Predicting Default Payments of Credit Card Clients

Sarbajeet Biswal

7 June 2018

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.4.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.3
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 3.4.2
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(arules)
```

```
## Warning: package 'arules' was built under R version 3.4.4
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'arules'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
## The following objects are masked from 'package:base':
##
##     abbreviate, write
```

```
df = read.csv("C:/Users/Administrator/Desktop/PYTHON/default_of_credit_card_clients.csv")

df1 = df
names(df1)
```

```
##  [1] "ID"                         "LIMIT_BAL"
##  [3] "SEX"                        "EDUCATION"
##  [5] "MARRIAGE"                   "AGE"
##  [7] "PAY_0"                      "PAY_2"
##  [9] "PAY_3"                      "PAY_4"
## [11] "PAY_5"                      "PAY_6"
## [13] "BILL_AMT1"                  "BILL_AMT2"
## [15] "BILL_AMT3"                  "BILL_AMT4"
## [17] "BILL_AMT5"                  "BILL_AMT6"
## [19] "PAY_AMT1"                   "PAY_AMT2"
## [21] "PAY_AMT3"                   "PAY_AMT4"
## [23] "PAY_AMT5"                   "PAY_AMT6"
## [25] "default.payment.next.month"
```

Data cleaning

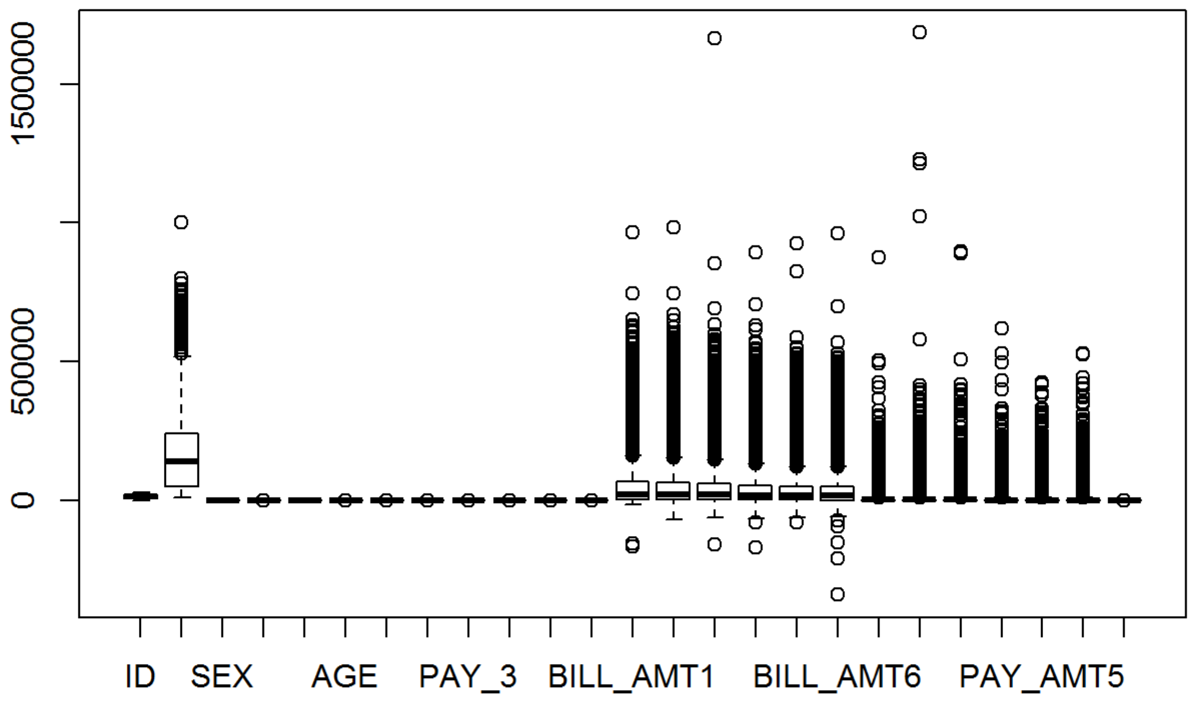
Checking null values

```
colSums(is.na(df1))
```

```
##                         ID                  LIMIT_BAL
##                          0                          0
##                        SEX                  EDUCATION
##                          0                          0
##                   MARRIAGE                        AGE
##                          0                          0
##                      PAY_0                      PAY_2
##                          0                          0
##                      PAY_3                      PAY_4
##                          0                          0
##                      PAY_5                      PAY_6
##                          0                          0
##                  BILL_AMT1                  BILL_AMT2
##                          0                          0
##                  BILL_AMT3                  BILL_AMT4
##                          0                          0
##                  BILL_AMT5                  BILL_AMT6
##                          0                          0
##                   PAY_AMT1                   PAY_AMT2
##                          0                          0
##                   PAY_AMT3                   PAY_AMT4
##                          0                          0
##                   PAY_AMT5                   PAY_AMT6
##                          0                          0
## default.payment.next.month
##                          0
```

Checking outliers

```
boxplot(df1)
```




```
summary(df1)
```

```
##        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            
```

Structure of data

```
str(df1)
```

```
## 'data.frame':    30000 obs. of  25 variables:
##  $ ID                        : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ LIMIT_BAL                 : int  20000 120000 90000 50000 50000 50000 500000 100000 140
000 20000 ...
##  $ SEX                       : int  2 2 2 2 1 1 1 2 2 1 ...
##  $ EDUCATION                 : int  2 2 2 2 2 1 1 2 3 3 ...
##  $ MARRIAGE                  : int  1 2 2 1 1 2 2 2 1 2 ...
##  $ AGE                       : int  24 26 34 37 57 37 29 23 28 35 ...
##  $ PAY_0                     : int  2 -1 0 0 -1 0 0 0 0 -2 ...
##  $ PAY_2                     : int  2 2 0 0 0 0 0 -1 0 -2 ...
##  $ PAY_3                     : int  -1 0 0 0 -1 0 0 -1 2 -2 ...
##  $ PAY_4                     : int  -1 0 0 0 0 0 0 0 0 -2 ...
##  $ PAY_5                     : int  -2 0 0 0 0 0 0 0 0 -1 ...
##  $ PAY_6                     : int  -2 2 0 0 0 0 0 -1 0 -1 ...
##  $ BILL_AMT1                 : int  3913 2682 29239 46990 8617 64400 367965 11876 11285 0
...
##  $ BILL_AMT2                 : int  3102 1725 14027 48233 5670 57069 412023 380 14096 0
...
##  $ BILL_AMT3                 : int  689 2682 13559 49291 35835 57608 445007 601 12108 0
...
##  $ BILL_AMT4                 : int  0 3272 14331 28314 20940 19394 542653 221 12211 0 ...
##  $ BILL_AMT5                 : int  0 3455 14948 28959 19146 19619 483003 -159 11793 13007
...
##  $ BILL_AMT6                 : int  0 3261 15549 29547 19131 20024 473944 567 3719 13912
...
##  $ PAY_AMT1                  : int  0 0 1518 2000 2000 2500 55000 380 3329 0 ...
##  $ PAY_AMT2                  : int  689 1000 1500 2019 36681 1815 40000 601 0 0 ...
##  $ PAY_AMT3                  : int  0 1000 1000 1200 10000 657 38000 0 432 0 ...
##  $ PAY_AMT4                  : int  0 1000 1000 1100 9000 1000 20239 581 1000 13007 ...
##  $ PAY_AMT5                  : int  0 0 1000 1069 689 1000 13750 1687 1000 1122 ...
##  $ PAY_AMT6                  : int  0 2000 5000 1000 679 800 13770 1542 1000 0 ...
##  $ default.payment.next.month: int  1 1 0 0 0 0 0 0 0 0 ...
```

I lowercase the column name, and rename the column names when required, In particular, remarkably this dataset misses a column PAY_1. In the analysis here below we assume that PAY_0 is actually pay_1, to be consider the repayment of the month prior to the month where we calculate the defaulting and removing the duplicate rows.

```
df1 = df

names(df1) = tolower(names(df1))

names(df1)[7] = "pay_1"

# Remove "id" column
df1 <- df1[c(2:25)]


colSums(is.na(df1))
```

```
##                 limit_bal                        sex
##                         0                          0
##                 education                   marriage
##                         0                          0
##                       age                      pay_1
##                         0                          0
##                     pay_2                      pay_3
##                         0                          0
##                     pay_4                      pay_5
##                         0                          0
##                     pay_6                  bill_amt1
##                         0                          0
##                 bill_amt2                  bill_amt3
##                         0                          0
##                 bill_amt4                  bill_amt5
##                         0                          0
##                 bill_amt6                   pay_amt1
##                         0                          0
##                  pay_amt2                   pay_amt3
##                         0                          0
##                  pay_amt4                   pay_amt5
##                         0                          0
##                  pay_amt6 default.payment.next.month
##                         0                          0
```

```
df1 = df1[!duplicated(df1),]
```

```
cat("Explanatory variables: ", ncol(df1)-1, "\n\n")
```

```
## Explanatory variables:  23
```

```
cat("Number of Observations: ", nrow(df1), "\n\n")
```

```
## Number of Observations:  29965
```

```
df1$default.payment.next.month <- as.factor(df1$default.payment.next.month)

names(df1)[24] <- "target"

# create a "target" column for our own convenience
cat("Target variable:  'default.payment.next.month' -> 'target' \n\n")
```

```
## Target variable:  'default.payment.next.month' -> 'target'
```

Descriptive Analytics

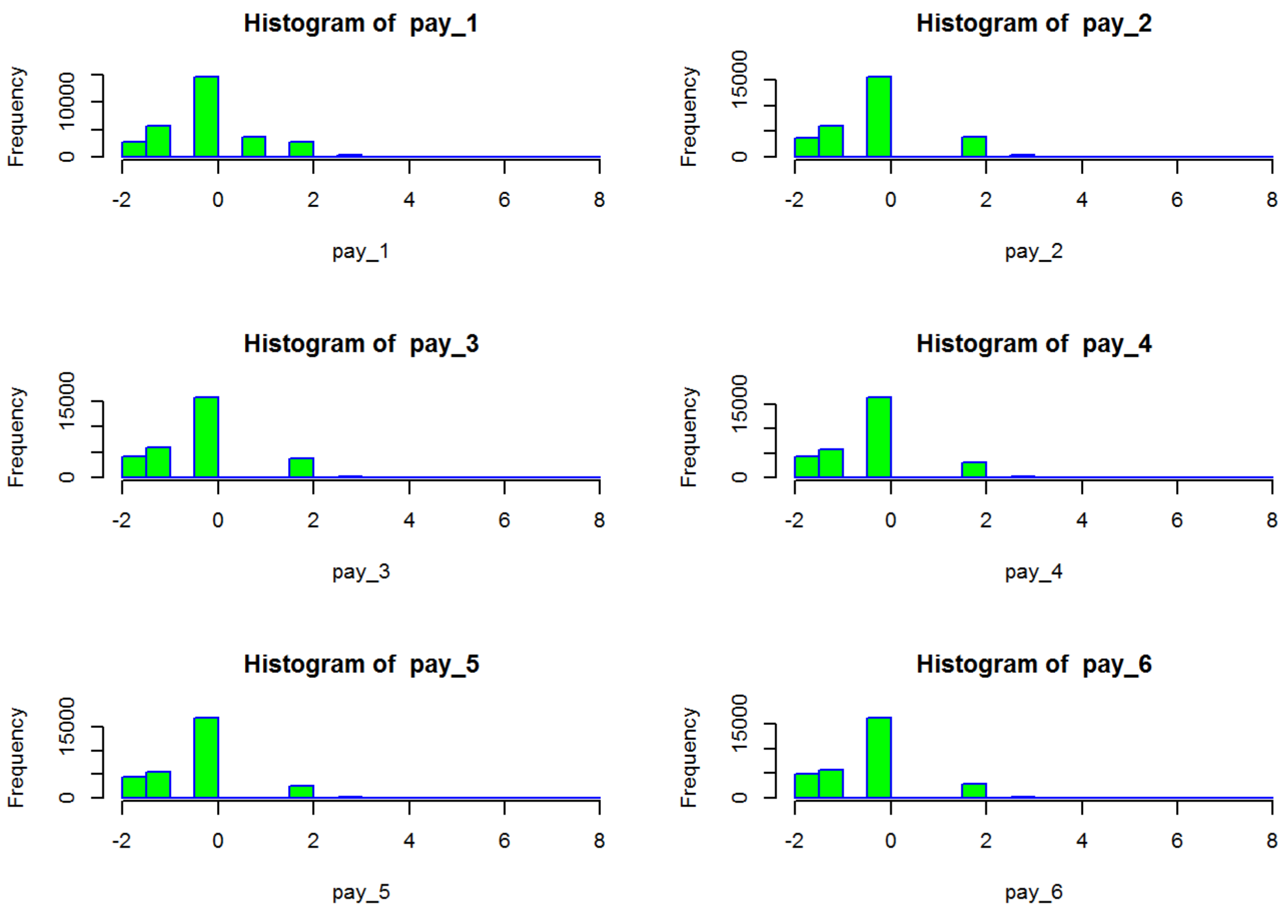
Payment Delays:

Let’s start by looking at the past payment delays

```
#names(df1)

head(df1[c(6:11)],10)
```

```
##    pay_1 pay_2 pay_3 pay_4 pay_5 pay_6
## 1      2     2    -1    -1    -2    -2
## 2     -1     2     0     0     0     2
## 3      0     0     0     0     0     0
## 4      0     0     0     0     0     0
## 5     -1     0    -1     0     0     0
## 6      0     0     0     0     0     0
## 7      0     0     0     0     0     0
## 8      0    -1    -1     0     0    -1
## 9      0     0     2     0     0     0
## 10    -2    -2    -2    -2    -1    -1
```

```
# pay status columns

library(ggplot2)
par(mfrow=c(3,2))
for(i in 6:11)
{

 hist(df1[,i],main = paste("Histogram of ",names(df1)[i]),labels = FALSE,xlab = names(df1)
[i],col = "green",border = "blue")

}
```



Standing credit

Let's look now at how the debts/credit is accumulating over the months, credit to be repaid is a positive number here.

```
summary(df1[,c(12:17)])
```

```
##    bill_amt1         bill_amt2         bill_amt3          bill_amt4
##  Min.   :-165580   Min.   :-69777   Min.   :-157264   Min.   :-170000
##  1st Qu.:   3595   1st Qu.:  3010   1st Qu.:   2711   1st Qu.:   2360
##  Median :  22438   Median : 21295   Median :  20135   Median :  19081
##  Mean   :  51283   Mean   : 49236   Mean   :  47068   Mean   :  43313
##  3rd Qu.:  67260   3rd Qu.: 64109   3rd Qu.:  60201   3rd Qu.:  54601
##  Max.   : 964511   Max.   :983931   Max.   :1664089   Max.   : 891586
##    bill_amt5         bill_amt6
##  Min.   :-81334   Min.   :-339603
##  1st Qu.:  1787   1st Qu.:   1262
##  Median : 18130   Median :  17124
##  Mean   : 40358   Mean   :  38917
##  3rd Qu.: 50247   3rd Qu.:  49252
##  Max.   :927171   Max.   : 961664
```

```
head(df1[,c(12:17)],10)
```

```
##    bill_amt1 bill_amt2 bill_amt3 bill_amt4 bill_amt5 bill_amt6
## 1       3913      3102       689         0         0         0
## 2       2682      1725      2682      3272      3455      3261
## 3      29239     14027     13559     14331     14948     15549
## 4      46990     48233     49291     28314     28959     29547
## 5       8617      5670     35835     20940     19146     19131
## 6      64400     57069     57608     19394     19619     20024
## 7     367965    412023    445007    542653    483003    473944
## 8      11876       380       601       221      -159       567
## 9      11285     14096     12108     12211     11793      3719
## 10         0         0         0         0     13007     13912
```

Payments in the previous months

Let's have a quick look at how the payments are performed in the previous month.

```
# pay status columns

summary(df1[,c(18:23)])
```

```
##     pay_amt1         pay_amt2          pay_amt3         pay_amt4     
##  Min.   :     0   Min.   :      0   Min.   :     0   Min.   :     0  
##  1st Qu.:  1000   1st Qu.:    850   1st Qu.:   390   1st Qu.:   300  
##  Median :  2102   Median :   2010   Median :  1804   Median :  1500  
##  Mean   :  5670   Mean   :   5928   Mean   :  5232   Mean   :  4832  
##  3rd Qu.:  5008   3rd Qu.:   5000   3rd Qu.:  4512   3rd Qu.:  4016  
##  Max.   :873552   Max.   :1684259   Max.   :896040   Max.   :621000  
##     pay_amt5         pay_amt6     
##  Min.   :     0   Min.   :     0  
##  1st Qu.:   261   1st Qu.:   131  
##  Median :  1500   Median :  1500  
##  Mean   :  4805   Mean   :  5222  
##  3rd Qu.:  4042   3rd Qu.:  4000  
##  Max.   :426529   Max.   :528666
```

```
head(df1[,c(18:23)],10)
```

```
##    pay_amt1 pay_amt2 pay_amt3 pay_amt4 pay_amt5 pay_amt6
## 1         0      689        0        0        0        0
## 2         0     1000     1000     1000        0     2000
## 3      1518     1500     1000     1000     1000     5000
## 4      2000     2019     1200     1100     1069     1000
## 5      2000    36681    10000     9000      689      679
## 6      2500     1815      657     1000     1000      800
## 7     55000    40000    38000    20239    13750    13770
## 8       380      601        0      581     1687     1542
## 9      3329        0      432     1000     1000     1000
## 10        0        0        0    13007     1122        0
```

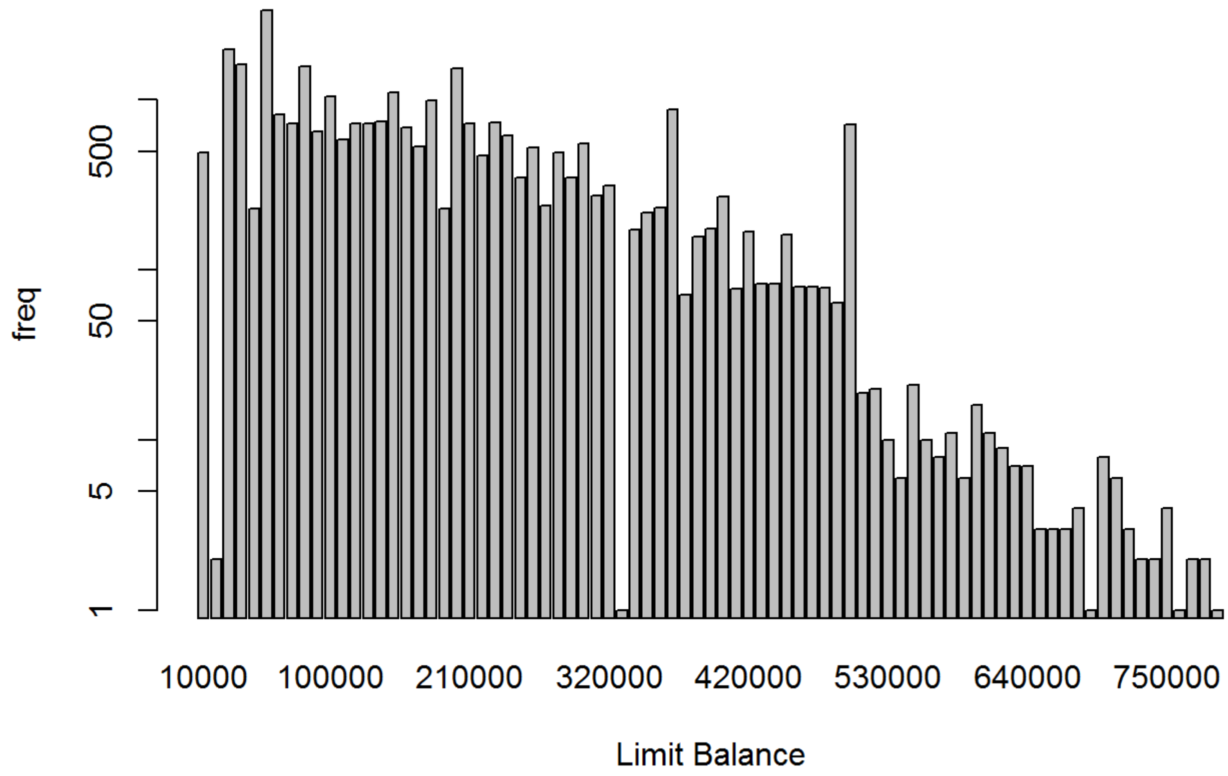
```
summary(df1$limit_bal)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   10000   50000  140000  167442  240000 1000000
```

```
cat("\n Standard deviation:",sd(df1$limit_bal),"\n\n")
```

```
## 
##  Standard deviation: 129760.1
```

```
# Limit balance

counts <- table(df1$limit_bal)
barplot(counts,log = "y",ylab = "freq",xlab = "Limit Balance")
```

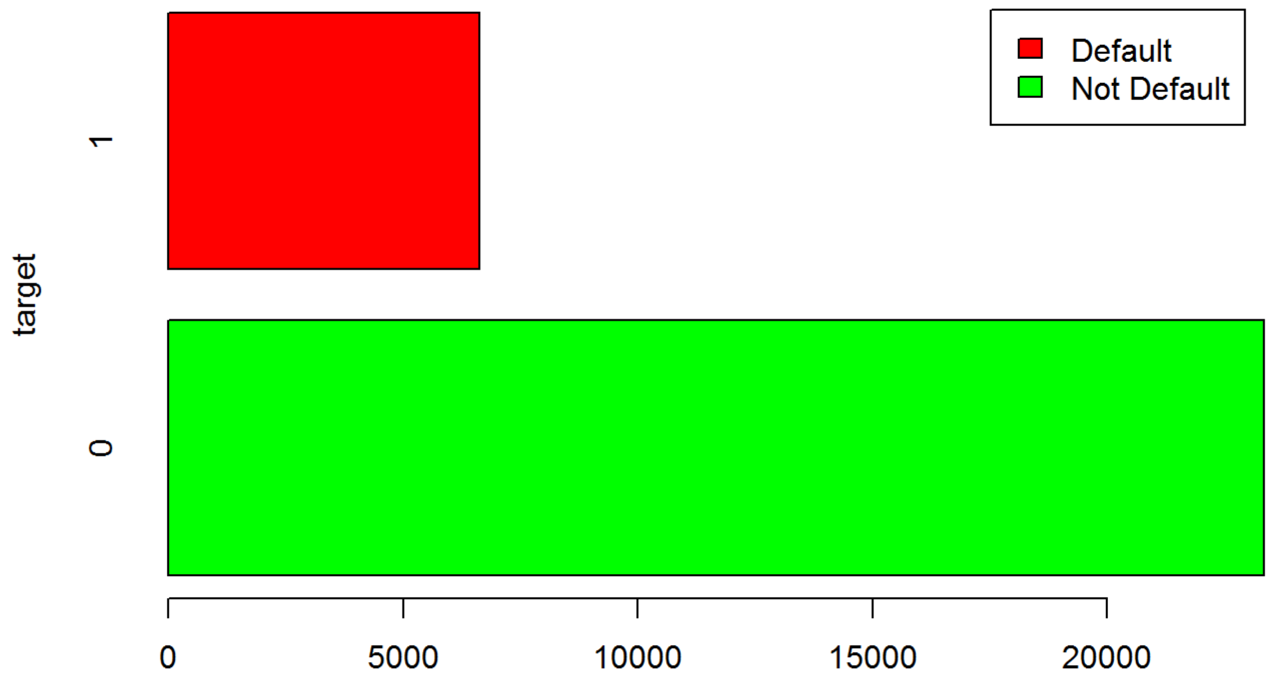


Explore Defaulting

First off, let’s start with a zoomed out view on the problem. We want to predict defaulting, Let’s answer the following questions:

how many cases do we have on our dataset to work with? What is the breakdown depending on some of the variables available?

```
d = df1

d = table(d$target)

barplot(d,horiz = TRUE,ylab = "target",legend.text = c("Not Default","Default"),col = c("gree
n","red"))
```

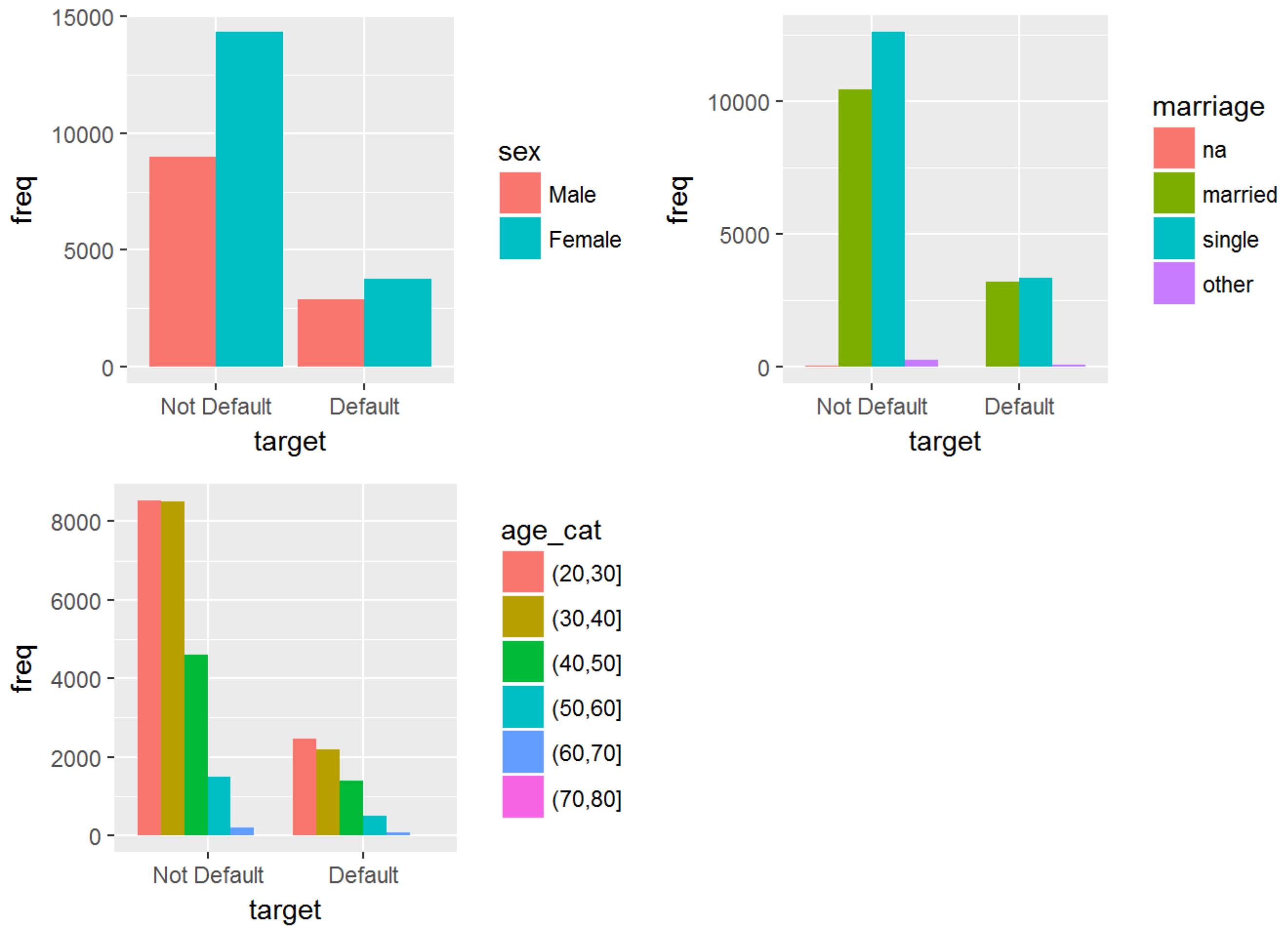


Explore some statistics of defaulting using the categorical variables

Let’s have a look at a number of histograms to see how defaulting correlated with the categorical variables available, by converting target, sex, marriage, age to categories

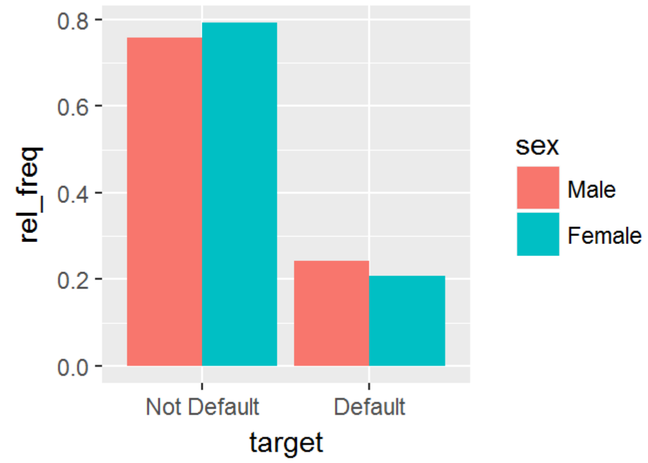
Absolute statistics

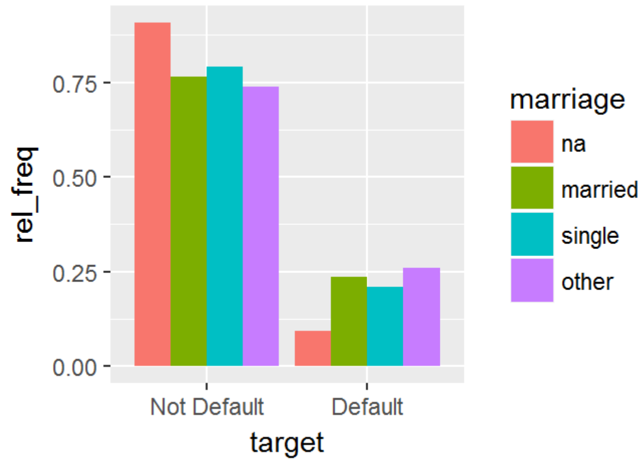
```
e = df1

e$target = factor(e$target,levels=c(0,1),labels = c("Not Default","Default"))

e$sex = factor(e$sex,levels=c(1,2),labels = c("Male","Female"))

e$marriage = factor(df1$marriage,levels = c(0,1,2,3),labels = c("na","married","single","other"))

e1 = e %>% group_by(target,sex) %>% summarise(freq = n())

e2 = e %>% group_by(target,marriage) %>% summarise(freq = n())

e$age_cat = cut(e$age,breaks = seq(0,100,10),include.lowest = TRUE)

e3 = e %>% group_by(target,age_cat) %>% summarise(freq = n())

plot1 = ggplot(e1,aes(x=target,y=freq,fill=sex)) + geom_bar(stat = 'identity',position = 'dodge')
plot2 = ggplot(e2,aes(x=target,y=freq,fill=marriage)) + geom_bar(stat = 'identity',position = 'dodge')
plot3 = ggplot(e3,aes(x=target,y=freq,fill=age_cat)) + geom_bar(stat = 'identity',position = 'dodge')

grid.arrange(plot1, plot2,plot3,ncol=2)
```

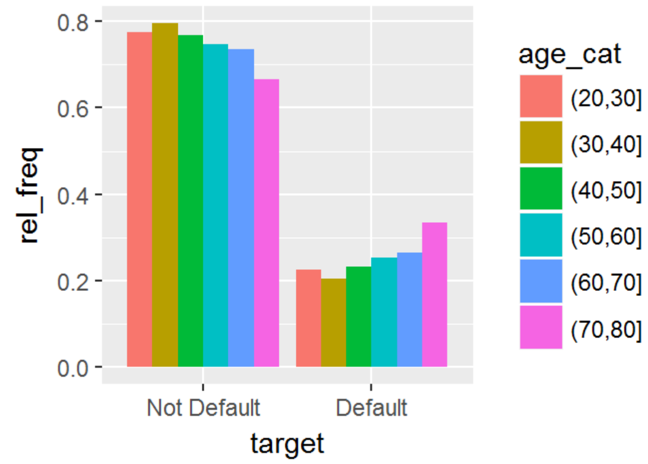


Statistics relative to the population

```
e = df1

e$target = factor(e$target,levels=c(0,1),labels = c("Not Default","Default"))

e$sex = factor(e$sex,levels=c(1,2),labels = c("Male","Female"))

e$marriage = factor(df1$marriage,levels = c(0,1,2,3),labels = c("na","married","single","othe
r"))

e1 = e %>% group_by(target,sex) %>% summarise(freq = n())
e11 = e %>% group_by(sex) %>% summarise(freq1 = n())
e1$rel_freq = e1$freq/e11$freq1

e2 = e %>% group_by(target,marriage) %>% summarise(freq = n())
e11 = e %>% group_by(marriage) %>% summarise(freq1 = n())
e2$rel_freq = e2$freq/e11$freq1

e$age_cat = cut(e$age,breaks = seq(0,100,10),include.lowest = TRUE)

e3 = e %>% group_by(target,age_cat) %>% summarise(freq = n())
e11 = e %>% group_by(age_cat) %>% summarise(freq1 = n())
e3$rel_freq = e3$freq/e11$freq1

plot1 = ggplot(e1,aes(x=target,y=rel_freq,fill=sex)) + geom_bar(stat = 'identity',position =
'dodge')
plot2 = ggplot(e2,aes(x=target,y=rel_freq,fill=marriage)) + geom_bar(stat = 'identity',positi
on = 'dodge')
plot3 = ggplot(e3,aes(x=target,y=rel_freq,fill=age_cat)) + geom_bar(stat = 'identity',positio
n = 'dodge')

grid.arrange(plot1, plot2,plot3,ncol=2)
```







Feature engineering

Splitting the dataset into the Training set and Test set

```
d = df1

#d[,c(1,5,12:23)] = scale(d[,c(1,5,12:23)])

library(caTools)
```

```
## Warning: package 'caTools' was built under R version 3.4.2
```

```
set.seed(123)
split = sample.split(d, SplitRatio = 0.7)

train = subset(d, split==T)
test = subset(d, split==F)

 train[,c(1,5,12:23)] = scale(train[,c(1,5,12:23)])
 test[,c(1,5,12:23)] = scale(test[,c(1,5,12:23)])
```

Models

Support Vector Machine (SVM)

```
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.4.2
```

```
model1 = svm(formula = target ~ ., data = train, type = 'C-classification', kernel = 'linear'
)

prob_pred = predict(model1,newdata = test[,-24])

 cm = table(prob_pred,test[,24])

# cm = table(prob_pred,test[,10])

accuracy=sum(diag(cm))/sum(cm)

cat("SVM model's accuracy: ",accuracy)
```

```
## SVM model's accuracy:  0.8068682
```

Logistic Regression

```
 model2 = glm(formula = target ~ ., data = train, family = binomial)
```

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

```
#model2 = glm(formula = target ~ limit_bal+education+marriage+age+pay_1+pay_2+pay_3+bill_amt1
+pay_amt1+pay_amt2, data = train, family = binomial)

prob_pred = predict(model2,newdata = test[,-24],type = 'response')

y_pred = ifelse(prob_pred > 0.5,1,0)

 cm = table(y_pred,test[,24])

# cm = table(prob_pred,test[,10])

accuracy=sum(diag(cm))/sum(cm)

cat("Logistic Regression model's accuracy: ",accuracy)
```

```
## Logistic Regression model's accuracy:  0.8129756
```

```
#str(d)
```

Naive Bayes

```r
library(e1071)

model3 = naiveBayes(formula = target ~ ., data = train)

prob_pred = predict(model3,newdata = test[,-24])

 cm = table(prob_pred,test[,24])

# cm = table(prob_pred,test[,10])

accuracy=sum(diag(cm))/sum(cm)

cat("Naive Bayes model's accuracy: ",accuracy)
```

```
## Naive Bayes model's accuracy:  0.7300761
```

Decision Tree

```r
library(rpart)

model4 = rpart(formula = target ~ ., data = train)

prob_pred = predict(model4,newdata = test[,-24],type = 'class')

 cm = table(prob_pred,test[,24])

# cm = table(prob_pred,test[,10])

accuracy=sum(diag(cm))/sum(cm)

cat("Decission Tree Classification model's accuracy: ",accuracy)
```

```
## Decission Tree Classification model's accuracy:  0.8201842
```