Report on Default of Credit Card Clients Dataset

Masayoshi Sato

2021/6/26

Introduction

Finance is one of fields where machine learning is used. It uses a huge amount of data and is also surrounded by a lot of uncertainties. In this paper, we will deal with a problem many credit card companies have been facing. Can we predict whether a credit user will pay their debt or fail based on objective data? Traditionally, finding a credible borrower have been a kind of know-how, or skill and experience nurtured by financial institutions. Instead, we try to build machine learning models. using a dataset which is open to public.

The dataset we use, "Default of Credit Card Clients Dataset" is stored in Kaggle website. It was collected in Taiwan in 2005. It has 24 variables, such as age, education, and payment condition. Outcome has two results, "0" non-default, "1" default. Each data was anonymously collected and labeled with individual ID.

Our goal is to find a classification model which predicts the most accurate outcome, default or not. We need to bear it in mind that its distribution of these outcomes is imbalanced. Namely, the number of default clients are small compared to non- default clients. To address the issue, we will use other criteria, balanced accuracy.

We will use three machine learning models, logistic regression, decision tree, and random forest. If necessary, we will tune their parameters to find the best solution. Our procedures are as follows :

1. Data exploration and data cleansing
2. Splitting the dataset into train_set, validation_set, and test_set
3. Applying models, logistic regression, decision tree, and random forest
4. Considering models performance, and evaluating

This paper is written as a final assignment in "HarvardX PH125.9x Data Science: Capstone."

Packages and Dataset

In this paper, we use R packages, "tidyverse[1]", "DataExplorer[2]", "gridExtra[3]", "rpart[4]", "caret[5]", and "ranger[6]".

We use a dataset stored in Kaggle[7]website. In the description, it says, "This 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." It is CSV file.

[1] https://cran.r-project.org/web/packages/tidyverse/index.html
[2] https://cran.r-project.org/web/packages/DataExplorer/index.html
[3] https://cran.r-project.org/web/packages/gridExtra/index.html
[4] https://cran.r-project.org/web/packages/rpart/index.html
[5] https://cran.r-project.org/web/packages/caret/index.html
[6] https://cran.r-project.org/web/packages/ranger/index.html
[7] https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset

Kaggle requires registration to download the data. For the sake of convenience, the data file is stored in my GitHub repository[8].

Data Exploration

First, we need to check the downloaded dataset. Columns are as follows.

```
## spec_tbl_df [30,000 x 25] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ID                        : num [1:30000] 1 2 3 4 5 6 7 8 9 10 ...
##  $ LIMIT_BAL                 : num [1:30000] 20000 120000 90000 50000 50000 50000 500000 100000 1400
##  $ SEX                       : num [1:30000] 2 2 2 2 1 1 1 2 2 1 ...
##  $ EDUCATION                 : num [1:30000] 2 2 2 2 2 1 1 2 3 3 ...
##  $ MARRIAGE                  : num [1:30000] 1 2 2 1 1 2 2 2 1 2 ...
##  $ AGE                       : num [1:30000] 24 26 34 37 57 37 29 23 28 35 ...
##  $ PAY_0                     : num [1:30000] 2 -1 0 0 -1 0 0 0 0 -2 ...
##  $ PAY_2                     : num [1:30000] 2 2 0 0 0 0 0 -1 0 -2 ...
##  $ PAY_3                     : num [1:30000] -1 0 0 0 -1 0 0 -1 2 -2 ...
##  $ PAY_4                     : num [1:30000] -1 0 0 0 0 0 0 0 0 -2 ...
##  $ PAY_5                     : num [1:30000] -2 0 0 0 0 0 0 0 0 -1 ...
##  $ PAY_6                     : num [1:30000] -2 2 0 0 0 0 0 -1 0 -1 ...
##  $ BILL_AMT1                 : num [1:30000] 3913 2682 29239 46990 8617 ...
##  $ BILL_AMT2                 : num [1:30000] 3102 1725 14027 48233 5670 ...
##  $ BILL_AMT3                 : num [1:30000] 689 2682 13559 49291 35835 ...
##  $ BILL_AMT4                 : num [1:30000] 0 3272 14331 28314 20940 ...
##  $ BILL_AMT5                 : num [1:30000] 0 3455 14948 28959 19146 ...
##  $ BILL_AMT6                 : num [1:30000] 0 3261 15549 29547 19131 ...
##  $ PAY_AMT1                  : num [1:30000] 0 0 1518 2000 2000 ...
##  $ PAY_AMT2                  : num [1:30000] 689 1000 1500 2019 36681 ...
##  $ PAY_AMT3                  : num [1:30000] 0 1000 1000 1200 10000 657 38000 0 432 0 ...
##  $ PAY_AMT4                  : num [1:30000] 0 1000 1000 1100 9000 ...
##  $ PAY_AMT5                  : num [1:30000] 0 0 1000 1069 689 ...
##  $ PAY_AMT6                  : num [1:30000] 0 2000 5000 1000 679 ...
##  $ default.payment.next.month: num [1:30000] 1 1 0 0 0 0 0 0 0 0 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   ID = col_double(),
##   ..   LIMIT_BAL = col_double(),
##   ..   SEX = col_double(),
##   ..   EDUCATION = col_double(),
##   ..   MARRIAGE = col_double(),
##   ..   AGE = col_double(),
##   ..   PAY_0 = col_double(),
##   ..   PAY_2 = col_double(),
##   ..   PAY_3 = col_double(),
##   ..   PAY_4 = col_double(),
##   ..   PAY_5 = col_double(),
##   ..   PAY_6 = col_double(),
##   ..   BILL_AMT1 = col_double(),
##   ..   BILL_AMT2 = col_double(),
##   ..   BILL_AMT3 = col_double(),
##   ..   BILL_AMT4 = col_double(),
##   ..   BILL_AMT5 = col_double(),
```

[8]https://github.com/masa951125/Final_project/raw/main/UCI_Credit_Card.csv

```
##   ..   BILL_AMT6 = col_double(),
##   ..   PAY_AMT1 = col_double(),
##   ..   PAY_AMT2 = col_double(),
##   ..   PAY_AMT3 = col_double(),
##   ..   PAY_AMT4 = col_double(),
##   ..   PAY_AMT5 = col_double(),
##   ..   PAY_AMT6 = col_double(),
##   ..   default.payment.next.month = col_double()
##   .. )
```

It has 30000 rows and 25 columns. "Default.payment.next.month" is an outcome . Other features seem to be either numerical or categorical data. "SEX", "EDUCATION", "MARRIAGE", "PAY_0" -"PAY_6", and "default.payment.next.month" look like categorical data, as their values are limited number of integers. Other features seem to be numerical.

We know there are no NAs, Nulls in the dataset.

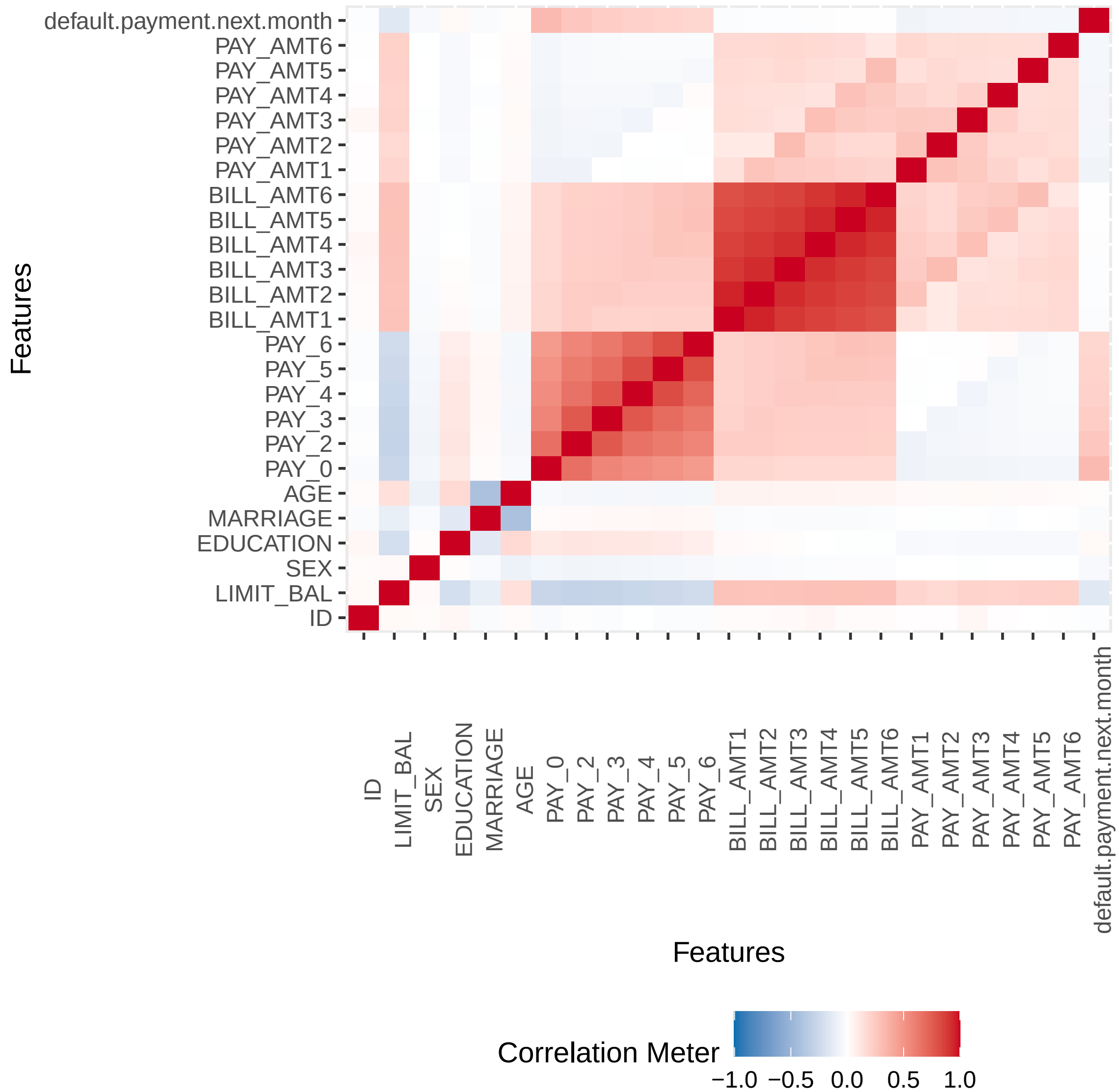
```r
#Number of NAs
sum(is.na(original_default))
```

```
## [1] 0
```

```r
#Number of Nulls
sum(is.null(original_default))
```

```
## [1] 0
```

How these predictors are correlated? We use "plot_correlation" function to investigate this.



Takeaways from this are;

1. Outcome (default.payment.next.month) has a strong positive correlation with PAY.
2. Overall, LIMIT_BAL has a relatively strong correlation with other factors (except SEX).
3. EDUCATION, MARRIAGE, AGE have relatively strong correlation with one another.
4. EDUCATION and AGE have a relatively weak correlation with PAY and BILL_AMT respectively.
5. PAY and BILL_AMT, BILL_AMT and PAY_AMT have strong correlation.

Then, we will look into these features further.

1 Outcome

First, we look into the outcome, "default.payment.next.month". The data description says, "Default payment, 1=yes, 0=no.[9]" We show the proportion of "0","1".

[9] https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset

```
##
##      0      1
## 0.7788 0.2212
```

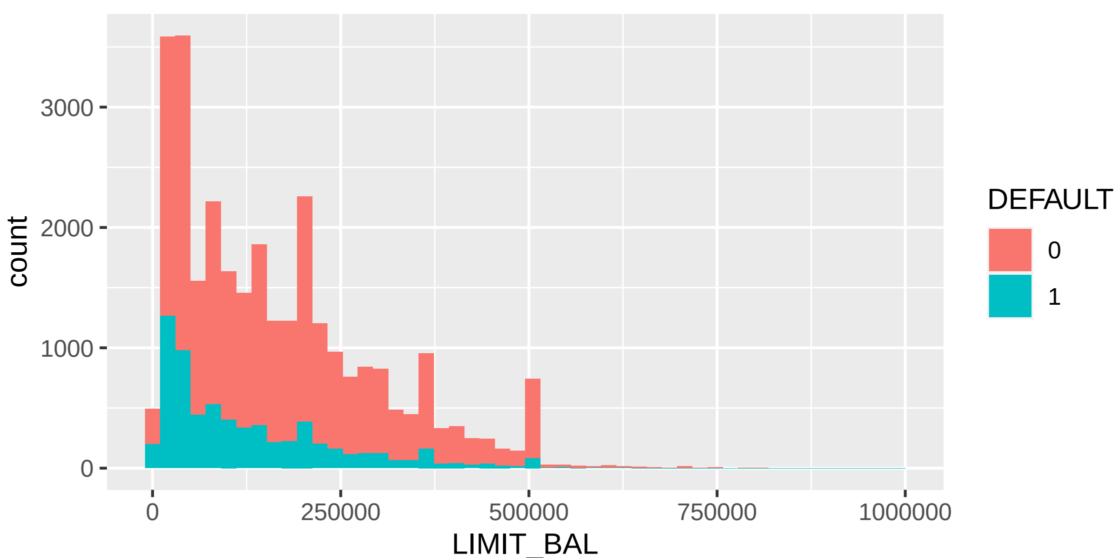
This means that if we predict all the outcome as “0”, we will get 77.9% accuracy. We need to take into account this fact. We change the name, “default.payment.next.month”, to“DEFAULT” for the sake of convenience. Also, we change this numeric variable into factor.

2 “LIMIT_BAL”

This is an “amount of given credit in NT dollars (includes individual and family/supplementary credit)[10]” It is numerical data.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10000   50000  140000  167484  240000 1000000
```

We draw its distribution filling the proportion of default.



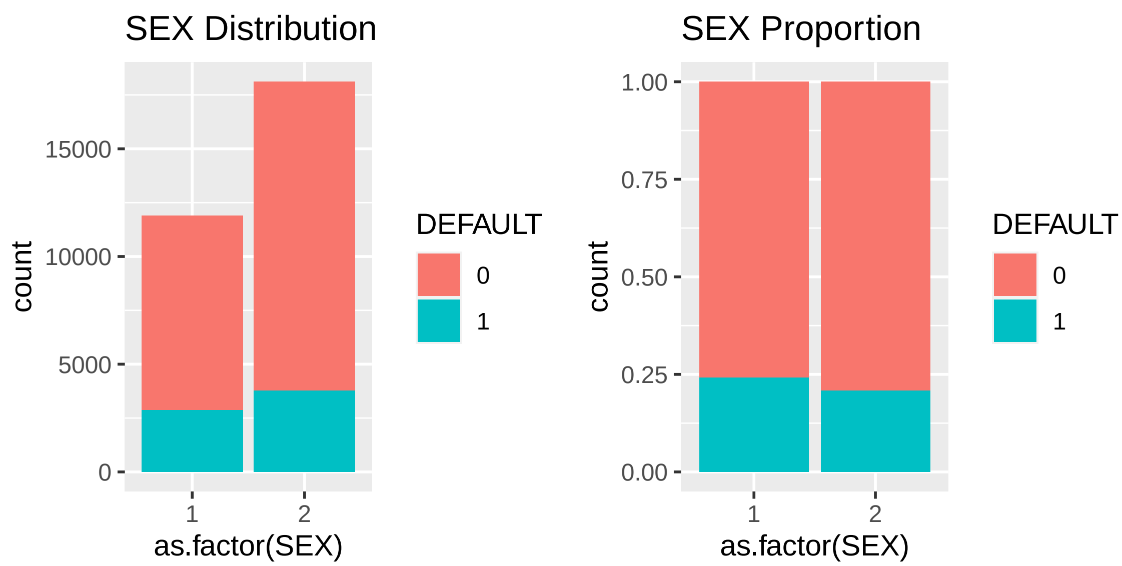
Distribution is skewed right. Default clients seem to be gathered around lower range of LIMIT_BAL values.

3 “SEX”

The values “1”, “2” correspond to male and female respectively[11]. Male is 40% and female is 60%.

We draw its distribution and its proportion in terms of default rates.

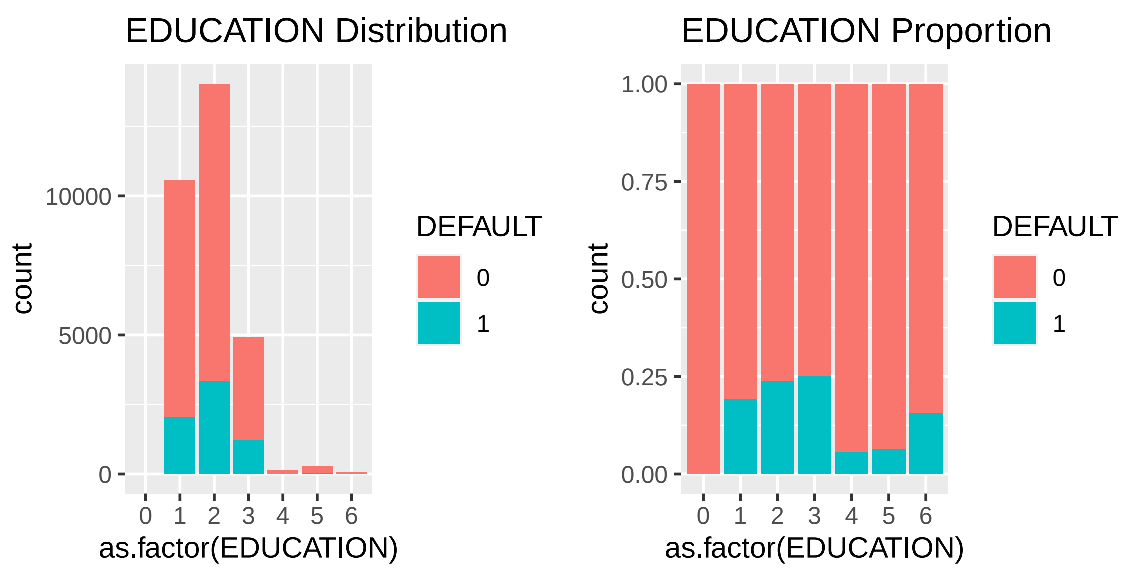
[10]https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset NT stands for “New Taiwan”.
[11]https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset



There seemed to be little difference between genders. This categorical variable is somewhat irrelevant to the outcome.

4 "EDUCATION"

In this variable, values are "1", "2", "3", "4", "5", "6". They are categorical values. The numbers have meanings as follows ; 1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, 6=unknown[12]. We plot its distribution and stacked bar graph.



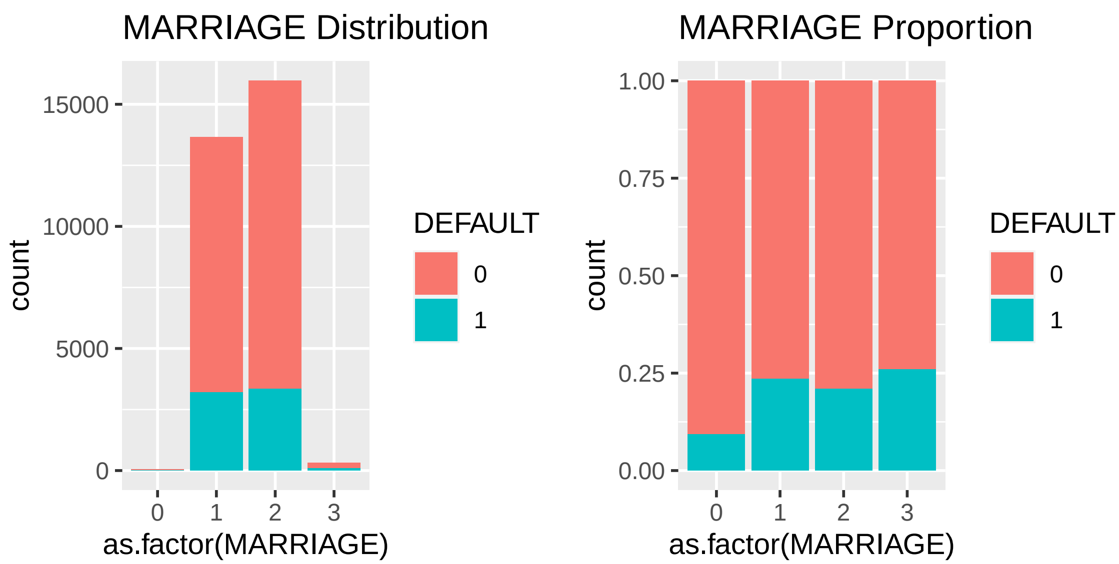
People whose final education is high school have relatively high default rate. On the other hand, people whose final education is graduate school have low default rate.

[12]https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset

5 "MARRIAGE"

This is also categorical data. The value number means clients' marital status[13]. 1=married, 2=single, 3=others. There are 54 individuals whose values are 0.

We draw its distribution graph and stacked bar graph.



There seems to be little difference among the groups.

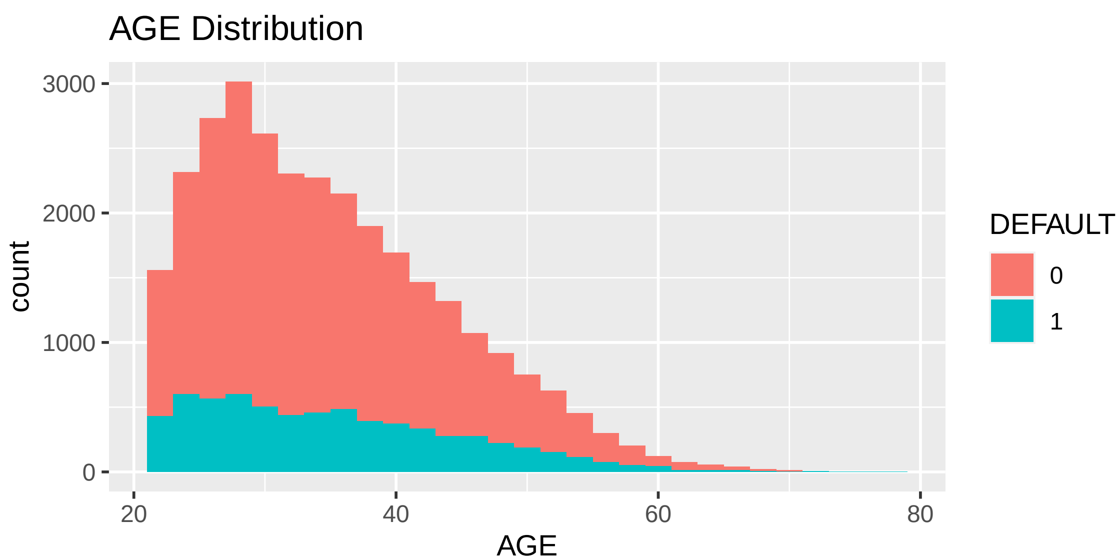
6 "AGE"

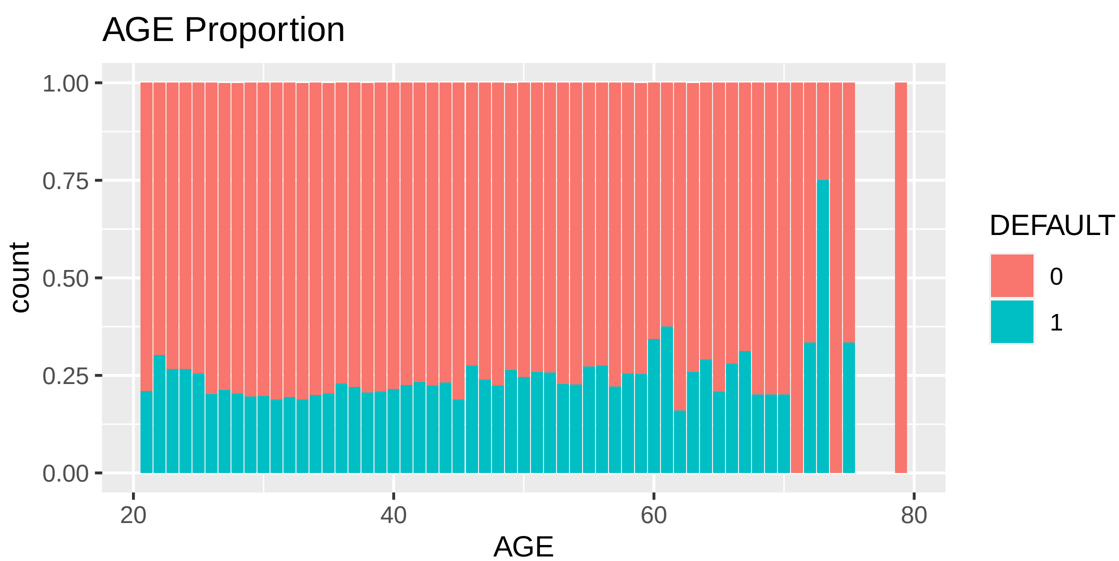
As its name indicates, it is numerical data.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.00   28.00   34.00   35.49   41.00   79.00
```

We plots its distribution as well as its default rate.

[13] https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset





The number of clients are decreasing as they get older. Regarding default clients, younger people in their early 20s and older age groups around 60s show higher proportions than other age groups.

7 "PAY"

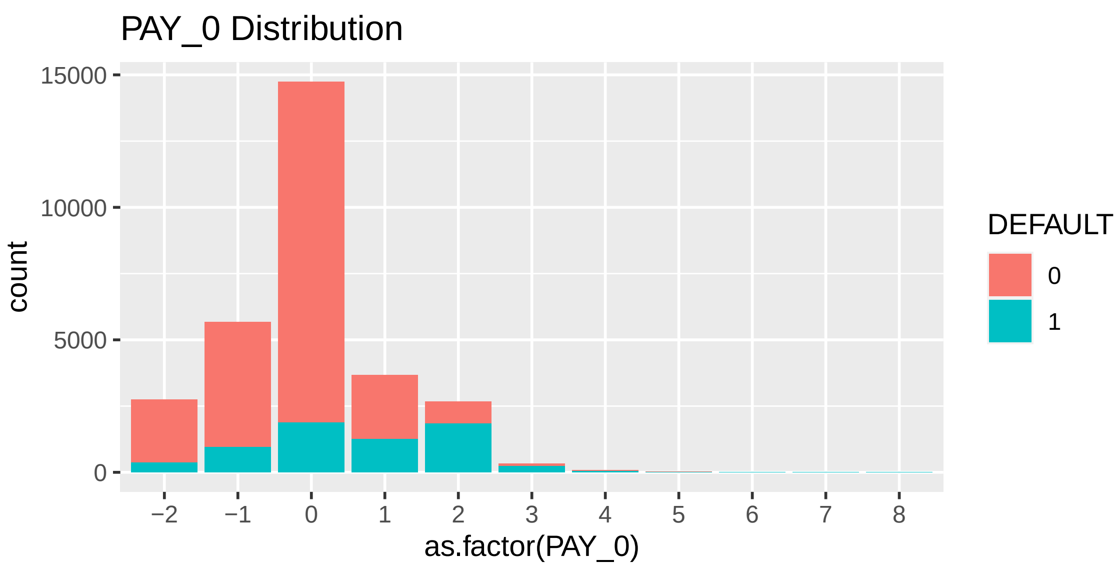
Variable from PAY_0, PAY_2 ~PAY_6 have the same values, as are explained in the description[14]. They are categorical data. PAY_0 means repayment status in September, 2005. Then go back in time by a month until April, 2005. Values are ;

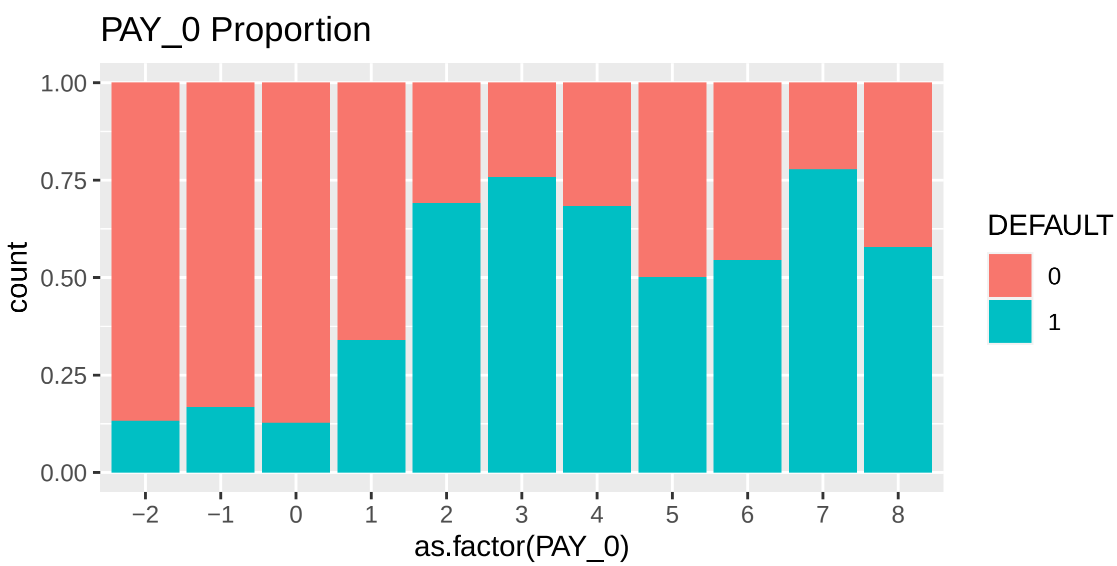
```
##  [1]  2 -1  0 -2  1  3  4  8  7  5  6
```

They mean; "-1"= pay duly, "1"= payment delay for 1 month, "2" = payment delay for 2 months, ... 9 = payment delay for 9 months and above. "0" and "-2" are not defined.

We draw PAY_0 distribution and stacked bar graph.hey are categorical data.

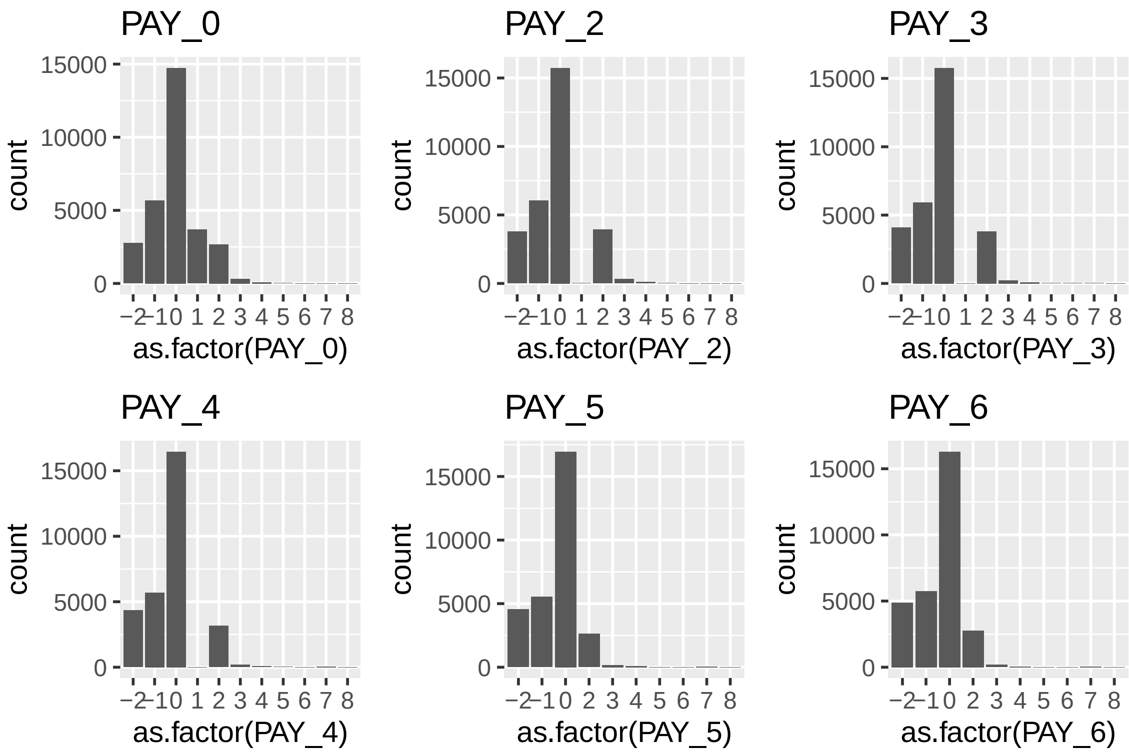
[14]https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset





We are not sure what “-2” and “0” mean. But from these two graphs we understand that as payment delay becomes longer, clients are more likely to come to a default.

PAY_2 ~ PAY_6 ’s structures are almost as the same as PAY_0. We show their distribution.



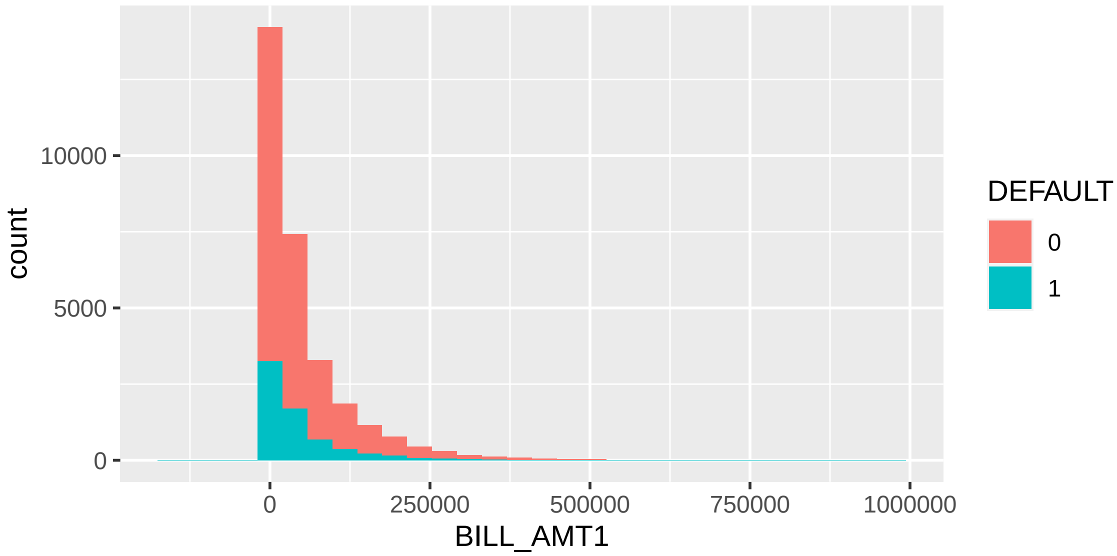
8 "BILL_AMT"

This variable means an amount of bill statement. This is numerical data.

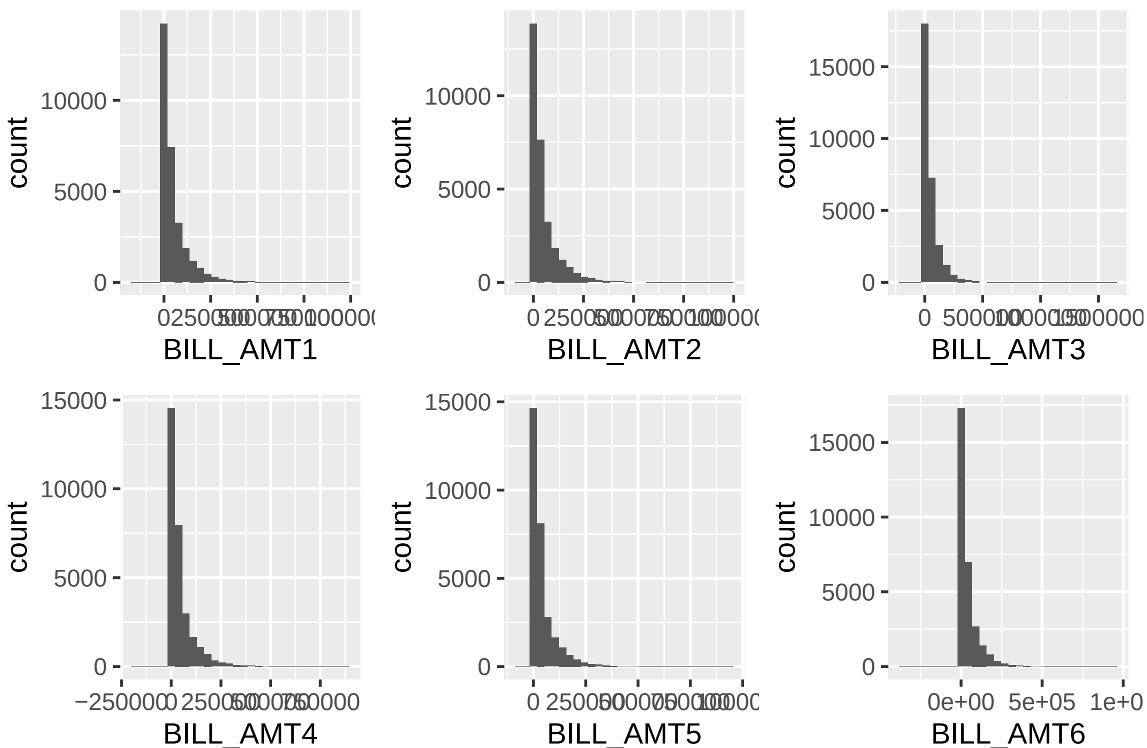
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -165580    3559   22382   51223   67091  964511
```

BILL_AMT1 is a record in September, 2005[15]. Then go back in time by a month until April, 2005. Likewise previous variables, we plot BILL_AMT distribution.

[15]https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset



From BILL_AMT1 to BILL_AMT6, their structures are almost the same as are shown in following plots.



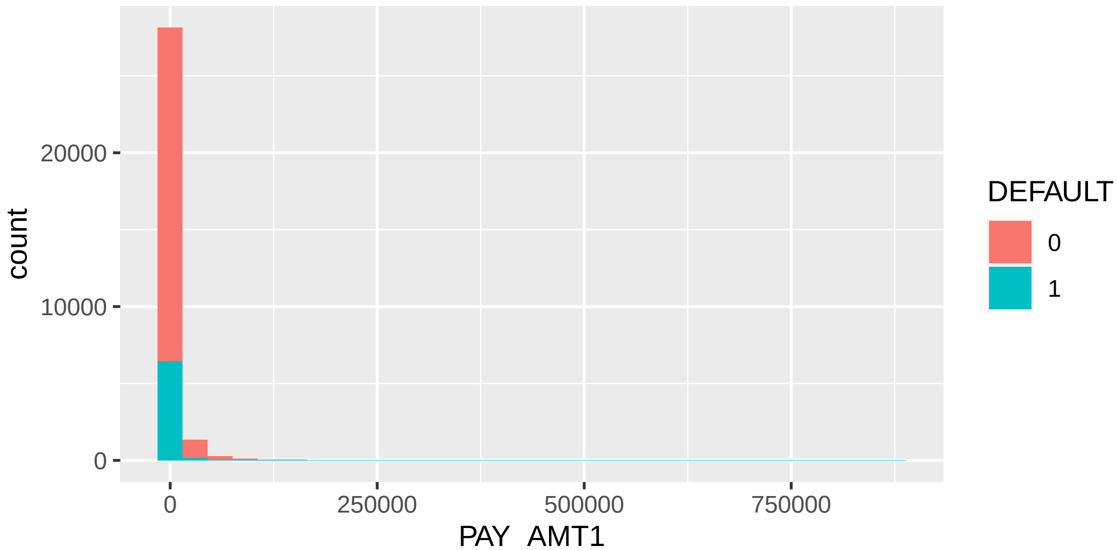
9 "PAY_AMT"

This variable means an amount of previous payment. This is numerical data. This is a summary of PAY_AMT1.

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0    1000    2100    5664    5006  873552
```

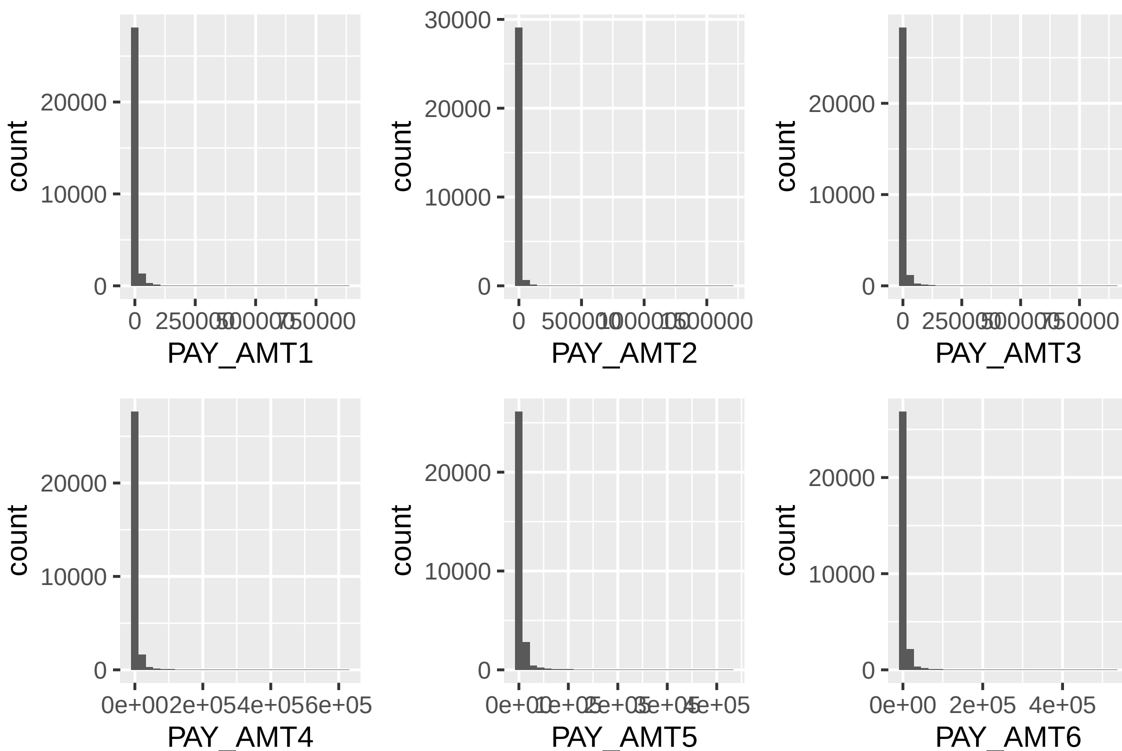
PAY_AMT1 is an amount of previous payment in September, 2005[16]. Likewise BILL_AMT, PAY_AMT goes back in time by a month from August to April, 2005 which is PAY_AMT6.

Here is PAY_AMT1's plot.



As we have seen a distribution summary, it is right skewed significantly. Maximum amount is more than 850,000 NT dollars, but its mean and median are 5664 and 2100 respectively.

From PAY_AMT1 to PAY_AMT6, their structures are almost the same as are shown in following plots.



[16] https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset

Data Preparation

Before going further, we will arrange our dataset to make it easier to investigate. First, we remove ID column, as it is irrelevant to the outcome.

Currently our data is composed of numerical values. Categorical values need to be changed to factors. As we saw in data exploration, SEX,EDUCATION,MARRIAGE, PAY_0~PAY_6 are categorical values.

Regarding numerical values, "LIMIT_BAL" ranges from 10000 to 1000000, but "AGE" from 21 to 79 as we saw in the previous section. When doing regression, these wide varieties of ranges of variables cause inaccuracy. Thus we "scale" these variables, using following formula.

$$Transformed.Values = \frac{Values - Mean}{Standard.Deviation}$$

Then we get variables whose mean is 0, and standard deviation is 1.

Then, we split the data into two. As we have relatively a large amount of data, 30000, the proportion we use is 80% and 20%. The smaller dataset will be used when evaluating a model at the final stage. We call it test_set.

We are going to use decision tree, and random forest later. They can be tuned to produce improved results by finding hyperparameters. If we tune the model using hyperparameters again and again, this might cause overfitting. To avoid this , we split the larger dataset again and produce two datasets, "train_set" and "validation_set". Split proportion is the same as before, 80% and 20%.

The three datasets have the almost same outcome ratio.

```
## # A tibble: 2 x 4
##   outcome train_set validation_set test_set
##     <dbl> <table>   <table>        <table>
## 1       0 0.7788311 0.7787961      0.7787035
## 2       1 0.2211689 0.2212039      0.2212965
```

Model analysis

1 Evaluation metrics

Generally speaking, overall accuracy is used to evaluate a model. But this dataset has imbalanced proportion of outcomes. Guessing all responses are 0, we calculate its confusion matrix, accuracy, sensitivity and specificity.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3739 1062
##          1    0    0
##
##                Accuracy : 0.7788
##                  95% CI : (0.7668, 0.7905)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 0.5082
##
##                   Kappa : 0
```

```
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.7788
##          Neg Pred Value :    NaN
##              Prevalence : 0.7788
##          Detection Rate : 0.7788
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

Even such a guess produces fairly good results., sensitivity (True Positive / True Positive+ False Negative) is 1.0, in contrast specificity (True Negative / True Negative + False Positive) is 0. Balanced accuracy (arithmetic mean of sensitivity and specificity) is 0.5, which is really a random guess. As such, the credit company falsely give credit to a lot of clients who will fail to repay a debt. The loss for the company would be huge.

Therefore, our goal is to pursue more accurate accuracy than guessing all responses are 0, taking account of improving specificity. We will use both *accuracy* and *balanced accuracy* to evaluate each model.

2 Logistic regression

As this is a classification, we use logistic regression. we use "glm" function. There are 24 predictors in the train_set. We use "step regression" to find the best logistic regression model.

Stepwise regression explanation. First we make null-model and full-model.

Forward and backward stepwise algorithm.

Predict by using validation_set. First we predict probabilities and then classify them using cut-off 0.5.

To show accuracy we use confusionMatrix function in caret library.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3567  714
##          1  172  348
##
##                Accuracy : 0.8155
##                  95% CI : (0.8042, 0.8263)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 2.329e-10
##
##                   Kappa : 0.3446
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9540
##             Specificity : 0.3277
```

```
##          Pos Pred Value : 0.8332
##          Neg Pred Value : 0.6692
##              Prevalence : 0.7788
##          Detection Rate : 0.7430
##    Detection Prevalence : 0.8917
##       Balanced Accuracy : 0.6408
##
##        'Positive' Class : 0
##
```

Make a table.

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9539984	0.3276836	0.640841

3 Decision tree default model

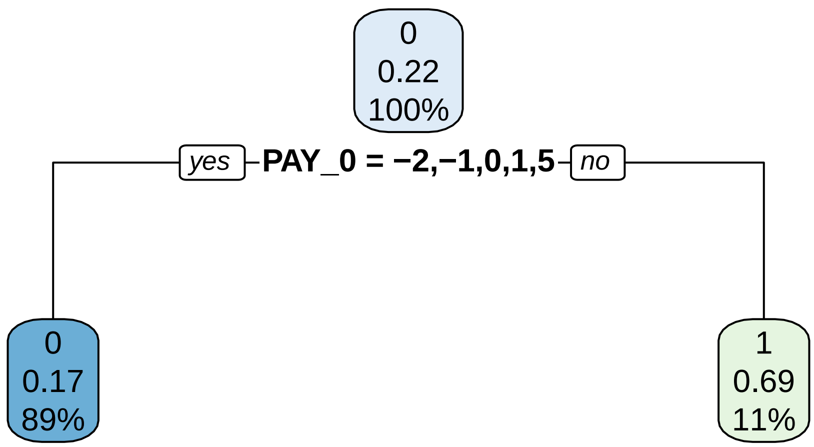
Use CART classification and regression tree. Rpart ~ using default minsplit=20, cp=0.01.

Predict.

Confusion Matrix.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3597  736
##          1  142  326
##
##                Accuracy : 0.8171
##                  95% CI : (0.8059, 0.828)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 3.487e-11
##
##                   Kappa : 0.3363
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9620
##             Specificity : 0.3070
##          Pos Pred Value : 0.8301
##          Neg Pred Value : 0.6966
##              Prevalence : 0.7788
##          Detection Rate : 0.7492
##    Detection Prevalence : 0.9025
##       Balanced Accuracy : 0.6345
##
##        'Positive' Class : 0
##
```

Draw decision tree rpart.plot is good function to show decision tree clearly.



Find used features.

```
##      PAY_0      PAY_4      PAY_5      PAY_6      PAY_3      PAY_2
## 1000.94794   38.19276   36.20872   26.78453   25.29650   21.82443
```

This model illustrates that PAY_0 is overwhelmingly important.

Make a table

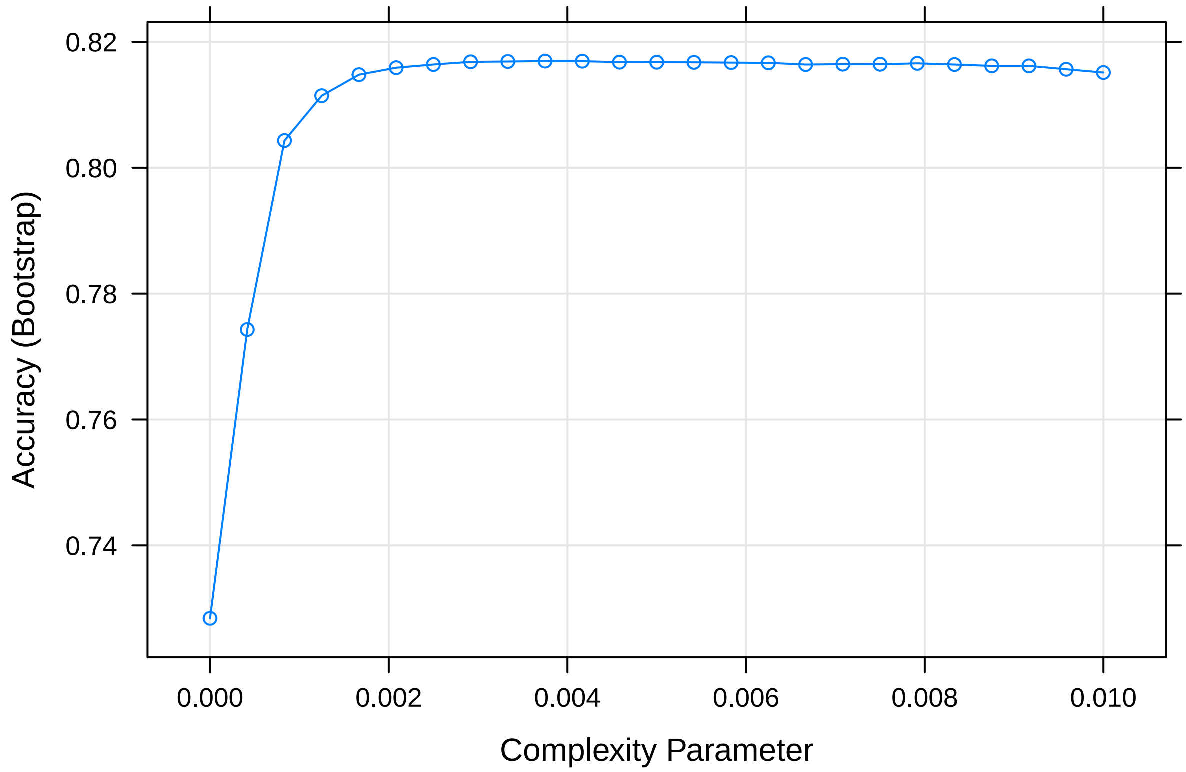
method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9539984	0.3276836	0.640841
CART default	0.8171214	0.9620219	0.3069680	0.634495

4 Decision tree further tuning

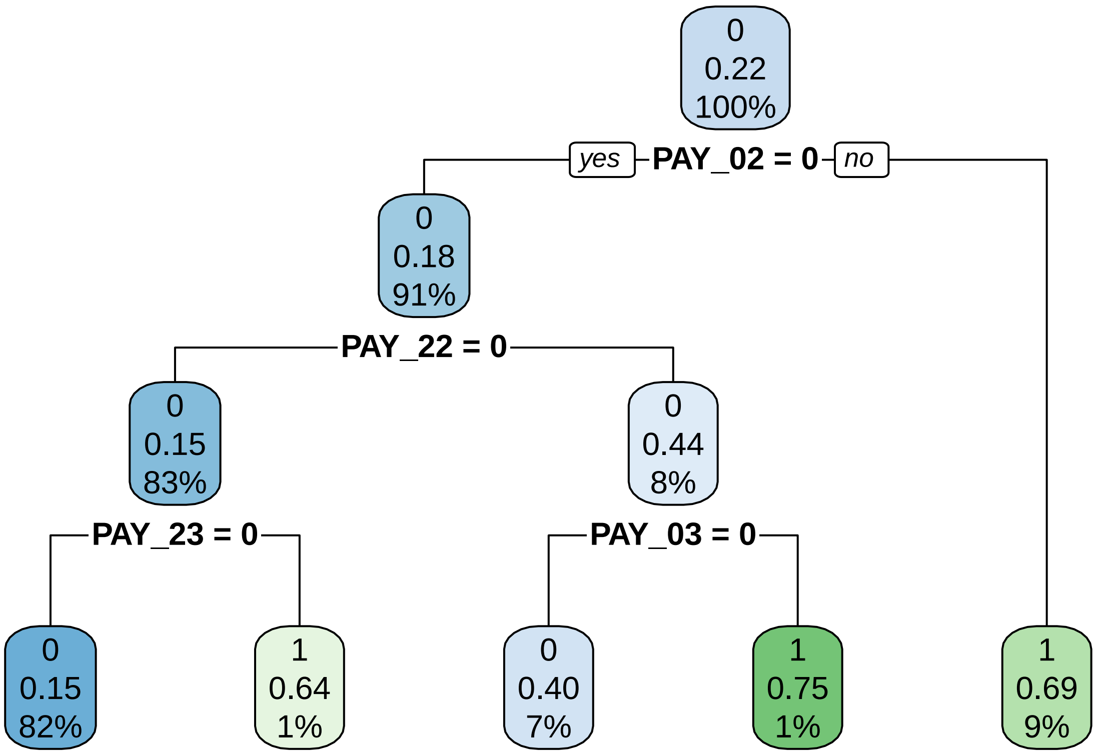
We use "train" function in "caret" package. and tune cp. Cross validation

rpart ~tuning using smaller cp, less than 0.01

Plot cp.



Draw decision tree. using rpart.plot.



Note: numeric values are scaled

Prediction.

Confusion matrix

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3587  730
##          1  152  332
##
##                Accuracy : 0.8163
##                  95% CI : (0.805, 0.8272)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 9.111e-11
##
##                   Kappa : 0.3378
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9593
##             Specificity : 0.3126
##          Pos Pred Value : 0.8309
##          Neg Pred Value : 0.6860
##              Prevalence : 0.7788
```

```
##          Detection Rate : 0.7471
##    Detection Prevalence : 0.8992
##       Balanced Accuracy : 0.6360
##
##        'Positive' Class : 0
##
```

Make a table.

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9539984	0.3276836	0.6408410
CART default	0.8171214	0.9620219	0.3069680	0.6344950
CART tuned cp	0.8162883	0.9593474	0.3126177	0.6359826

5 Random forest default

Using "ranger".

Model details.

```
## Ranger result
##
## Call:
##  ranger(formula = DEFAULT ~ ., data = train_set, probability = F)
##
## Type:                             Classification
## Number of trees:                  500
## Sample size:                      19198
## Number of independent variables:  23
## Mtry:                             4
## Target node size:                 1
## Variable importance mode:         none
## Splitrule:                        gini
## OOB prediction error:             18.30 %
```

Prediction.

Confusion matrix

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3564  698
##          1  175  364
##
##                Accuracy : 0.8182
##                  95% CI : (0.807, 0.829)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 1.018e-11
##
##                   Kappa : 0.3593
```

```
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9532
##             Specificity : 0.3427
##          Pos Pred Value : 0.8362
##          Neg Pred Value : 0.6753
##              Prevalence : 0.7788
##          Detection Rate : 0.7423
##    Detection Prevalence : 0.8877
##       Balanced Accuracy : 0.6480
##
##        'Positive' Class : 0
##
```

Make a table.

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9539984	0.3276836	0.6408410
CART default	0.8171214	0.9620219	0.3069680	0.6344950
CART tuned cp	0.8162883	0.9593474	0.3126177	0.6359826
random forest default	0.8181629	0.9531960	0.3427495	0.6479728

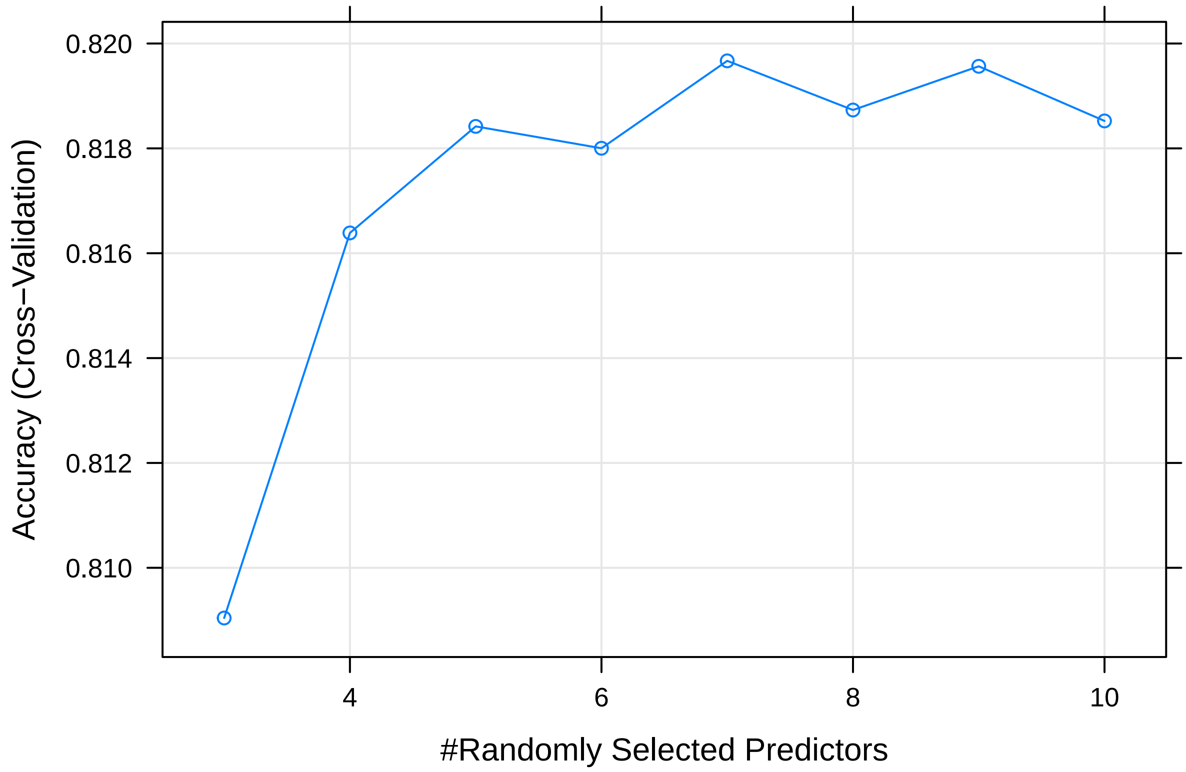
6 Random forest cross validation

Grid search

```
##    model     parameter                         label forReg forClass probModel
## 1 ranger          mtry #Randomly Selected Predictors   TRUE     TRUE      TRUE
## 2 ranger     splitrule                Splitting Rule   TRUE     TRUE      TRUE
## 3 ranger min.node.size             Minimal Node Size   TRUE     TRUE      TRUE
```

Make a model.

Plot.



Prediction.

Confusion Matrix

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3578  726
##          1  161  336
##
##                Accuracy : 0.8152
##                  95% CI : (0.804, 0.8261)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 2.935e-10
##
##                   Kappa : 0.3376
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9569
##             Specificity : 0.3164
##          Pos Pred Value : 0.8313
##          Neg Pred Value : 0.6761
##              Prevalence : 0.7788
##          Detection Rate : 0.7453
##    Detection Prevalence : 0.8965
```

```
##       Balanced Accuracy : 0.6367
##
##        'Positive' Class : 0
##
```

Make a table.

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9539984	0.3276836	0.6408410
CART default	0.8171214	0.9620219	0.3069680	0.6344950
CART tuned cp	0.8162883	0.9593474	0.3126177	0.6359826
random forest default	0.8181629	0.9531960	0.3427495	0.6479728
random forest tuned	0.8152468	0.9569404	0.3163842	0.6366623

Evaluation

Best performance in terms of balanced accuracy is "random forest default model" Best performance in terms of accuracy is "CART default model" Then evaluate by using test_set.

Make a table.

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
CART default	0.8226962	0.9621228	0.3320783	0.6471006
Random forest default	0.8178637	0.9492831	0.3554217	0.6523524

Conclusion

###