Project 4 Lending Club Loan Status

Wenjing Du (wenjing4) *Xinyi Song (xinyis8)*

In this project, we are provided with historical loan data issued by Lending Club. The goal is to build a model to predict the chance of default for a loan. This report is divided into three parts. The first part is about data pre-processing. In the second part, we provided technical detail of the three models we used. Finally, we evaluated the performance of the three models based on.

Data Pre-processing

The dataset in this project has 844006 rows with 30 features in total including the response 'loan_statue'. And there are three levels in 'loan_status': 'Default', 'Charged Off' and 'Fully paid'. After observing the whole dataset, we decide to remove the following categorical variables: "zip_code", "grade", "emp_title", "earliest_cr_line", "title" and "emp_length" since there are too many levels in these variables, we cannot create dummy variables required for our model with so many levels. Besides, we set the missing values of numerical variables with their median and as for categorical variables, we set their missing values with the most frequent level. Additionally, for our response 'loan_status', if it is 'Fully paid', we set it as 0, otherwise we set it as 1.

Here we split the whole dataset into three sets of training/test pairs using 'Project4_test_id.csv', where 'Project4_test_id.csv' has three columns and each column contains the ids of test samples.

Implementation

In this project, the dependent variable here is limited to three categories: 'Default', 'Charged Off' and 'Fully paid', after data preprocessing, we simplify it to a binary classification problem, where $Y \epsilon \{0,1\}$. Therefore, we used three kinds of classification models: Logistic Regression Model with Elastic Net Penalty, Linear Discriminant Analysis and Tree Model and predicted their loan status and evaluated their performance based on average Log-loss.

Performance Evaluation

Logistic Regression Model with Ridge Penalty

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables,

and the elastic net is a regularized regression method that linearly combines the L1 and L2 penalties of the lasso and the ridge methods. We used the ten-fold cross validation to choose the optimal λ by calling the function *cv.glmnet*, and then fitted the logistic regression model with sentiment versus key features by calling the *glmnet* function in package "glmnet" and setting "family=binomial" for canonical link function of logistic regression.
The reason that we chose to use Ridge penalty here for better model performance is as follows: Lasso regression could help us do model selection by shrinking coefficients to zero. However, the lasso tends to select one variable from a group, ignore the others and fails to do group selection. Besides, the number of selected features is bounded by the number of samples. On the contrary, Ridge Regression only shrinks the coefficients of the variable depends on its importance to the model accuracy rather than providing 0. Therefore, by applying elastic net model, we could obtain both variable selection features of lasso penalty and effective regularization characteristics of ridge regression, remove the limitation on the number of selected variables and obtain grouping effect. Since in this project, the design matrix of predictors (features) could be extremely sparse, to avoid the bound of number of variables and do group selection, we used the ten-fold-cross validation to choose optimal alpha, here this alpha is 0 which indicates ridge penalty.
However, the result of Log-loss is not as expected.

Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a method evaluating how well a group of variables supports an a priori grouping of objects. It is based on work by Fisher (1936) and is closely related to other linear methods such as multiple linear regression, principal components analysis (PCA), and factor analysis (FA)[2].
In LDA, a grouping variable is treated as the response variable and is expected to be categorical. Groupings should reflect ecologically relevant knowledge, such as sampling environment or method, or reflect the results of an exploratory method such as cluster analysis or non-metric dimensional scaling. LDA assumes that the observations conform to Gaussian distribution and each classifier shares the same covariance matrix.
Linear Discriminant Analysis models the distribution of predictors separately in each of the response classes, and then it uses Bayes' theorem to estimate the probability. Comparing with other classification algorithms such as random forests, it is much more interpretable, and the prediction process is easier and more efficient.
Here we fitted the Linear Discriminant Analysis model by calling the function *lda* in the R package "MASS" based on our train data and used test data to do prediction. Still, the result of Log-loss is not as expected.
After reflecting our data preprocess and the results, since the main reason for us to remove variables with too many levels is to create dummy variables for logistic regression and linear discriminant analysis, we guessed that when removing these variables, we might lose some information, therefore, we try another method which can deal with categorical variables with many levels and do not require dummy variables. We use boosting tree and transform our train dataset with data.matrix.

Tree Model

For boosting tree, trees are grown sequentially: each tree is grown using information from previously grown trees and each tree is fitted on a modified version of the original data set. For better performance, we select the XGBoost algorithm. This algorithm uses a more regularized model formalization to control over-fitting. The package we used is the 'xgboost'. Before we apply this method, we noticed that there are quite a few categorical variables existing in this dataset, and some of them have hundreds of levels. Therefore, we use the function 'data.matrix' to convert the data frame to a numeric matrix. Then we decided the parameters. We set the max number of iterations as 100, eta as 0.1, subsample=0.5 which means that XGBoost would randomly sample half of the training data prior to growing trees. and this will prevent overfitting. The maximum depth is set to be the default value 6. Finally we run the model and the performance is evaluated in the next section

Performance Evaluation

Average Log-loss for Three Splits

The loss function we used to evaluate the performance of the XGBoost algorithm is the log-loss. The Log-loss of the three splits using tree model is as following:

Table I Log-loss of Tree Model of Three Splits

	Test 1	Test 2	Test 3
Log-loss	0.448372	0.4510186	0.4492939

Here the average of Log-loss of these three test splits is 0.4495615, which reaches the benchmark of less than 0.45.

Prediction on 2018 Q3 and Q4

The response variable "loan_status" in the two test datasets, LoanStats_2018Q3 and LoanStats_2018Q4, has 7 levels, while our training set only has 3 levels. Therefore, though we use the complete test data set to predict, we remove the observations with levels not in the training data when generating the confusion matrix and calculate the Log-loss. The confusion matrice and values of Log-loss of our prediction on two datasets are listed below. We can see that the xgboost model performs well on LoanStats_2018Q4, and slightly worse on LoanStats_2018Q3.

Q3		test	
		0	1
prediction	0	9276	1010
	1	15	8

Q4		test	
		0	1
prediction	0	4893	120
	1	15	2

Log-loss = 0.5240503 Log-loss = 0.2312642

We have also selected 4 samples, first two from LoanStats_2018Q3 and the other two from LoanStats_2018Q4, to see whether our model makes reasonable prediction.

As for how our model gives the predictions, the logic of boosting tree model is as following:

1. Fit a model to the data, F1(x)=Y
2. Fit a model to the residuals, h1(x)=Y-F1(x)
3. Generate a new model, F2(x)=F1(x)+h1(x) (Note: F2 is boosted version of F1) and goes on …Fm(x)=F(m-1)(x)+h(m-1)(x)

Notice, here hm is just a model and Gradient boosting is a framework where we can plug in any model and plugging in tree-based model gives us better results.

Here our problem is to predict the lending club loan status of a person based on the information of features provided. Using an XGboost algorithm, the first set of predictions is assumed to be the mean value of loan status. Using this mean value and the actual value, the error can be calculated. The next tree uses these errors as the target variable. The leaf nodes will have predicted error values for each row. These predicted values (may be positive or negative) are added to the mean value resulting in a new set of loan status predictions. These are compared with the actual values again and errors are calculated. The complete process in repeated till the error values do not change.

Among all the variables, we think there are 4 variables with significant influence on the loan status: annual income, sub-grade, term, and verification status. The meaning of these 4 variables are listed below. In table 2, we provide the values of these variables of the 4 selected samples, and also the predicted probability. By common sense we know that those who have higher annual income are more likely to fully paid their loans. If the income is verified by the lending club, the income information is more reliable. Sub-grade is assigned from A to G and from 1 to 5. A1 is the best sub-grade, while G5 is the worst. Also, larger number of payments leads to higher risk of default. According to the table below, the predicted probability does match our understanding on the value of these variables.

Table II Samples selected from LoanStats_2018Q3 and LoanStats_2018Q4

Sample	id	Annual_Inc	sub_grade	term	Verification_Status	Probability
Sample1	100509	37000	F3	60 months	Not Verified	0.7059537
Sample2	114329	1800000	A1	36 months	Source Verified	0.01888227
Sample3	108637	40000	F3	60 months	Source Verified	0.7226423
Sample4	64639	6100000	A1	36 months	Source Verified	0.04522669

Variable Description:

- **Annual_Inc**: The self-reported annual income provided by the borrower during registration.
- **Verification_Status**: Indicates if income was verified by [Lending Club], not verified, or if the income source was verified.
- **Sub_grade**: LendingClub assigned loan subgrade.
- **Term**: The number of payments on the loan. Values are in months and can be either 36 or 60.