.csv
Main tables – our train and test samples
Target (binary)
Info about loan and loan applicant at application time
SK_ID_CURR
SK_ID_CURR
SK_ID_CURR
SK_ID_CURR
bureau.csv
Application data from previous loans that client got from other institutions and that were reported to Credit Bureau
One row per client's loan in Credit Bureau
previous_application.csv
Application data of client's previous loans in Home Credit
Info about the previous loan parameters and client info at time of previous application
One row per previous application
SK_ID_PREV
SK_ID_BUREAU
SK_ID_PREV
bureau_balance.csv
Monthly balance of credits in Credit Bureau
Behavioral data
POS_CASH_balance.csv
Monthly balance of client's previous loans in Home Credit
Behavioral data
instalments_payments.csv
Past payment data for each installments of previous credits in Home Credit related to loans in our sample
Behavioral data
credit_card_balance.csv
Monthly balance of client's previous credit card loans in Home Credit
Behavioral data

1.2 Problems

The objective of the competition is to use historical loan application data to predict whether or not an applicant will be able to repay a loan. This is a standard supervised classification task:

Supervised: The labels are included in the training data and the goal is to train a model to learn to predict the labels from the features

Classification: The label is a binary variable, 0 (will repay loan on time), 1 (will have difficulty in repaying loan).

2 Feature Engineering

In this section, we will give the answer to the question known as how to find the final data set for fitting Machine Learning models. Generally, we take the process of data analyzing and feature selection at first to get a better insight about what data we have. After that based on the explanation in Fico's 5 factors, we have a theoretical understanding of the components of rating a credit score, and from that, we generate more features from the initial data set to develop and extend more details about the knowledge domain of credit scoring.

To do feature selection from each data table, we expand the dataset following Exploratory Data Analysis (EDA) techniques to get a better understanding about the figures of descriptive statistics and applying data-plotting function for geometric analyzing. EDA is an open-ended process where we calculate statistics and make figures to find trends, anomalies, patterns, or relationships within the data. The goal of EDA is to learn what our data can tell us. It generally starts out with a high-level overview, then narrows into specific areas as we find intriguing areas of the data. Moreover, we also compute the correlation matrix to analyze and investigate the relationship or correlation coefficients between features in this step. A correlation matrix is not solely used as a way to summarize data, but also works as an input into a more advanced analysis, and as a diagnostic for advanced analyses.

Feature extraction as well as feature generation method starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations. Feature extraction is a dimensionality reduction process, where an initial set of raw variables is reduced to more manageable groups (features) for processing, while still accurately and completely describing the original data set. In this step, we also approach the feature generating process and extend new features based on internal data or external data, hence, the initial data set will be transformed into a better dimension and give numerous pieces of better information.

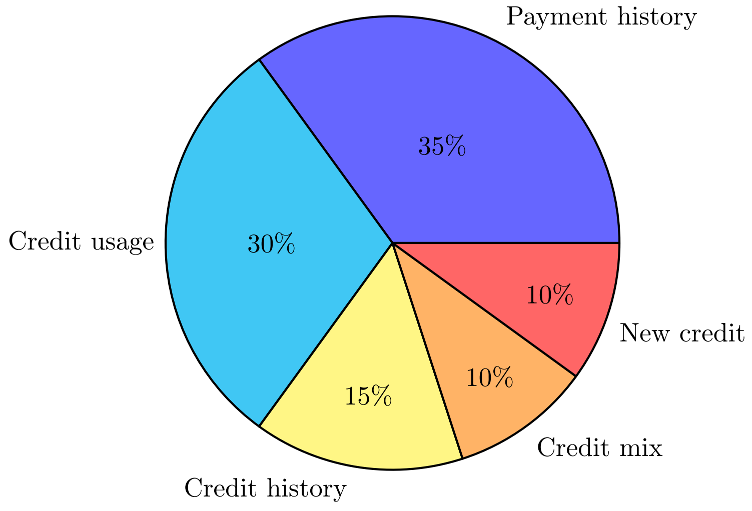
By approaching the knowledge of rating credit score in Fico's theory (the details are shown in the next section), we can expand and generate more domain knowledge features by taking several data generation techniques:

- Taking transformation: Square, Square-Root, Power, Max, Min, Median, Mean, Variance, etc.
- Feature Calculation: Sum, Minus (Difference), Ratio (Division), Multiplication, Power, etc.
- Groupby: Category feature, Time-based feature, Time-series feature, group of features extending Trending info, group of features extending Social Network info.

Additionally, when exploring and cleaning the data, we tend to keep all features and data points. Because XGBoost can estimate scores of features based on Random Forest framework and handle with missing values, thus we can apply this Machine Learning Algorithm to point out important features supporting the selection and analyzing process without replacing the missing value entries in the tables.

2.1 FICO's factors and Domain Knowledge Feature

FICO is the biggest name in town when it comes to credit scores. Most major card issuers and lenders in the U.S. use FICO's traditional model to decide whether to extend credit to consumers and at what interest rate. According to the company's website, 90 percent of all lending decisions in the U.S. use FICO scores, and more than 27 million scores are sold each day. Using the information in a borrower's [credit report](http://www.creditcards.com/glossary/term-credit-report.php), FICO breaks that information into categories. Those five components each get different weights.



2.2 Handling with the dataset

After exploring the initial data set by taking EDA, we can separate the dataset into 3 main types of features: normalized features, categorical features and continuous features. By calculating correlation, we can see relationship amongst features and relation to how an applicant repaying a loan. The relationship is not very strong (in fact that they are all considered very weak, but these variables will still be useful for a machine learning model to predict whether or not an applicant will repay a loan on time. If some features have negative correlations with the target, indicating that the client is more likely to repay the loan. Furthermore, we can extend the information into many valuable figures in credit scoring domain by taking transformation or making polynomial features.

Generating new domainn knowledge features. To increase details in credit utilization (usage) side is shown in the 2nd Fico's factor, we can extend several features to some financial ratio for showing more info about client's financial status, available money or potential income. We can also apply feature groupby to expand Social Network info.

3 Modelling

After finishing the data engineering process, we can move to the next step as fitting model and tuning model. As mentioned above, the problem is classification, thus we can immediately think about several models such as logistic regression, decision tree, and random forest classification. However, to handle this big data set problem, we should approach to the more powerful and state-of-the-art model, therefore, we choose two models to solve this problem, there are XGBoost and LightGBM.

XGBoost (eXtreme Gradient Boosting) is a scalable and accurate implementation of gradient boosting machines and it has proven to push the limits of computing power for boosted trees algorithms as it was built and developed for the sole purpose of model performance and computational speed. Specifically, it was engineered to exploit every bit of memory and hardware resources for tree boosting algorithms. The algorithm of XGBoost is level-wise free growth.

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks.

3.1 Algorithm comparison

XGBoost (level-wise) and Light-GBM (leaf-wise), both of these algorithms can be considered a kind of approach under the framework namely Gradient Boosting Decision Trees (which we will henceforth be referred to as GBDTs). Leaf wise splits lead to increase in complexity and may lead to over fitting and it can be overcome by specifying another parameter max-depth which specifies the depth to which splitting will occur. Below, we will see the steps to install Light GBM and run a model using it. We will be comparing the results with XGBoost results to prove that you should take Light GBM in a "LIGHT MANNER". While other algorithms grow trees horizontally, Light GBM grows tree vertically meaning that Light GBM grows tree leaf-wise while other algorithms grow level-wise. It, in order to grow, will choose the leaf that has a max delta loss. When growing the same leaf, leaf-wise algorithm can reduce more loss when compared to a level-wise algorithm.

3.2 Finding the Best Split

The key challenge in training a Gradient Boosting Decision Trees (GBDT) is the process of finding the best split for each leaf. Modern datasets tend to be both large in the number of samples and the number of features. For instance, a term frequency – inverse document frequency matrix of a million documents with a vocabulary size of 1 million would have trillion entries. Thus, a naive GBDT would take forever to train on such datasets. There is no method that can find the best split while avoiding going through all features of all data points. Therefore, the various methods that XGBoost and Light-GBM present are methods of finding the approximate best split, including Histogram-based methods, Ignoring sparse inputs, Subsampling the data, Exclusive Feature Bundling (Light-GBM).

3.3 The Improvement

Generally, LightGBM improves on XGBoost. The LightGBM paper uses XG-Boost as a baseline and outperforms it in training speed and the dataset sizes it can handle. The accuracies are comparable. LightGBM in some cases reaches its top accuracy in under a minute and while only reading a fraction of the whole dataset. This goes to show the power of approximation algorithms and intelligently sampling a dataset to extract the most information as fast as possible.

3.4 Tuning Model

There are four of the most popular approaches to tuning the hyper-parameters of a machine learning model.

- *Manual*: select hyper-parameters based on intuition/experience/guessing, train the model with the hyper-parameters, and score on the validation data. Repeat process until you run out of patience or are satisfied with the results.
- *Grid Search*: set up a grid of hyper-parameter values and for each combination, train a model and score on the validation data. In this approach, every single combination of hyper-parameters values is tried which can be very inefficient.
- *Random search*: set up a grid of hyper-parameter values and select random combinations to train the model and score. The number of search iterations is set based on time/resources.
- *Automated Hyper-parameter Tuning*: use methods such as gradient descent, Bayesian Optimization, or evolutionary algorithms to conduct a guided search for the best hyper-parameters.

4 Conclusion

4.1 Metric: ROC AUC

The Area Under the Curve (AUC) explains itself by its name, which is simply the area under the ROC curve. (This is the integral of the curve.) This metric is between 0 and 1 with a better model scoring higher. A model that simply guesses at random will have an ROC AUC of 0.5.

When we measure a classifier according to the ROC AUC, we do not generation 0 or 1 predictions, but rather a probability between 0 and 1. This may be confusing because we usually like to think in terms of accuracy, but when we get into problems with inbalanced classes (we will see this is the case), accuracy is not the best metric. For example, if I wanted to build a model that could detect terrorists with 99.9999% accuracy, I would simply make a model that predicted every single person was not a terrorist. Clearly, this would not be effective (the recall would be zero) and we use more advanced metrics such as ROC AUC or the F1 score to more accurately reflect the performance of a classifier.

Therefore, a model with a high ROC AUC will also have a high accuracy, but the ROC AUC is a better representation of model performance.

4.2 Final results

Process	AUC scores
Only application.csv	0.69
full data	0.72
Extended data-1	0.76
Extended data-2	0.78
Extended data and Optimized parameters	0.797

4.3 Future Works

Our work is just a scratch on the surface of the Data Science and AI applications for financial industry. There are numerous things to do which begins from thinking again about the feature engineering and extraction to model selection and tuning parameters. To continue improving the solution of this problem, we write out some work for the future:

- Research more info and knowledge about feature in credit scoring domain.
- Collecting and exploring more data in credit rating framework.
- Improving computer (CPU/GPU) to fitting model in a larger dataset.
- Strengthen experiment and theoretical knowledge about algorithms and models to have a wider understanding and can modify parameters in a better way.

References

- See my Python codes on Github: https://github.com/thuat86/6010U-final-project.
- See my PowerPoint presentation on Google Drive: https://drive.google.com/file/d/1GG-tDsTlAIar0XDwsMiKgQINggPNUwBb/view.