**Case Study - Phase 3**

**Yuhao Zhu**

**3.1 Predictive Models of Default**

**1. (i) How did you set up your model training and evaluation?**

Train and test data are generated using both random split and temporal split methods. Specifically, prepare\_data method enables us to split the original dataset into train and test data, and the train data would account for about 70% of the whole data set. The way to split the data is using either random or temporal, following the instructions. As for the evaluation, we use accuracy to decide the best model, and using five-fold cross validation to tune hyper-parameters.

**(ii) Which model hyper-parameters did you tune (for each model)?**

Here we used multiple models, each model has different hyper-parameters to tune.

* For Naïve Bayes, there is no hyper-parameters to tune;
* For l1 with Logistic Regression, we tuned ‘C’ from 0.00001 to 100000;
* For l2 with Logistic Regression, we also tuned ‘C’ from 0.0001 to 100000;
* For Decision Tree, we tuned max depth from 1 to 20;
* For Random Forest, we tuned estimators number from [10,20,50,100,150], and max depth from 3 to 10;
* For Multi-Layer Perceptron, we tuned activation functions using ['identity', 'relu', 'logistic'] and alpha from 0.0001 to 0.1;
* For Bagging, we tuned max features from 6 to 15;
* For Gradient Boosting, we tuned number of estimators from [10,20,50,100,150], and max depth from 3 to 7;

**(iii) Which performance measures did you use? Report your evaluation results.**

We calculated Accuracy score, as well as Precision, Recall, F1 score and AUC score.

The results are shown below:

**Table 1. Results with ‘random’ split**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Naïve Bayes** | **L1 LR** | **L2 LR** | **Decision Tree** | **Random Forest** | **MLP** | **Bagging** | **Gradient Boosting** |
| **Accuracy** | 0.88955 | 0.89255 | 0.8938 | 0.88785 | 0.8915 | 0.8932 | 0.881 | 0.8938 |
| **Precision** | 0.8945 | 0.8992 | 0.8961 | 0.8861 | 0.8893 | 0.8960 | 0.8774 | 0.8965 |
| **Recall** | 0.8895 | 0.8925 | 0.8938 | 0.8879 | 0.8915 | 0/8932 | 0.8810 | 0.8938 |
| **F1** | 0.8916 | 0.8952 | 0.8949 | 0.8869 | 0.8903 | 0.8945 | 0.8789 | 0.8950 |
| **AUC** | 0.93 | 0.93 | 0.93 | 0.92 | 0.93 | 0.93 | 0.91 | 0.93 |

**Table 2. Results with ‘time’ split**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Naïve Bayes** | **L1 LR** | **L2 LR** | **Decision Tree** | **Random Forest** | **MLP** | **Bagging** | **Gradient Boosting** |
| **Accuracy** | 0.89855 | 0.90525 | 0.90535 | 0.9039 | 0.90565 | 0.90585 | 0.8872 | 0.90185 |
| **Precision** | 0.9000 | 0.9092 | 0.9093 | 0.9067 | 0.9091 | 0.9101 | 0.8842 | 0.9024 |
| **Recall** | 0.8989 | 0.9052 | 0.9053 | 0.9039 | 0.9056 | 0.9059 | 0.8872 | 0.9019 |
| **F1** | 0.8994 | 0.9069 | 0.9070 | 0.9051 | 0.9071 | 0.9076 | 0.8854 | 0.9021 |
| **AUC** | 0.93 | 0.94 | 0.94 | 0.93 | 0.94 | 0.94 | 0.92 | 0.94 |

From the table above we can see that all the methods perform quite similar, with an accuracy score around 0.89, among which l2 regularization with Logistic Regression and Multi-Layer Perceptron perform relatively better.

1. **What are some advantages and disadvantages of using these data splitting procedures?**

The “Temporal train/test split” method has better interpretability that we can predict the characteristics and derived statistics of later loan using earlier loan data. If it works well, we can get better predictions of new loan assets as compared with the “Random train/test split method”.

On the other hand, if the model cannot remain consistent time stability, different time periods could have huge impact on the characteristics of loans that they should be considered as independent features. In this situation, the “Random train/test split” can derive better predictive models as compared with the “Temporal train/test split method”.

From the results of classification models here, models with Random splits have relatively worse performance as lower accuracy and AUC than those with Temporal splits.

**3. (i). Provide a list of aforementioned features that are derived by LendingClub and any other features that correlate/reflect those.**

Two features are statistics defined by LendingClub: "grade", "dti".  
"grade“: LC assigned loan grade  
"dti": A ratio calculated using the borrower’s total monthly debt payments on the total debt obligations, excluding mortgage and the requested LC loan, divided by the borrower’s self-reported monthly income.

**(ii). Fit a (L1 or L2 regularized) Logistic Regression model using only one of the features you identified in (i). What is the predictive power as compared to that for the models you trained in part 1?**

**Table 3. Results with LendingClub features**

|  |  |  |
| --- | --- | --- |
|  | **L1 LR** | **L2 LR** |
| **Accuracy** | 0.8181 | 0.8181 |
| **Precision** | 0.6693 | 0.6693 |
| **Recall** | 0.8181 | 0.8181 |
| **F1** | 0.7362 | 0.7362 |
| **AUC** | 0.50 | 0.41 |

I choose the 'dit' to fit the logistic regression models. According to the result above, the results show that the predictive power using 'dti' alone is significantly lower than using all features. The accuracy score of one feature is about 81.81%, while the models fit earlier with all features could achieve the accuracy of about 89.30%. Although accuracy scores here are still above 0.8.It is almost meaningless, because the AUC is much lower, with only 0.5 for L1 and 0.41 for L2, which indicates that almost no reasonable information is caught.

**(iii). Remove the features you identified in (i), refit your models onto remaining features and report new performance measure(s). What are your conclusions?**

**Table 4. Results after removing LendingClub features**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Naïve Bayes** | **L1 LR** | **L2 LR** | **Decision Tree** | **Random Forest** | **MLP** | **Bagging** | **Gradient Boosting** |
| **Accuracy** | 0.8643 | 0.8934 | 0.89612 | 0.88705 | 0.89425 | 0.89005 | 0.8207 | 0.8935 |
| **Precision** | 0.8684 | 0.8964 | 0.8959 | 0.8822 | 0.8936 | 0.8981 | 0.7989 | 0.8975 |
| **Recall** | 0.8643 | 0.89934 | 0.8931 | 0.8871 | 0.8942 | 0.8901 | 0.8207 | 0.8935 |
| **F1** | 0.8662 | 0.8947 | 0.8944 | 0.8838 | 0.8939 | 0.8931 | 0.8056 | 0.8952 |
| **AUC** | 0.90 | 0.94 | 0.94 | 0.93 | 0.94 | 0.93 | 0.78 | 0.94 |

The accuracy is very similar to the result if we also include the LendingClub (LC) features, showing that the features which are not lending club features are those which have a good prediction ability. However, the accuracy of bagging decrease more compared with other model. I think it is because bagging is to combine several not-so-good model together and generate a much better model. Not using lending club feature would decrease the number of not-so-good model and worse the final model.

Generally speaking, the models without LC derived features have better performance than the models fit earlier. We should use the models without LC derived features to better describe the dataset and get better results.

**4. Generate My Model and compare scores.**

As is shown from the table above, L2 Regularization with Logistic Regression has the best performance, so we choose L2 with LR as My Model.

There are two ways to assess the extent to which My Model's scores agree with the grades assigned by LendingClub.

1. First, the pre-defined functions here define the scipy kendalltau score as the similarity to LendingClub grades. The Kendall’s tau is defined as tau = (P - Q) / sqrt((P + Q + T) \* (P + Q + U)), where P is the number of concordant pairs, Q the number of discordant pairs, T the number of ties only in x, and U the number of ties only in y. If a tie occurs for the same pair in both xand y, it is not added to either T or U.

2. Since LendingClub assigned grades from A to G, I can assume that all loans are equally assigned to each grade in the rank of its probability of default or not. Based on this, I ranked the prediction probabilities of My Model's results, equally divided them into seven groups, and assigned the groups with predicted grades from A to G. Then, I can get the similarity comparisons between My Model's predicted grades and their actual grades.

The first method results in a similarity score of 32.668% and the second method results in a similarity score of 11.595%. Both methods show that My Model's scores agree with the grades assigned by LendingClub in a small and non-significant degree.

**5. Time stability analysis**

In this part, two datasets are filtered, one with training data from 2010, testing data from 2017; another with training data from 2016 and testing data from 2017. The results are shown as below:

**Table 5. Time stability analysis**

|  |  |  |
| --- | --- | --- |
|  | **2010** | **2016** |
| **Accuracy** | 0.8936 | 0.8936 |
| **Precision** | 0.8955 | 0.8955 |
| **Recall** | 0.8936 | 0.8936 |
| **F1** | 0.8945 | 0.8945 |
| **AUC** | 0.93 | 0.93 |

My model is stable and both models trained on data in 2010 and on data in 2016 perform similarly on data in 2017. The results table above clearly shows that the optimal parameter 'C' in both models are 100.0 and all performance scores are exactly the same. The model is quite robust across time, both 2010 and 2016 training data can give good prediction results. The model is stable regarding the time.

**6. Original data analysis**

After testing the models, we test the selected model on original dataset. The results are shown as below:

**Table 6. Original data analysis**

|  |  |  |
| --- | --- | --- |
|  | **Random** | **Temporal** |
| **Accuracy** | 0.9424 | 0.95165 |
| **Precision** | 0.9425 | 0.9623 |
| **Recall** | 0.9424 | 0.9516 |
| **F1** | 0.9424 | 0.9554 |
| **AUC** | 0.96 | 0.97 |

According to the result table, we found that the performances are surprisingly good. Compared with results we generated earlier, all the metrics improve about 5%, with an AUC reached 0.97 in temporal method. This may be due to more information are included with the whole dataset. Even the outlier include certain information for prediction. Plus, since we choose L2-Regularization with Logistic Regression, which would conduct some degree of feature selection, even though it would not completely exclude features, it may give lower weight to the features which have large variance (just my guess). So even some outliers may make the prediction worse, the model can still benefit more from the information contained in the outliers.

**3.2 Investment Strategies**

**7. Results on three regression models**

**Table 7.1 Performance for each return calculation (all data)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **M1** | **M2** | **M3(2.4%)** | **M4(6%)** |
| **L1 regressor** | -0.23 | -0.02 | -0.21 | -0.16 |
| **L2 regressor** | -0.23 | -0.02 | -0.21 | -0.16 |
| **Neural Network regressor** | -0.23 | -0.00 | -0.11 | -0.11 |
| **Random Forest regressor** | -0.22 | -0.02 | -0.21 | -0.16 |

**Table 7.2 Performance for each return calculation (default loans)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **M1** | **M2** | **M3(2.4%)** | **M4(6%)** |
| **L1 regressor** | -0.43 | -0.01 | -0.34 | -0.24 |
| **L2 regressor** | -0.43 | -0.01 | -0.35 | -0.24 |
| **Neural Network regressor** | -0.58 | -0.01 | -0.37 | -0.28 |
| **Random Forest regressor** | -0.43 | -0.01 | -0.34 | -0.24 |

**Table 7.3 Performance for each return calculation (non-default loans)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **M1** | **M2** | **M3(2.4%)** | **M4(6%)** |
| **L1 regressor** | -1.00 | -0.99 | -1.43 | -1.36 |
| **L2 regressor** | -1.00 | -0.99 | -1.44 | -1.37 |
| **Neural Network regressor** | -1.59 | -1.32 | -1.05 | -1.04 |
| **Random Forest regressor** | -1.01 | -0.99 | -1.42 | -1.35 |

All the regression performs really bad, the R square is pretty low across all the model. The performance is worse when predicting the non-default loans. It shows that the features we have could not have the ability to predict the return or there are still too much noise in the data to perform a good prediction. When we filter the data, we do not remove too much outliers, which might be the reason why we have such a bad performance in regression. The model may learn the noise but not capture the general trend. In addition, it also could be the case that the general trend could not be told by the features we have, the return seems to be more random or arbitrary. Or there are no clear pattern for return.

**8. Investment strategies**

**Table 8. return calculation**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Strategy** | **M1** | **M2** | **M3(2.4%)** | **M4(6%)** |
| **Rand** | -0.0314 | 0.029 | -0.00 | 0.033 |
| **Def** | -0.0302 | 0.028 | -0.002 | 0.035 |
| **Ret** | -0.0368 | 0.0326 | 0.002 | 0.0348 |
| **DefRet** | -0.0366 | 0.0249 | -0.0053 | 0.0373 |
| **BEST** | Def | Ret | Ref | DefRet |

i) Please refer to the table above

Ii) Please refer to the table above

Iii) Return-based model perform best, even though it really does not perform well, and I could not say there is significant difference between four strategy. Rand strategy performs worst and has the lowest return. It might cause loss more since it does not utilize any information to make the strategy. However, the difference is not so big. I think the data might tell us it is hard to come up with a solution for investment, since there are so much uncertainty going on

**9. Sensitivity test of portfolio size**

The graph shows that the investment return increase significantly if we increase the portfolio from 1000 to 2000. It means that when portfolio is small, adding number of loans can increase the return. However, when portfolio increase too much, the investment return decreased. It matches with the questions that the good loans are running out to invest in. I think it means that we would like to have a balance of the portfolio, so that we could balance the risk for a specific investment, and invest in potentially good loans.

