Report on Default of Credit Card Clients Dataset

Masayoshi Sato

2021/6/26

Contents

Introduction

Finance thought to be a field where machine learning can be effective. It is surrounded by uncertainties, such as economic downturn, a collapse of markets. Individuals are no exception, we are not sure that a person is credible enough to lend money. They might be a deadbeat, or struggle in a significant debt, even though they look credible. This is the reason why machine learning plays a role in predicting uncertain economic future.

In this paper, we will deal with a familiar problem. Based on objective data, can we predict effectively whether a credit user will pay their debt or result in default? Traditionally, finding a credible borrower have been a skill and experience nurtured by financial institutions, like banks, credit company. Instead, we will look into open data, and using machine learning models, such as logistic regression, decision tree, and random forest. We will fit the data with these models and find the model which will show the most accurate prediction.

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.

Our goal is to find a classification model which predicts the most accurate outcome. However, its distribution of outcome 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. Overall 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.

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

[1]https://cran.r-project.org/web/packages/tidyverse/index.html
[2]https://cran.r-project.org/web/packages/DataExplorer/vignettes/dataexplorer-intro.html
[3]https://cran.r-project.org/web/packages/gridExtra/index.html
[4]https://cran.r-project.org/web/packages/rpart/rpart.pdf
[5]https://topepo.github.io/caret/
[6]https://cran.r-project.org/web/packages/ranger/ranger.pdf
Compared to "randomForest", "ranger" is very quick and easy to operate.
[7]https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset
[8]https://github.com/masa951125/Final_project/raw/main/UCI_Credit_Card.csv

Data Exploration

First, we need to check the downloaded dataset. It has 30000 rows and 25 columns. First six rows are like these.

```
## # A tibble: 6 x 25
##      ID LIMIT_BAL   SEX EDUCATION MARRIAGE   AGE PAY_0 PAY_2 PAY_3 PAY_4 PAY_5
##   <dbl>     <dbl> <dbl>     <dbl>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1     1     20000     2         2        1    24     2     2    -1    -1    -2
## 2     2    120000     2         2        2    26    -1     2     0     0     0
## 3     3     90000     2         2        2    34     0     0     0     0     0
## 4     4     50000     2         2        1    37     0     0     0     0     0
## 5     5     50000     1         2        1    57    -1     0    -1     0     0
## 6     6     50000     1         1        2    37     0     0     0     0     0
## # ... with 14 more variables: PAY_6 <dbl>, BILL_AMT1 <dbl>, BILL_AMT2 <dbl>,
## #   BILL_AMT3 <dbl>, BILL_AMT4 <dbl>, BILL_AMT5 <dbl>, BILL_AMT6 <dbl>,
## #   PAY_AMT1 <dbl>, PAY_AMT2 <dbl>, PAY_AMT3 <dbl>, PAY_AMT4 <dbl>,
## #   PAY_AMT5 <dbl>, PAY_AMT6 <dbl>, default.payment.next.month <dbl>
```

We look into it further using "str" and "summary" function.

```
## tibble [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 14000
##  $ 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(),
```

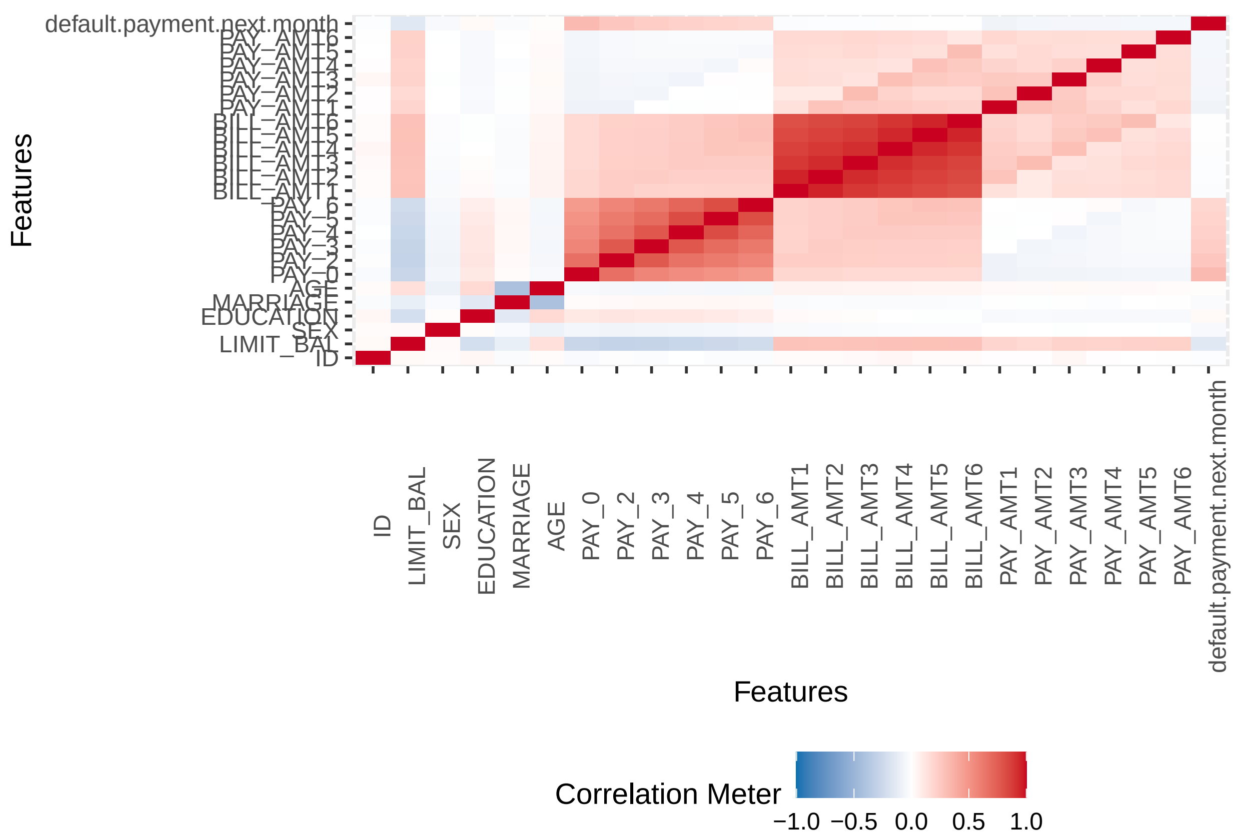
```
##   ..   MARRIAGE = col_double(),
##   ..   AGE = col_double(),
##   ..   PAY_0 = col_double(),
##   ..   PAY_2 = col_double(),
##   ..   PAY_3 = col_double(),
##   ..   PAY_4 = col_double(),
##   ..   PAY_5 = col_double(),
##   ..   PAY_6 = col_double(),
##   ..   BILL_AMT1 = col_double(),
##   ..   BILL_AMT2 = col_double(),
##   ..   BILL_AMT3 = col_double(),
##   ..   BILL_AMT4 = col_double(),
##   ..   BILL_AMT5 = col_double(),
##   ..   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()
##   .. )
```

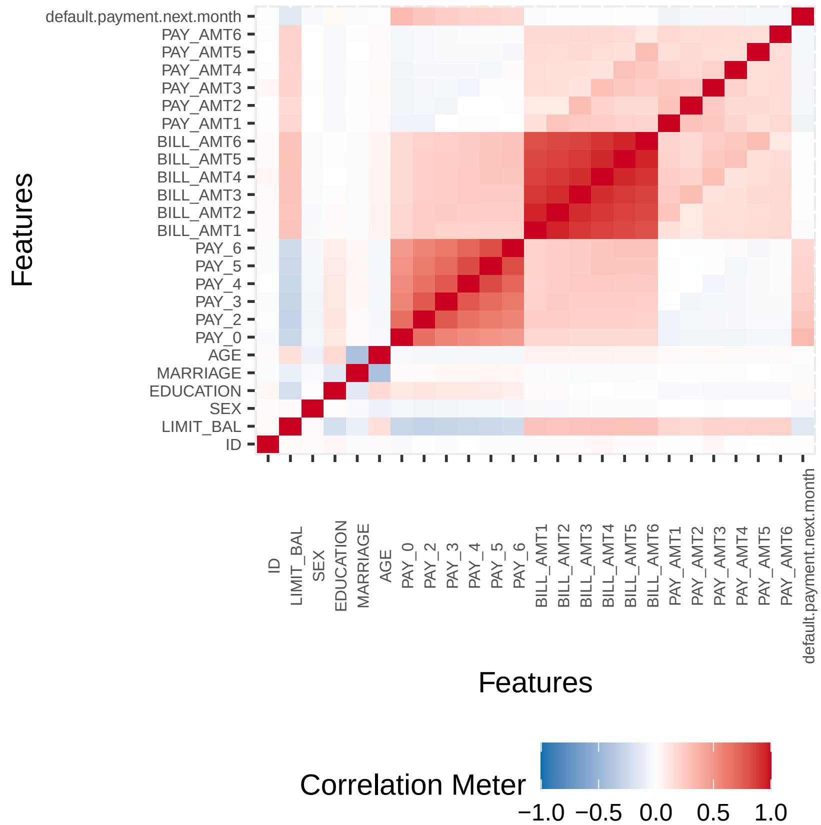
```
##        ID          LIMIT_BAL            SEX          EDUCATION
##  Min.   :    1   Min.   :  10000   Min.   :1.000   Min.   :0.000
##  1st Qu.: 7501   1st Qu.:  50000   1st Qu.:1.000   1st Qu.:1.000
##  Median :15000   Median : 140000   Median :2.000   Median :2.000
##  Mean   :15000   Mean   : 167484   Mean   :1.604   Mean   :1.853
##  3rd Qu.:22500   3rd Qu.: 240000   3rd Qu.:2.000   3rd Qu.:2.000
##  Max.   :30000   Max.   :1000000   Max.   :2.000   Max.   :6.000
##     MARRIAGE          AGE            PAY_0             PAY_2
##  Min.   :0.000   Min.   :21.00   Min.   :-2.0000   Min.   :-2.0000
##  1st Qu.:1.000   1st Qu.:28.00   1st Qu.:-1.0000   1st Qu.:-1.0000
##  Median :2.000   Median :34.00   Median : 0.0000   Median : 0.0000
##  Mean   :1.552   Mean   :35.49   Mean   :-0.0167   Mean   :-0.1338
##  3rd Qu.:2.000   3rd Qu.:41.00   3rd Qu.: 0.0000   3rd Qu.: 0.0000
##  Max.   :3.000   Max.   :79.00   Max.   : 8.0000   Max.   : 8.0000
##      PAY_3             PAY_4             PAY_5             PAY_6
##  Min.   :-2.0000   Min.   :-2.0000   Min.   :-2.0000   Min.   :-2.0000
##  1st Qu.:-1.0000   1st Qu.:-1.0000   1st Qu.:-1.0000   1st Qu.:-1.0000
##  Median : 0.0000   Median : 0.0000   Median : 0.0000   Median : 0.0000
##  Mean   :-0.1662   Mean   :-0.2207   Mean   :-0.2662   Mean   :-0.2911
##  3rd Qu.: 0.0000   3rd Qu.: 0.0000   3rd Qu.: 0.0000   3rd Qu.: 0.0000
##  Max.   : 8.0000   Max.   : 8.0000   Max.   : 8.0000   Max.   : 8.0000
##    BILL_AMT1         BILL_AMT2        BILL_AMT3         BILL_AMT4
##  Min.   :-165580   Min.   :-69777   Min.   :-157264   Min.   :-170000
##  1st Qu.:   3559   1st Qu.:  2985   1st Qu.:   2666   1st Qu.:   2327
##  Median :  22382   Median : 21200   Median :  20089   Median :  19052
##  Mean   :  51223   Mean   : 49179   Mean   :  47013   Mean   :  43263
##  3rd Qu.:  67091   3rd Qu.: 64006   3rd Qu.:  60165   3rd Qu.:  54506
##  Max.   : 964511   Max.   :983931   Max.   :1664089   Max.   : 891586
##    BILL_AMT5        BILL_AMT6          PAY_AMT1         PAY_AMT2
##  Min.   :-81334   Min.   :-339603   Min.   :     0   Min.   :      0
##  1st Qu.:  1763   1st Qu.:   1256   1st Qu.:  1000   1st Qu.:    833
```

```
##  Median : 18105   Median :  17071   Median :  2100   Median :   2009
##  Mean   : 40311   Mean   :  38872   Mean   :  5664   Mean   :   5921
##  3rd Qu.: 50191   3rd Qu.:  49198   3rd Qu.:  5006   3rd Qu.:   5000
##  Max.   :927171   Max.   : 961664   Max.   :873552   Max.   :1684259
##     PAY_AMT3          PAY_AMT4          PAY_AMT5           PAY_AMT6
##  Min.   :     0   Min.   :     0   Min.   :     0.0   Min.   :     0.0
##  1st Qu.:   390   1st Qu.:   296   1st Qu.:   252.5   1st Qu.:   117.8
##  Median :  1800   Median :  1500   Median :  1500.0   Median :  1500.0
##  Mean   :  5226   Mean   :  4826   Mean   :  4799.4   Mean   :  5215.5
##  3rd Qu.:  4505   3rd Qu.:  4013   3rd Qu.:  4031.5   3rd Qu.:  4000.0
##  Max.   :896040   Max.   :621000   Max.   :426529.0   Max.   :528666.0
##  default.payment.next.month
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.2212
##  3rd Qu.:0.0000
##  Max.   :1.0000
```

"Default.payment.next.month" is an outcome of the dataset which has values of 0 and 1. Other features seem to be either numerical or categorical data. "SEX", "EDUCATION", "MARRIAGE" and "PAY_0" -"PAY_6" look like categorical data, as their values are limited number of integers. Other features seem to be numerical. From checking its summary, we understand there are no NAs in the dataset.

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





Takeaways from this are;

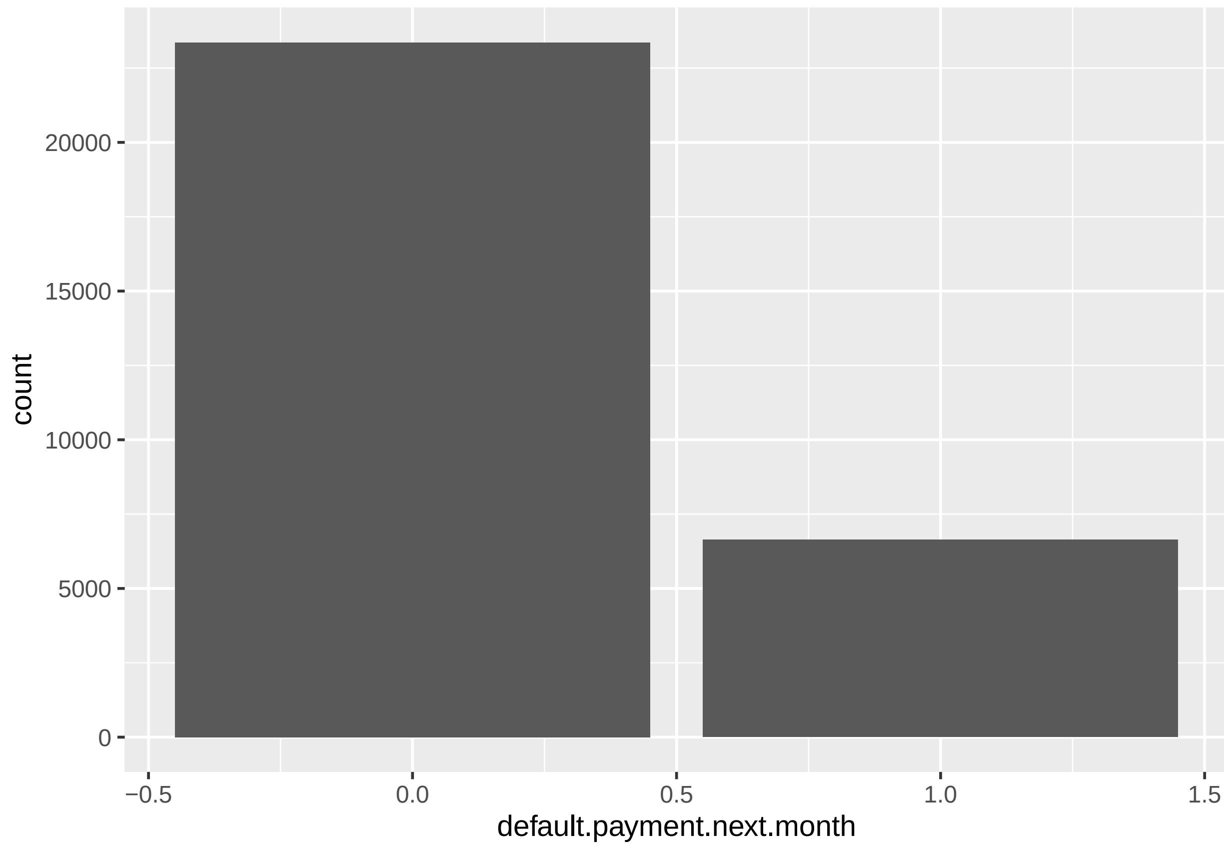
1. Outcome (default.payment.next.month) has a strong positive correlation with PAY.
2. Overall, LIMIT_BAL has a relatively strong correlation with other factors (except SEX).
3. EDUCATION, MARRIAGE, AGE have relatively strong correlation with one another.
4. EDUCATION and AGE have a relatively weak correlation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

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



This variable is unevenly distributed. Then we see the proportion of "0","1".

```
##
##      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.

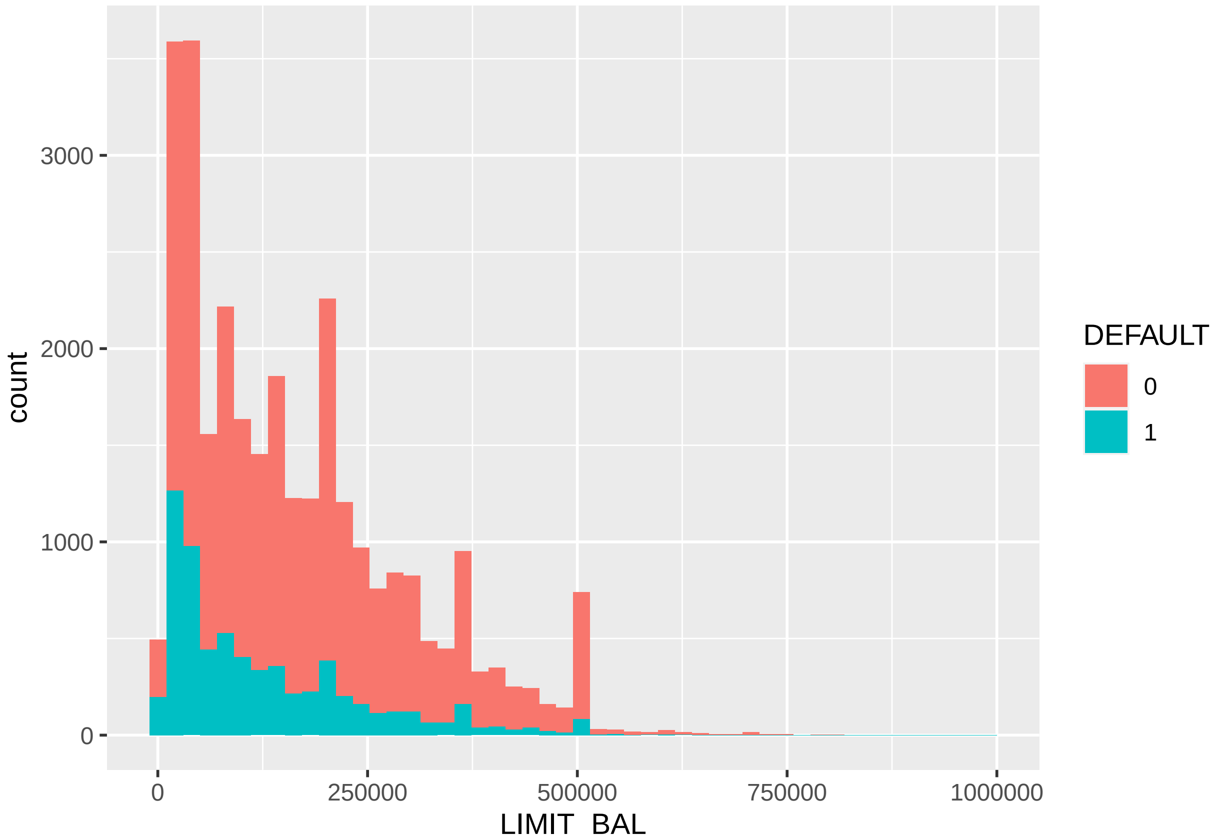
2 "LIMIT_BAL"

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

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

We draw its distribution filling the proportion of default.

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



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

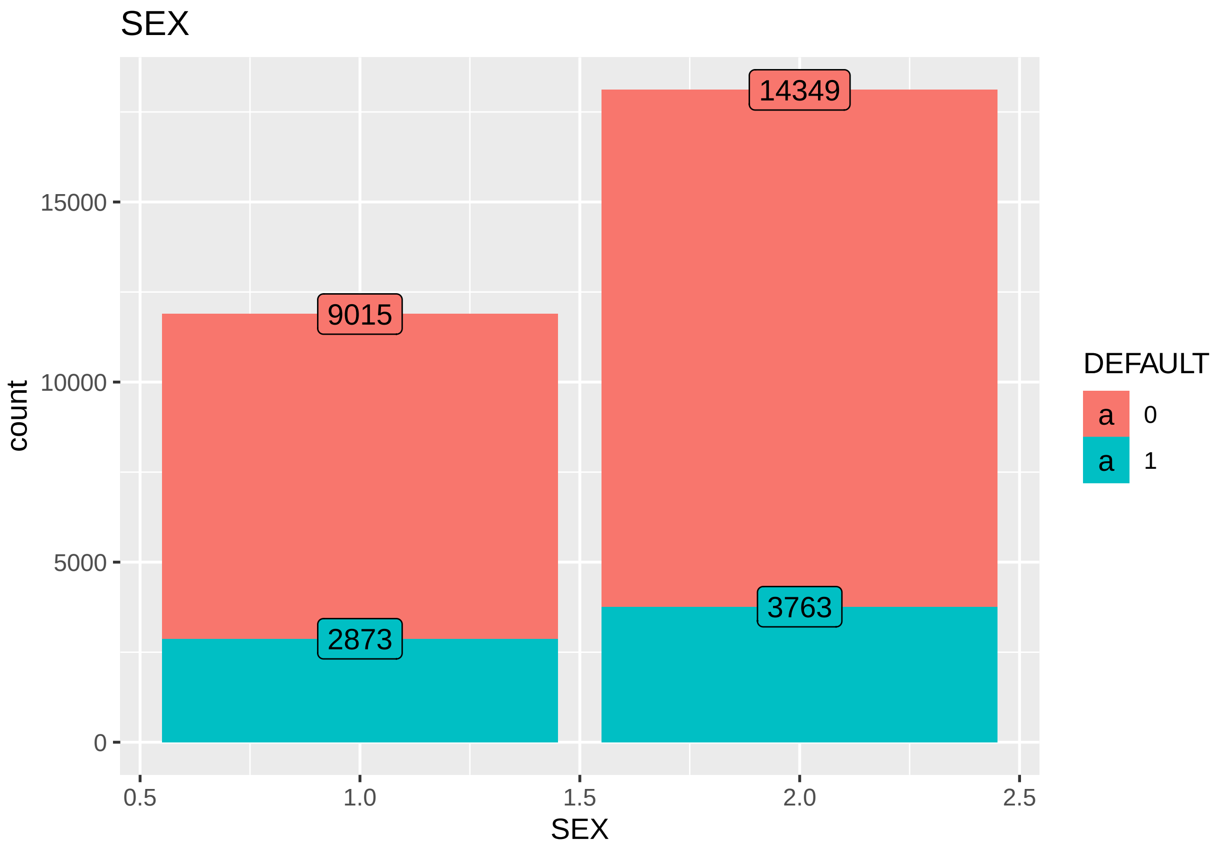
3 "SEX"

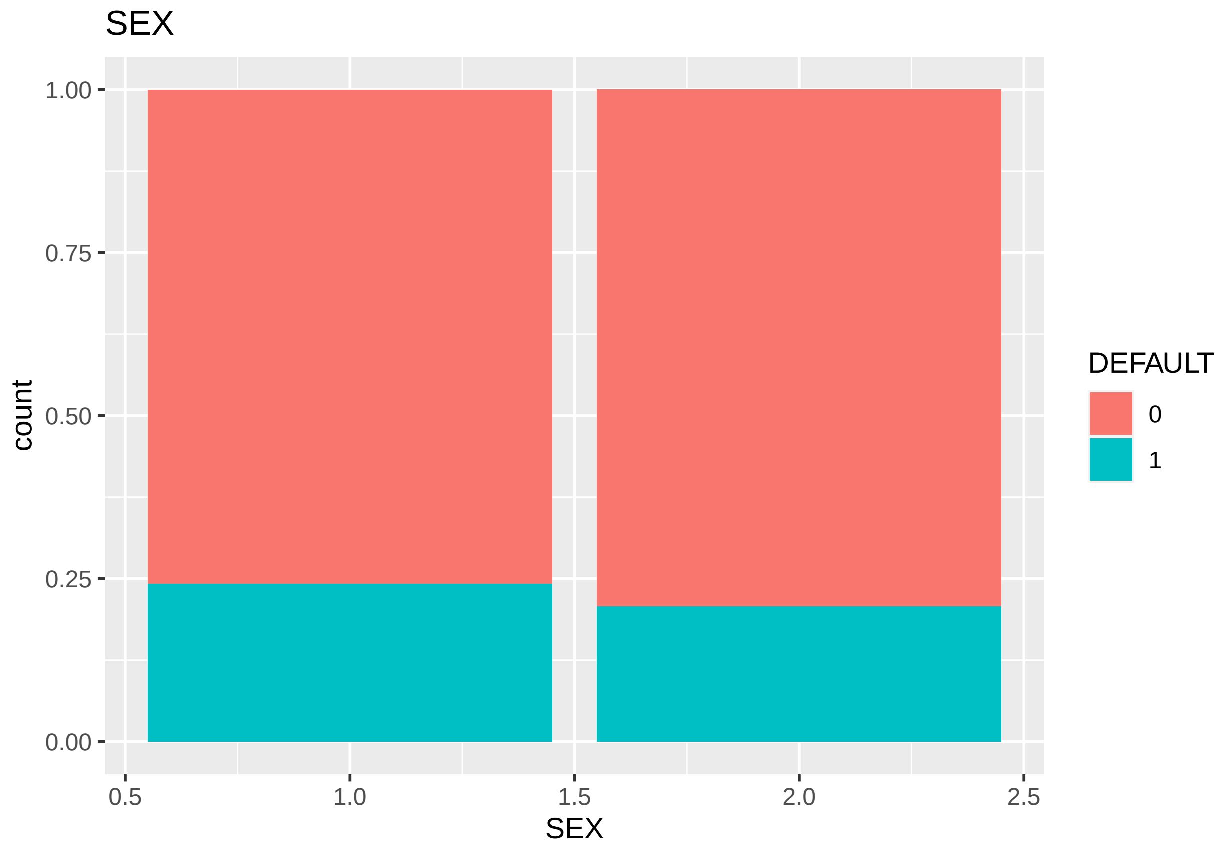
The values "1", "2" correspond to male and female respectively[11]. Male is 40% and female is 60%.

```
##
##         1         2
## 0.3962667 0.6037333
```

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

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


SEX
count
15000
10000
5000
0
9015
2873
14349
3763
0.5
1.0
1.5
2.0
2.5
SEX
DEFAULT
0
1

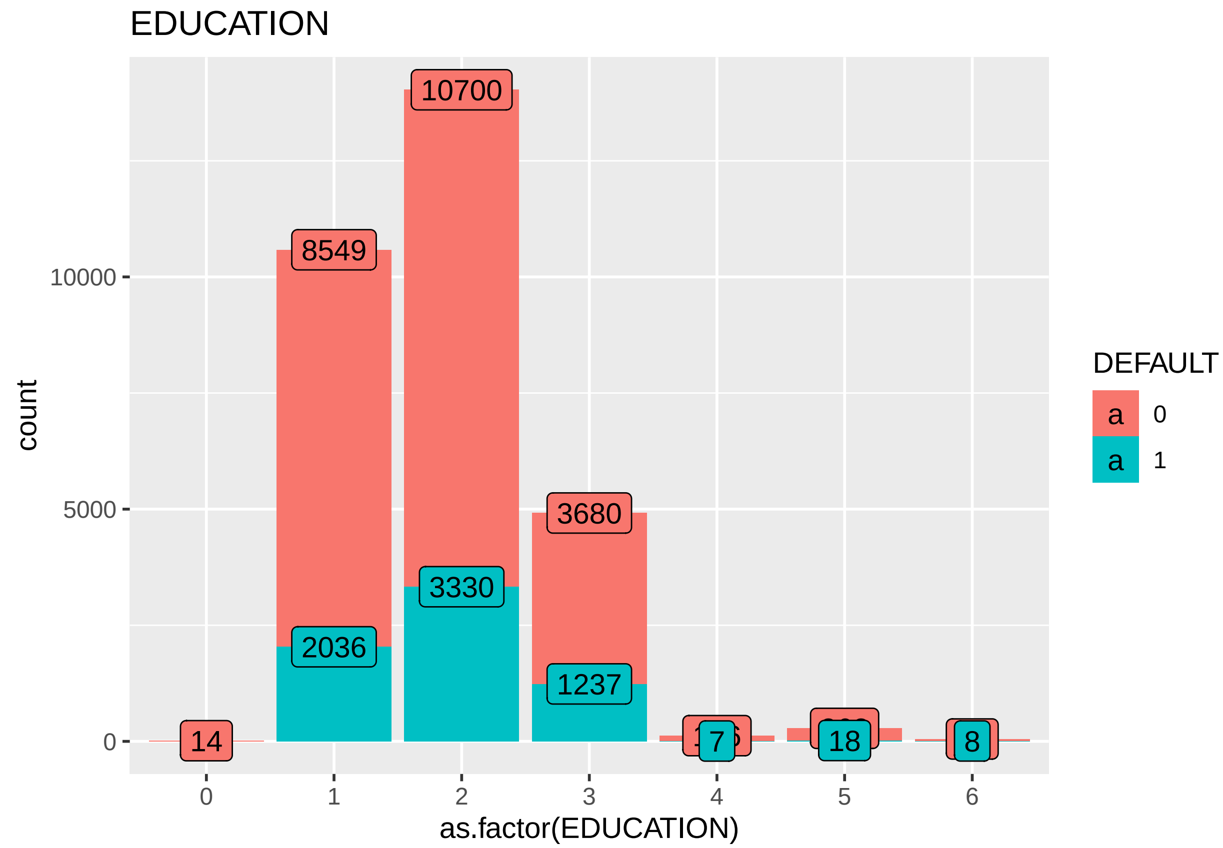


There seemed to be little difference between genders. This categorical variable is somewhat irrelevant to the outcome.

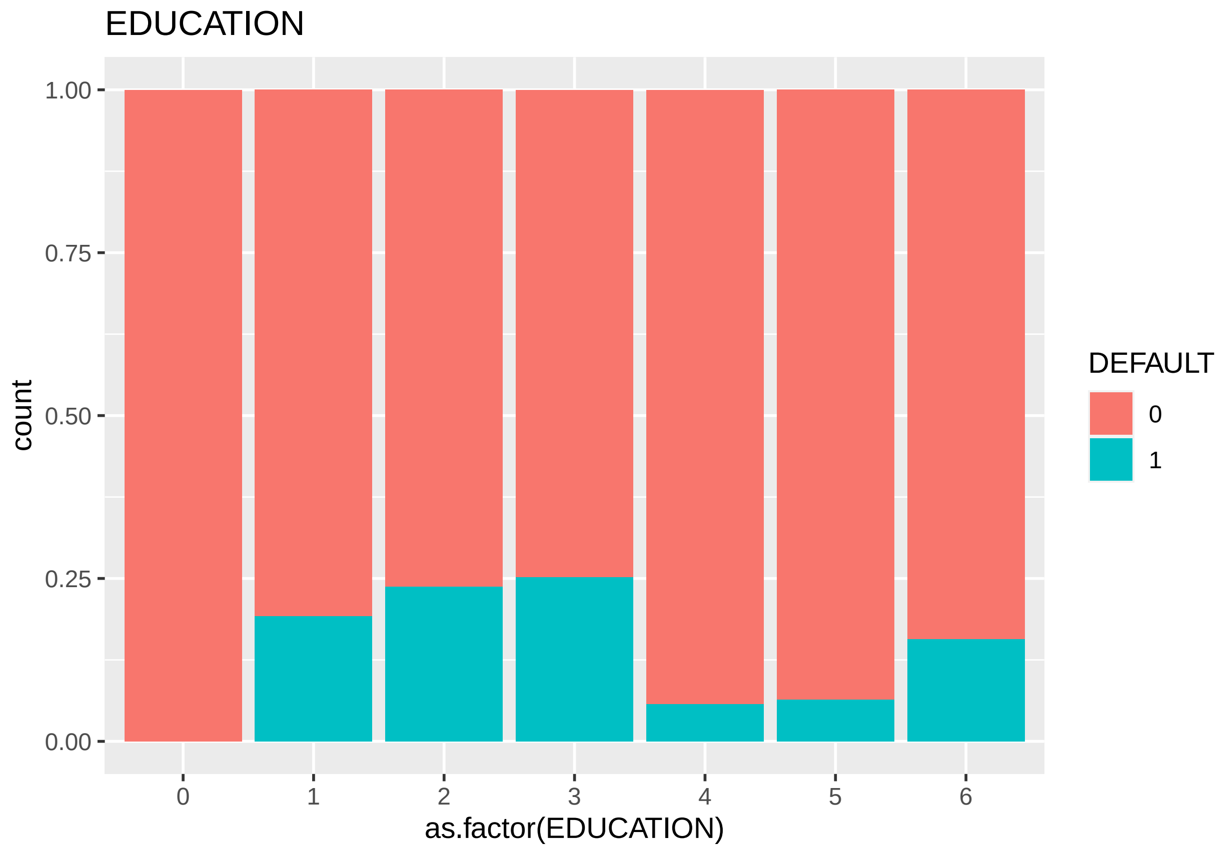
4 "EDUCATION"

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.

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



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.

5 "MARRIAGE"

Kaggle's data explanation says;

marital status. 1=married, 2=single, 3=others.

```
summary(original_default$ MARRIAGE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   2.000   1.552   2.000   3.000
```

```
unique(original_default$ MARRIAGE)
```

```
## [1] 1 2 3 0
```

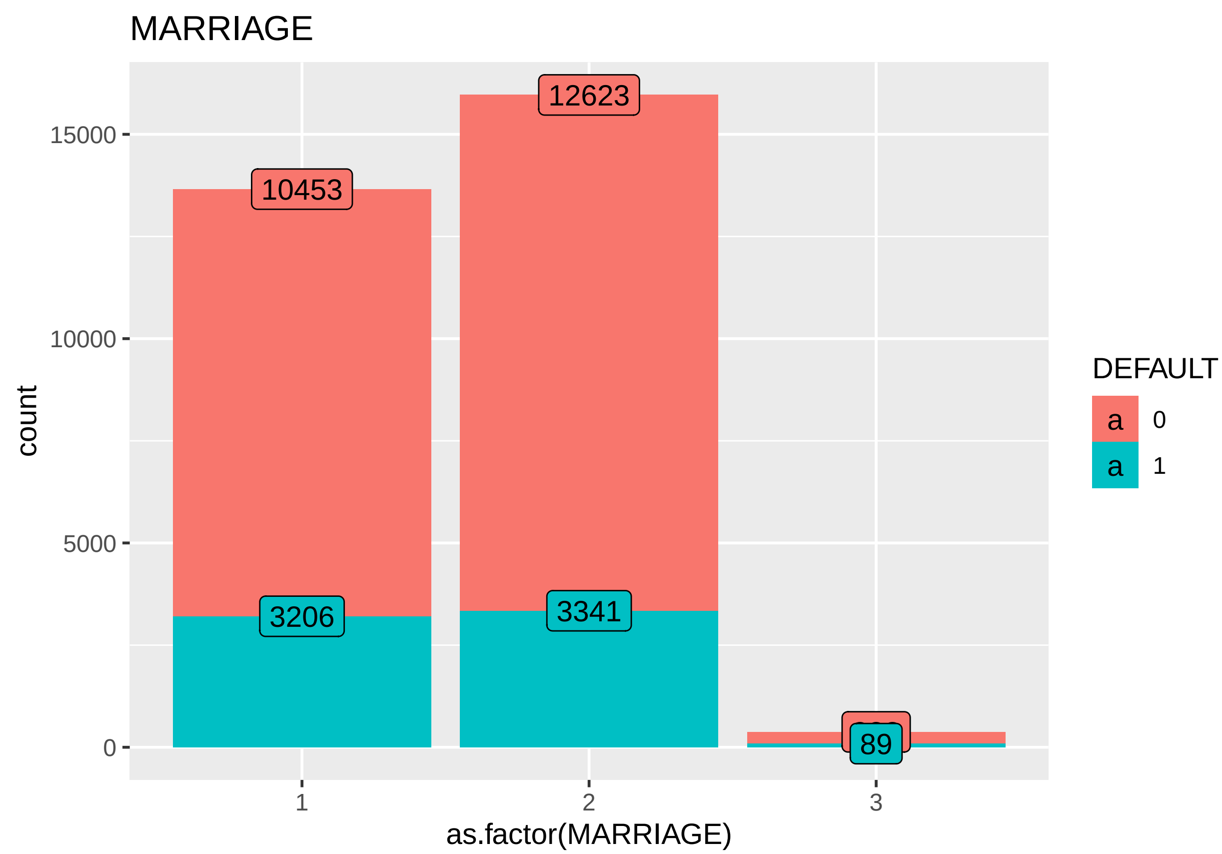
categorical data

0 is not defined. 0 can be included in 3.

```
original_default$MARRIAGE <- ifelse(original_default$MARRIAGE== 0, 3,
                                    original_default$MARRIAGE)
```

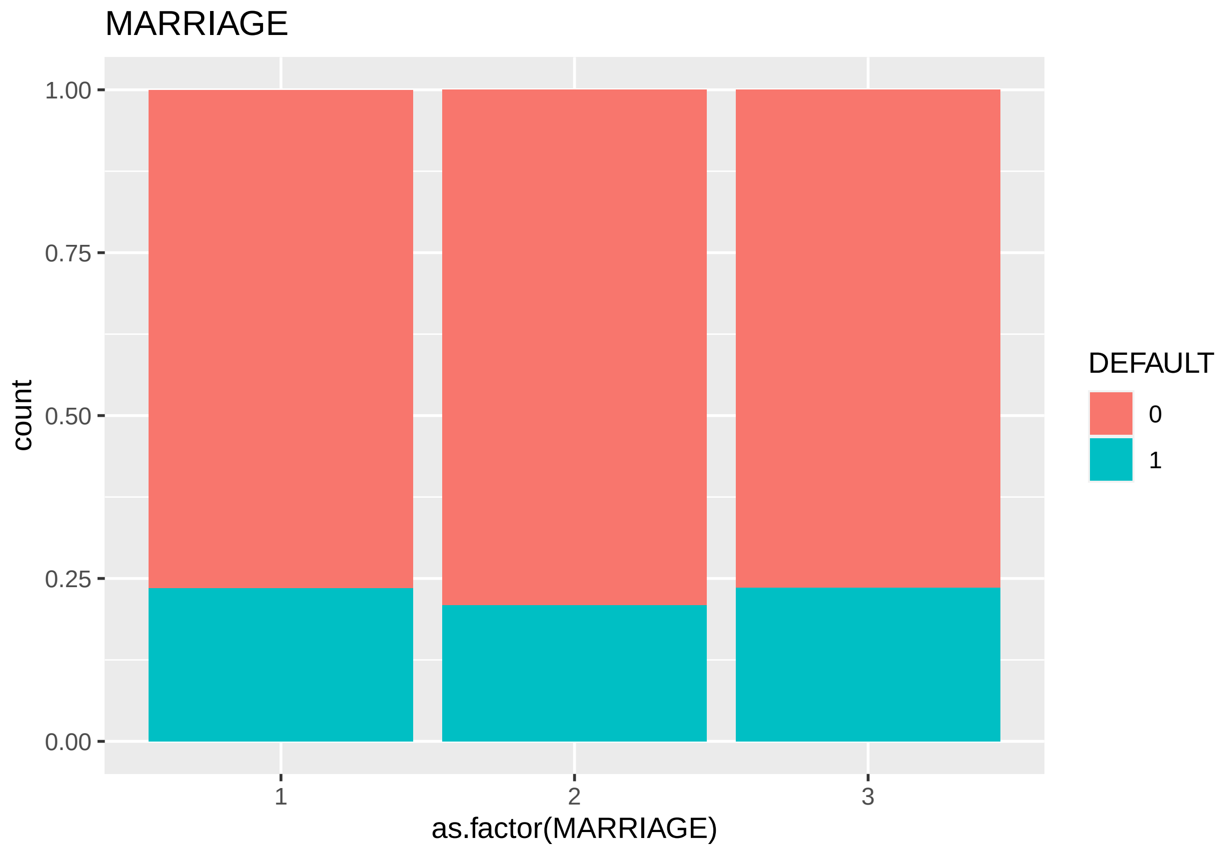
Plot.

```
original_default %>% ggplot(aes(x=as.factor(MARRIAGE), fill= DEFAULT)) +
  geom_bar() +
  ggtitle("MARRIAGE")+
  stat_count(aes(label = ..count..), geom = "label")# illustrate numbers
```



Stack bar graph

```
original_default %>% ggplot(aes(x=as.factor(MARRIAGE), fill= DEFAULT)) +
  geom_bar(position="fill") +
  ggtitle("MARRIAGE")
```



There seems to be little difference among the groups.

6 "AGE"

```
summary(original_default$AGE)
```

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

numeric data

Plot.

```
ggplot(data=original_default, aes(AGE, fill=DEFAULT)) +geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

7 "PAY"

Kaggle's data explanation says;

PAY_0 means repayment status in September, 2005.

-1=pay duly, 1=payment delay for one month, 2=payment delay for two months, … 8=payment delay for eight months, 9=payment delay for nine months and above. Regarding values from PAY_2 to PAY_6, the scales are the same as PAY_0. As the number increases, the date of repayment status goes back in time by a month until April, 2005 which is PAY_6.

. PAY_0.

```
summary(original_default$PAY_0)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
## -2.0000 -1.0000  0.0000 -0.0167  0.0000  8.0000
```

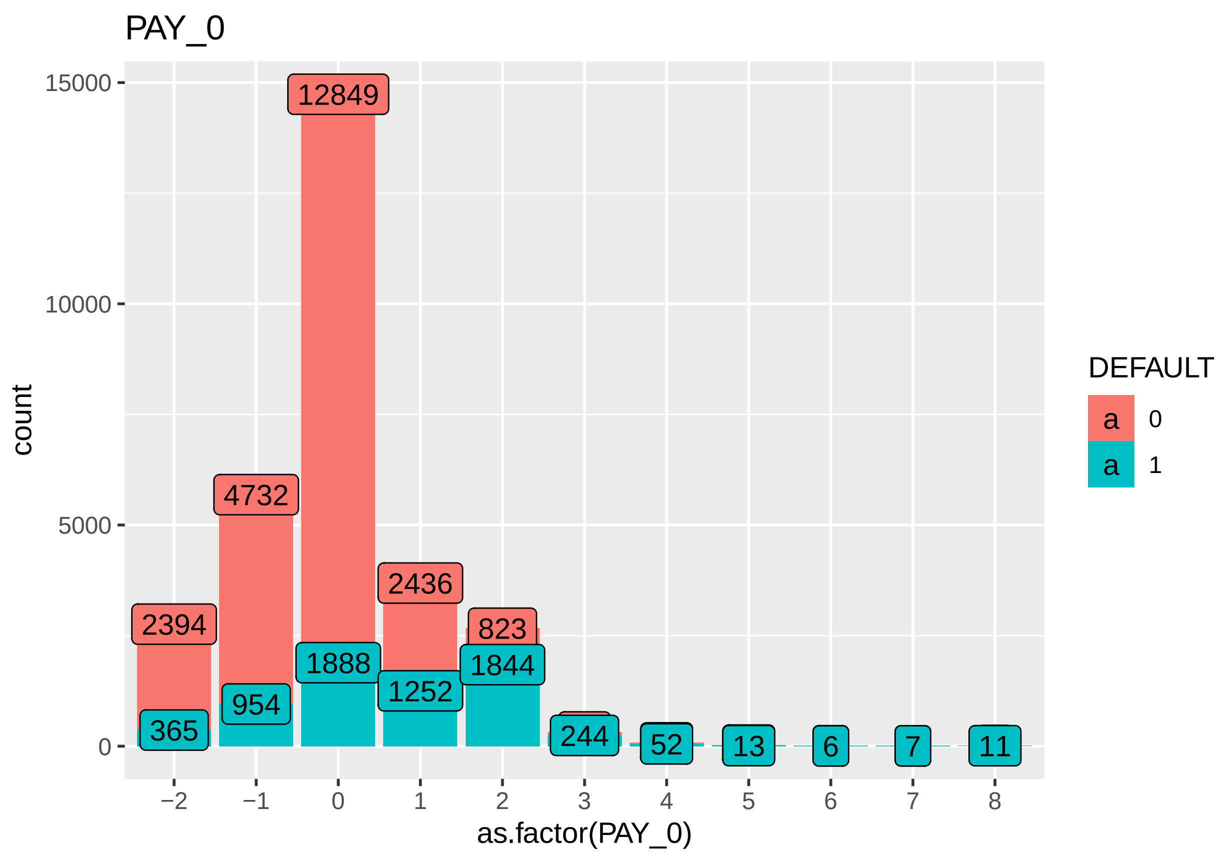
```
unique(original_default$PAY_0)
```

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

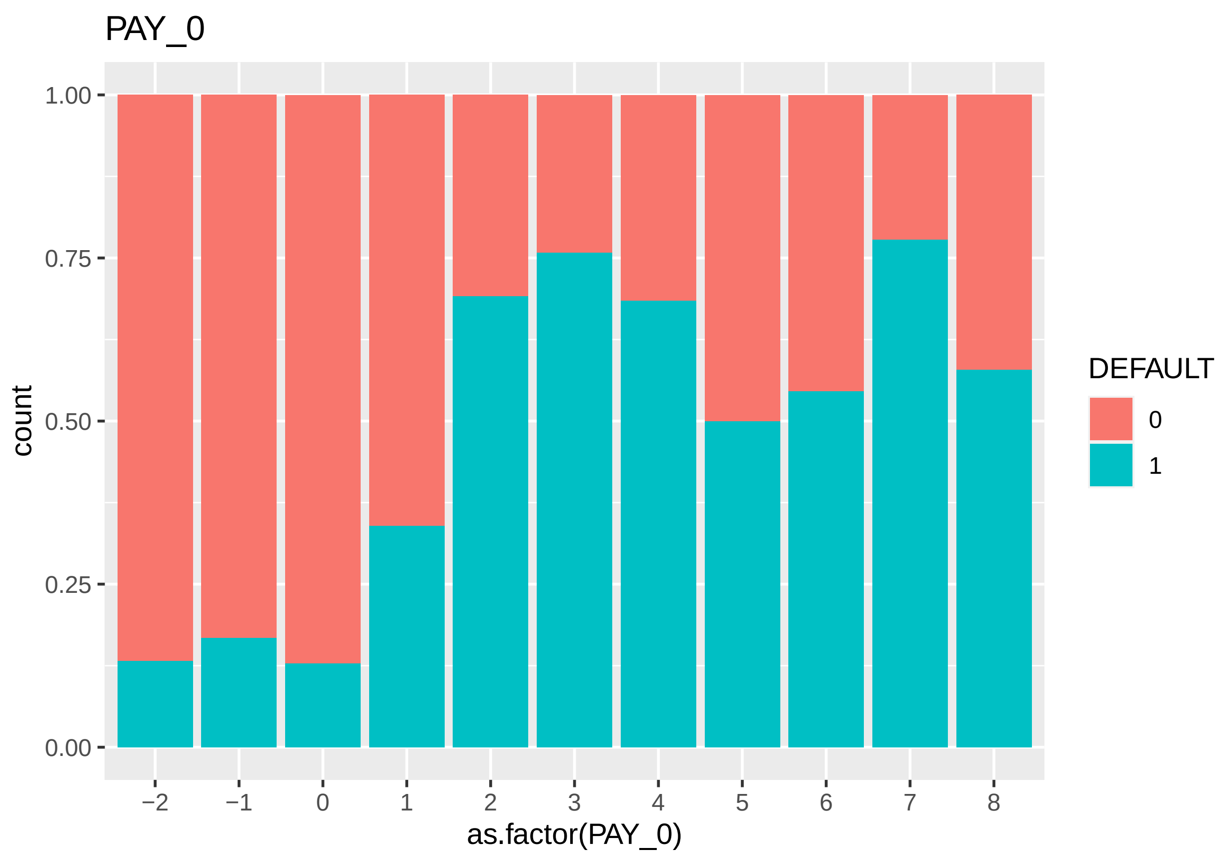
They are categorical data.

Plot.

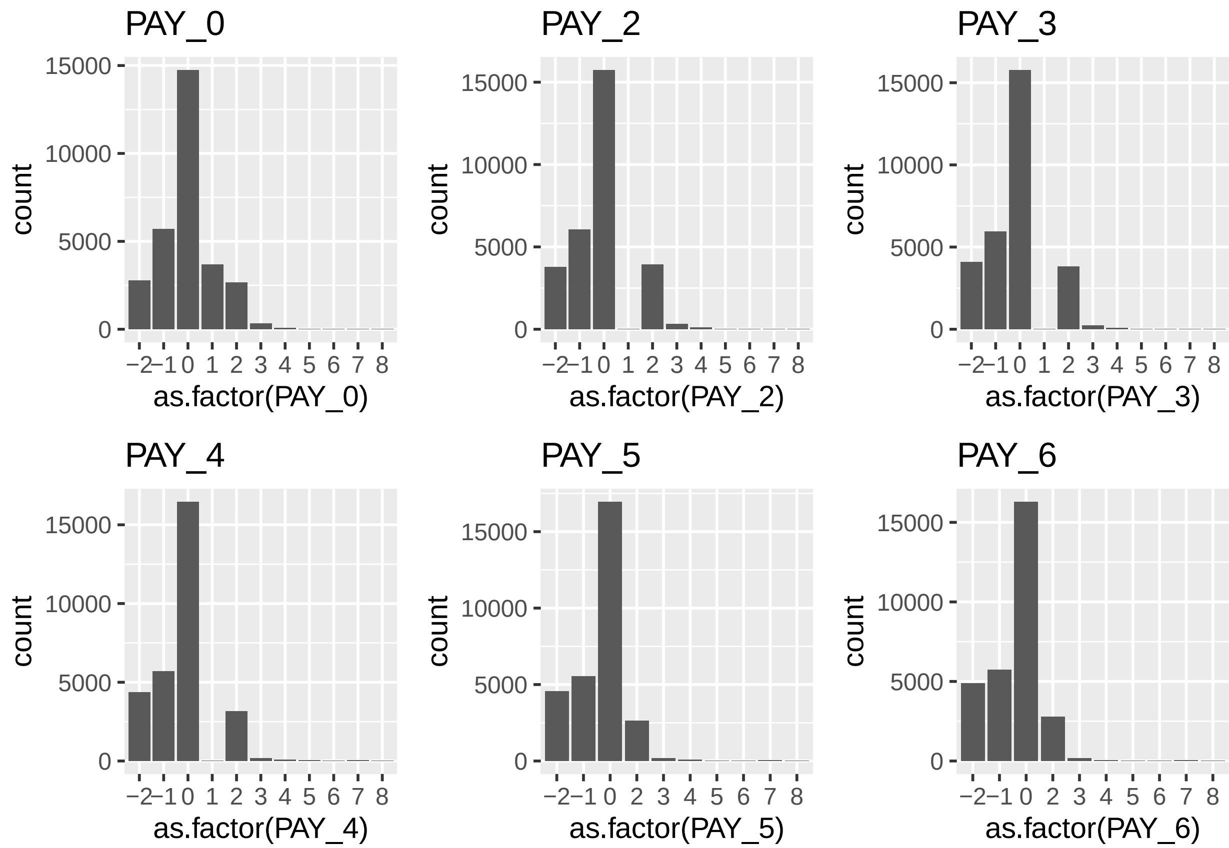
```
original_default %>% ggplot(aes(x=as.factor(PAY_0), fill= DEFAULT)) +
  geom_bar() +
  ggtitle("PAY_0")+
  stat_count(aes(label = ..count..), geom = "label")# illustrate numbers
```



Stack bar graph PAY_0.



PAY_2 ~ PAY_6 's structures are almost as same as PAY_0. Show distribution.



8 "BILL_AMT"

Kaggle's data explanation says;

BILL_AMT1 is an amount of bill statement in September, 2005 (NT dollar). Likewise PAY, BILL_AMT goes back in time by a month from August to April, 2005 which is BILL_AMT6.

```
summary(original_default$BILL_AMT1)
```

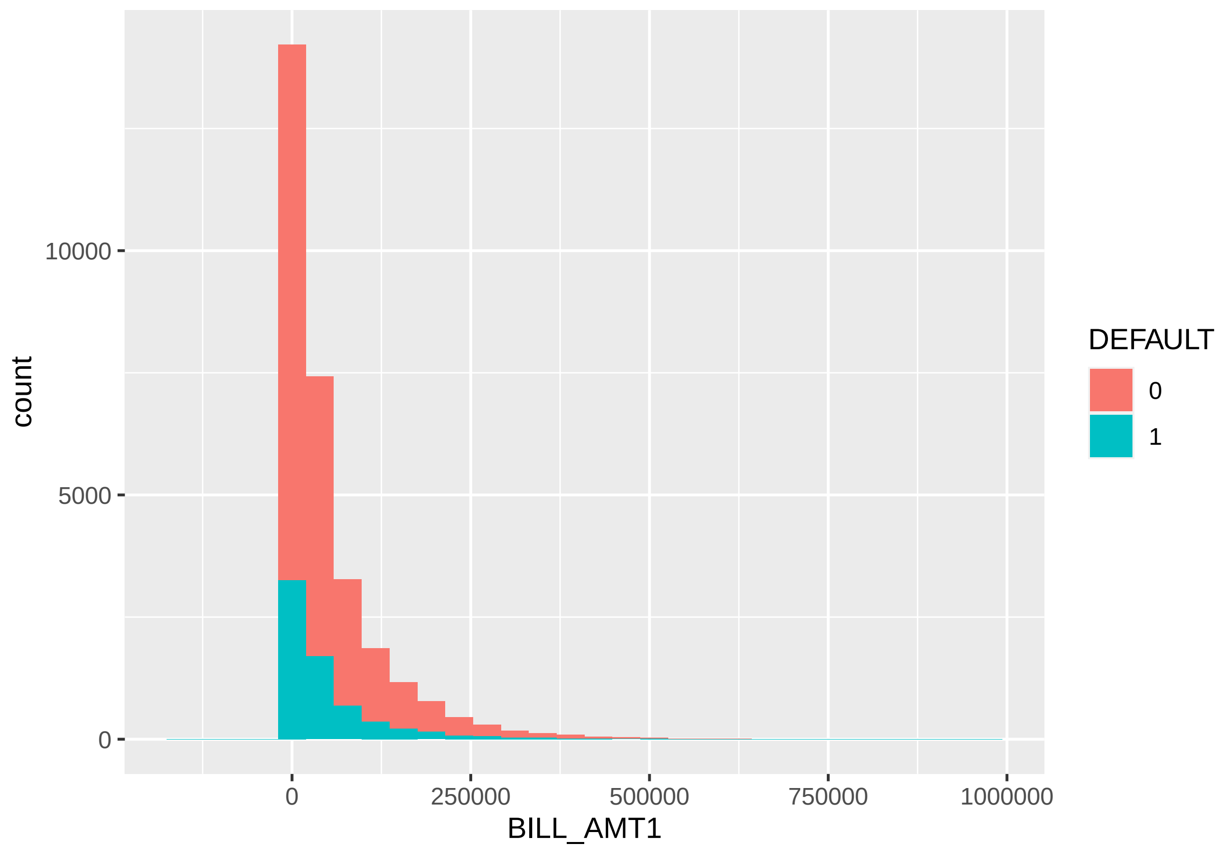
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -165580    3559   22382   51223   67091  964511
```

These are numerical data.

Here is BILL_AMT1's plot.

```
ggplot(data=original_default, aes(BILL_AMT1,fill= DEFAULT)) +geom_histogram()
```

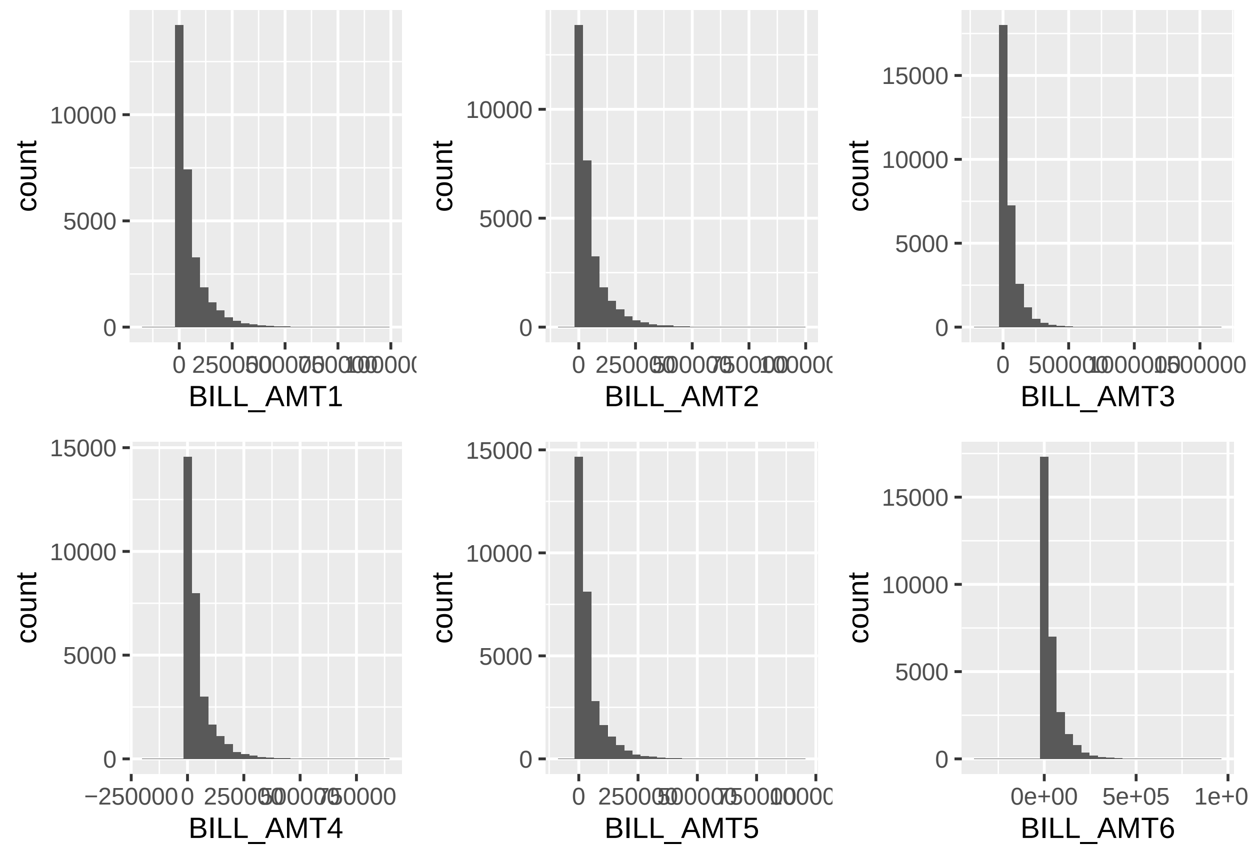
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



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

```
b1 <- ggplot(data=original_default, aes(BILL_AMT1)) +geom_histogram()
b2 <- ggplot(data=original_default, aes(BILL_AMT2)) +geom_histogram()
b3 <- ggplot(data=original_default, aes(BILL_AMT3)) +geom_histogram()
b4 <- ggplot(data=original_default, aes(BILL_AMT4)) +geom_histogram()
b5 <- ggplot(data=original_default, aes(BILL_AMT5)) +geom_histogram()
b6 <- ggplot(data=original_default, aes(BILL_AMT6)) +geom_histogram()
grid.arrange(b1,b2,b3,b4,b5,b6, nrow=2, ncol=3)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



9 "PAY_AMT"

Kaggle's data explanation says;

PAY_AMT1 is an amount of previous payment in September, 2005 (NT dollar). Likewise BILL_AMT, PAY_AMT goes back in time by a month from August to April, 2005 which is PAY_AMT6.

```
summary(original_default$PAY_AMT1)
```

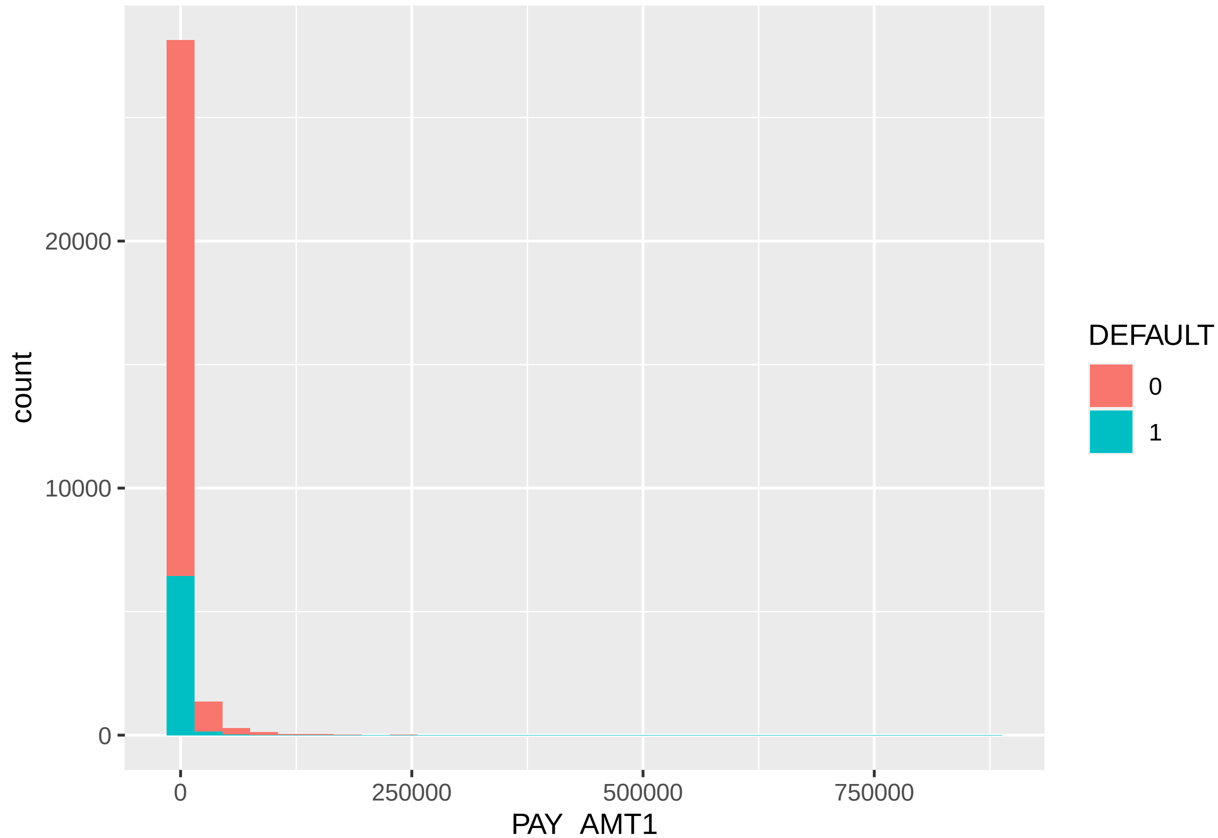
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0    1000    2100    5664    5006  873552
```

They are numerical data.

Here is PAY_AMT1's plot.

```
ggplot(data=original_default, aes(PAY_AMT1,fill= DEFAULT)) +geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

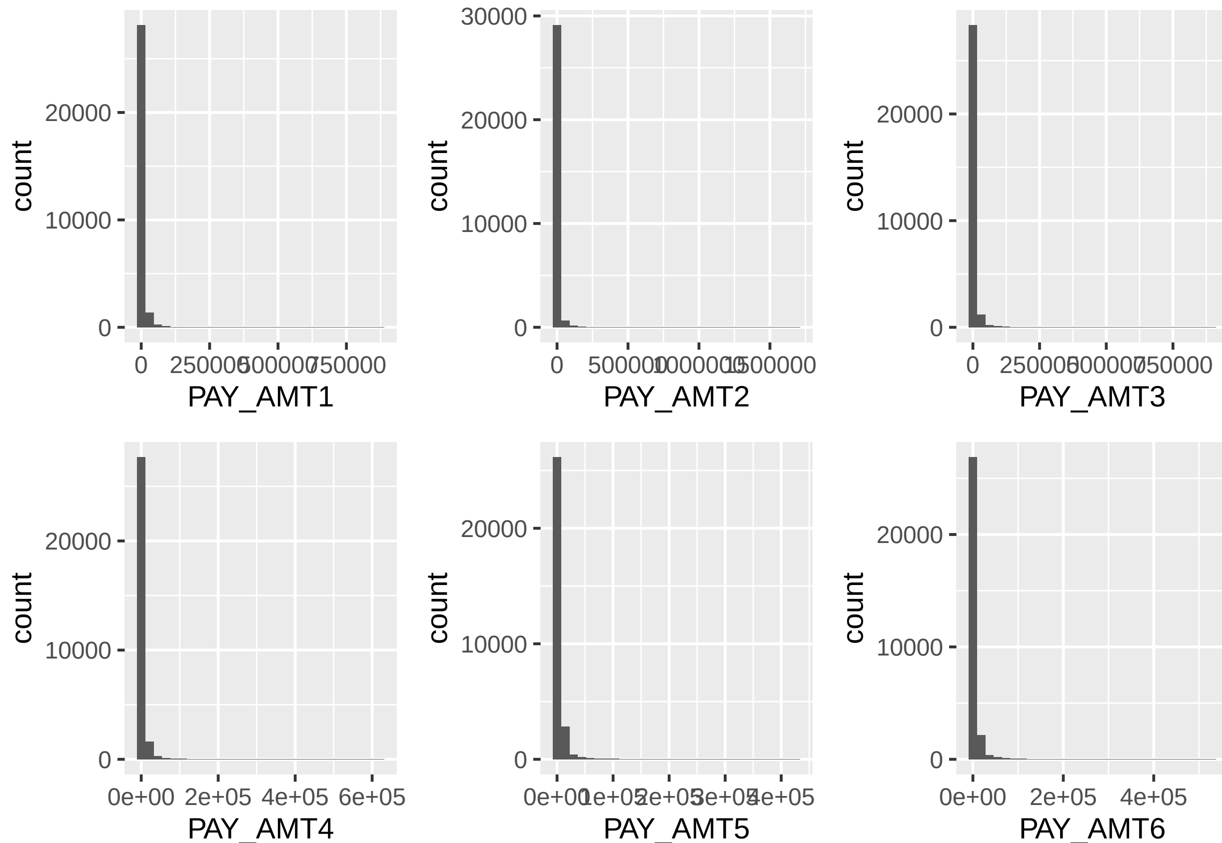


From PAY_AMT1 to PAY_AMT6, their structures are almost the same as are shown in following plots.

```
p1 <- ggplot(data=original_default, aes(PAY_AMT1)) +geom_histogram()
p2 <- ggplot(data=original_default, aes(PAY_AMT2)) +geom_histogram()
p3 <- ggplot(data=original_default, aes(PAY_AMT3)) +geom_histogram()
p4 <- ggplot(data=original_default, aes(PAY_AMT4)) +geom_histogram()
p5 <- ggplot(data=original_default, aes(PAY_AMT5)) +geom_histogram()
p6 <- ggplot(data=original_default, aes(PAY_AMT6)) +geom_histogram()

grid.arrange(p1,p2,p3,p4,p5,p6, nrow=2, ncol=3)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Data Preparation

Remove ID

```
original_default <- original_default %>% select(-ID)
```

Categorical data, change numeric to factor. SEX,EDUCATION,MARRIAGE, PAY_0~PAY_6 are categorical data

```
original_default <- original_default %>%
  mutate(SEX = as.factor(SEX),
         EDUCATION = as.factor(EDUCATION),
         MARRIAGE = as.factor(MARRIAGE),
         PAY_0 = as.factor(PAY_0),
         PAY_2 = as.factor(PAY_2),
         PAY_3 = as.factor(PAY_3),
         PAY_4 = as.factor(PAY_4),
         PAY_5 = as.factor(PAY_5),
         PAY_6 = as.factor(PAY_6) )
```

Scaling. We use "scale" function to standardize predictors. Categorical data columns. we assume these can be defined as factors.

```r
cat_col <- c("SEX", "EDUCATION", "MARRIAGE",
             "PAY_0", "PAY_2", "PAY_3", "PAY_4", "PAY_5", "PAY_6", "DEFAULT")

#all columns
all_col <- names(original_default)

#numerical data columns
num_col <- all_col[-which(all_col %in% cat_col)]

#scaling numerical data
original_default[num_col] <-original_default %>% select(-all_of(cat_col)) %>% scale()
```

Check the dataset.

```r
str(original_default)
```

```
## tibble [30,000 x 24] (S3: tbl_df/tbl/data.frame)
##  $ LIMIT_BAL: num [1:30000] -1.137 -0.366 -0.597 -0.905 -0.905 ...
##  $ SEX      : Factor w/ 2 levels "1","2": 2 2 2 2 1 1 1 2 2 1 ...
##  $ EDUCATION: Factor w/ 7 levels "0","1","2","3",..: 3 3 3 3 3 2 2 3 4 4 ...
##  $ MARRIAGE : Factor w/ 3 levels "1","2","3": 1 2 2 1 1 2 2 2 1 2 ...
##  $ AGE      : num [1:30000] -1.246 -1.029 -0.161 0.164 2.334 ...
##  $ PAY_0    : Factor w/ 11 levels "-2","-1","0",..: 5 2 3 3 2 3 3 3 3 1 ...
##  $ PAY_2    : Factor w/ 11 levels "-2","-1","0",..: 5 5 3 3 3 3 3 2 3 1 ...
##  $ PAY_3    : Factor w/ 11 levels "-2","-1","0",..: 2 3 3 3 2 3 3 2 5 1 ...
##  $ PAY_4    : Factor w/ 11 levels "-2","-1","0",..: 2 3 3 3 3 3 3 3 3 1 ...
##  $ PAY_5    : Factor w/ 10 levels "-2","-1","0",..: 1 3 3 3 3 3 3 3 3 2 ...
##  $ PAY_6    : Factor w/ 10 levels "-2","-1","0",..: 1 4 3 3 3 3 3 2 3 2 ...
##  $ BILL_AMT1: num [1:30000] -0.6425 -0.6592 -0.2986 -0.0575 -0.5786 ...
##  $ BILL_AMT2: num [1:30000] -0.6474 -0.6667 -0.4939 -0.0133 -0.6113 ...
##  $ BILL_AMT3: num [1:30000] -0.668 -0.6392 -0.4824 0.0328 -0.1612 ...
##  $ BILL_AMT4: num [1:30000] -0.672 -0.622 -0.45 -0.232 -0.347 ...
##  $ BILL_AMT5: num [1:30000] -0.663 -0.606 -0.417 -0.187 -0.348 ...
##  $ BILL_AMT6: num [1:30000] -0.653 -0.598 -0.392 -0.157 -0.331 ...
##  $ PAY_AMT1 : num [1:30000] -0.342 -0.342 -0.25 -0.221 -0.221 ...
##  $ PAY_AMT2 : num [1:30000] -0.227 -0.214 -0.192 -0.169 1.335 ...
##  $ PAY_AMT3 : num [1:30000] -0.297 -0.24 -0.24 -0.229 0.271 ...
##  $ PAY_AMT4 : num [1:30000] -0.308 -0.244 -0.244 -0.238 0.266 ...
##  $ PAY_AMT5 : num [1:30000] -0.314 -0.314 -0.249 -0.244 -0.269 ...
##  $ PAY_AMT6 : num [1:30000] -0.2934 -0.1809 -0.0121 -0.2371 -0.2552 ...
##  $ DEFAULT  : Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
```

```r
summary(original_default)
```

```
##    LIMIT_BAL        SEX       EDUCATION MARRIAGE       AGE
##  Min.   :-1.2138   1:11888   0:   14    1:13659   Min.   :-1.5715
##  1st Qu.:-0.9055   2:18112   1:10585    2:15964   1st Qu.:-0.8121
##  Median :-0.2118             2:14030    3:  377   Median :-0.1612
##  Mean   : 0.0000             3: 4917              Mean   : 0.0000
##  3rd Qu.: 0.5589             4:  123              3rd Qu.: 0.5982
##  Max.   : 6.4164             5:  280              Max.   : 4.7207
##                              6:   51
```

```
##      PAY_0           PAY_2           PAY_3           PAY_4
##  0       :14737  0       :15730  0       :15764  0       :16455
##  -1      : 5686  -1      : 6050  -1      : 5938  -1      : 5687
##  1       : 3688  2       : 3927  -2      : 4085  -2      : 4348
##  -2      : 2759  -2      : 3782  2       : 3819  2       : 3159
##  2       : 2667  3       :  326  3       :  240  3       :  180
##  3       :  322  4       :   99  4       :   76  4       :   69
##  (Other):  141  (Other):   86  (Other):   78  (Other):  102
##      PAY_5           PAY_6          BILL_AMT1         BILL_AMT2
##  0       :16947  0       :16286  Min.   :-2.9443   Min.   :-1.6713
##  -1      : 5539  -1      : 5740  1st Qu.:-0.6473   1st Qu.:-0.6490
##  -2      : 4546  -2      : 4895  Median :-0.3917   Median :-0.3931
##  2       : 2626  2       : 2766  Mean   : 0.0000   Mean   : 0.0000
##  3       :  178  3       :  184  3rd Qu.: 0.2155   3rd Qu.: 0.2083
##  4       :   84  4       :   49  Max.   :12.4028   Max.   :13.1334
##  (Other):   80  (Other):   80
##    BILL_AMT3         BILL_AMT4         BILL_AMT5         BILL_AMT6
##  Min.   :-2.9456   Min.   :-3.3150   Min.   :-2.0008   Min.   :-6.3551
##  1st Qu.:-0.6395   1st Qu.:-0.6363   1st Qu.:-0.6340   1st Qu.:-0.6316
##  Median :-0.3882   Median :-0.3763   Median :-0.3653   Median :-0.3661
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.: 0.1896   3rd Qu.: 0.1748   3rd Qu.: 0.1625   3rd Qu.: 0.1734
##  Max.   :23.3178   Max.   :13.1865   Max.   :14.5872   Max.   :15.4950
##
##     PAY_AMT1          PAY_AMT2           PAY_AMT3           PAY_AMT4
##  Min.   :-0.3419   Min.   :-0.25699   Min.   :-0.29680   Min.   :-0.30806
##  1st Qu.:-0.2816   1st Qu.:-0.22083   1st Qu.:-0.27465   1st Qu.:-0.28916
##  Median :-0.2152   Median :-0.16979   Median :-0.19456   Median :-0.21231
##  Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.00000   Mean   : 0.00000
##  3rd Qu.:-0.0397   3rd Qu.:-0.03998   3rd Qu.:-0.04093   3rd Qu.:-0.05188
##  Max.   :52.3983   Max.   :72.84177   Max.   :50.59444   Max.   :39.33152
##
##     PAY_AMT5           PAY_AMT6         DEFAULT
##  Min.   :-0.31413   Min.   :-0.29338   0:23364
##  1st Qu.:-0.29760   1st Qu.:-0.28675   1: 6636
##  Median :-0.21595   Median :-0.20900
##  Mean   : 0.00000   Mean   : 0.00000
##  3rd Qu.:-0.05026   3rd Qu.:-0.06837
##  Max.   :27.60317   Max.   :29.44461
##
```

Spliting into train_set,validation_set, test_set.

First we split data into test_set, and default.Test_set will be only used as evaluation. We use "createDataPartition" function in "caret" package. Set seed 2021.

```
set.seed(2021, sample.kind = "Rounding")
```

```
## Warning in set.seed(2021, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
index_1 <- createDataPartition(original_default$DEFAULT, p=0.2, list=F, times=1)
test_set <- original_default[index_1,]
default <- original_default[-index_1,]
```

As we tune hyperparameters, we split default into train_set and validation_set. Validation set will be used when tuning models.

```
set.seed(2021, sample.kind = "Rounding")
```

```
## Warning in set.seed(2021, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
index_2 <- createDataPartition(default$DEFAULT, p=0.2, list=F, times=1)
validation_set <-default[index_2,]
train_set <- default[-index_2,]
```

Check default ratio.

```
#train_set
prop.table(table(train_set$DEFAULT))
```

```
##
##         0         1
## 0.7788311 0.2211689
```

```
#validation_set
prop.table(table(validation_set$DEFAULT))
```

```
##
##         0         1
## 0.7787961 0.2212039
```

```
#test_set
prop.table(table(test_set$DEFAULT))
```

```
##
##         0         1
## 0.7787035 0.2212965
```

Almost similar ratio.

Model analysis

1 Baseline prediction

All predicted as non_default make factor vectors.

```
base_pred <-factor(numeric(length(test_set$DEFAULT)),levels=c("0","1"))
```

Confusion matrix.

```r
confusionMatrix(base_pred, test_set$DEFAULT)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 4673 1328
##          1    0    0
##
##                Accuracy : 0.7787
##                  95% CI : (0.768, 0.7892)
##     No Information Rate : 0.7787
##     P-Value [Acc > NIR] : 0.5074
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.7787
##          Neg Pred Value :    NaN
##              Prevalence : 0.7787
##          Detection Rate : 0.7787
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

We need to find models which exceed these values(except sensitivity). In this model, sensitivity is 1, but specificity is 0. This means the credit company falsely give credit to a person who fail to repay a debt. The loss for the company would be huge.

evaluation method

as this is a classification problem, we calculate accuracy using confusion matrix. However, as is shown in this baseline prediction, default rate is imbalanced. As well as accuracy, we will pay attention to specificity and balanced accuracy.

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.

```r
#a null model with no predictors
null_model <- glm(DEFAULT~1, data = train_set, family = binomial(link = "logit"))

#a full model using all of the potential predictors
full_model <- glm(DEFAULT~., data = train_set, family = binomial(link = "logit"))
```

Forward and backward stepwise algorithm.

```r
step_mdl   <- step(null_model,
                   scope = list(lower = null_model, upper = full_model),
                   direction = "both")
```

```
## Start:  AIC=20289.81
## DEFAULT ~ 1

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

##              Df Deviance   AIC
## + PAY_0      10    17383 17405
## + PAY_2      10    18439 18461
## + PAY_3      10    18834 18856
## + PAY_4      10    18980 19002
## + PAY_5       9    19077 19097
## + PAY_6       9    19232 19252
## + LIMIT_BAL   1    19768 19772
## + PAY_AMT2    1    20063 20067
## + PAY_AMT1    1    20085 20089
## + PAY_AMT3    1    20112 20116
## + PAY_AMT5    1    20164 20168
## + PAY_AMT4    1    20177 20181
## + EDUCATION   6    20169 20183
## + PAY_AMT6    1    20220 20224
## + SEX         1    20257 20261
## + MARRIAGE    2    20277 20283
## + BILL_AMT1   1    20279 20283
## + BILL_AMT3   1    20284 20288
## + BILL_AMT2   1    20284 20288
## + BILL_AMT4   1    20285 20289
## <none>             20288 20290
## + BILL_AMT5   1    20286 20290
## + BILL_AMT6   1    20286 20290
## + AGE         1    20288 20292
##
## Step:  AIC=17404.59
## DEFAULT ~ PAY_0

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

##              Df Deviance   AIC
## + PAY_4      10    17107 17149
## + PAY_5       9    17114 17154
## + PAY_3      10    17128 17170
## + PAY_6       9    17137 17177
## + LIMIT_BAL   1    17178 17202
## + PAY_2       9    17243 17283
## + PAY_AMT2    1    17294 17318
## + PAY_AMT3    1    17312 17336
## + PAY_AMT1    1    17321 17345
## + PAY_AMT5    1    17329 17353
## + EDUCATION   6    17320 17354
```

```
## + PAY_AMT4   1    17338 17362
## + PAY_AMT6   1    17352 17376
## + SEX        1    17364 17388
## + BILL_AMT5  1    17377 17401
## + BILL_AMT6  1    17377 17401
## + MARRIAGE   2    17375 17401
## + BILL_AMT4  1    17379 17403
## + BILL_AMT3  1    17379 17403
## + BILL_AMT1  1    17380 17404
## <none>            17383 17405
## + BILL_AMT2  1    17381 17405
## + AGE        1    17381 17405
## - PAY_0     10    20288 20290
##
## Step:  AIC=17148.59
## DEFAULT ~ PAY_0 + PAY_4
```

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

```
##             Df Deviance   AIC
## + LIMIT_BAL  1    16957 17001
## + PAY_6      9    16990 17050
## + PAY_AMT2   1    17019 17063
## + PAY_5      9    17027 17087
## + PAY_3     10    17027 17089
## + PAY_AMT1   1    17051 17095
## + PAY_AMT5   1    17062 17106
## + PAY_AMT3   1    17067 17111
## + PAY_2      9    17052 17112
## + EDUCATION  6    17058 17112
## + PAY_AMT4   1    17075 17119
## + PAY_AMT6   1    17082 17126
## + SEX        1    17091 17135
## + BILL_AMT6  1    17095 17139
## + BILL_AMT5  1    17096 17140
## + BILL_AMT4  1    17099 17143
## + MARRIAGE   2    17099 17145
## + BILL_AMT3  1    17101 17145
## + BILL_AMT1  1    17104 17148
## + BILL_AMT2  1    17105 17149
## <none>            17107 17149
## + AGE        1    17105 17149
## - PAY_4     10    17383 17405
## - PAY_0     10    18980 19002
##
## Step:  AIC=17001.21
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL
##
##             Df Deviance   AIC
## + PAY_6      9    16855 16917
## + PAY_5      9    16884 16946
## + PAY_3     10    16895 16959
## + PAY_AMT2   1    16916 16962
## + PAY_AMT1   1    16934 16980
```

```
## + PAY_2      9    16919 16981
## + EDUCATION  6    16925 16981
## + BILL_AMT2  1    16936 16982
## + BILL_AMT1  1    16937 16983
## + PAY_AMT5   1    16941 16987
## + SEX        1    16943 16989
## + MARRIAGE   2    16941 16989
## + BILL_AMT3  1    16943 16989
## + PAY_AMT3   1    16944 16990
## + BILL_AMT4  1    16946 16992
## + PAY_AMT4   1    16947 16993
## + AGE        1    16950 16996
## + BILL_AMT5  1    16950 16996
## + BILL_AMT6  1    16952 16998
## + PAY_AMT6   1    16952 16998
## <none>            16957 17001
## - LIMIT_BAL  1    17107 17149
## - PAY_4     10    17178 17202
## - PAY_0     10    18714 18738
##
## Step:  AIC=16916.88
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6
##
##             Df Deviance   AIC
## + PAY_AMT2   1    16818 16882
## + PAY_3     10    16803 16885
## + BILL_AMT2  1    16828 16892
## + BILL_AMT1  1    16830 16894
## + EDUCATION  6    16823 16897
## + PAY_AMT1   1    16834 16898
## + BILL_AMT3  1    16837 16901
## + PAY_2      9    16823 16903
## + BILL_AMT4  1    16840 16904
## + MARRIAGE   2    16839 16905
## + SEX        1    16842 16906
## + PAY_AMT3   1    16843 16907
## + PAY_AMT5   1    16844 16908
## + BILL_AMT5  1    16845 16909
## + PAY_AMT4   1    16847 16911
## + AGE        1    16847 16911
## + PAY_5      9    16831 16911
## + BILL_AMT6  1    16848 16912
## + PAY_AMT6   1    16851 16915
## <none>            16855 16917
## - PAY_6      9    16957 17001
## - PAY_4     10    16975 17017
## - LIMIT_BAL  1    16990 17050
## - PAY_0     10    18464 18506
##
## Step:  AIC=16881.94
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2

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

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

##             Df Deviance   AIC
## + BILL_AMT3  1    16780 16846
## + BILL_AMT2  1    16785 16851
## + BILL_AMT1  1    16787 16853
## + BILL_AMT4  1    16792 16858
## + PAY_3     10    16777 16861
## + EDUCATION  6    16788 16864
## + BILL_AMT5  1    16800 16866
## + BILL_AMT6  1    16805 16871
## + MARRIAGE   2    16803 16871
## + SEX        1    16805 16871
## + PAY_AMT1   1    16806 16872
## + PAY_2      9    16790 16872
## + AGE        1    16810 16876
## + PAY_5      9    16794 16876
## + PAY_AMT5   1    16811 16877
## + PAY_AMT3   1    16812 16878
## + PAY_AMT4   1    16813 16879
## <none>            16818 16882
## + PAY_AMT6   1    16816 16882
## - PAY_AMT2   1    16855 16917
## - PAY_6      9    16916 16962
## - LIMIT_BAL  1    16912 16974
## - PAY_4     10    16943 16987
## - PAY_0     10    18402 18446
##
## Step:  AIC=16845.65
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3

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

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

##             Df Deviance   AIC
## + PAY_AMT1   1    16758 16826
## + EDUCATION  6    16749 16827
## + PAY_3     10    16744 16830
## + SEX        1    16767 16835
## + MARRIAGE   2    16765 16835
## + PAY_AMT5   1    16768 16836
## + PAY_AMT3   1    16770 16838
## + PAY_AMT4   1    16773 16841
## + AGE        1    16773 16841
## + PAY_5      9    16757 16841
## + PAY_2      9    16758 16842
## + BILL_AMT6  1    16774 16842
## + BILL_AMT5  1    16774 16842
## + PAY_AMT6   1    16776 16844
## <none>            16780 16846
## + BILL_AMT4  1    16778 16846
## + BILL_AMT2  1    16780 16848
```

```
## + BILL_AMT1  1    16780 16848
## - BILL_AMT3  1    16818 16882
## - PAY_AMT2   1    16837 16901
## - PAY_6      9    16882 16930
## - PAY_4     10    16895 16941
## - LIMIT_BAL  1    16909 16973
## - PAY_0     10    18342 18388
##
## Step:  AIC=16826.41
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1

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

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

##             Df Deviance   AIC
## + PAY_3     10    16720 16808
## + EDUCATION  6    16729 16809
## + SEX        1    16746 16816
## + MARRIAGE   2    16745 16817
## + PAY_AMT5   1    16749 16819
## + AGE        1    16752 16822
## + PAY_AMT3   1    16752 16822
## + PAY_5      9    16736 16822
## + BILL_AMT6  1    16753 16823
## + BILL_AMT5  1    16753 16823
## + PAY_AMT4   1    16753 16823
## + PAY_AMT6   1    16756 16826
## <none>            16758 16826
## + BILL_AMT4  1    16757 16827
## + PAY_2      9    16741 16827
## + BILL_AMT1  1    16758 16828
## + BILL_AMT2  1    16758 16828
## - PAY_AMT1   1    16780 16846
## - BILL_AMT3  1    16806 16872
## - PAY_AMT2   1    16807 16873
## - PAY_6      9    16860 16910
## - PAY_4     10    16873 16921
## - LIMIT_BAL  1    16879 16945
## - PAY_0     10    18295 18343
##
## Step:  AIC=16807.47
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1 + PAY_3
##
##             Df Deviance   AIC
## + EDUCATION  6    16691 16791
## + SEX        1    16707 16797
## + MARRIAGE   2    16706 16798
## + PAY_AMT5   1    16710 16800
## + PAY_AMT3   1    16713 16803
## + AGE        1    16713 16803
## + PAY_5      9    16697 16803
```

```
## + BILL_AMT6  1    16714 16804
## + BILL_AMT5  1    16714 16804
## + PAY_AMT4   1    16714 16804
## + PAY_AMT6   1    16717 16807
## <none>           16720 16808
## + BILL_AMT4  1    16718 16808
## + BILL_AMT1  1    16719 16809
## + BILL_AMT2  1    16719 16809
## + PAY_2      9    16712 16818
## - PAY_3     10    16758 16826
## - PAY_AMT1   1    16744 16830
## - PAY_AMT2   1    16756 16842
## - PAY_4     10    16778 16846
## - BILL_AMT3  1    16761 16847
## - PAY_6      9    16813 16883
## - LIMIT_BAL  1    16829 16915
## - PAY_0     10    18044 18112
##
## Step:  AIC=16790.71
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1 + PAY_3 + EDUCATION
##
##             Df Deviance   AIC
## + SEX        1    16678 16780
## + MARRIAGE   2    16678 16782
## + PAY_AMT5   1    16681 16783
## + AGE        1    16684 16786
## + PAY_5      9    16668 16786
## + PAY_AMT3   1    16684 16786
## + BILL_AMT6  1    16684 16786
## + BILL_AMT5  1    16685 16787
## + PAY_AMT4   1    16685 16787
## + PAY_AMT6   1    16688 16790
## <none>           16691 16791
## + BILL_AMT4  1    16689 16791
## + BILL_AMT1  1    16690 16792
## + BILL_AMT2  1    16690 16792
## + PAY_2      9    16683 16801
## - EDUCATION  6    16720 16808
## - PAY_3     10    16729 16809
## - PAY_AMT1   1    16714 16812
## - PAY_AMT2   1    16726 16824
## - PAY_4     10    16748 16828
## - BILL_AMT3  1    16733 16831
## - PAY_6      9    16784 16866
## - LIMIT_BAL  1    16792 16890
## - PAY_0     10    18014 18094
##
## Step:  AIC=16780.35
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1 + PAY_3 + EDUCATION + SEX
##
##             Df Deviance   AIC
## + MARRIAGE   2    16665 16771
```

```
## + PAY_AMT5   1    16668 16772
## + PAY_AMT3   1    16672 16776
## + PAY_5      9    16656 16776
## + BILL_AMT6  1    16672 16776
## + PAY_AMT4   1    16673 16777
## + BILL_AMT5  1    16673 16777
## + AGE        1    16673 16777
## + PAY_AMT6   1    16675 16779
## <none>            16678 16780
## + BILL_AMT4  1    16677 16781
## + BILL_AMT1  1    16678 16782
## + BILL_AMT2  1    16678 16782
## + PAY_2      9    16670 16790
## - SEX        1    16691 16791
## - EDUCATION  6    16707 16797
## - PAY_3     10    16716 16798
## - PAY_AMT1   1    16702 16802
## - PAY_AMT2   1    16714 16814
## - PAY_4     10    16736 16818
## - BILL_AMT3  1    16721 16821
## - PAY_6      9    16769 16853
## - LIMIT_BAL  1    16778 16878
## - PAY_0     10    18003 18085
##
## Step:  AIC=16770.47
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1 + PAY_3 + EDUCATION + SEX + MARRIAGE
##
##             Df Deviance   AIC
## + PAY_AMT5   1    16655 16763
## + PAY_5      9    16642 16766
## + PAY_AMT3   1    16658 16766
## + BILL_AMT6  1    16659 16767
## + PAY_AMT4   1    16659 16767
## + BILL_AMT5  1    16659 16767
## + PAY_AMT6   1    16662 16770
## <none>            16665 16771
## + BILL_AMT4  1    16663 16771
## + BILL_AMT1  1    16664 16772
## + AGE        1    16664 16772
## + BILL_AMT2  1    16664 16772
## + PAY_2      9    16656 16780
## - MARRIAGE   2    16678 16780
## - SEX        1    16678 16782
## - EDUCATION  6    16693 16787
## - PAY_3     10    16702 16788
## - PAY_AMT1   1    16688 16792
## - PAY_AMT2   1    16700 16804
## - PAY_4     10    16722 16808
## - BILL_AMT3  1    16706 16810
## - PAY_6      9    16755 16843
## - LIMIT_BAL  1    16772 16876
## - PAY_0     10    17984 18070
##
```

```
## Step:  AIC=16762.82
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1 + PAY_3 + EDUCATION + SEX + MARRIAGE + PAY_AMT5
##
##              Df Deviance   AIC
## + PAY_5      9    16632 16758
## + BILL_AMT5  1    16650 16760
## + PAY_AMT3   1    16650 16760
## + PAY_AMT4   1    16651 16761
## + PAY_AMT6   1    16653 16763
## + BILL_AMT6  1    16653 16763
## <none>            16655 16763
## + BILL_AMT4  1    16654 16764
## + AGE        1    16654 16764
## + BILL_AMT1  1    16654 16764
## + BILL_AMT2  1    16654 16764
## - PAY_AMT5   1    16665 16771
## + PAY_2      9    16646 16772
## - MARRIAGE   2    16668 16772
## - SEX        1    16668 16774
## - EDUCATION  6    16683 16779
## - PAY_3     10    16692 16780
## - PAY_AMT1   1    16676 16782
## - PAY_AMT2   1    16686 16792
## - PAY_4     10    16713 16801
## - BILL_AMT3  1    16700 16806
## - PAY_6      9    16743 16833
## - LIMIT_BAL  1    16755 16861
## - PAY_0     10    17973 18061
##
## Step:  AIC=16757.69
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1 + PAY_3 + EDUCATION + SEX + MARRIAGE + PAY_AMT5 +
##     PAY_5

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

##              Df Deviance   AIC
## + PAY_AMT3   1    16626 16754
## + BILL_AMT5  1    16626 16754
## + PAY_AMT4   1    16629 16757
## + BILL_AMT6  1    16629 16757
## + PAY_AMT6   1    16629 16757
## <none>            16632 16758
## + BILL_AMT4  1    16630 16758
## + AGE        1    16631 16759
## + BILL_AMT1  1    16631 16759
## + BILL_AMT2  1    16631 16759
## - PAY_5      9    16655 16763
## - PAY_AMT5   1    16642 16766
## + PAY_2      9    16623 16767
## - MARRIAGE   2    16646 16768
## - SEX        1    16645 16769
## - EDUCATION  6    16660 16774
```

```
## - PAY_4     10    16669 16775
## - PAY_3     10    16670 16776
## - PAY_AMT1   1    16652 16776
## - PAY_AMT2   1    16662 16786
## - PAY_6      9    16678 16786
## - BILL_AMT3  1    16676 16800
## - LIMIT_BAL  1    16730 16854
## - PAY_0     10    17924 18030
##
## Step:  AIC=16753.49
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1 + PAY_3 + EDUCATION + SEX + MARRIAGE + PAY_AMT5 +
##     PAY_5 + PAY_AMT3

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

##             Df Deviance   AIC
## + BILL_AMT5  1    16622 16752
## + PAY_AMT4   1    16623 16753
## <none>            16626 16754
## + PAY_AMT6   1    16624 16754
## + BILL_AMT6  1    16624 16754
## + BILL_AMT2  1    16625 16755
## + AGE        1    16625 16755
## + BILL_AMT1  1    16625 16755
## + BILL_AMT4  1    16625 16755
## - PAY_AMT3   1    16632 16758
## - PAY_5      9    16650 16760
## - PAY_AMT5   1    16634 16760
## - MARRIAGE   2    16639 16763
## + PAY_2      9    16617 16763
## - SEX        1    16639 16765
## - PAY_4     10    16660 16768
## - PAY_AMT1   1    16643 16769
## - EDUCATION  6    16653 16769
## - PAY_3     10    16664 16772
## - PAY_AMT2   1    16653 16779
## - PAY_6      9    16671 16781
## - BILL_AMT3  1    16671 16797
## - LIMIT_BAL  1    16717 16843
## - PAY_0     10    17915 18023
##
## Step:  AIC=16752.38
## DEFAULT ~ PAY_0 + PAY_4 + LIMIT_BAL + PAY_6 + PAY_AMT2 + BILL_AMT3 +
##     PAY_AMT1 + PAY_3 + EDUCATION + SEX + MARRIAGE + PAY_AMT5 +
##     PAY_5 + PAY_AMT3 + BILL_AMT5

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

##             Df Deviance   AIC
## <none>            16622 16752
## + PAY_AMT6   1    16621 16753
## + BILL_AMT4  1    16622 16754
```

```
## + BILL_AMT2  1    16622 16754
## - BILL_AMT5  1    16626 16754
## + PAY_AMT4   1    16622 16754
## + AGE        1    16622 16754
## + BILL_AMT1  1    16622 16754
## + BILL_AMT6  1    16622 16754
## - PAY_AMT3   1    16626 16754
## - PAY_5      9    16647 16759
## - PAY_AMT5   1    16631 16759
## - MARRIAGE   2    16636 16762
## + PAY_2      9    16614 16762
## - SEX        1    16636 16764
## - PAY_4     10    16657 16767
## - EDUCATION  6    16650 16768
## - PAY_AMT1   1    16641 16769
## - PAY_3     10    16661 16771
## - BILL_AMT3  1    16646 16774
## - PAY_6      9    16667 16779
## - PAY_AMT2   1    16653 16781
## - LIMIT_BAL  1    16710 16838
## - PAY_0     10    17914 18024
```

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

```
step_prob <- predict(step_mdl, validation_set,type="response")
step_pred <- ifelse(step_prob >0.5,1,0)
```

To show accuracy we use confusionMatrix function in caret library.

```
confusionMatrix(as.factor(step_pred), validation_set$DEFAULT)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3568  715
##          1  171  347
##
##                Accuracy : 0.8155
##                  95% CI : (0.8042, 0.8263)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 2.329e-10
##
##                   Kappa : 0.3441
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9543
##             Specificity : 0.3267
##          Pos Pred Value : 0.8331
##          Neg Pred Value : 0.6699
##              Prevalence : 0.7788
##          Detection Rate : 0.7432
```

```
##    Detection Prevalence : 0.8921
##       Balanced Accuracy : 0.6405
##
##        'Positive' Class : 0
##
```

Make a table.

```
results <- tibble(method = "logistic regression",
                  Accuracy =confusionMatrix(as.factor(step_pred), validation_set$DEFAULT)$overall[1],
                  Sensitivity =confusionMatrix(as.factor(step_pred), validation_set$DEFAULT)$byClass[1]
                  Specificity =confusionMatrix(as.factor(step_pred), validation_set$DEFAULT)$byClass[2]
                  Balanced_Accuracy =confusionMatrix(as.factor(step_pred), validation_set$DEFAULT)$byCl

results %>% knitr::kable()
```

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9542658	0.326742	0.6405039

3 Decision tree default model

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

```
set.seed(2021, sample.kind = "Rounding")
```

```
## Warning in set.seed(2021, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
rpart_mdl <-rpart(DEFAULT ~ .,data = train_set)
```

Predict.

```
rpart_pred <- predict(rpart_mdl, validation_set, type="class")
```

Confusion Matrix.

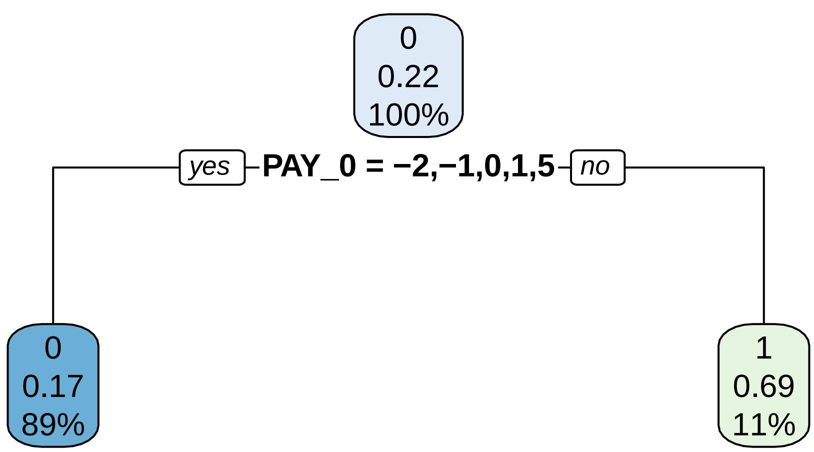
```
confusionMatrix(rpart_pred, validation_set$DEFAULT)
```

```
## 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.

```
rpart.plot(rpart_mdl)
```



Find used features.

```
rpart_mdl$variable.importance
```

```
##      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

```
results <- bind_rows(
  results,
  tibble(method="CART default",
         Accuracy = confusionMatrix(rpart_pred,
                                    validation_set$DEFAULT)$overall[1],
         Sensitivity = confusionMatrix(rpart_pred,
                                       validation_set$DEFAULT)$byClass[1],
         Specificity = confusionMatrix(rpart_pred,
                                       validation_set$DEFAULT)$byClass[2],
         Balanced_Accuracy = confusionMatrix(rpart_pred,
                                             validation_set$DEFAULT)$byClass[11]))

results %>% knitr::kable()
```

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9542658	0.326742	0.6405039
CART default	0.8171214	0.9620219	0.306968	0.6344950

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

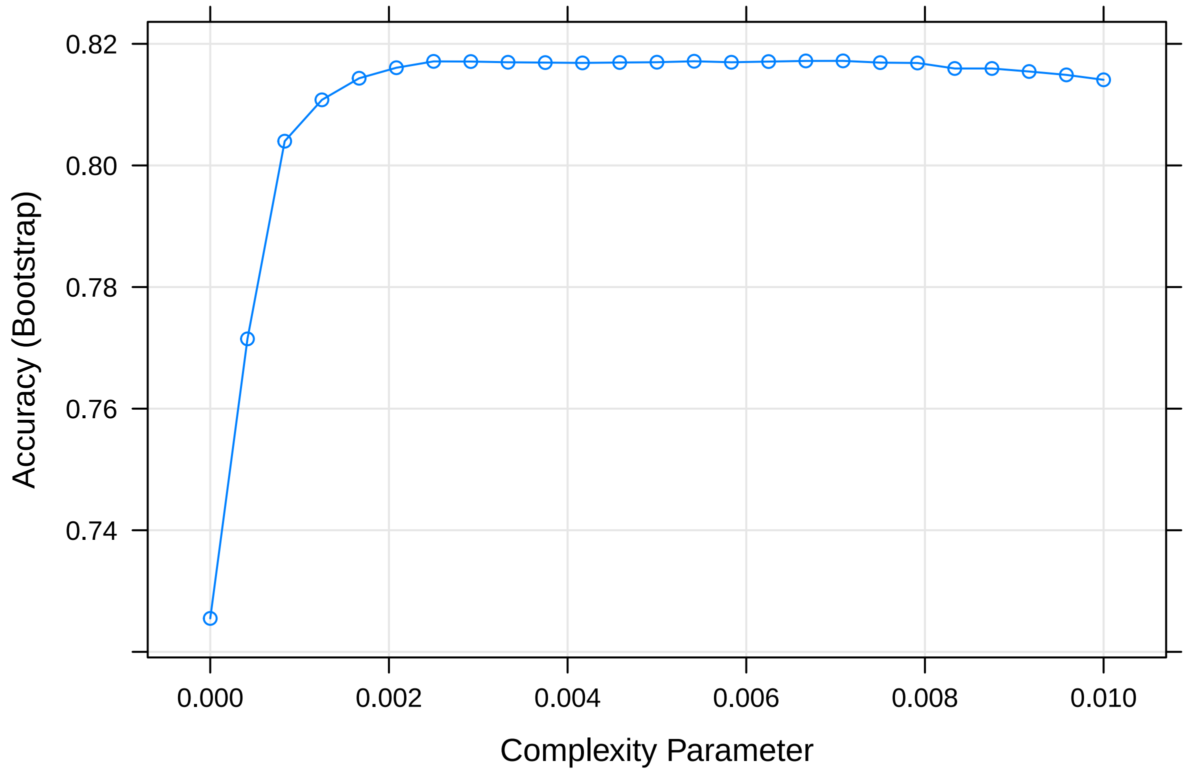
```
set.seed(2021, sample.kind = "Rounding")
```

```
## Warning in set.seed(2021, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
rpart_tuned_mdl <- train(DEFAULT ~ .,
                      method = "rpart",
                      tuneGrid = data.frame(cp = seq(0, 0.01, len = 25)),
                      control = rpart.control(minsplit = 0),
                      data = train_set)
```

Plot cp.

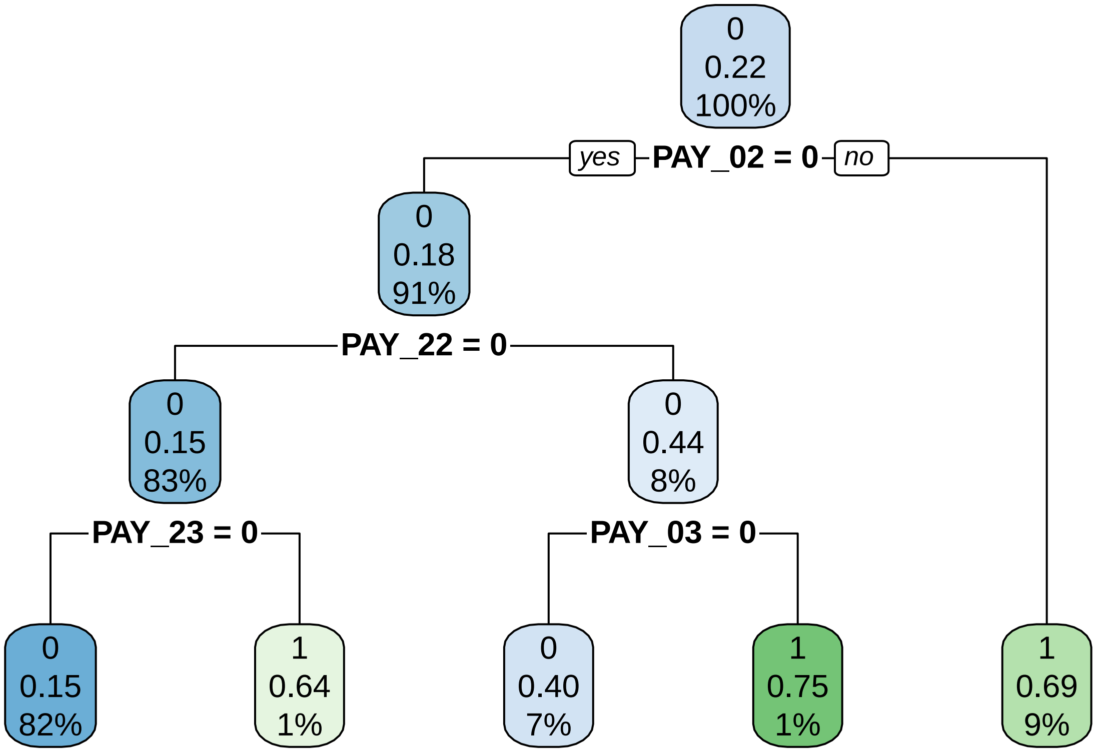
```
plot(rpart_tuned_mdl)
```




```
opt_cp <-rpart_tuned_mdl$bestTune
```

Draw decision tree. using rpart.plot.

```
rpart.plot(rpart_tuned_mdl$finalModel)
```



Note: numeric values are scaled

Prediction.

```
rpart_tuned_pred <- predict(rpart_tuned_mdl, validation_set)
```

Confusion matrix

```
confusionMatrix(rpart_tuned_pred, validation_set$DEFAULT)
```

```
## 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.

```
results <- bind_rows(
  results,
  tibble(method="CART tuned cp",
         Accuracy = confusionMatrix(rpart_tuned_pred, validation_set$DEFAULT)$overall[1],
         Sensitivity = confusionMatrix(rpart_tuned_pred, validation_set$DEFAULT)$byClass[1],
         Specificity = confusionMatrix(rpart_tuned_pred, validation_set$DEFAULT)$byClass[2],
         Balanced_Accuracy = confusionMatrix(rpart_tuned_pred, validation_set$DEFAULT)$byClass[11]) )

results %>% knitr::kable()
```

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9542658	0.3267420	0.6405039
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".

```
set.seed(2021, sample.kind = "Rounding")
```

```
## Warning in set.seed(2021, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
rf_mdl <- ranger(
  formula = DEFAULT ~ .,
  data = train_set,
  probability = F)
```

Model details.

```
rf_mdl
```

```
## 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:             18.18 %
```

Prediction.

```
rf_pred <- predict(rf_mdl, validation_set)$predictions
```

Confusion matrix

```
confusionMatrix(rf_pred, validation_set$DEFAULT)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3556  699
##          1  183  363
##
##                Accuracy : 0.8163
##                  95% CI : (0.805, 0.8272)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 9.111e-11
##
##                   Kappa : 0.3545
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9511
##             Specificity : 0.3418
##          Pos Pred Value : 0.8357
##          Neg Pred Value : 0.6648
##              Prevalence : 0.7788
##          Detection Rate : 0.7407
##    Detection Prevalence : 0.8863
##       Balanced Accuracy : 0.6464
##
##        'Positive' Class : 0
##
```

Make a table.

```r
results <- bind_rows(
  results,
  tibble(method="random forest default",
         Accuracy = confusionMatrix(rf_pred, validation_set$DEFAULT)$overall[1],
         Sensitivity = confusionMatrix(rf_pred, validation_set$DEFAULT)$byClass[1],
         Specificity = confusionMatrix(rf_pred, validation_set$DEFAULT)$byClass[2],
         Balanced_Accuracy = confusionMatrix(rf_pred, validation_set$DEFAULT)$byClass[11]) )

results %>% knitr::kable()
```

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9542658	0.3267420	0.6405039
CART default	0.8171214	0.9620219	0.3069680	0.6344950
CART tuned cp	0.8162883	0.9593474	0.3126177	0.6359826
random forest default	0.8162883	0.9510564	0.3418079	0.6464322

6 Random forest cross validation

Grid search

```r
modelLookup("ranger")
```

```
##    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.

```r
set.seed(2021, sample.kind = "Rounding")
```

```
## Warning in set.seed(2021, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

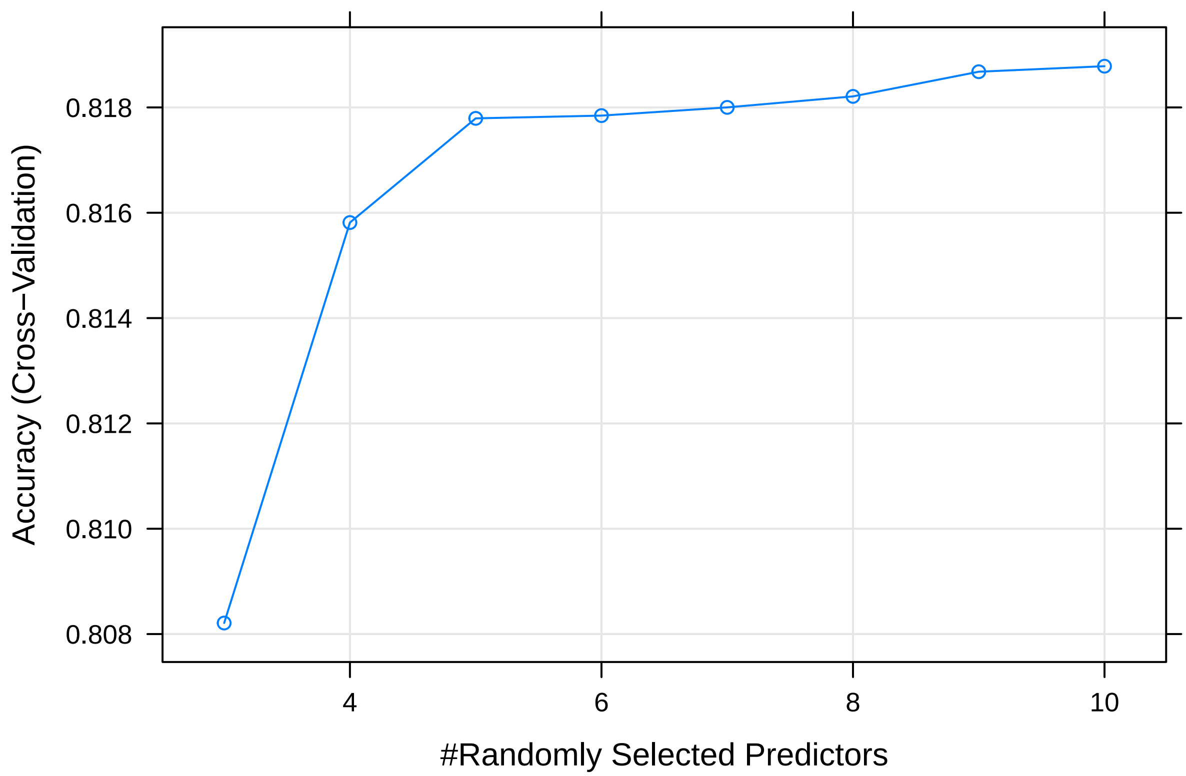
```r
rf_cv_mdl <- train( DEFAULT~ .,
                    data = train_set,
                    method = 'ranger',
                    metric = 'Accuracy',
                    num.trees = 1000,
                    tuneGrid = expand.grid(
                      mtry = 3:10, splitrule = 'gini', min.node.size = 1),
                    trControl = trainControl(method = 'cv', number = 5))
```

```
## Growing trees.. Progress: 100%. Estimated remaining time: 0 seconds.
## Growing trees.. Progress: 93%. Estimated remaining time: 2 seconds.
## Growing trees.. Progress: 82%. Estimated remaining time: 6 seconds.
## Growing trees.. Progress: 93%. Estimated remaining time: 2 seconds.
## Growing trees.. Progress: 82%. Estimated remaining time: 6 seconds.
## Growing trees.. Progress: 92%. Estimated remaining time: 2 seconds.
```

```
## Growing trees.. Progress: 79%. Estimated remaining time: 8 seconds.
## Growing trees.. Progress: 92%. Estimated remaining time: 2 seconds.
## Growing trees.. Progress: 81%. Estimated remaining time: 7 seconds.
## Growing trees.. Progress: 90%. Estimated remaining time: 3 seconds.
## Growing trees.. Progress: 82%. Estimated remaining time: 6 seconds.
## Growing trees.. Progress: 61%. Estimated remaining time: 19 seconds.
```

Plot.

```
plot(rf_cv_mdl)
```



Prediction.

```
rf_cv_pred <- predict(rf_cv_mdl, validation_set)
```

Confusion Matrix

```
confusionMatrix(rf_cv_pred, validation_set$DEFAULT)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3574  706
##          1  165  356
```

```
##
##                Accuracy : 0.8186
##                  95% CI : (0.8074, 0.8294)
##     No Information Rate : 0.7788
##     P-Value [Acc > NIR] : 6.162e-12
##
##                   Kappa : 0.356
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9559
##             Specificity : 0.3352
##          Pos Pred Value : 0.8350
##          Neg Pred Value : 0.6833
##              Prevalence : 0.7788
##          Detection Rate : 0.7444
##    Detection Prevalence : 0.8915
##       Balanced Accuracy : 0.6455
##
##        'Positive' Class : 0
##
```

Make a table.

```
results <- bind_rows( results,
                      tibble(
                        method="random forest tuned ",
Accuracy = confusionMatrix(rf_cv_pred, validation_set$DEFAULT)$overall[1],
Sensitivity = confusionMatrix(rf_cv_pred, validation_set$DEFAULT)$byClass[1],
Specificity = confusionMatrix(rf_cv_pred, validation_set$DEFAULT)$byClass[2],
Balanced_Accuracy= confusionMatrix(rf_cv_pred, validation_set$DEFAULT)$byClass[11]) )

results %>% knitr::kable()
```

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
logistic regresion	0.8154551	0.9542658	0.3267420	0.6405039
CART default	0.8171214	0.9620219	0.3069680	0.6344950
CART tuned cp	0.8162883	0.9593474	0.3126177	0.6359826
random forest default	0.8162883	0.9510564	0.3418079	0.6464322
random forest tuned	0.8185795	0.9558706	0.3352166	0.6455436

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.

```
final_pred_rpart <- predict(rpart_mdl, test_set,type="class")
confusionMatrix(final_pred_rpart, test_set$DEFAULT)
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction    0    1
##          0 4496  887
##          1  177  441
##
##                Accuracy : 0.8227
##                  95% CI : (0.8128, 0.8323)
##     No Information Rate : 0.7787
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3638
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9621
##             Specificity : 0.3321
##          Pos Pred Value : 0.8352
##          Neg Pred Value : 0.7136
##              Prevalence : 0.7787
##          Detection Rate : 0.7492
##    Detection Prevalence : 0.8970
##       Balanced Accuracy : 0.6471
##
##        'Positive' Class : 0
##
```

```
final_pred_rf <-predict(rf_mdl, test_set)$predictions
confusionMatrix(final_pred_rf, test_set$DEFAULT)$byClass
```

```
##          Sensitivity          Specificity       Pos Pred Value
##            0.9494971            0.3524096            0.8376439
##       Neg Pred Value            Precision               Recall
##            0.6647727            0.8376439            0.9494971
##                   F1           Prevalence       Detection Rate
##            0.8900702            0.7787035            0.7393768
## Detection Prevalence    Balanced Accuracy
##            0.8826862            0.6509534
```

Make a table.

```
final_results <- tibble( method ="CART default",
                         Accuracy =confusionMatrix(final_pred_rpart, test_set$DEFAULT)$overall[1],
                         Sensitivity =confusionMatrix(final_pred_rpart, test_set$DEFAULT)$byClass[1],
                         Specificity =confusionMatrix(final_pred_rpart, test_set$DEFAULT)$byClass[2],
                         Balanced_Accuracy = confusionMatrix(final_pred_rpart, test_set$DEFAULT)$byClas

final_results <-  bind_rows( final_results,
                             tibble( method ="Random forest default",
                             Accuracy =confusionMatrix(final_pred_rf, test_set$DEFAULT)$overall[1],
                             Sensitivity =confusionMatrix(final_pred_rf, test_set$DEFAULT)$byClass[1],
                             Specificity =confusionMatrix(final_pred_rf, test_set$DEFAULT)$byClass[2],
                             Balanced_Accuracy = confusionMatrix(final_pred_rf, test_set$DEFAULT)$byCla
```

```r
final_results %>% knitr::kable()
```

method	Accuracy	Sensitivity	Specificity	Balanced_Accuracy
CART default	0.8226962	0.9621228	0.3320783	0.6471006
Random forest default	0.8173638	0.9494971	0.3524096	0.6509534

Conclusion

###