Repayment Prediction of Overdue Payments of Online Loan Platform Users

ECE 6143
Final project

Yipeng Qu (N12062723)

Yan Zhou(N19862016)

Abstract

Nowadays users' dishonesty incidents occur frequently with more and more online lenders appear. Therefore it becomes more important to know how to lower the probability of this kind of event. Our project is based on the data set of lending club . We preprocess the data at first, then do feature engineering. Moreover, we will predict the probability of the events of past-due payment with the algorithms of LogisticRegression, SVM and LightGBM. At last, we will use ROC-AUC to evaluate the effectiveness of the algorithm.

Milestones

The implement of our project could be separated into 4 steps:

1. Collect the raw data.
2. Get featured data by cleaning the raw data, filling the missing values and doing feature engineering.
3. Get the predicted data by using the algorithms of LogisticRegression, SVM, and LightGBM
4. Evaluate each predicted result from each model by using ROC-AUC.

Data

Lending Club Dataset:

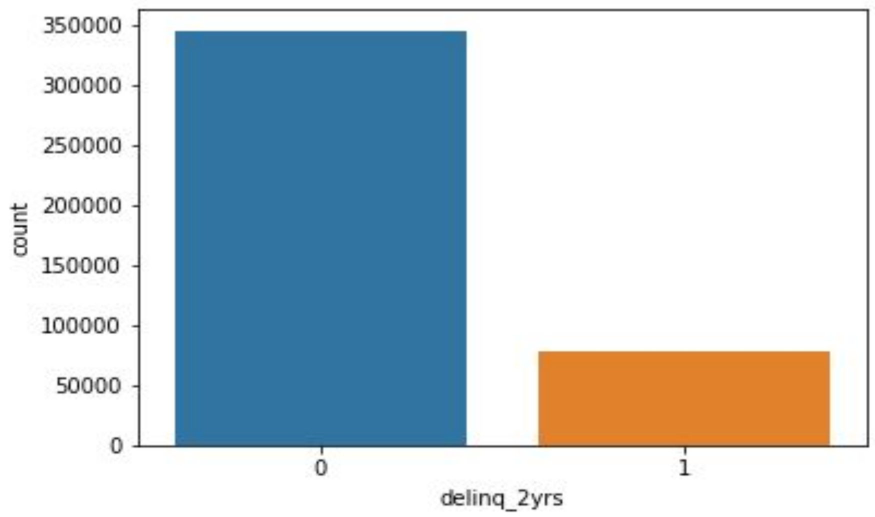
- Features：A total of 119 original features, including user basic information and user loan records.
- Label：Whether there has been a 30-day overdue repayment event in the past two years, 1 or 0.

Data preprocessing

- Drop useless columns
  - The missing percentage is too high
  - There’s only one value
  - Overfitting
- Text all in lowercase
- Filling missing value
  - Categorical：fill with mode
  - Numerical：fill with mean

EDA (Exploratory Data Analysis)

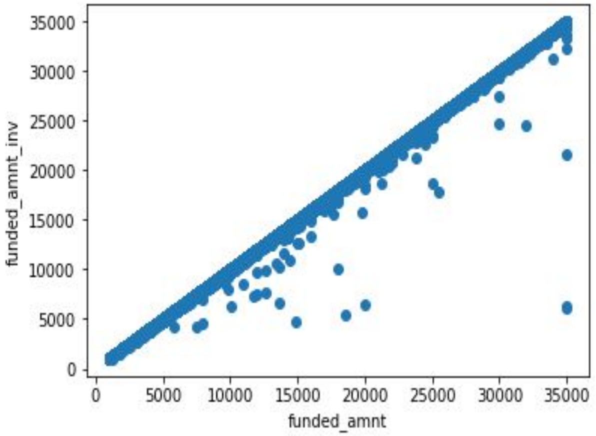
- Positive and negative sample ratio
- AUC metric is the best choice for class imbalance

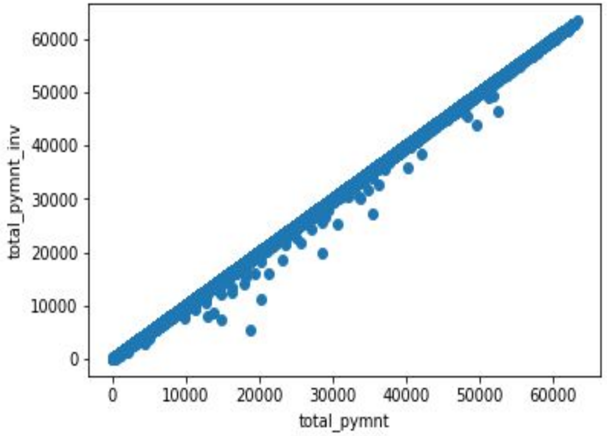


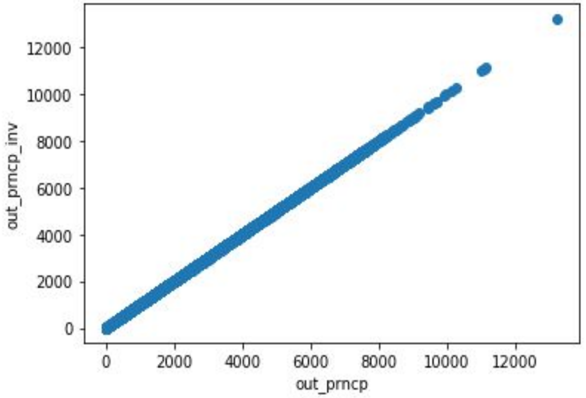
Drop redundant columns

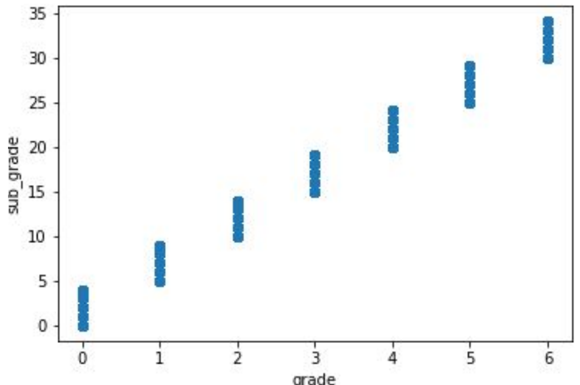
- “funded_amnt” has a strong linear correlation with “funded_amnt_inv”, so drop “funded_amnt”.

- Drop “total_pymnt”, “out_prncp” and “grade”









Feature Engineering

- Logistic regression and SVM
- Categorical —— one hot encoding
- Numerical —— standard scaling

$$x' = \frac{x - \bar{x}}{\sigma}$$

Where x is the original feature vector, $\bar{x} = \mathbf{average}(x)$ is the mean of that feature vector, and σ is its standard deviation.

Standard scaling

Model

- Logistic Regression

Logistic function: $f(z) = 1/(1 + e^{-z})$

- SVM

Given data $(\boldsymbol{x}_i, y_i)$

Optimization $\min_{w,b} J(\boldsymbol{w}, b)$

$$J(\boldsymbol{w}, b) = C \sum_{i=1}^{N} \max(0, 1 - y_i(\boldsymbol{w}^T \boldsymbol{x}_i + b)) + \frac{1}{2} ||\boldsymbol{w}||^2$$

Hinge loss term
Attempts to reduce
Misclassifications

C controls final margin

margin=$1/||\boldsymbol{w}||$

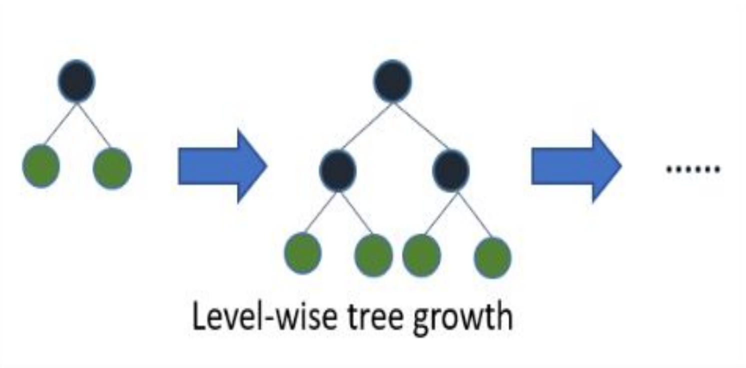
Model

- LightGBM

Optimization in Accuracy

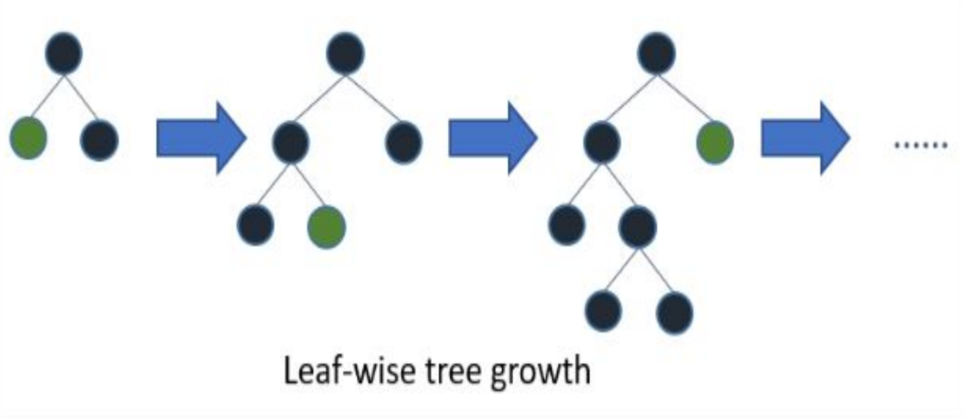
Leaf-wise (Best-first) Tree Growth

Most decision tree learning algorithms grow trees by level (depth)-wise, like the following image:



LightGBM grows trees leaf-wise (best-first)[7]. It will choose the leaf with max delta loss to grow. Holding `#leaf` fixed, leaf-wise algorithms tend to achieve lower loss than level-wise algorithms.

Leaf-wise may cause over-fitting when `#data` is small, so LightGBM includes the `max_depth` parameter to limit tree depth. However, trees still grow leaf-wise even when `max_depth` is specified.



Loss function

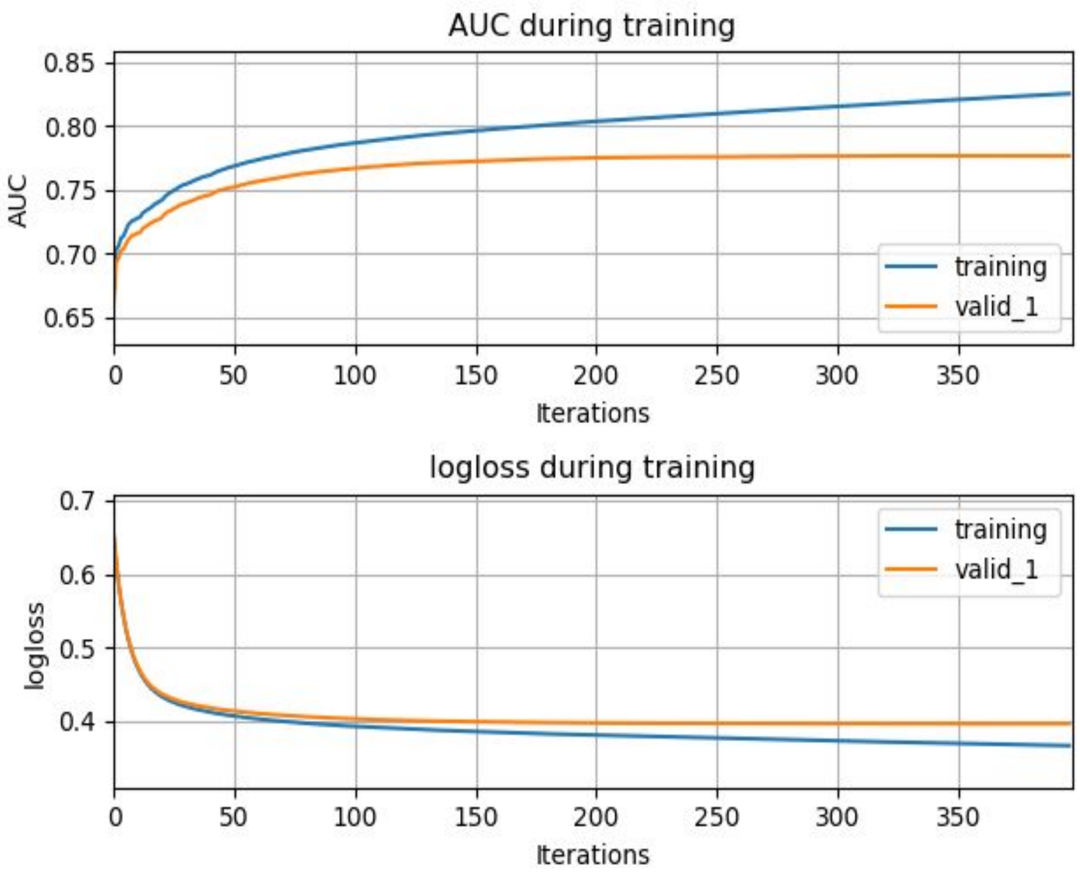
- Binary logloss (Logistic regression,lightgbm)

$$-\frac{1}{N}\sum_{i=1}^{N}(y_i \log p_i + (1-y_i)\log(1-p_i))$$

yi is ground truth of sample_i, pi is the probability that the sample_i is predicted to be 1 , N is the number of samples

Training process

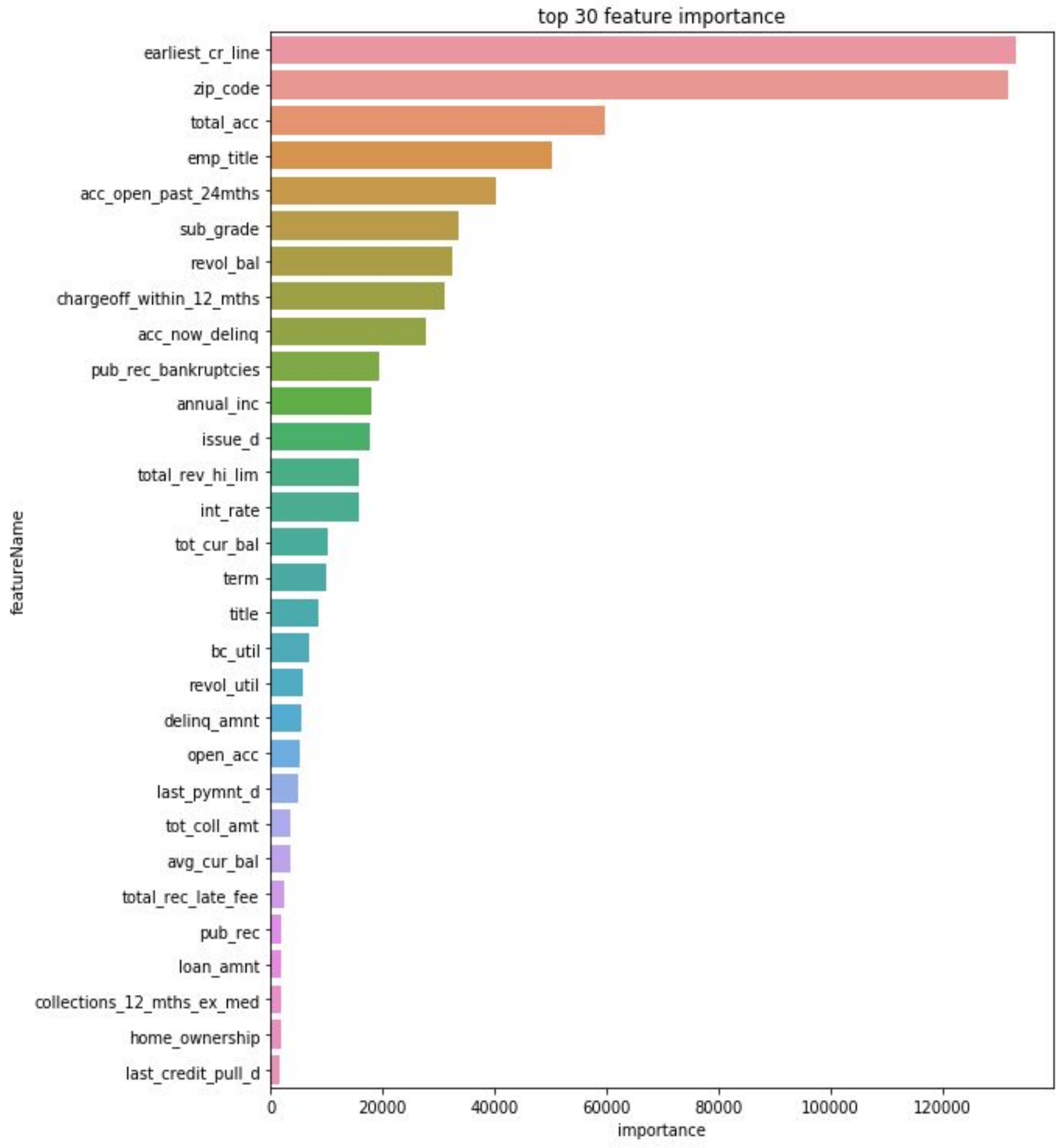
- 5-fold cross validation, auc metric
- Compute the auc of each fold
- Average auc as the final result



Result

AUC:

- Logistic Regression: 0.731
- SVM: 0.470
- LightGBM: 0.754



Top 30 feature importance of lightgbm model

References

1. “LendingClub Statistics”, *LendingClub*, https://www.lendingclub.com/info/download-data.action
2. “Logistic regression”, *Wikipedia*, https://en.wikipedia.org/wiki/Logistic_regression
3. “Support-vector machine”, *Wikipedia*, https://en.wikipedia.org/wiki/Support_vector_machine
4. “Gradient boosting”, *Wikipedia*, https://en.wikipedia.org/wiki/Gradient_boosting#Gradient_tree_boosting
5. “The Area Under an ROC Curve”, http://gim.unmc.edu/dxtests/roc3.htm
6. “Advances in Neural Information Processing Systems 30 (NIPS 2017), pp. 3149-3157”. *NIPS Proceedingsβ*, Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, Tie-Yan Liu. LightGBM: A Highly Efficient Gradient Boosting Decision Tree
7. “Welcome to LightGBM’s documentation!”, *LightGBM*, https://lightgbm.readthedocs.io/en/latest/index.html
8. “sklearn.preprocessing.StandardScaler”, *Scikit learn*, https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html