Credit Card Default Analysis

MDML 2047 Final Project

Ruoyu Zhu (rz1403)

Introduction

Every year, credit scoring methodologies evaluate the risk in billions of dollars in loans. And the accuracy of the evaluation determines the profit or loss of a financial institution. Because of that, advanced machine learning methods are quickly finding applications throughout the financial services industry and achieved great predictive successes. Credit card department, which generates profit from the interest rate, is highly relying on the result of machine learning model when making lending decision.

Data Selection and Data Description

Data Selection

Due to legal issues and privacy reasons, real-world credit card data is hard to find, and it is impossible to script transaction history and demography data from any online open source. To keep the analysis realistic, we found a dataset contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005.

The original dataset can be found here at the UCI Machine Learning Repository. (Reference: Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.)

Data Description

There are 25 variables in tha dataset. Our goal is to predict whether the customer will default next month. So the target variable is *"default.payment.next.month"*, where 1 = will default next month, and 0 = won't default next month.

There are 24 predictors:

- ID: ID of each client
- LIMIT_BAL: Amount of given credit in NT dollars (includes individual and family/supplementary credit
- SEX: Gender (1=male, 2=female)
- EDUCATION: (1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, 6=unknown)
- MARRIAGE: Marital status (1=married, 2=single, 3=divorce, 0=others)
- AGE: Age in years
- PAY_0: Repayment status in September, 2005 (-2: No consumption; -1: Paid in full; 0: The use of revolving credit; 1 = payment delay for one month;2 = payment delay for two months; . . .; 8 = payment delay for eight months and above)
- PAY_2: Repayment status in August, 2005 (scale same as above)
- …
- PAY_6: Repayment status in April, 2005 (scale same as above)
- BILL_AMT1: Amount of bill statement in September, 2005 (NT dollar)
- BILL_AMT2: Amount of bill statement in August, 2005 (NT dollar)
- …

- BILL_AMT6: Amount of bill statement in April, 2005 (NT dollar)
- PAY_AMT1: Amount of previous payment in September, 2005 (NT dollar)
- …
- PAY_AMT6: Amount of previous payment in April, 2005 (NT dollar)

Data Cleaning

Drop NAs

There are ~20 instances contain NA in some columns. Since we have a relatively large database(~30k rows), then droping 20 rows won't affect the model performance. So we decide to drop the rows contain NA.

```
data <- na.omit(data)
```

Rename

To make variable names easy to interpret, we rename each column.

```
data <- plyr::rename(data, c('ID' = 'id','LIMIT_BAL' = 'limit_balance',
                              'SEX' = 'sex', 'EDUCATION' = 'edu',
                             'MARRIAGE' ='marriage','AGE' = 'age',
                             'PAY_0' = 'pay.9','PAY_2' = 'pay.8',
                             'PAY_3' = 'pay.7','PAY_4' = 'pay.6',
                             'PAY_5' = 'pay.5', 'PAY_6' = 'pay.4',
                             'default.payment.next.month' = 'default'))
data <- plyr::rename(data, c('BILL_AMT1' = 'bill.amt.9', 'BILL_AMT2' = 'bill.amt.8',
                             'BILL_AMT3' = 'bill.amt.7','BILL_AMT4' = 'bill.amt.6',
                             'BILL_AMT5' = 'bill.amt.5','BILL_AMT6' = 'bill.amt.4'))
data <- plyr::rename(data, c('PAY_AMT1' = 'pay.amt.9', 'PAY_AMT2' = 'pay.amt.8',
                             'PAY_AMT3' = 'pay.amt.7','PAY_AMT4' = 'pay.amt.6',
                             'PAY_AMT5' = 'pay.amt.5','PAY_AMT6' = 'pay.amt.4'))
```

Clean Data of Education Level

According to the dictionary: Education: 1 = graduate school; 2 = university; 3 = high school; 0, 4, 5, 6 = others. Since we do not know the difference among 0,4,5,6 and regression model will treate the variable "edu"(education level) monotonic, then we assign 4 to education level if original education level = others.

```
data$edu[data$edu == 0] <- 4
data$edu[data$edu == 5] <- 4
data$edu[data$edu == 6] <- 4
```

Convert numberic to factor

Variables 'edu'(education level) and 'marriage' (Marital status) needs to change from numerical to factors.

```
data$edu <- as.factor(data$edu)
data$sex <- as.factor(data$sex)
data$marriage <- as.factor(data$marriage)
```

Make a Balanced Dataset

Our target variable (default) are not balanced. There are 23364 0s and only 6636 1s. Unblanced data affects generalizability of results and potentially the identifiability of model parameters.

Thus, we keep all 1s and randomly select the same amount of 0s to build a balanced dataset.

```
table(data_balance$default)
```

```
##
##    0    1
## 6636 6636
```

Summary of cleaned data

Here's the sample data after cleaning

```
sample_n(data_balance, 3)
```

```
##          id limit_balance sex edu marriage age pay.9 pay.8 pay.7 pay.6
## 21864 28096        150000   2   2        2  41    -1    -1    -1    -1
## 707    3192         70000   2   2        2  23     1     2     0     0
## 13061  5954        120000   2   1        2  27     1    -2    -2    -2
##       pay.5 pay.4 bill.amt.9 bill.amt.8 bill.amt.7 bill.amt.6 bill.amt.5
## 21864    -1    -1        316        316        316        316        466
## 707       2     2      17461      16892      18013      19315      19859
## 13061    -2    -2          0          0          0          0          0
##       bill.amt.4 pay.amt.9 pay.amt.8 pay.amt.7 pay.amt.6 pay.amt.5
## 21864       8057       316       316       316       466      8057
## 707        19390         0      1400      1600      1000         0
## 13061          0         0         0         0         0         0
##       pay.amt.4 default
## 21864       316       0
## 707        1000       1
## 13061         0       1
```

Train Test Split

We take 75% of the data for training and the rest 25% data for testing. Due to the computational constraint, we do not use k-fold to do the cross-validation.

Models and Plots

Baseline Model

We use simple logistic regression without any payment information, i.e. our only variables are 'limit_balance', 'sex', 'edu', 'age'.

```
model.base <- glm(default ~limit_balance + sex + edu + age,data=train.base,family=binomial)
```

```
##   (Intercept) limit_balance          sex2          edu2          edu3
##  2.434123e-01 -3.315290e-06 -1.444412e-01  6.791049e-02  7.133403e-02
##          edu4           age
## -1.325888e+00  9.061638e-03

## the auc score of baseline model is  62.52007
```

K-Nearest-Neighbor

We first use knn (k = 10) to predict the target. However, the result of KNN is even lower than the baseline model.

```
model.knn <- knn(train = train.knn[,-25], test = test.knn[,-25],cl =train.knn[,25], k= 10)
```

```
## the auc score of knn is 60.76668
```

Logistic Regression (with Ridge Regulation)

First we use all 24 features to build the LR model and see how it works. Then, we use ridge and lasso regulation to imporve the model.

```r
model_lr <- glm(default ~.-id ,data=train_lr,family=binomial)
```

```
## the auc score of logistic regression model is  72.37112
```

```r
ridge.model = glmnet(X, y, family = 'binomial',alpha = 0,lambda = 0.01)
lasso.model = glmnet(X, y, family = 'binomial',alpha = 1,lambda =0.01)
```

```
## the auc score of logistic regression model (after ridge regulatoin) is  72.10301
```

```
## the auc score of logistic regression model (after lasso regulatoin) is  71.90278
```

Logistic Regression after backward stepwise selection

We can see from the logisitic regression that some coeffecient are not significant. We do a feature selection to pick the most useful feature to rebuild the logistic regression model.

Here, we use backward stepwise selection.

```r
model_lr.step <- model_lr %>% stepAIC(trace = FALSE)
```

```
##
## Call:
## glm(formula = default ~ limit_balance + sex + edu + marriage +
##     age + pay.9 + pay.8 + pay.7 + pay.5 + bill.amt.9 + bill.amt.7 +
##     pay.amt.9 + pay.amt.8 + pay.amt.7 + pay.amt.5, family = binomial,
##     data = train_lr)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.3395  -1.0716   0.1204   1.0588   2.9837
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -7.675e-01  6.211e-01  -1.236 0.216569
## limit_balance -7.080e-07  2.247e-07  -3.151 0.001629 **
## sex2          -8.871e-02  4.500e-02  -1.971 0.048688 *
## edu2          -6.672e-02  5.146e-02  -1.296 0.194805
## edu3          -4.311e-02  6.986e-02  -0.617 0.537212
## edu4          -1.207e+00  2.332e-01  -5.175 2.28e-07 ***
## marriage1      9.301e-01  6.114e-01   1.521 0.128221
## marriage2      7.734e-01  6.116e-01   1.265 0.206016
## marriage3      9.358e-01  6.440e-01   1.453 0.146214
## age            7.509e-03  2.758e-03   2.723 0.006471 **
## pay.9          5.097e-01  2.481e-02  20.545  < 2e-16 ***
## pay.8          8.564e-02  2.917e-02   2.936 0.003327 **
## pay.7          9.772e-02  2.911e-02   3.357 0.000789 ***
## pay.5          3.826e-02  2.578e-02   1.484 0.137752
## bill.amt.9    -4.663e-06  9.029e-07  -5.164 2.42e-07 ***
## bill.amt.7     3.091e-06  1.008e-06   3.068 0.002158 **
## pay.amt.9     -1.046e-05  2.561e-06  -4.085 4.41e-05 ***
## pay.amt.8     -1.289e-05  2.420e-06  -5.325 1.01e-07 ***
## pay.amt.7     -3.480e-06  1.693e-06  -2.056 0.039793 *
## pay.amt.5     -3.380e-06  2.044e-06  -1.653 0.098244 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 13799  on 9953  degrees of freedom
## Residual deviance: 12088  on 9934  degrees of freedom
## AIC: 12128
##
## Number of Fisher Scoring iterations: 5
```

```
## the auc score after stepwise selection is  72.31967
```

Random Forest

Random forest is one of the most accurate learning algorithms available. For many data sets, it produces a highly accurate classifier. So we use random forest with ntree = 1000 on the model.

```
model.rf <- randomForest(default ~.-id, data = train.rf, ntree = 1000,importance = TRUE)
```

```
## the auc score of random forest model is  77.60296
```

AUC Plots for selected models

We compare all auc scores and plot ROC curves for selected model. Using logistic regression and random forest, the model performance is significantly better than the baseline model.

```
## the auc score of baseline model is  62.52007
```

```
## the auc score of knn is 60.76668
```

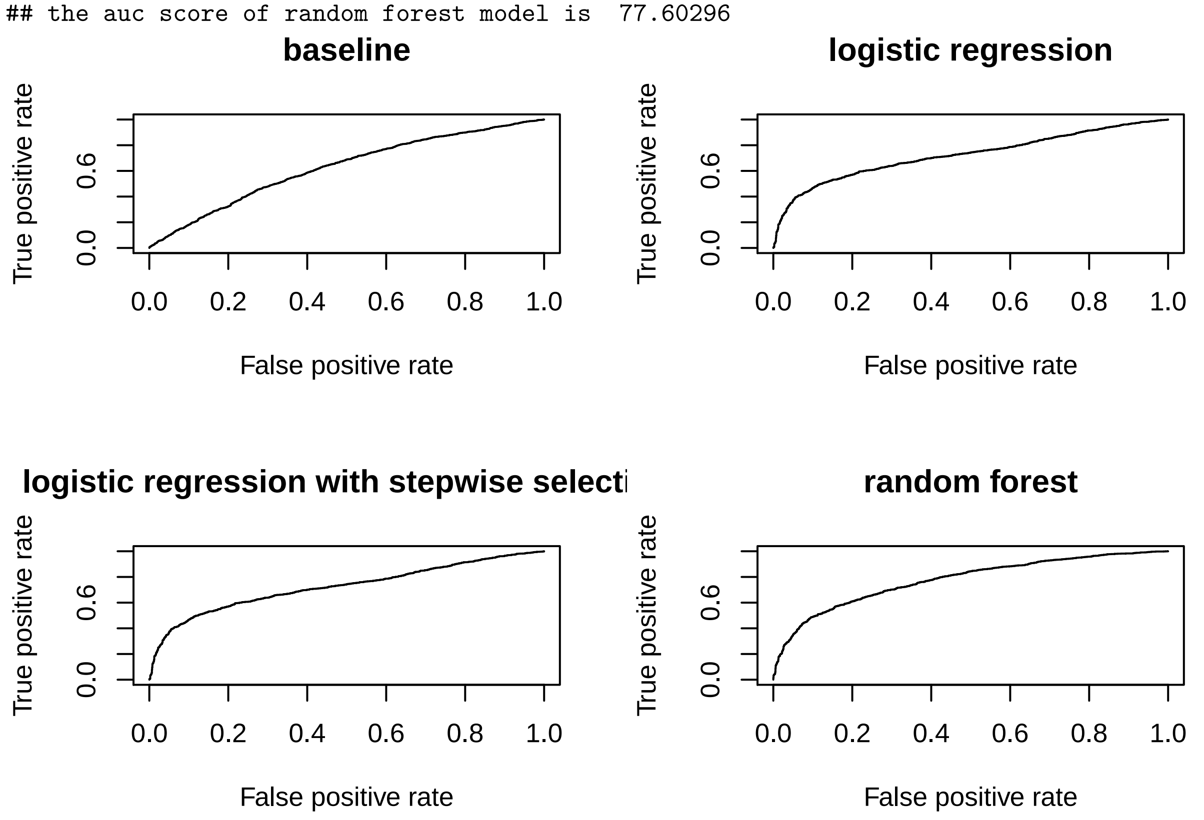
```
## the auc score of logistic regression model is  72.37112
```

```
## the auc score of logistic regression model (after ridge regulation) is  72.10301
```

```
## the auc score of logistic regression model (after lasso regulation) is  71.90278
```

```
## the auc score after stepwise selection is  72.31967
```

```
## the auc score of random forest model is  77.60296
```



Model extension

Predict use 5 months or less data

Looking at the dataset, we want to test out our model performance if we just use the data of the first k (k < 6) months to predict the next month payment. i.e. In the original data set, we use April 2005 to September 2005 (6 months total) data to predit payment default rate in October 2005. We now want to use data from April 2005 to August 2005 (5 months total) to predict September's payment.

Thus, we create a new target variable call 'default_5' (which means default or not using 5 months data) by using the given payment data on September. The new data looks like this:

```r
sample_n(data_balance.5m, 3)
```

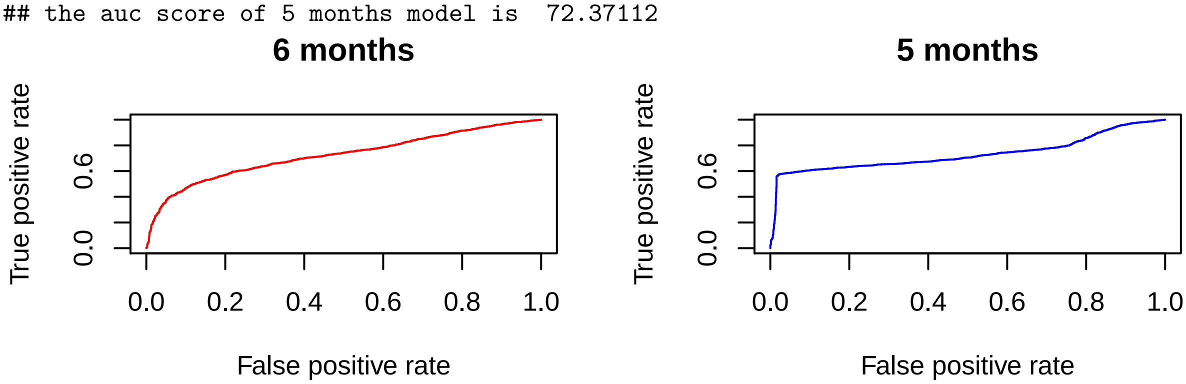
```
##          id limit_balance sex edu marriage age pay.8 pay.7 pay.6 pay.5
## 12790 16796        500000   1   2        2  30     0     0     0     0
## 1559   6583        200000   1   2        2  29     2     2     2     2
## 87      330        150000   1   1        1  40     2     2     2     2
##       pay.4 bill.amt.8 bill.amt.7 bill.amt.6 bill.amt.5 bill.amt.4
## 12790     0     135796     133712     136069      67837      69514
## 1559      2     174082     177684     180259     183704     186507
## 87        2     102586     100064     104975     107147     109428
##       pay.amt.8 pay.amt.7 pay.amt.6 pay.amt.5 pay.amt.4 default_5
## 12790      6000      5355      3500      2513      5000         0
## 1559       8000      7000      6500      6000      5600         1
## 87            0      8100      4000      4200      4200         1
```

Then we apply the same logistic regression model on the new data set, and plot the ROC curve for both 6 months and 5 months models. The auc score is close, and the ROC curve of the 5 months data has a sharp elbow when false positive rate is around 0.1. We can conclude from the graph that prediction using 5 months data is as good as using 6 months data.

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## the auc score of 5 months model is  73.25859
```

```
## the auc score of 5 months model is  72.37112
```



Conclusion and Future Work

Random forest has much higher AUC score than other models. Thus, we choose random forest as our final models. However, while making decisions for the new credit card application, regulators require financial institutions to provide reasons to customers when taking "adverse action", i.e. turning down a credit card application. Some possibilities include "The proportion of your revolving balances to total balances is too high" or "you recently inquired a new loan."

Currently, the black box models such as random forest are neither interpretable nor explainable. In settings where regulators or consumers demand explanations, more sophisticated machine learning techniques are needed. The techniques should offer both the promise of increased accuracy and explainability at the same time.