Project Code Final

2023-05-10

R Markdown

Package Libraries

```
rm(list = ls())

library(readr)



library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.1     ✓ purrr     1.0.1
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.1     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ── Conflicts ────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors

# install.packages("visdat")
library(visdat) # visualize missing values
# install.packages("caret")
library(caret)

## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift

library(ggpubr)
# install.packages("car")
library(car)

## Loading required package: carData
##
## Attaching package: 'car'
```

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

```
# install.packages("visdat")
library(moments)

library(dplyr)
# install.packages("MASS")
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(ggplot2)

library(gridExtra)
```

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

```
library("DescTools")
```

```
##
## Attaching package: 'DescTools'
##
## The following object is masked from 'package:car':
##
##     Recode
##
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
```

```
library(rpart)
library(rpart.plot)
```

Read Dataset

```
df <- read.csv("UCI_Credit_Card.csv")
df <- as_tibble(df)
glimpse(df)
```

```
## Rows: 30,000
## Columns: 25
## $ ID                        <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12,
13, …
## $ LIMIT_BAL                 <dbl> 20000, 120000, 90000, 50000, 50000,
50000, …
## $ SEX                       <int> 2, 2, 2, 2, 1, 1, 1, 2, 2, 1, 2, 2, 2,
1, 1…
## $ EDUCATION                 <int> 2, 2, 2, 2, 2, 1, 1, 2, 3, 3, 3, 1, 2,
2, 1…
## $ MARRIAGE                  <int> 1, 2, 2, 1, 1, 2, 2, 2, 1, 2, 2, 2, 2,
2, 2…
## $ AGE                       <int> 24, 26, 34, 37, 57, 37, 29, 23, 28, 35,
34,…
## $ PAY_0                     <int> 2, -1, 0, 0, -1, 0, 0, 0, 0, -2, 0, -1,
-1,…
## $ PAY_2                     <int> 2, 2, 0, 0, 0, 0, 0, -1, 0, -2, 0, -1,
0, 2…
## $ PAY_3                     <int> -1, 0, 0, 0, -1, 0, 0, -1, 2, -2, 2, -
1, -1…
## $ PAY_4                     <int> -1, 0, 0, 0, 0, 0, 0, 0, 0, -2, 0, -1,
-1, …
## $ PAY_5                     <int> -2, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, -1,
-1, …
## $ PAY_6                     <int> -2, 2, 0, 0, 0, 0, 0, -1, 0, -1, -1, 2,
-1,…
## $ BILL_AMT1                 <dbl> 3913, 2682, 29239, 46990, 8617, 64400,
3679…
## $ BILL_AMT2                 <dbl> 3102, 1725, 14027, 48233, 5670, 57069,
4120…
## $ BILL_AMT3                 <dbl> 689, 2682, 13559, 49291, 35835, 57608,
4450…
## $ BILL_AMT4                 <dbl> 0, 3272, 14331, 28314, 20940, 19394,
542653…
## $ BILL_AMT5                 <dbl> 0, 3455, 14948, 28959, 19146, 19619,
483003…
## $ BILL_AMT6                 <dbl> 0, 3261, 15549, 29547, 19131, 20024,
473944…
## $ PAY_AMT1                  <dbl> 0, 0, 1518, 2000, 2000, 2500, 55000,
380, 3…
## $ PAY_AMT2                  <dbl> 689, 1000, 1500, 2019, 36681, 1815,
40000, …
## $ PAY_AMT3                  <dbl> 0, 1000, 1000, 1200, 10000, 657, 38000,
0, …
## $ PAY_AMT4                  <dbl> 0, 1000, 1000, 1100, 9000, 1000, 20239,
```

```
581…
## $ PAY_AMT5                   <dbl> 0, 0, 1000, 1069, 689, 1000, 13750,
1687, 1…
## $ PAY_AMT6                   <dbl> 0, 2000, 5000, 1000, 679, 800, 13770,
1542,…
## $ default.payment.next.month <int> 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0…
```

Drop ID column

```
df <- df[,-1]
head(df)
```

```
## # A tibble: 6 × 24
##   LIMIT_BAL   SEX EDUCATION MARRIAGE   AGE PAY_0 PAY_2 PAY_3 PAY_4 PAY_5
PAY_6
##       <dbl> <int>     <int>    <int> <int> <int> <int> <int> <int> <int>
<int>
## 1     20000     2         2        1    24     2     2    -1    -1    -2
-2
## 2    120000     2         2        2    26    -1     2     0     0     0
2
## 3     90000     2         2        2    34     0     0     0     0     0
0
## 4     50000     2         2        1    37     0     0     0     0     0
0
## 5     50000     1         2        1    57    -1     0    -1     0     0
0
## 6     50000     1         1        2    37     0     0     0     0     0
0
## # … with 13 more variables: 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 <int>
```

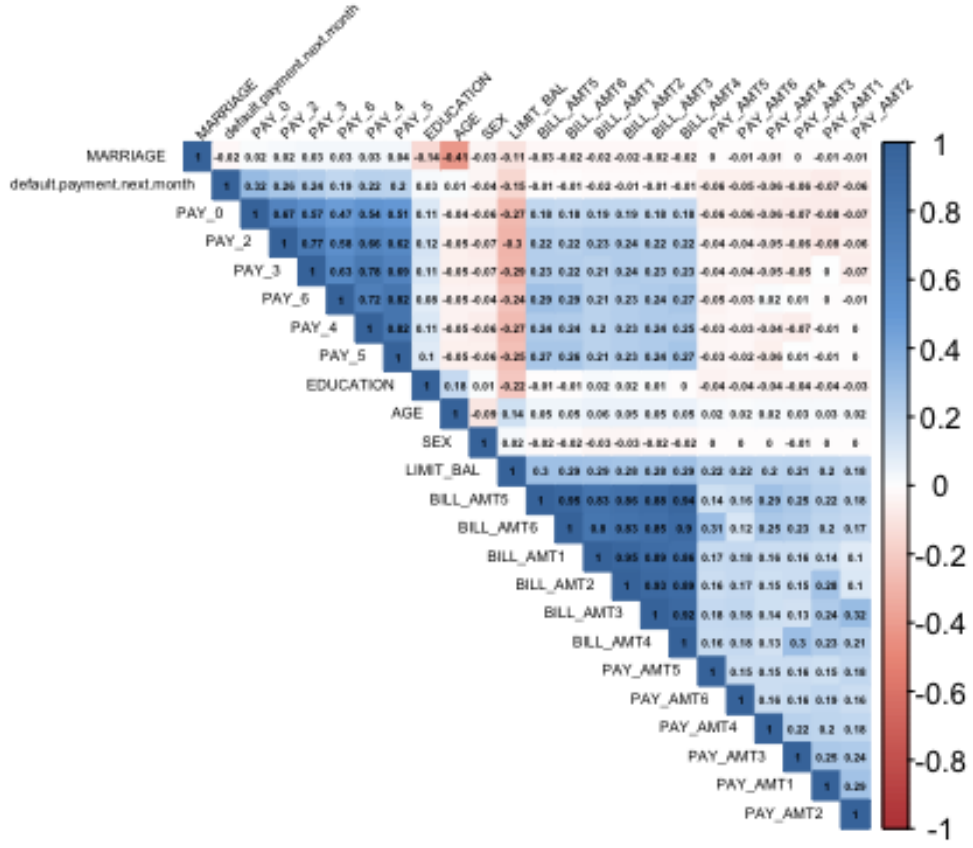
Find correlation between the target variables and independent variables

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
#Find the correlation of the dataset
corplotdf <- cor(df, method = "pearson")
col_gd <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD",
"#4477AA"))
corrplot(corplotdf, method = "color", col = col_gd(200),
         type = "upper", order = "hclust",
```

```
         addCoef.col = "Black",
         tl.col = "black", tl.srt = 45, number.cex = 0.3,tl.cex = 0.4)
```



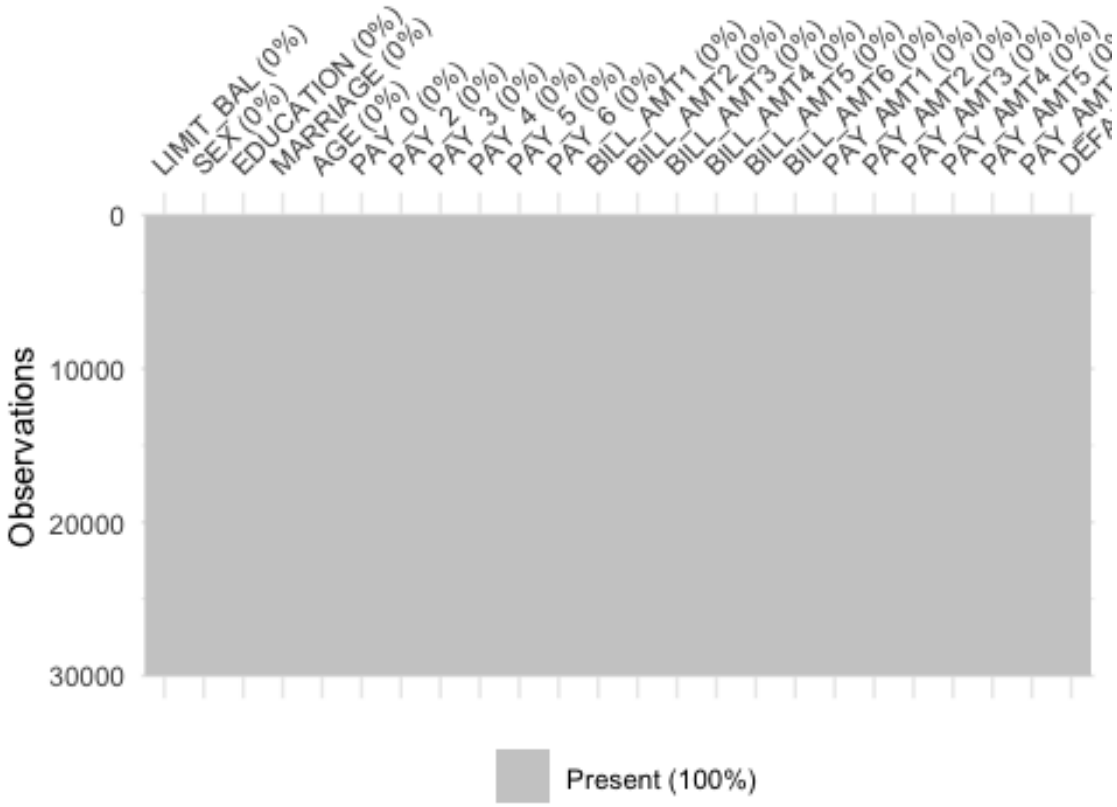
Rename response variable to DEFAULT and convert to Yes or No

```
df<-df %>%
   rename(DEFAULT = default.payment.next.month)

# Replace encoding with corresponding string for EDA, we will one-hot encode
later
df <- df %>%
  mutate(DEFAULT = as.factor(ifelse(DEFAULT == "1", "Yes", "No")),
         SEX = as.factor(ifelse(SEX == "1", "Male", "Female")),
         EDUCATION = as.factor(ifelse(EDUCATION == "1", "Graduate School",
                                      ifelse(EDUCATION == "2", "University",
                                             ifelse(EDUCATION == "3", "High
School",
                                                    ifelse(EDUCATION == "4",
"Other", "Unknown"))))),
         MARRIAGE = as.factor(ifelse(MARRIAGE == "1", "Married",
                                     ifelse(MARRIAGE == "2", "Single",
```

```
"Other")))
         )
```

Check Missing Values

```
sum(is.na(df))

## [1] 0

vis_miss(df)
```

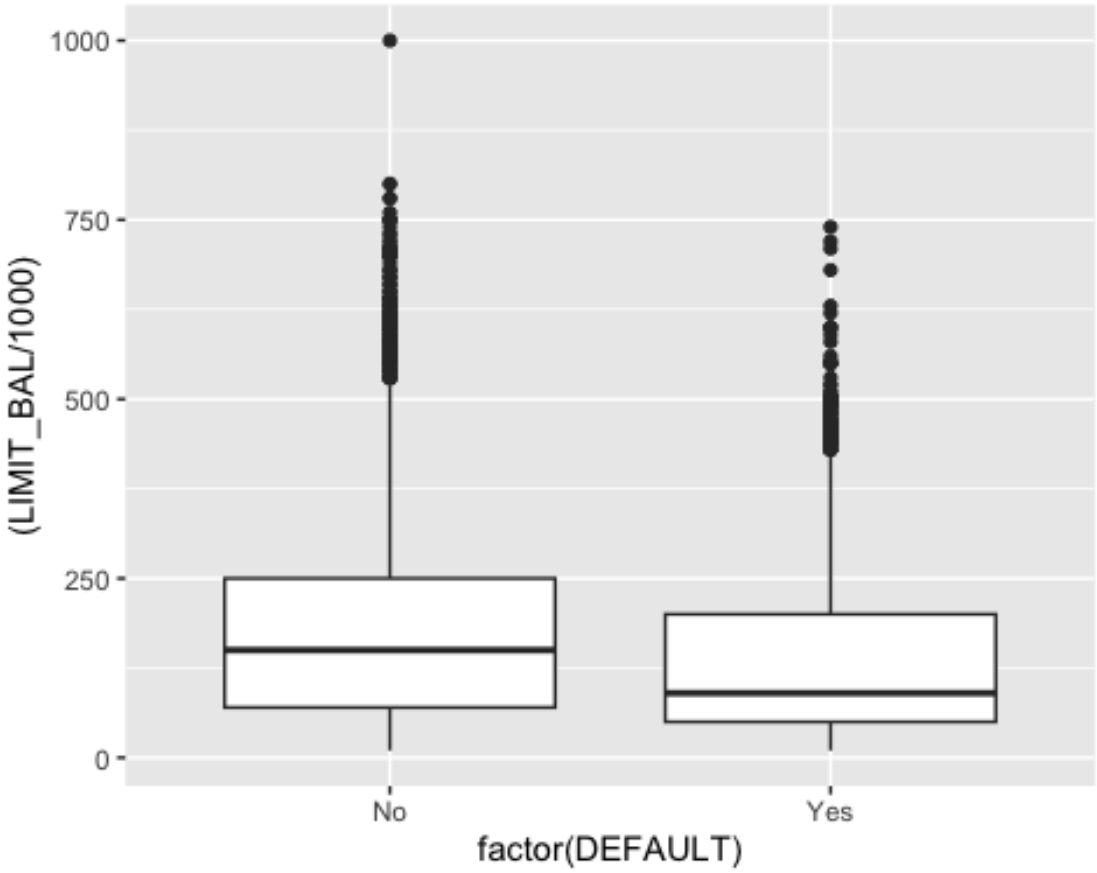


There is no missing values

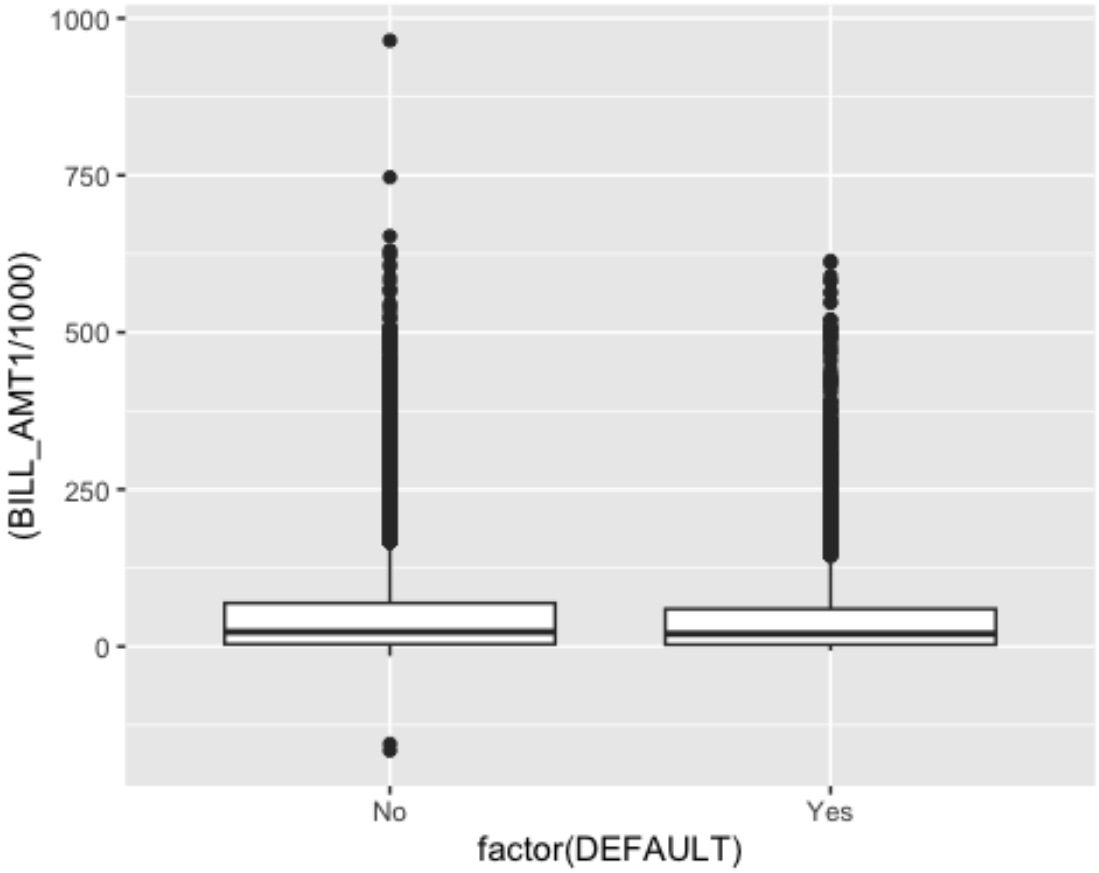
EDA

Visualization

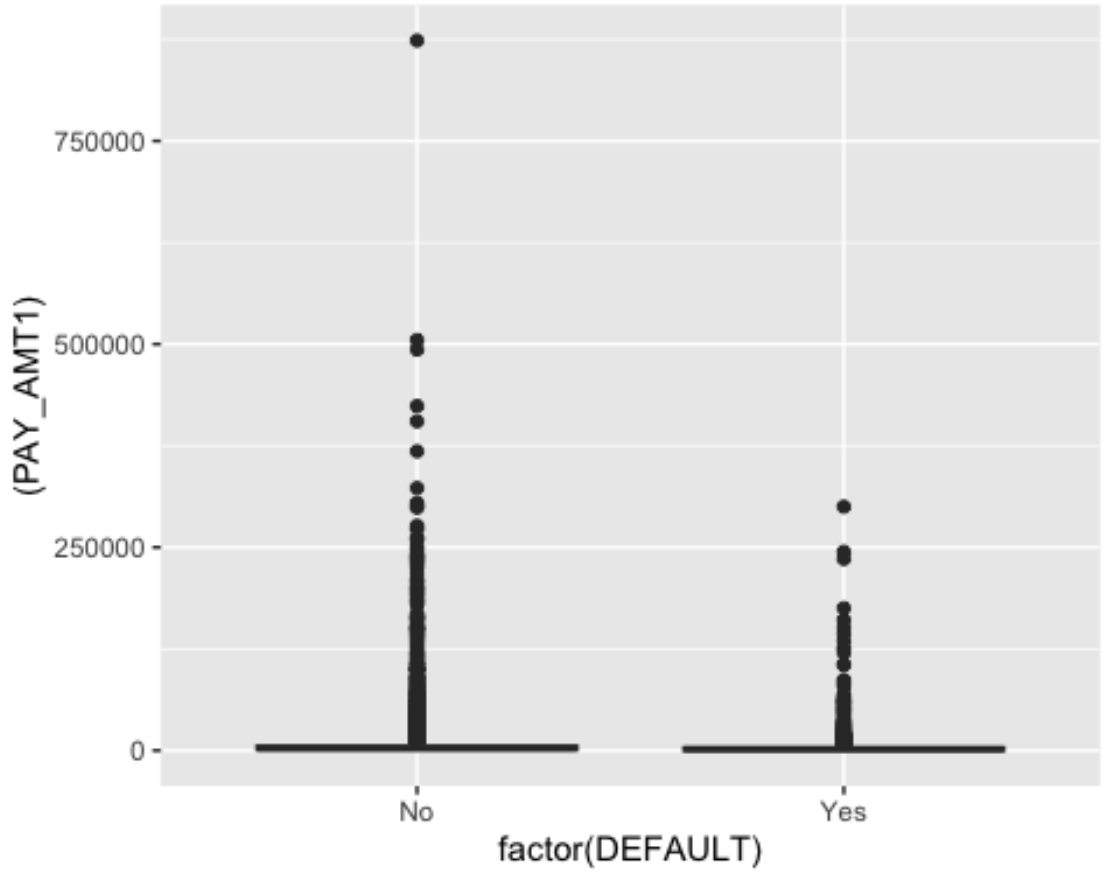
```
# default and limit_balance
d0 <- ggplot(df, aes(factor(DEFAULT), (LIMIT_BAL/1000))) + geom_boxplot()
d0
```



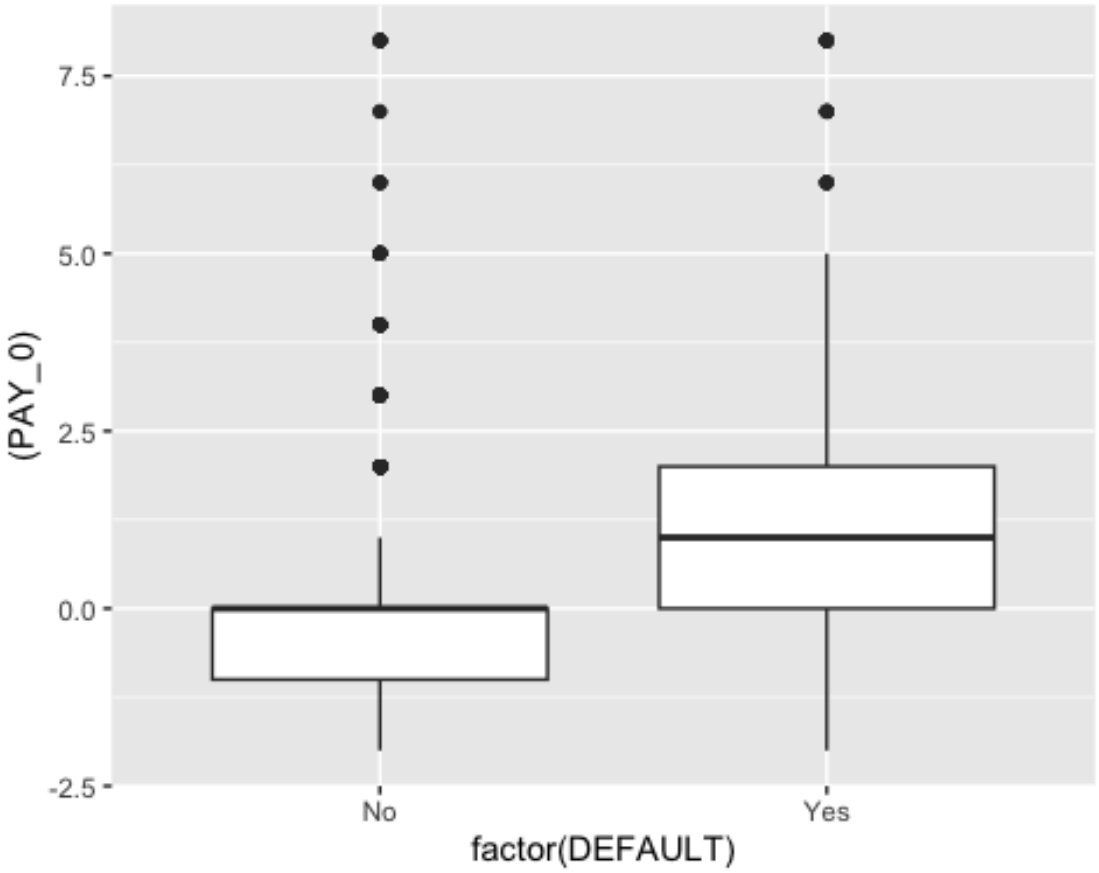

```
#default and bill payment
d1 <- ggplot(df, aes(factor(DEFAULT), (BILL_AMT1/1000))) + geom_boxplot()
d1
```



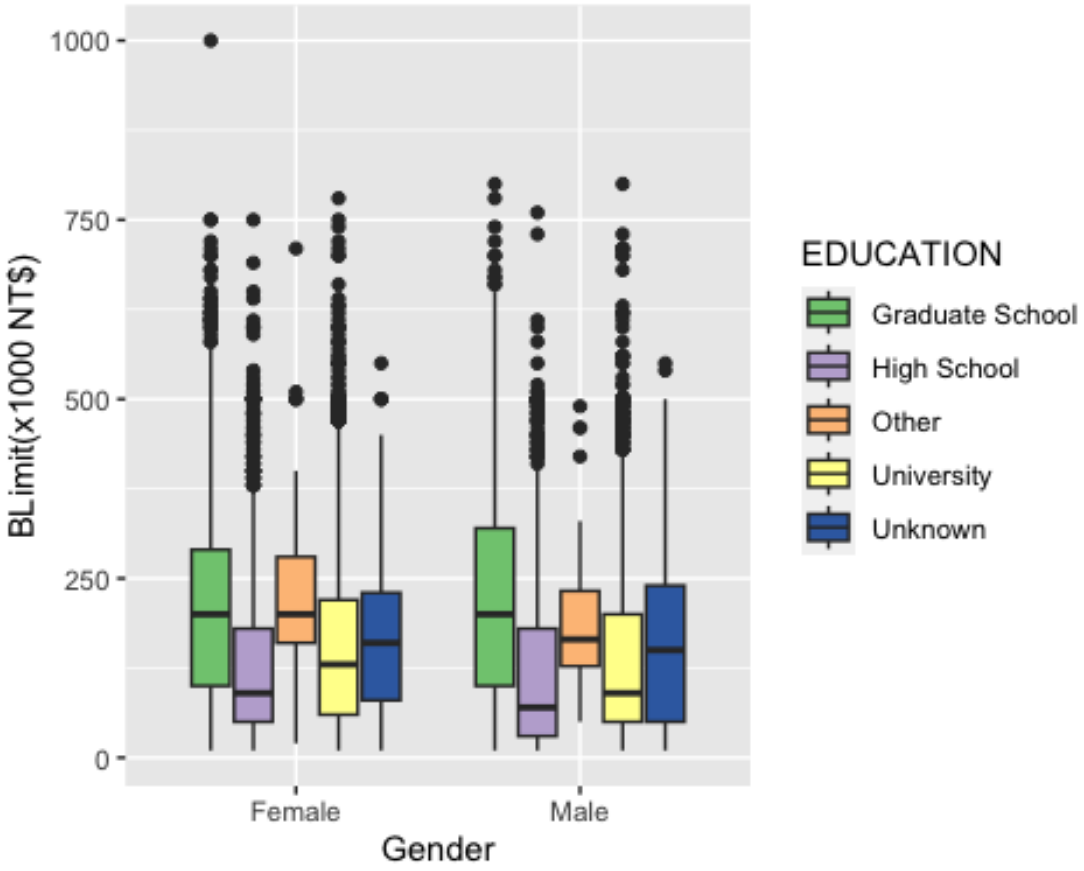

```
d8 <- ggplot(df, aes(factor(DEFAULT), (PAY_AMT1))) + geom_boxplot()
d8
```



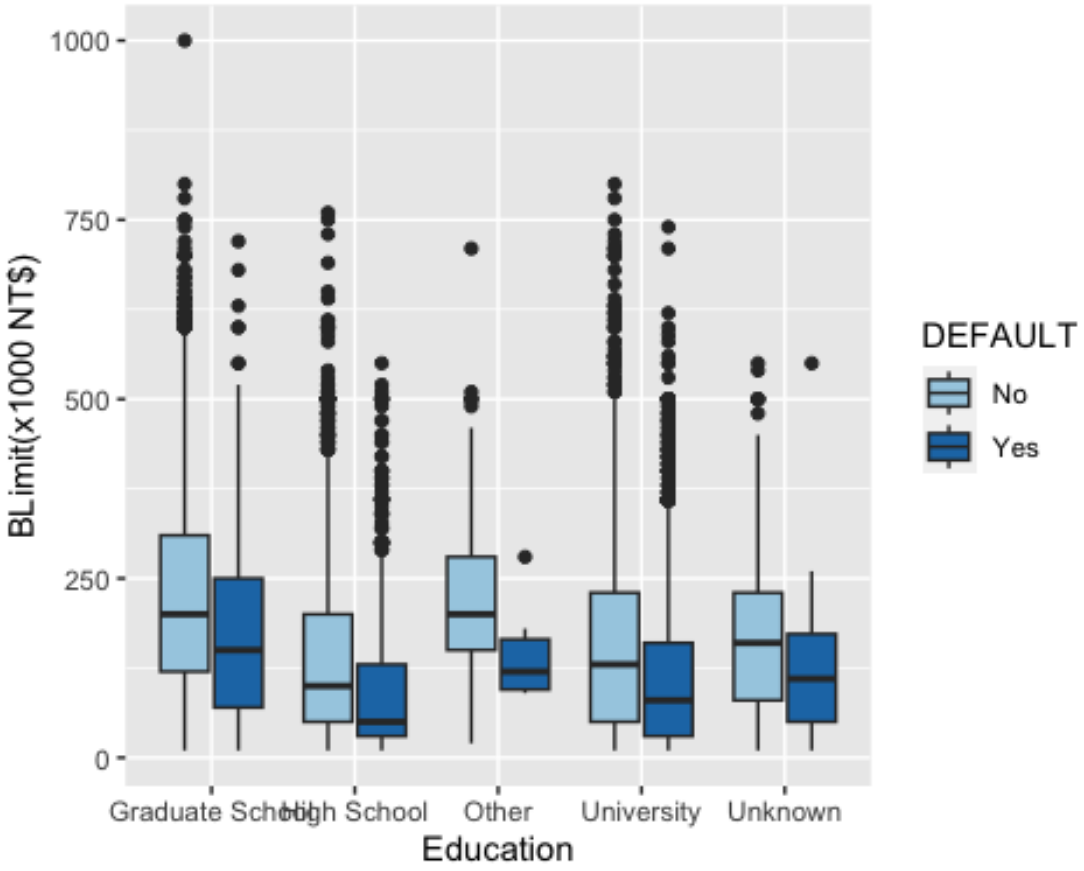

```
#default and bill payment
d7 <- ggplot(df, aes(factor(DEFAULT), (PAY_0))) + geom_boxplot()
d7
```



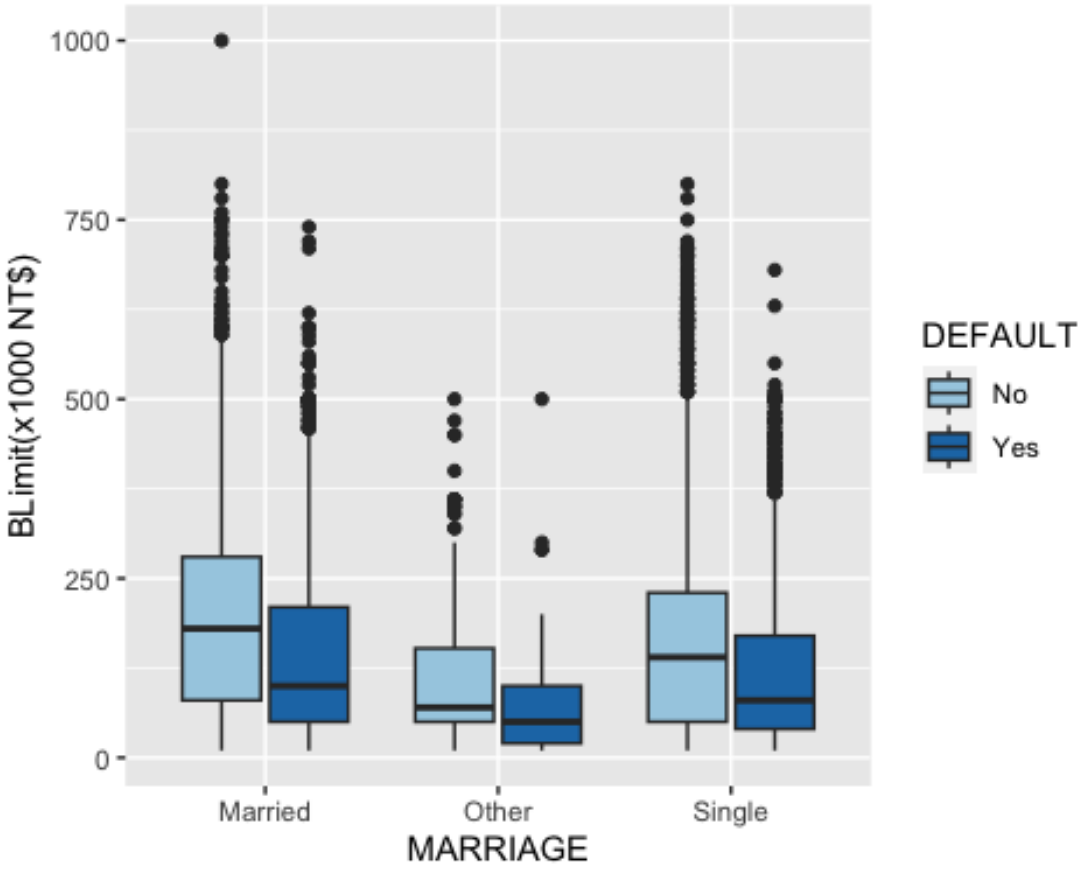

```
# SEX, limit balance education
d2 <- ggplot(df, aes(factor(SEX), (LIMIT_BAL/1000), fill=EDUCATION)) +
  geom_boxplot() +
  xlab("Gender") +
  ylab("BLimit(x1000 NT$)") +
  scale_fill_brewer(palette = "Accent")
d2
```



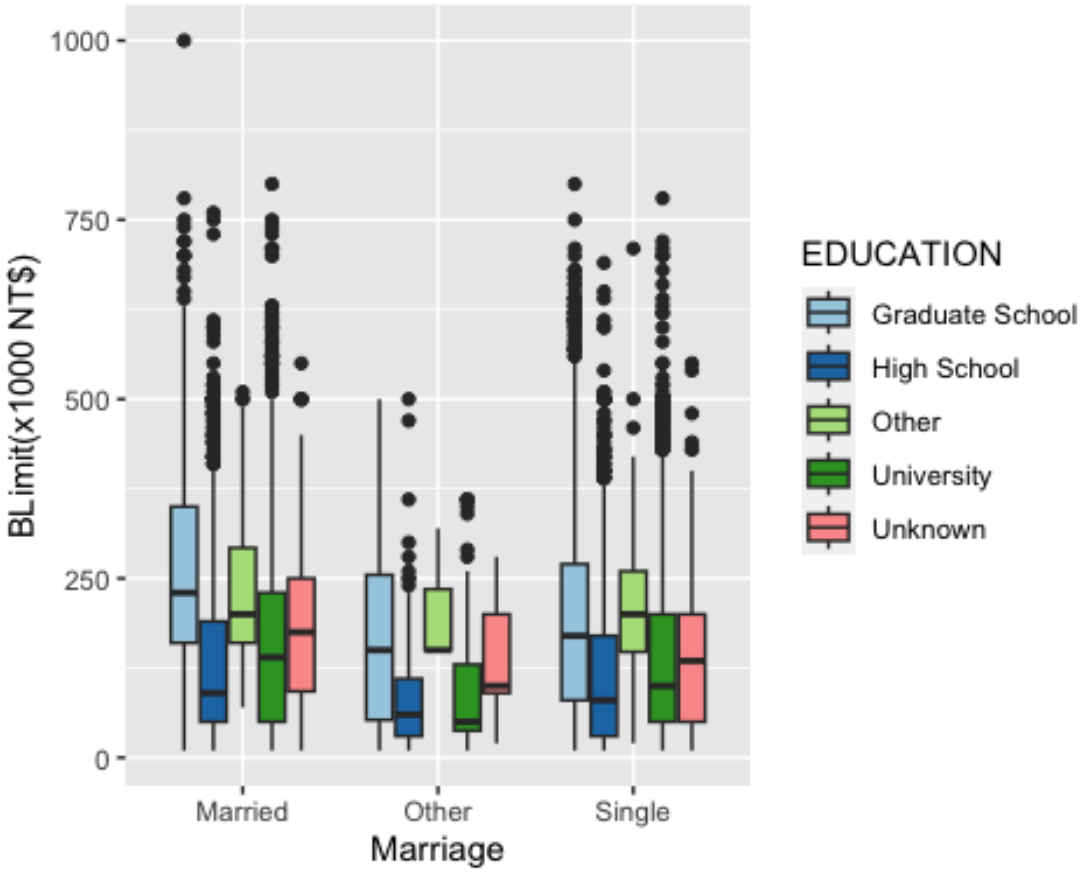

```r
# Balance limits ,education and gender
d3 <- ggplot(df, aes(factor(EDUCATION), (LIMIT_BAL/1000), fill=DEFAULT)) +
  geom_boxplot() +
  xlab("Education") +
  ylab("BLimit(x1000 NT$)") +
  scale_fill_brewer(palette = "Paired")
d3
```



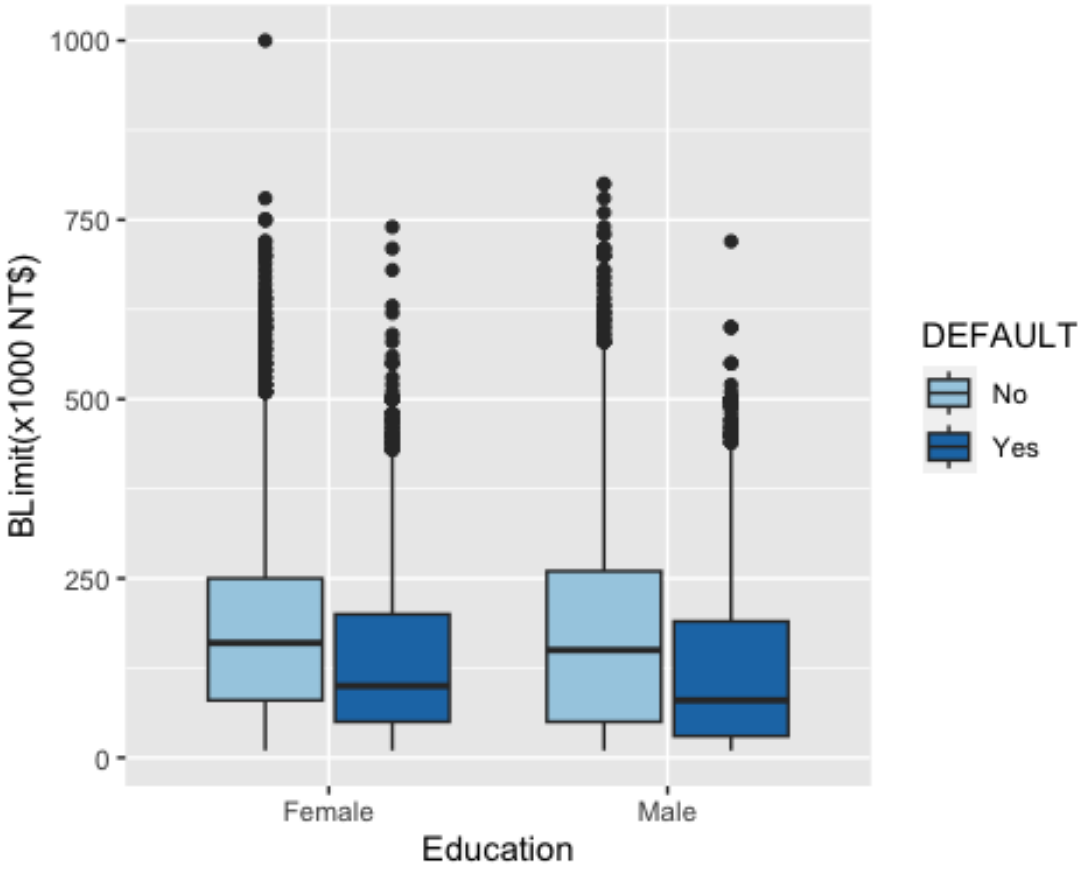

```
d3 <- ggplot(df, aes(factor(MARRIAGE), (LIMIT_BAL/1000), fill=DEFAULT)) +
  geom_boxplot() +
  xlab("MARRIAGE") +
  ylab("BLimit(x1000 NT$)") +
  scale_fill_brewer(palette = "Paired")
d3
```



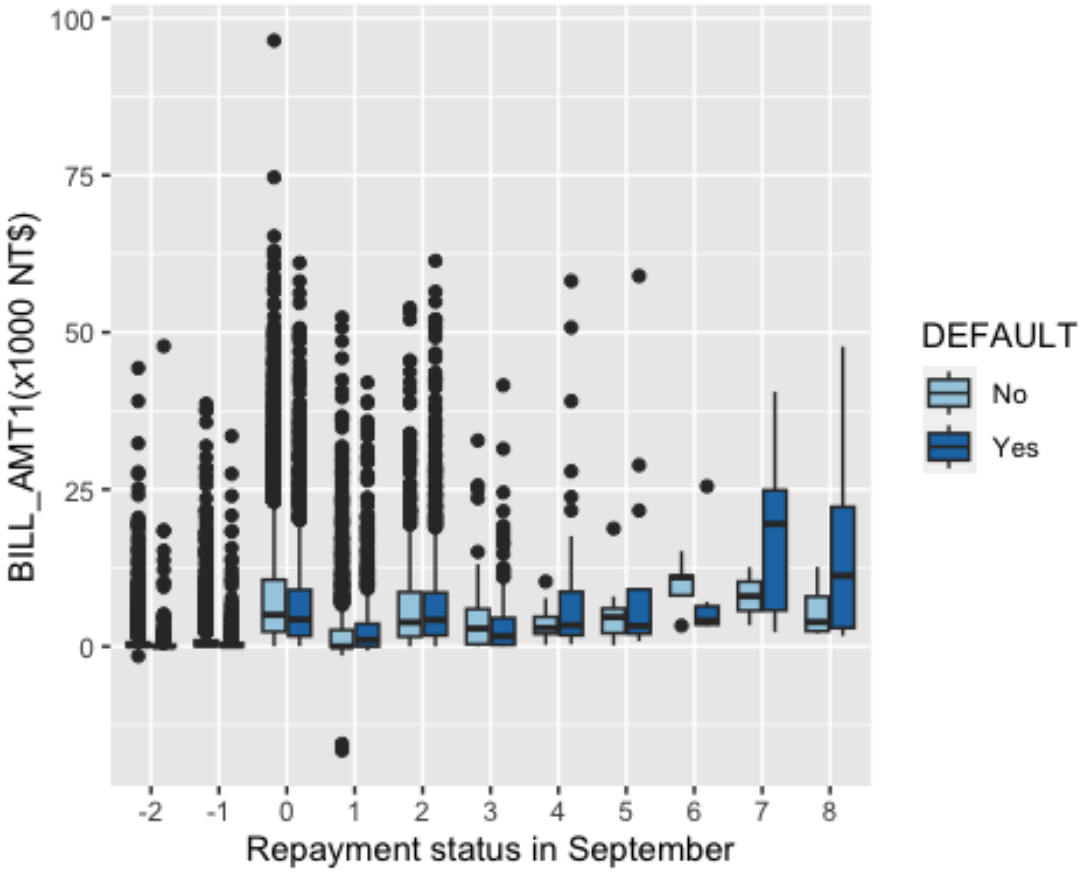

```r
# Balance limit, marraige, education
d4 <- ggplot(df, aes(factor(MARRIAGE), (LIMIT_BAL/1000), fill=EDUCATION)) +
  geom_boxplot() +
  xlab("Marriage") +
  ylab("BLimit(x1000 NT$)") +
  scale_fill_brewer(palette = "Paired")
d4
```



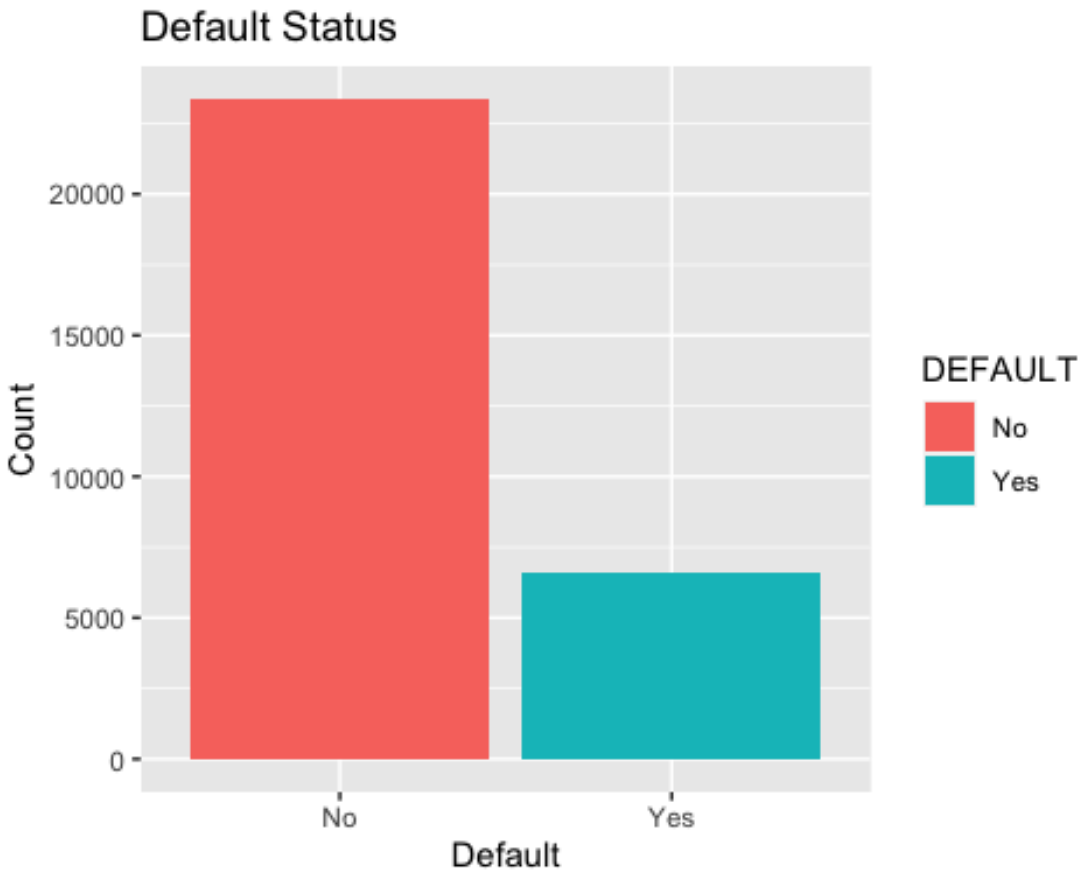

```
# Balance limits ,education and gender
d6 <- ggplot(df, aes(factor(SEX), (LIMIT_BAL/1000), fill=DEFAULT)) +
  geom_boxplot() +
  xlab("Education") +
  ylab("BLimit(x1000 NT$)") +
  scale_fill_brewer(palette = "Paired")
d6
```




```
d5 <- ggplot(df, aes(factor(PAY_0), (BILL_AMT1/10000), fill=DEFAULT)) +
  geom_boxplot() +
  xlab("Repayment status in September") +
  ylab("BILL_AMT1(x1000 NT$)") +
  scale_fill_brewer(palette = "Paired")
d5
```




```
ggplot(df, aes(x = DEFAULT, fill = DEFAULT)) +
  geom_bar() +
  labs(title = "Default Status",
       x = "Default",
       y = "Count")
```




```
# install.packages("gridExtra")  # Install the package
library(gridExtra)  # Load the package

graph1 <- ggplot(data=df, aes(x=BILL_AMT1,fill=DEFAULT)) + geom_histogram() +
  labs(title = "BILL_AMT1", x ="BILL_AMT1",fill = "DEFAULT") +
  scale_fill_manual(values=c("#56B4E9", "#FF9999")) +
  theme(axis.text.x = element_text(angle = 45,hjust=1))

graph2 <- ggplot(data=df, aes(x=BILL_AMT2,fill=DEFAULT)) + geom_histogram() +
  labs(title = "BILL_AMT2", x ="BILL_AMT2",fill = "DEFAULT") +
   scale_fill_manual(values=c("#56B4E9", "#FF9999"))
  theme(axis.text.x = element_text(angle = 45,hjust=1))

## List of 1
##  $ axis.text.x:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : num 45
##   ..$ lineheight   : NULL
```

```
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

```
graph3 <- ggplot(data=df, aes(x=BILL_AMT3,fill=DEFAULT)) + geom_histogram() +
  labs(title = "BILL_AMT3", x ="BILL_AMT3",fill = "DEFAULT") +
   scale_fill_manual(values=c("#56B4E9", "#FF9999"))
  theme(axis.text.x = element_text(angle = 45,hjust=1))
```

```
## List of 1
##  $ axis.text.x:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : num 45
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

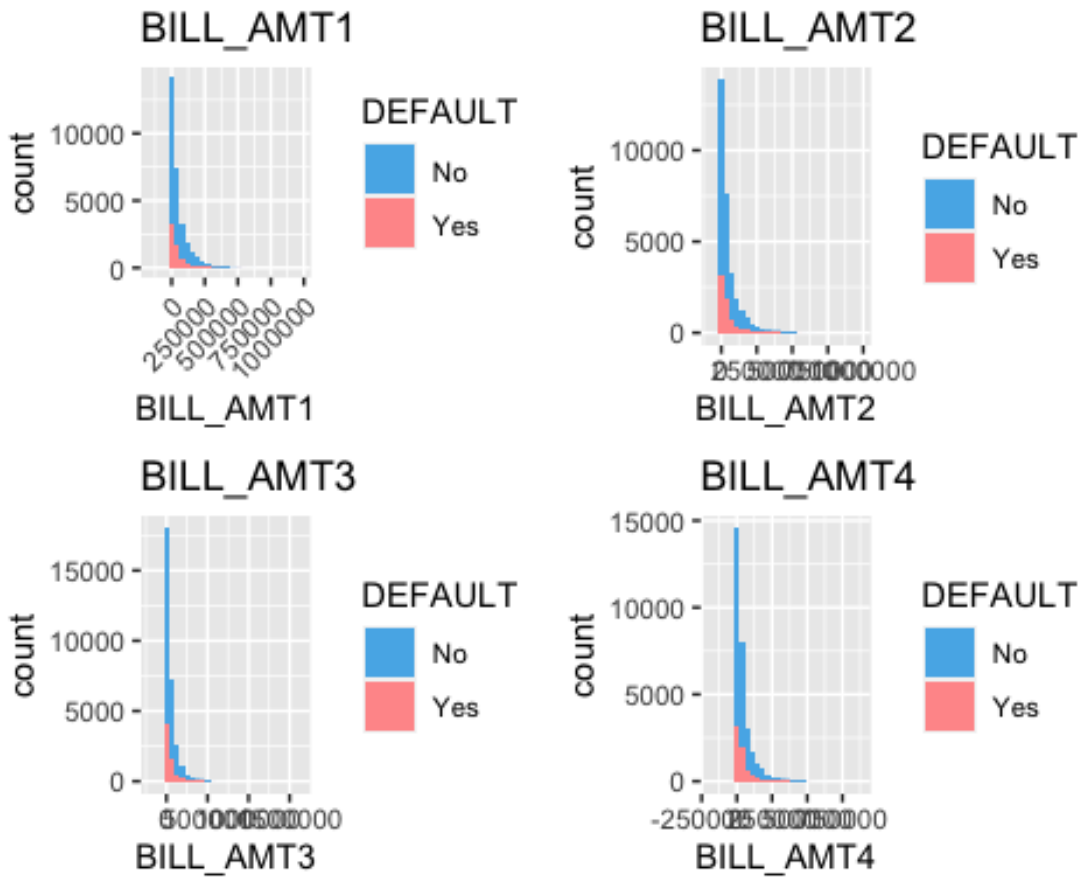
```
graph4 <- ggplot(data=df, aes(x=BILL_AMT4,fill=DEFAULT)) + geom_histogram() +
  labs(title = "BILL_AMT4", x ="BILL_AMT4",fill = "DEFAULT") +
   scale_fill_manual(values=c("#56B4E9", "#FF9999"))
  theme(axis.text.x = element_text(angle = 45,hjust=1))
```

```
## List of 1
##  $ axis.text.x:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : num 45
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
```

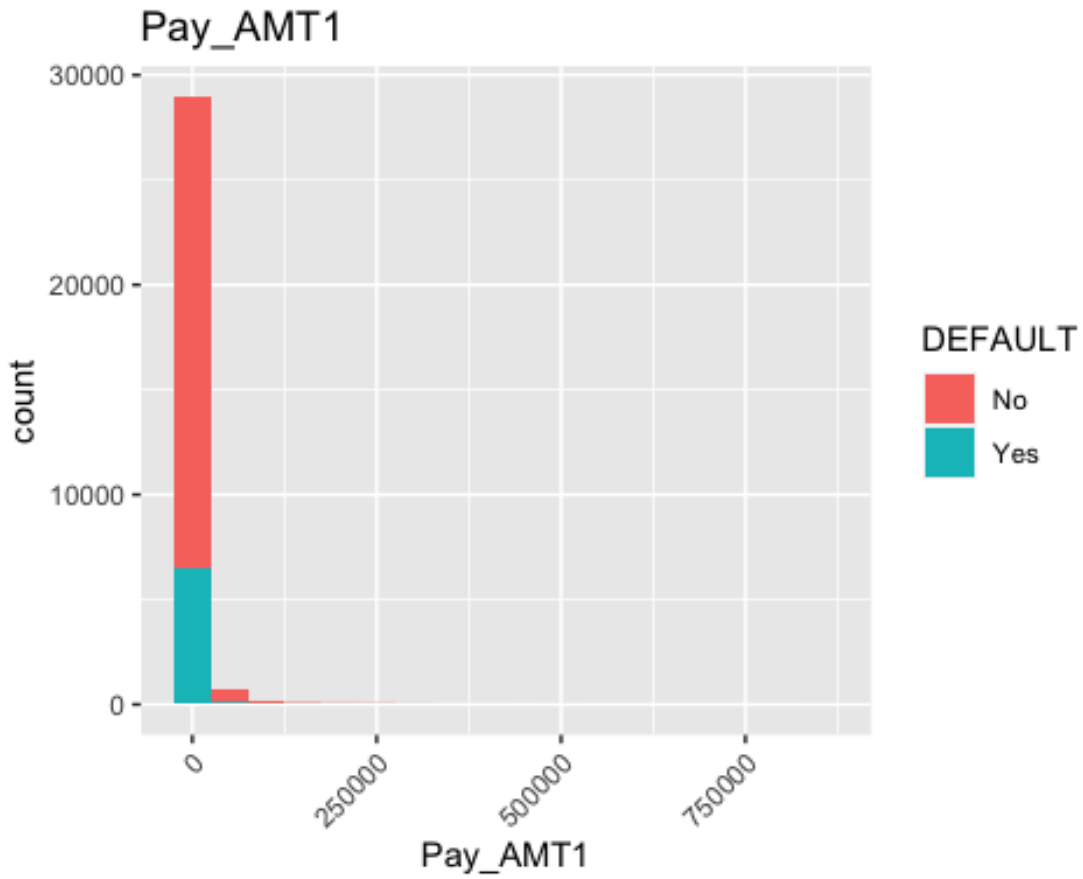
```
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

```
grid.arrange(graph1,graph2,graph3,graph4, ncol=2)
```

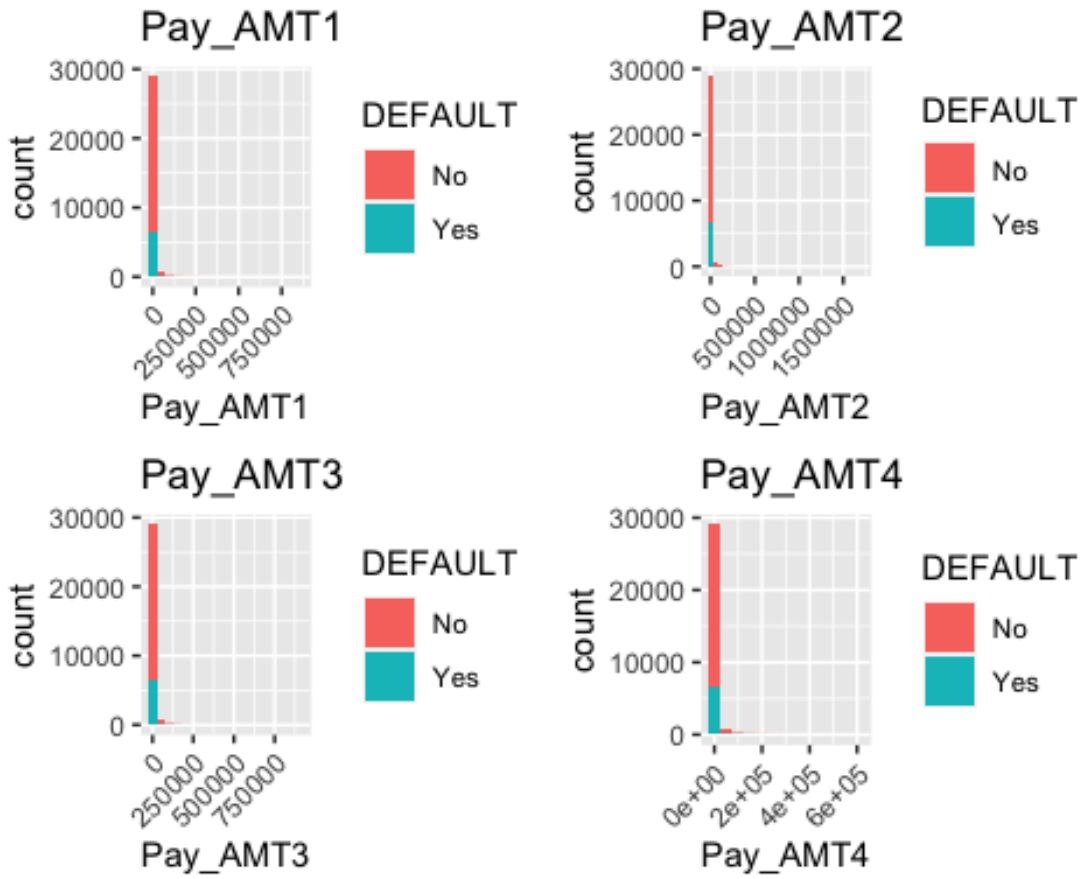
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
graph5 <- ggplot(data=df, aes(x=PAY_AMT1,fill=DEFAULT)) +
geom_histogram(binwidth = 50000) +
  labs(title = "Pay_AMT1", x ="Pay_AMT1",fill = "DEFAULT") +
  theme(axis.text.x = element_text(angle = 45,hjust=1))
graph5
```




```
graph6 <- ggplot(data=df, aes(x=PAY_AMT2,fill=DEFAULT)) +
geom_histogram(binwidth =  50000) +
  labs(title = "Pay_AMT2", x ="Pay_AMT2",fill = "DEFAULT") +
  theme(axis.text.x = element_text(angle = 45,hjust=1))

graph7 <- ggplot(data=df, aes(x=PAY_AMT3,fill=DEFAULT)) +
geom_histogram(binwidth =  50000) +
  labs(title = "Pay_AMT3", x ="Pay_AMT3",fill = "DEFAULT") +
  theme(axis.text.x = element_text(angle = 45,hjust=1))

graph8 <- ggplot(data=df, aes(x=PAY_AMT4,fill=DEFAULT)) +
geom_histogram(binwidth =  50000) +
  labs(title = "Pay_AMT4", x ="Pay_AMT4",fill = "DEFAULT") +
  theme(axis.text.x = element_text(angle = 45,hjust=1))

grid.arrange(graph5,graph6,graph7,graph8,ncol=2)
```




```
graph1 <- ggplot(data=df, aes(x=PAY_0,fill=DEFAULT)) + geom_histogram() +
  labs(title = "PAY_0", x ="PAY_0",fill = "DEFAULT") +
  scale_fill_manual(values=c("#56B4E9", "#FF9999")) +
  theme(axis.text.x = element_text(angle = 45,hjust=1))

graph2 <- ggplot(data=df, aes(x=PAY_2,fill=DEFAULT)) + geom_histogram() +
  labs(title = "PAY_2", x ="PAY_2",fill = "DEFAULT") +
   scale_fill_manual(values=c("#56B4E9", "#FF9999"))
  theme(axis.text.x = element_text(angle = 45,hjust=1))

## List of 1
##  $ axis.text.x:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : num 45
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
```

```
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

```
graph3 <- ggplot(data=df, aes(x=PAY_3,fill=DEFAULT)) + geom_histogram() +
  labs(title = "PAY_3", x ="PAY_3",fill = "DEFAULT") +
   scale_fill_manual(values=c("#56B4E9", "#FF9999"))
  theme(axis.text.x = element_text(angle = 45,hjust=1))
```

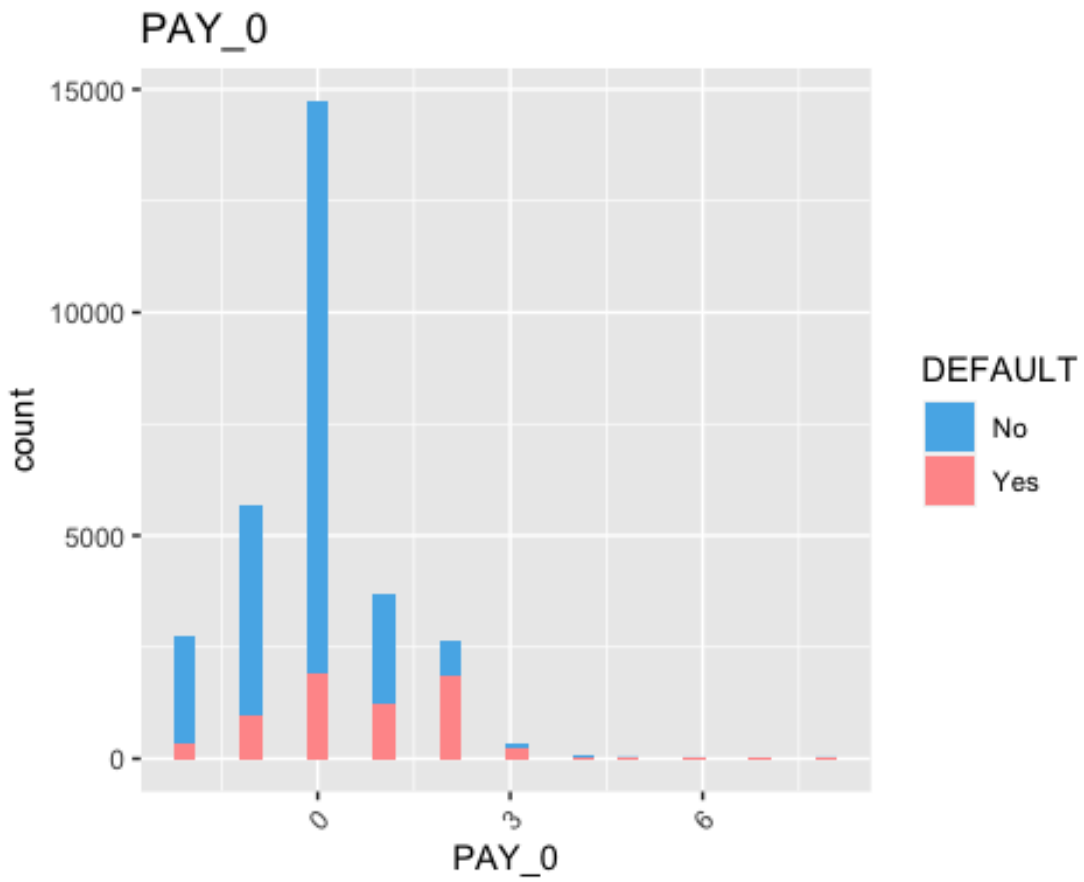
```
## List of 1
##  $ axis.text.x:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : num 45
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

```
graph4 <- ggplot(data=df, aes(x=PAY_4,fill=DEFAULT)) + geom_histogram() +
  labs(title = "PAY_4", x ="PAY_4",fill = "DEFAULT") +
   scale_fill_manual(values=c("#56B4E9", "#FF9999"))
  theme(axis.text.x = element_text(angle = 45,hjust=1))
```

```
## List of 1
##  $ axis.text.x:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : num 45
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

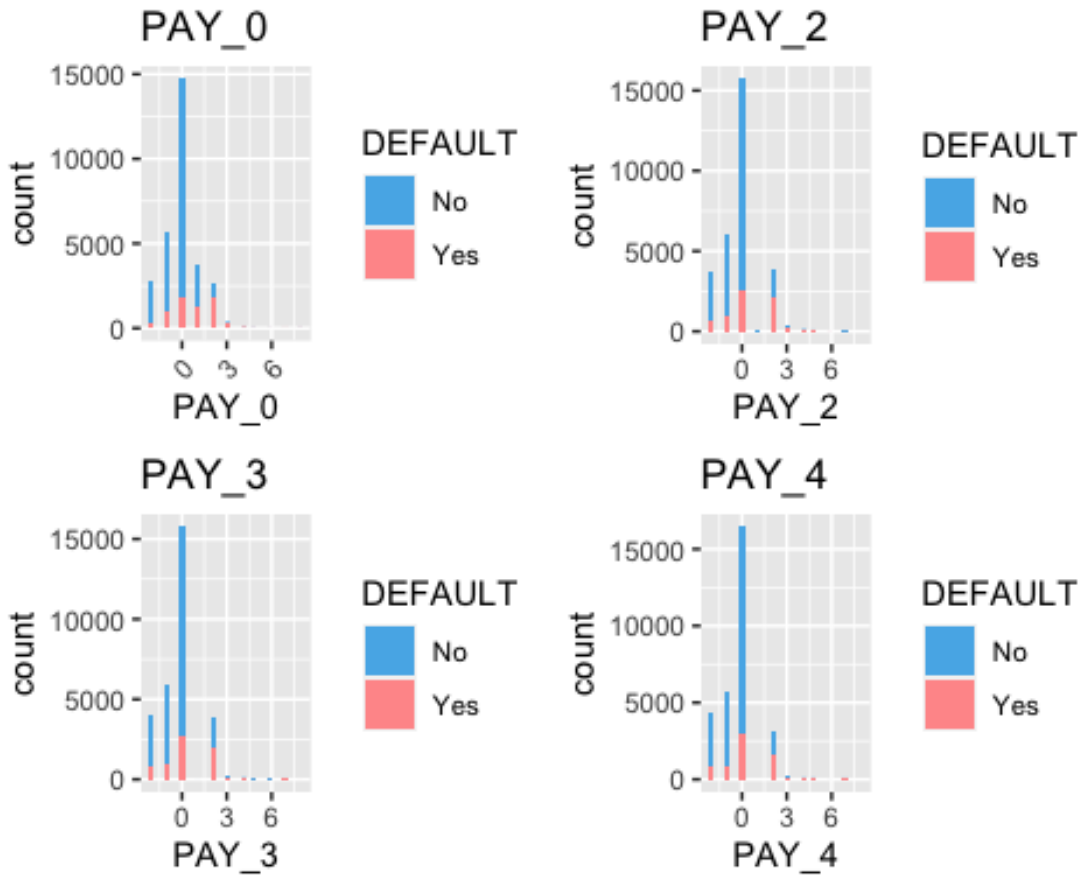
```
graph1
```

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

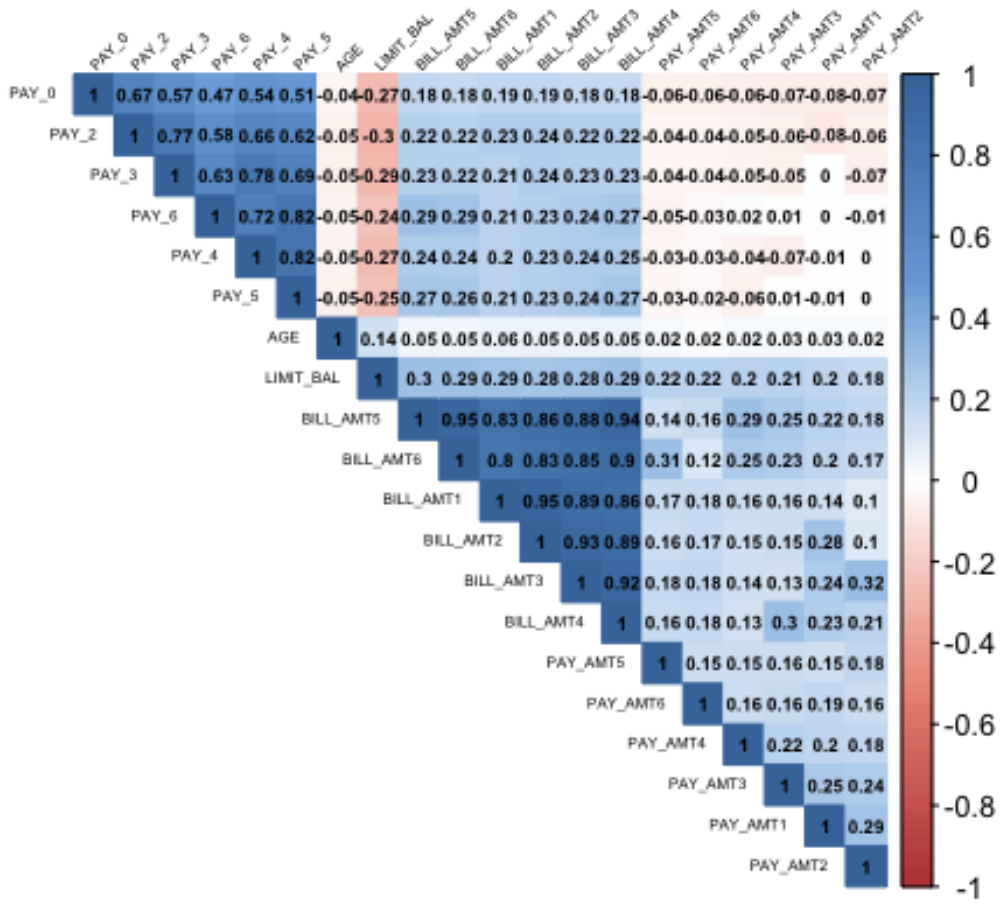



```
grid.arrange(graph1,graph2,graph3,graph4, ncol=2)
```

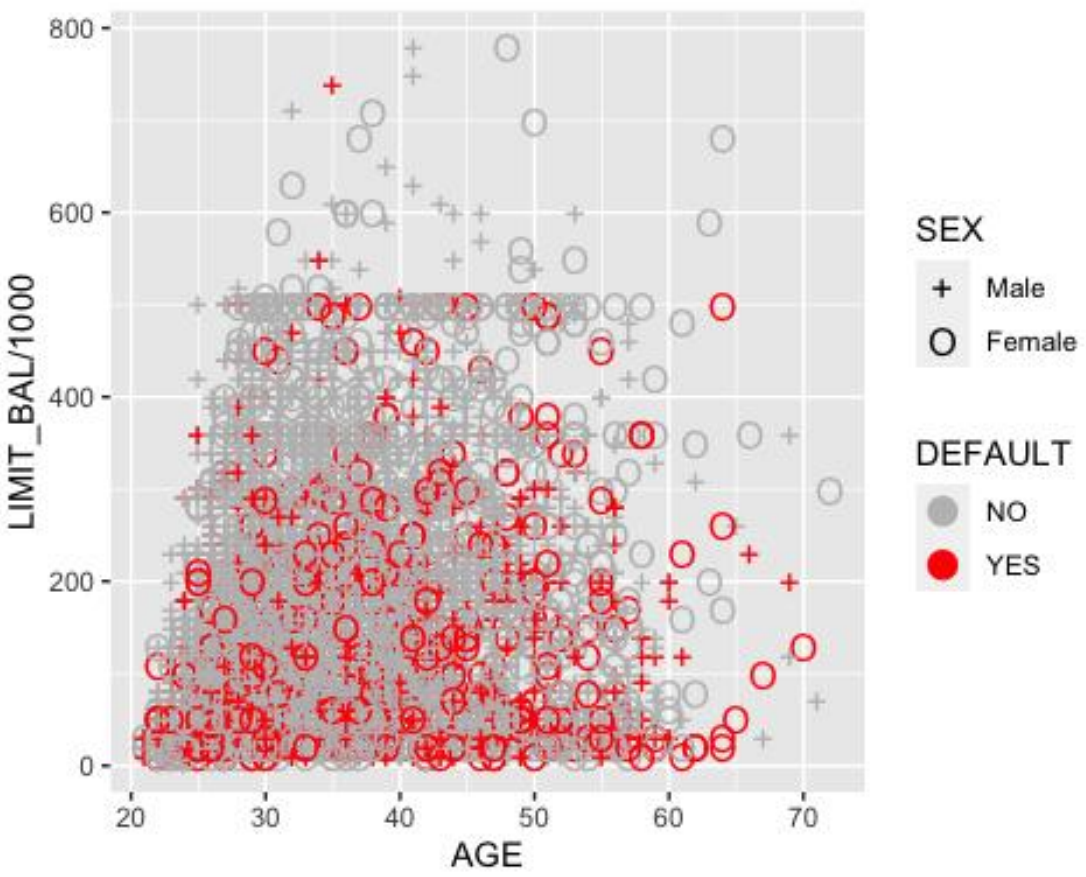
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



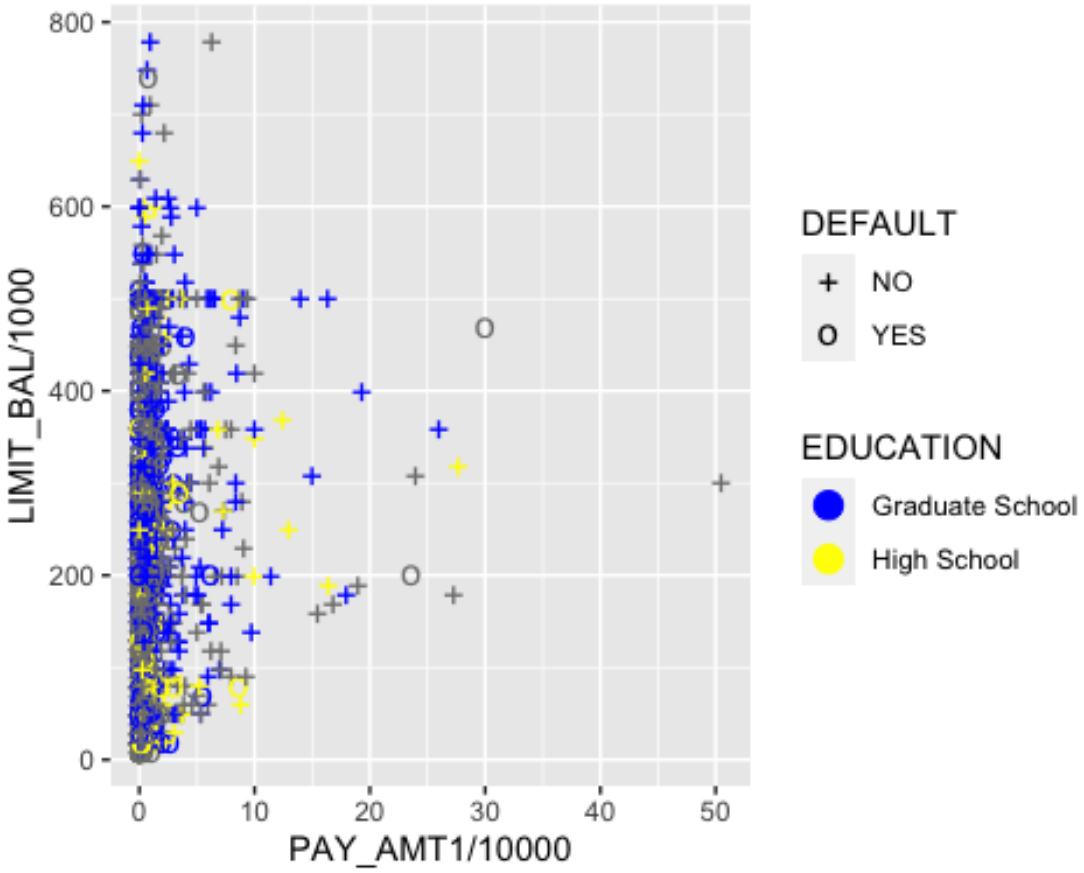

```r
# define numeric variable
df_quant <- df %>%
  select_if(is.numeric)

# install.packages("corrplot")
library(corrplot)

#Find the correlation of the dataset
corplotdf <- cor(df_quant, method = "pearson")
col_gd <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD",
"#4477AA"))

corrplot(corplotdf, method = "color", col = col_gd(200),
         type = "upper", order = "hclust",
         addCoef.col = "Black",
         tl.col = "black", tl.srt = 45, number.cex = 0.5,tl.cex = 0.4)
```



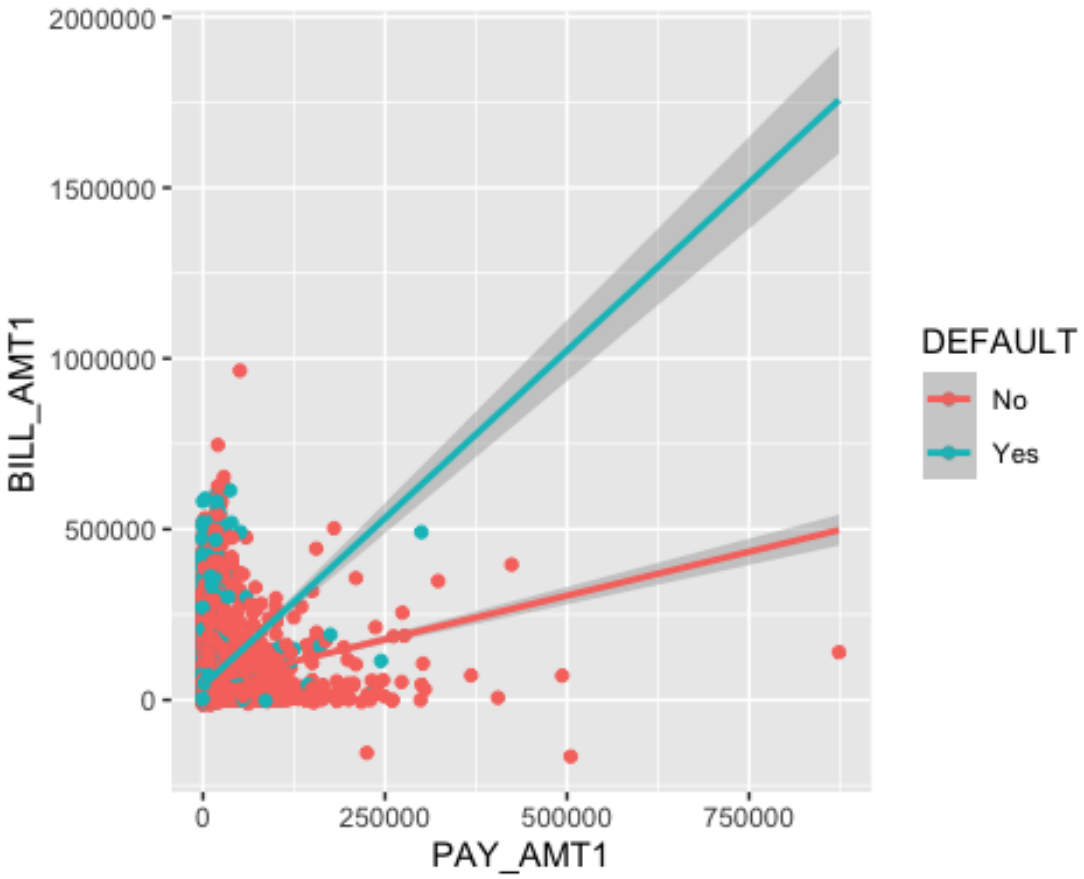

```
# install.packages("ggthemes")
library(ggthemes)

library(dplyr)
df_sampled <- df %>% sample_n(5000)

ggplot(data = df_sampled,aes(x=AGE,y
=LIMIT_BAL/1000,group=DEFAULT,color=DEFAULT,
                             shape=SEX))+
  scale_shape_manual(values=c('Male' = "O", 'Female' = "+"),labels=c("Male",
"Female"))+
  scale_color_manual(values=c('No' = "grey", 'Yes' = "red"),labels=c("NO",
"YES"))+
  geom_point(size=4)
```




```
ggplot(data = df_sampled,aes(x=PAY_AMT1/10000,y
=LIMIT_BAL/1000,group=DEFAULT,color=EDUCATION,
                             shape=DEFAULT))+
  scale_shape_manual(values=c('No' = "+", 'Yes' = "o"),labels=c("NO",
"YES"))+
  scale_color_manual(values=c('Graduate School' = "blue",'High
School'="yellow"),labels=c("Graduate School","High School"))+
  geom_point(size=4)
```




```
p1 <- ggplot(df, aes(x = PAY_AMT1, y = BILL_AMT1, col = DEFAULT)) +
  geom_point()+ geom_smooth(method=lm,fullrange=TRUE)
p1

## `geom_smooth()` using formula = 'y ~ x'
```



#Preprocessed

Convert datatype to `factor`

Let's convert some int variables into factors, which indicate categorical variables in R.

```
df<-df%>%
   mutate(
          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)
          )
```

##Hot-one coding

```
# One-Hot Encode Categorical Variables (except 'Pay_n', as they indicate
ordinal values)
# `fullRank = T` == `dropfirst=T`(in python), this is to prevent
multicollinearity
dmy <- dummyVars(" ~ SEX + EDUCATION + MARRIAGE", data = df, fullRank = T)
df_transformed <- data.frame(predict(dmy, newdata = df))
```

```
glimpse(df_transformed)
```

```
## Rows: 30,000
## Columns: 7
## $ SEX.Male              <dbl> 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1,
0, …
## $ EDUCATION.High.School <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
1, …
## $ EDUCATION.Other       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, …
## $ EDUCATION.University  <dbl> 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0,
0, …
## $ EDUCATION.Unknown     <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, …
## $ MARRIAGE.Other        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, …
## $ MARRIAGE.Single       <dbl> 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
0, …
```

```
# Drop SEX, EDUCATION, MARRIAGE from df
df_encoded <- df[, -c(2, 3, 4)]

# Combine df with encoded columns
df_encoded <- cbind(df_encoded, df_transformed)

df_encoded <- df_encoded%>%
   mutate(
          SEX.Male =as.factor(SEX.Male),
          EDUCATION.High.School=as.factor(EDUCATION.High.School),
          EDUCATION.Other=as.factor(EDUCATION.Other),
          EDUCATION.University =as.factor(EDUCATION.University),
          EDUCATION.Unknown =as.factor(EDUCATION.Unknown),
          MARRIAGE.Other=as.factor(MARRIAGE.Other),
          MARRIAGE.Single=as.factor(MARRIAGE.Single)
          )

glimpse(df_encoded)
```

```
## Rows: 30,000
## Columns: 28
## $ LIMIT_BAL             <dbl> 20000, 120000, 90000, 50000, 50000, 50000,
50000…
## $ AGE                   <int> 24, 26, 34, 37, 57, 37, 29, 23, 28, 35, 34,
51, …
## $ PAY_0                 <fct> 2, -1, 0, 0, -1, 0, 0, 0, 0, -2, 0, -1, -1,
1, 0…
## $ PAY_2                 <fct> 2, 2, 0, 0, 0, 0, 0, -1, 0, -2, 0, -1, 0, 2,
0, …
## $ PAY_3                 <fct> -1, 0, 0, 0, -1, 0, 0, -1, 2, -2, 2, -1, -1,
```

```
2, …
## $ PAY_4                  <fct> -1, 0, 0, 0, 0, 0, 0, 0, 0, -2, 0, -1, -1,
0, 0,…
## $ PAY_5                  <fct> -2, 0, 0, 0, 0, 0, 0, 0, 0, -1, 0, -1, -1,
0, 0,…
## $ PAY_6                  <fct> -2, 2, 0, 0, 0, 0, 0, -1, 0, -1, -1, 2, -1,
2, 0…
## $ BILL_AMT1              <dbl> 3913, 2682, 29239, 46990, 8617, 64400,
367965, 1…
## $ BILL_AMT2              <dbl> 3102, 1725, 14027, 48233, 5670, 57069,
412023, 3…
## $ BILL_AMT3              <dbl> 689, 2682, 13559, 49291, 35835, 57608,
445007, 6…
## $ BILL_AMT4              <dbl> 0, 3272, 14331, 28314, 20940, 19394, 542653,
221…
## $ BILL_AMT5              <dbl> 0, 3455, 14948, 28959, 19146, 19619, 483003,
-15…
## $ BILL_AMT6              <dbl> 0, 3261, 15549, 29547, 19131, 20024, 473944,
567…
## $ PAY_AMT1               <dbl> 0, 0, 1518, 2000, 2000, 2500, 55000, 380,
3329, …
## $ PAY_AMT2               <dbl> 689, 1000, 1500, 2019, 36681, 1815, 40000,
601, …
## $ PAY_AMT3               <dbl> 0, 1000, 1000, 1200, 10000, 657, 38000, 0,
432, …
## $ PAY_AMT4               <dbl> 0, 1000, 1000, 1100, 9000, 1000, 20239, 581,
100…
## $ PAY_AMT5               <dbl> 0, 0, 1000, 1069, 689, 1000, 13750, 1687,
1000, …
## $ PAY_AMT6               <dbl> 0, 2000, 5000, 1000, 679, 800, 13770, 1542,
1000…
## $ DEFAULT                <fct> Yes, Yes, No, No, No, No, No, No, No, No,
No, No…
## $ SEX.Male               <fct> 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1,
0, …
## $ EDUCATION.High.School <fct> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0,
1, …
## $ EDUCATION.Other       <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, …
## $ EDUCATION.University  <fct> 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0,
0, …
## $ EDUCATION.Unknown     <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, …
## $ MARRIAGE.Other        <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, …
## $ MARRIAGE.Single       <fct> 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1,
0, …
```

Skewness Correction

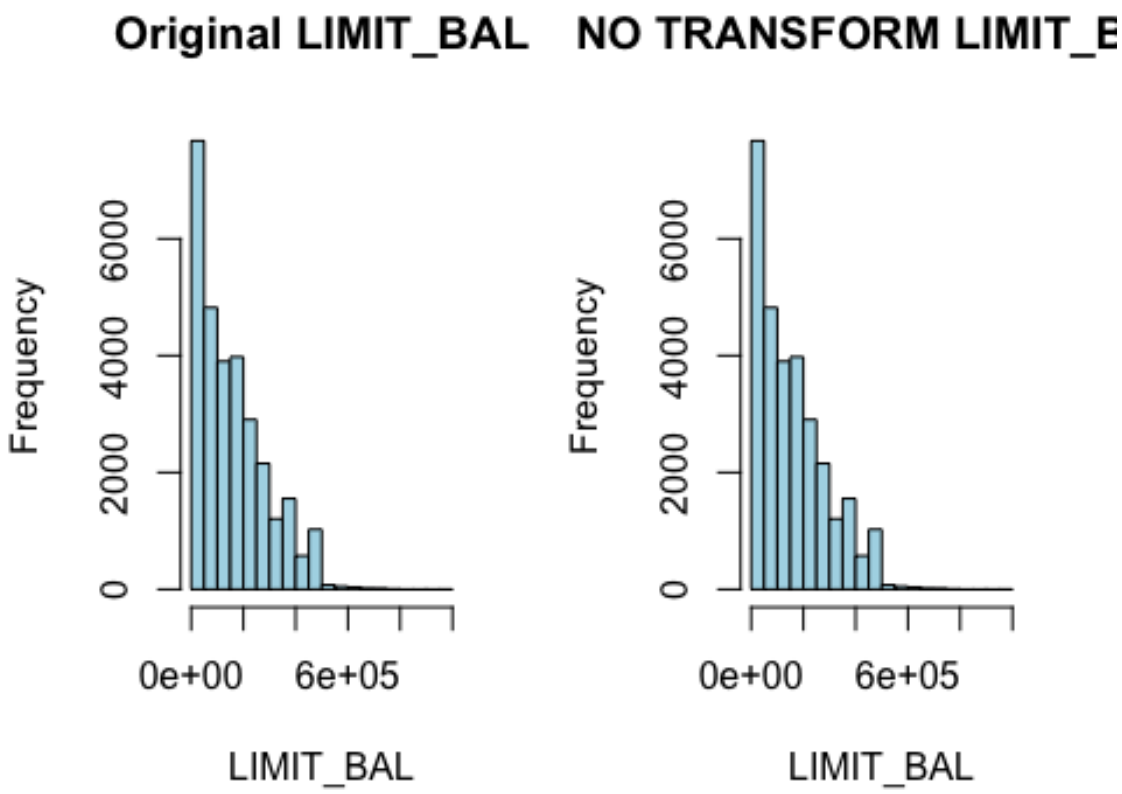
```
# Split features into df_quant (quant features), df_cat (qual features)
df_quant <- df_encoded %>%
  select_if(is.numeric)


df_cat <- df_encoded %>%
  select_if(is.factor)
```

Skewness Correction

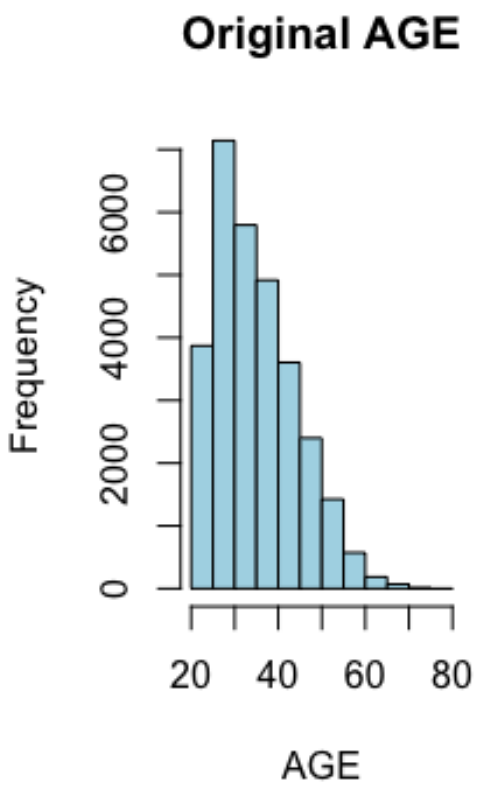
```
# Split features into df_quant (quant features), df_cat (qual features)
df_quant <- df_encoded %>%
  select_if(is.numeric)


df_cat <- df_encoded %>%
  select_if(is.factor)

# log/exponential transformation for positive/negative skewness
# Define skewness correction function
skew_autotransform <- function(DF, include=NULL, exclude=NULL, plot=FALSE,
threshold=1)
  {
  # Get list of column names that should be processed based on input
parameters
  if (is.null(include) & is.null(exclude)) {
    colnames <- names(DF)
  } else if (!is.null(include)) {
    colnames <- include
  } else if (!is.null(exclude)) {
    colnames <- setdiff(names(DF), exclude)
  } else {
    print('No columns to process!')
  }

  # Helper function that checks if all values are positive
  make_positive <- function(series) {
    minimum <- min(series)
    # If minimum is negative, offset all values by a constant to move all
values to positive territory
    if (minimum <= 0) {
      series <- series + abs(minimum) + 5 # offset with a large number
tailored for this dataset
    }
    return(series)
  }

  # Go through desired columns in DataFrame
  for (col in colnames) {
```

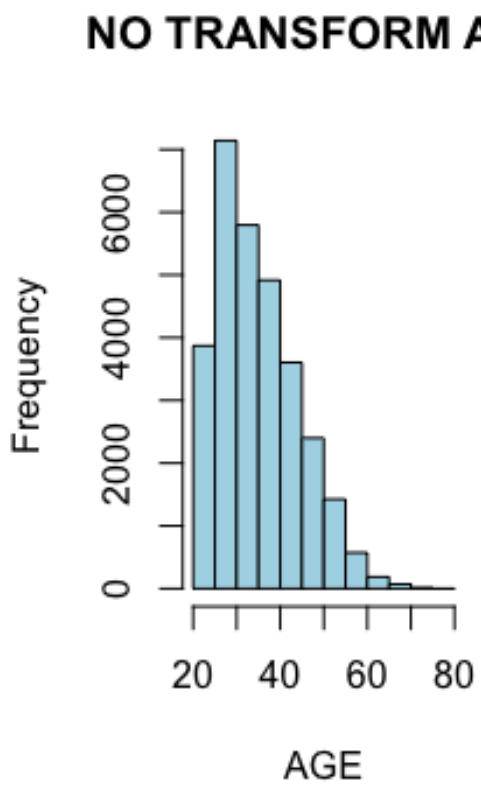
```
    # Get column skewness
    skew <- skewness(DF[[col]])
    transformed <- TRUE

    if (plot) {
      # Prep the plot of original data
      par(mfrow=c(1, 2))
      hist(DF[[col]], col="lightblue", main=paste0("Original ", col),
xlab=col)
    }

    # If skewness is larger than threshold and positively skewed; If yes,
apply appropriate transformation
    if (abs(skew) > threshold & skew > 0) {
      skewType <- 'positive'
      # Make sure all values are positive
      DF[[col]] <- make_positive(DF[[col]])

      # Apply log transformation
      DF[[col]] <- log(DF[[col]])
      skew_new <- skewness(DF[[col]])
    }
    else if (abs(skew) > threshold & skew < 0) {
      skewType <- 'negative'
      # Make sure all values are positive
      DF[[col]] <- make_positive(DF[[col]])

        # Apply exp transformation
      DF[[col]] <- DF[[col]]^10

      skew_new <- skewness(DF[[col]])

    } else {
      # Flag if no transformation was performed
      transformed <- FALSE
      skew_new <- skew
    }

    # Compare before and after if plot is True
    if (plot) {
      cat('\n ----------------------------------------------------')
      if (transformed) {
        cat('\n', col, 'had', skewType, 'skewness of', skew)
        cat('\n Transformation yielded skewness of', skew_new)
        hist(DF[[col]], col="salmon", main=paste0("Transformed ", col),
xlab=col)
      } else {
        cat('\n NO TRANSFORMATION APPLIED FOR', col, '. Skewness =', skew)
        hist(DF[[col]], col="lightblue", main=paste0("NO TRANSFORM ", col),
```

```
xlab=col)
      }
    }
  }

  return(DF)
}

# Excluding "BILL_AMT6", as corrected to be much higher value, set threshold
to be -1 to 1
df_quant_skewed <- skew_autotransform(df_quant, exclude="BILL_AMT6", plot =
T, threshold = 1)

##
##  ------------------------------------------------------
##  NO TRANSFORMATION APPLIED FOR LIMIT_BAL . Skewness = 0.9928173
```

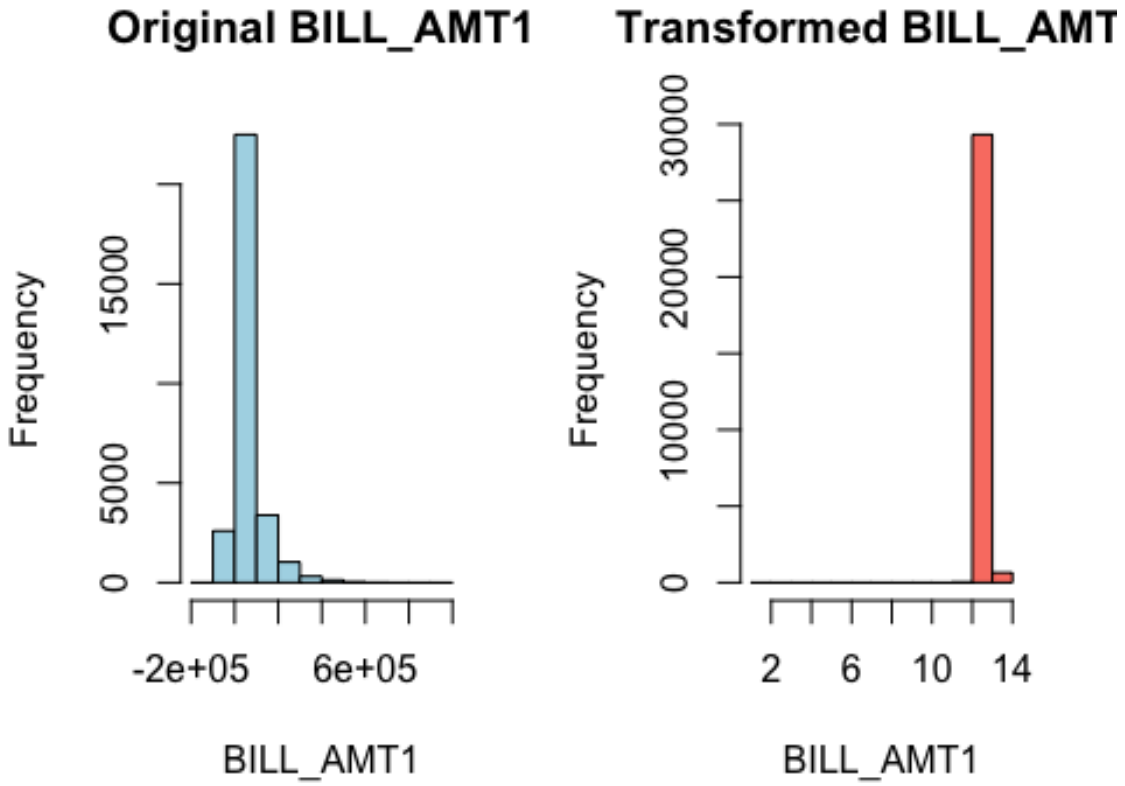



```
##
##  ------------------------------------------------------
##  NO TRANSFORMATION APPLIED FOR AGE . Skewness = 0.7322093
```

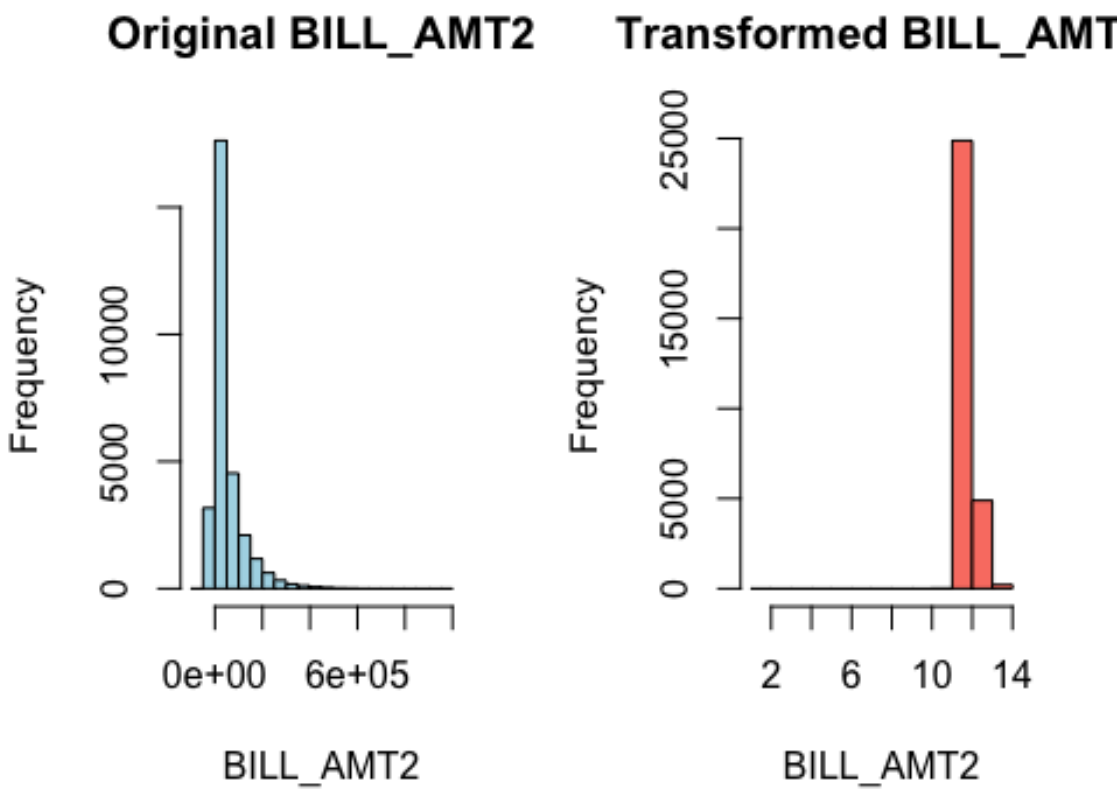




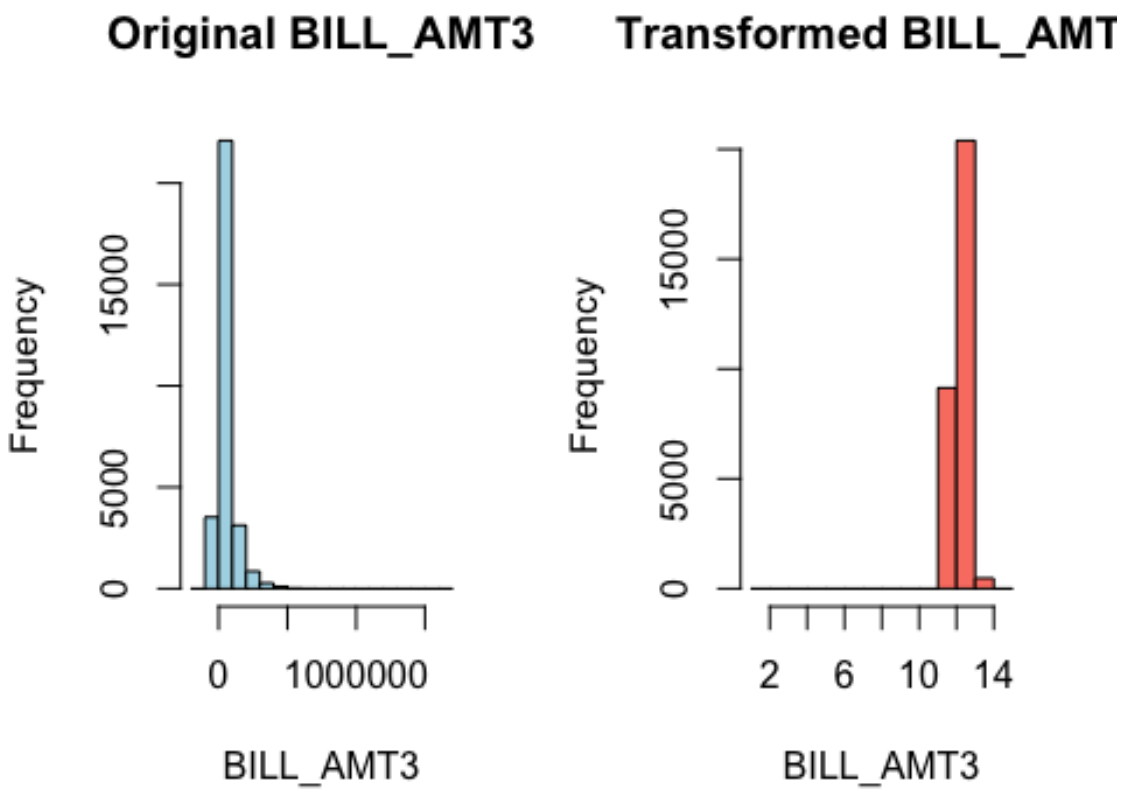

```
##
##  ------------------------------------------------------
##  BILL_AMT1 had positive skewness of 2.663728
##  Transformation yielded skewness of -0.5006624
```



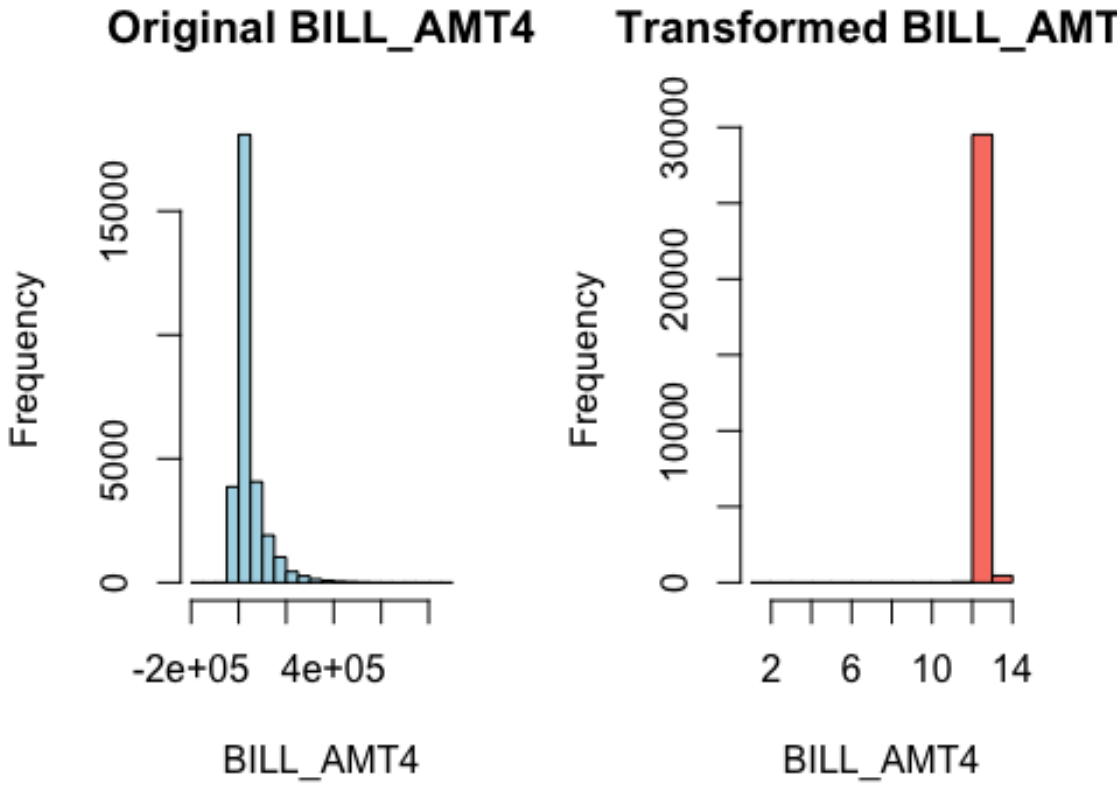

```
##
##  ------------------------------------------------------
##  BILL_AMT2 had positive skewness of 2.705086
##  Transformation yielded skewness of 0.7948691
```



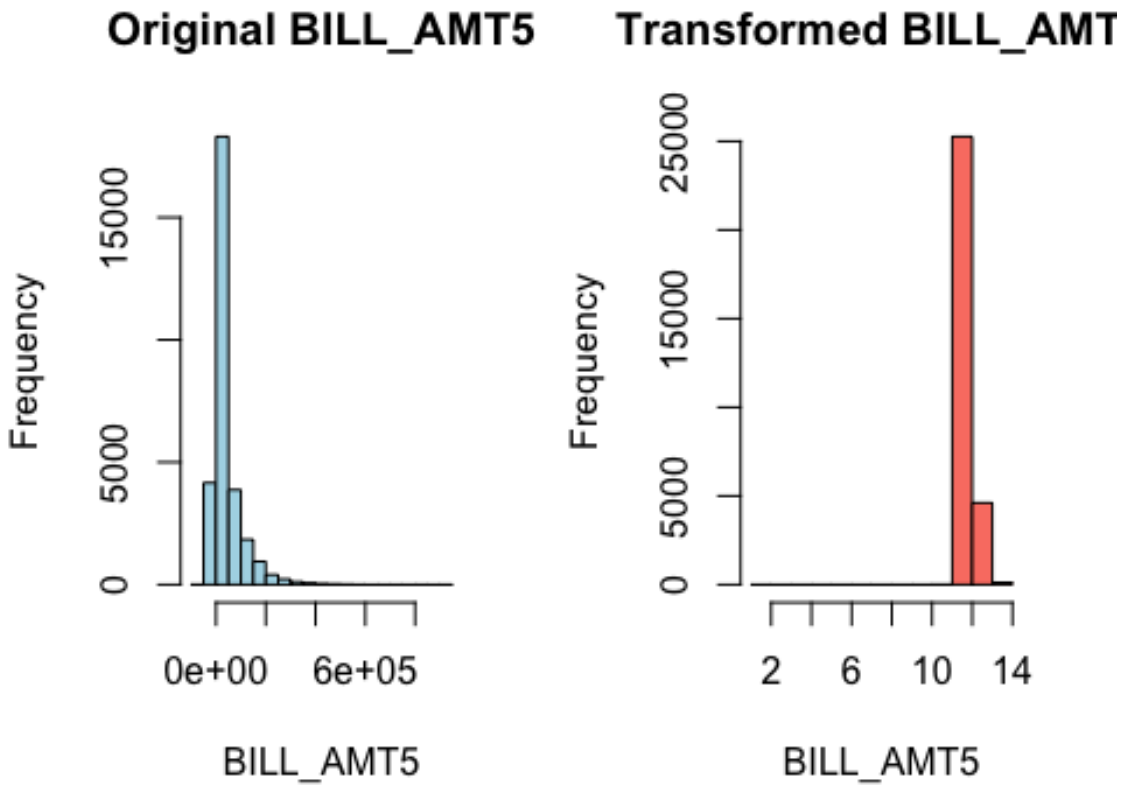

```
##
##  ------------------------------------------------------
##  BILL_AMT3 had positive skewness of 3.087676
##  Transformation yielded skewness of -0.4483056
```



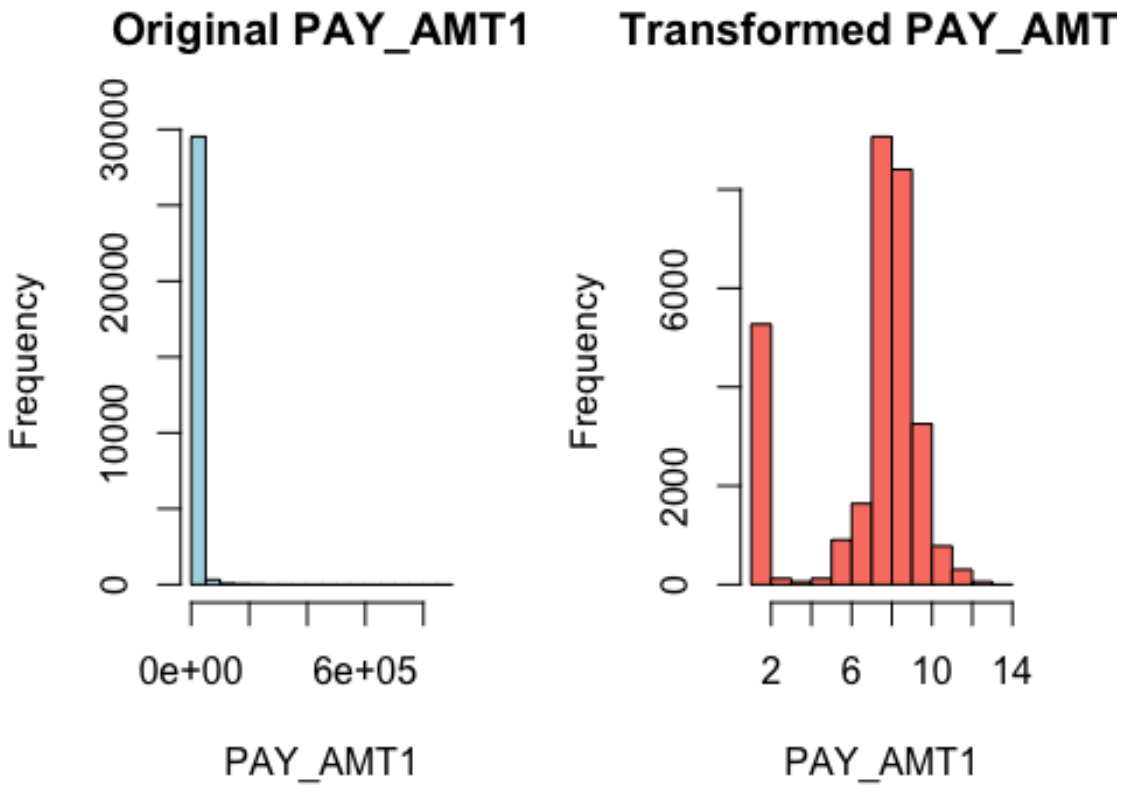

```
##
##  ------------------------------------------------------
##  BILL_AMT4 had positive skewness of 2.821824
##  Transformation yielded skewness of -1.087994
```



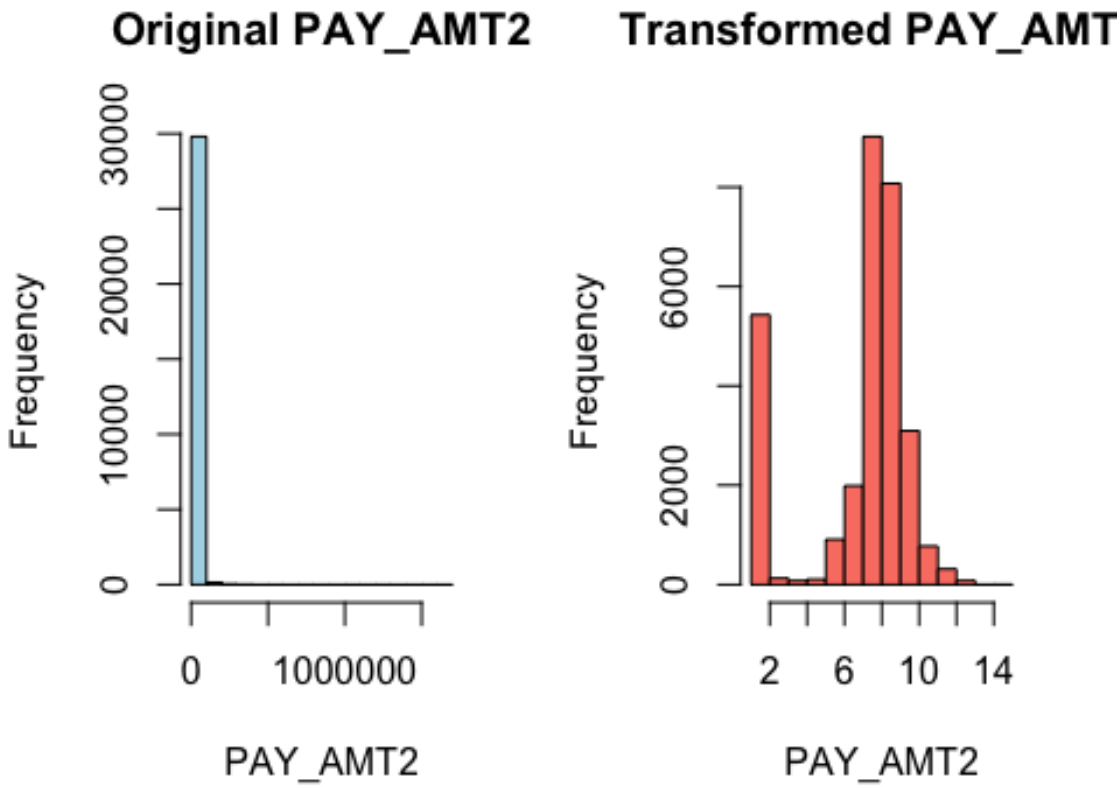

```
##
##  ------------------------------------------------------
##  BILL_AMT5 had positive skewness of 2.876236
##  Transformation yielded skewness of 0.7495502
```



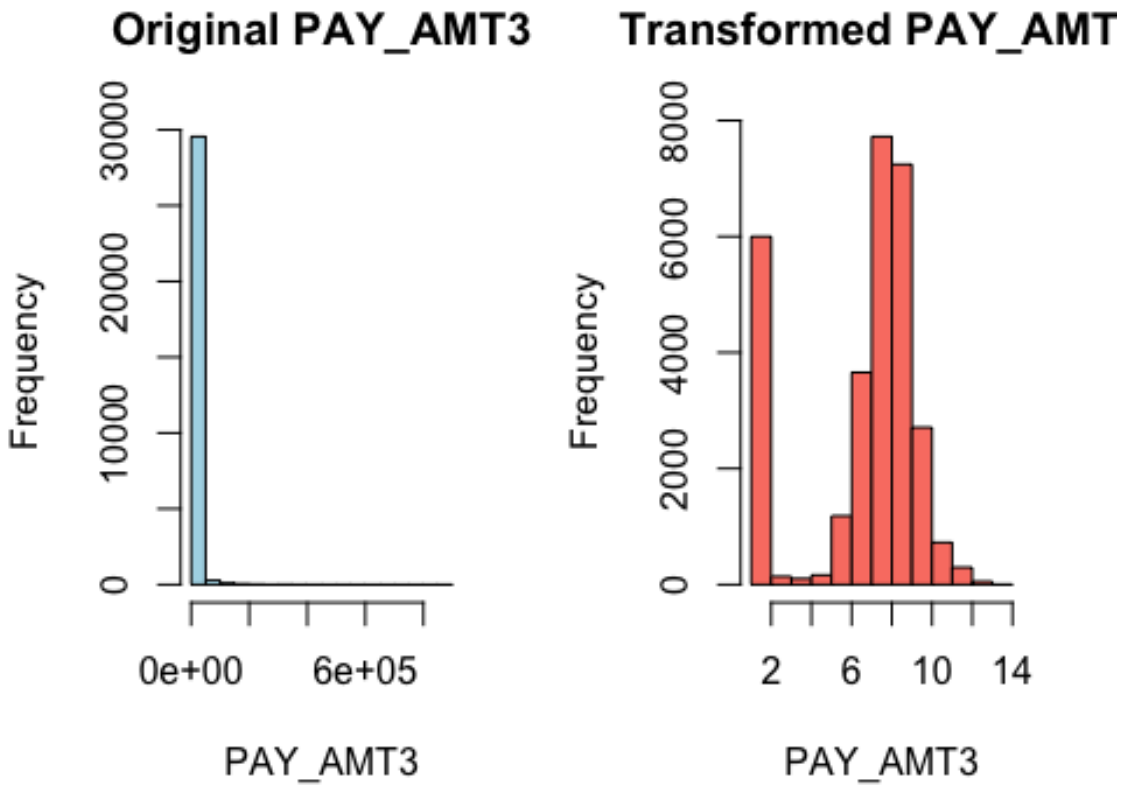

```
##
##  ------------------------------------------------------
##  PAY_AMT1 had positive skewness of 14.66763
##  Transformation yielded skewness of -1.119563
```



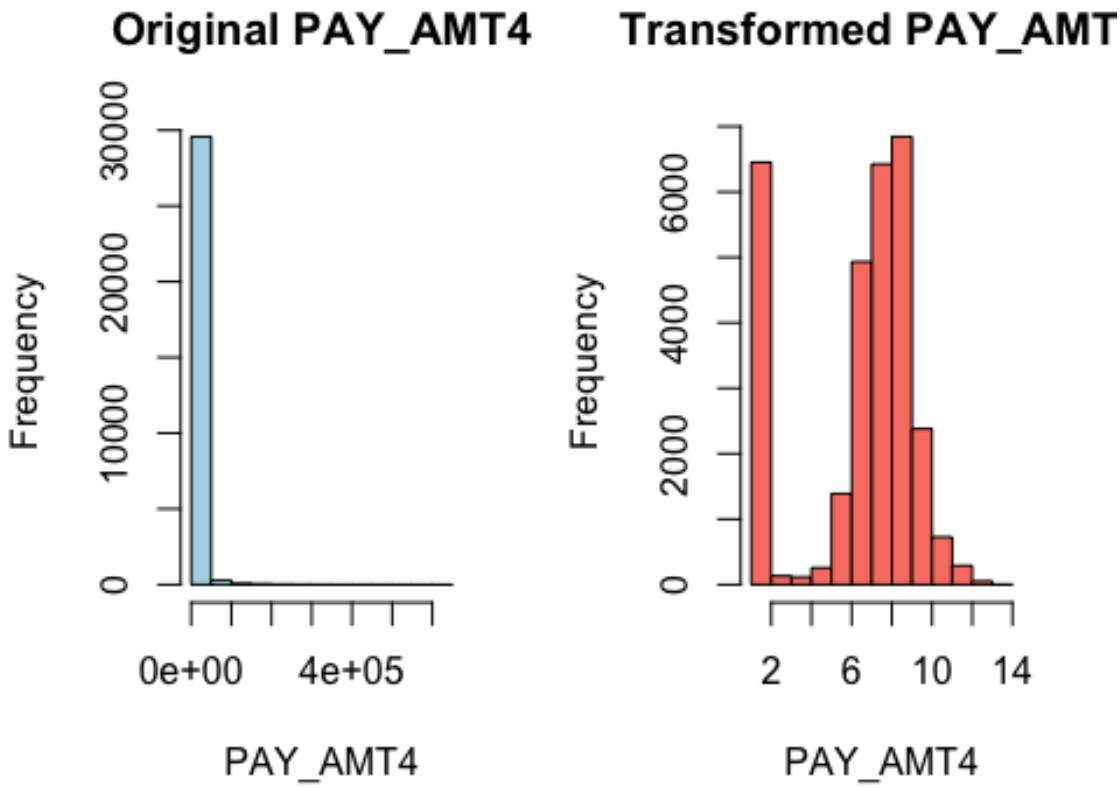

```
##
##  ------------------------------------------------------
##  PAY_AMT2 had positive skewness of 30.45229
##  Transformation yielded skewness of -1.063536
```



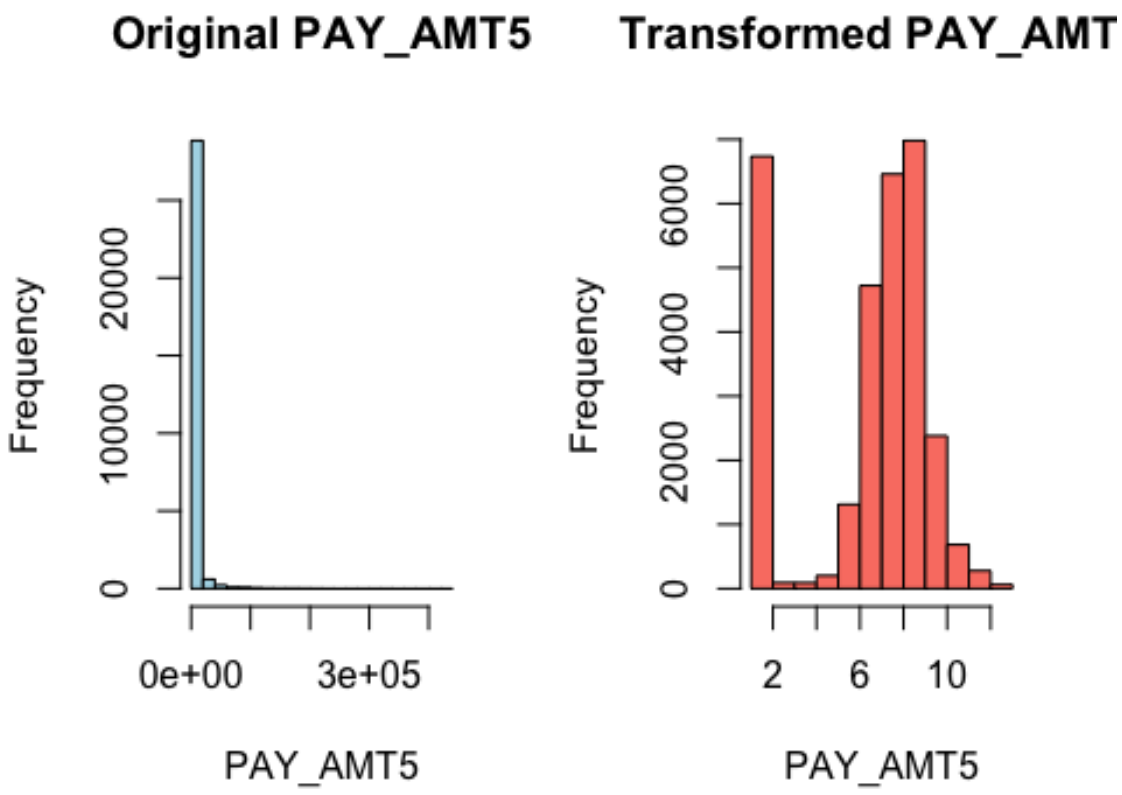

```
##
##  ------------------------------------------------------
##  PAY_AMT3 had positive skewness of 17.21577
##  Transformation yielded skewness of -0.899868
```



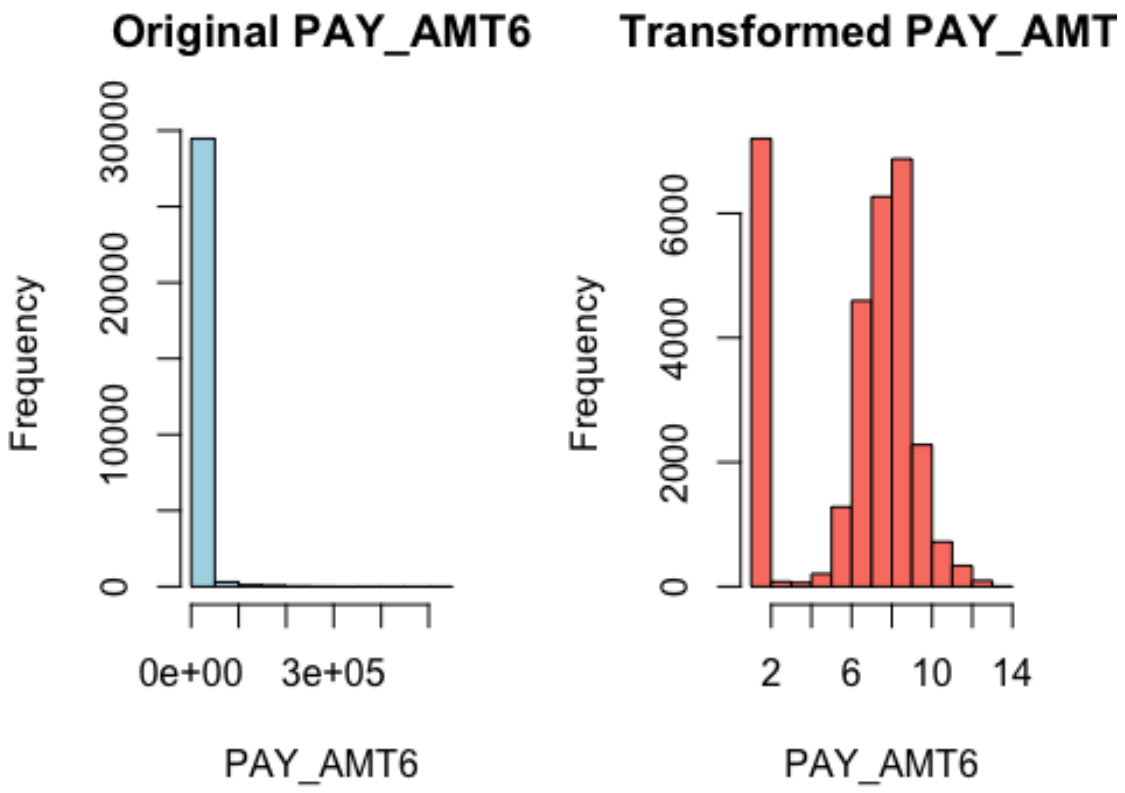

```
##
##  ------------------------------------------------------
##  PAY_AMT4 had positive skewness of 12.90434
##  Transformation yielded skewness of -0.7839622
```




```
##
##  ------------------------------------------------------
##  PAY_AMT5 had positive skewness of 11.12686
##  Transformation yielded skewness of -0.7681229
```




```
##
##  ------------------------------------------------------
##  PAY_AMT6 had positive skewness of 10.6402
##  Transformation yielded skewness of -0.6896096
```



Outlier Removal

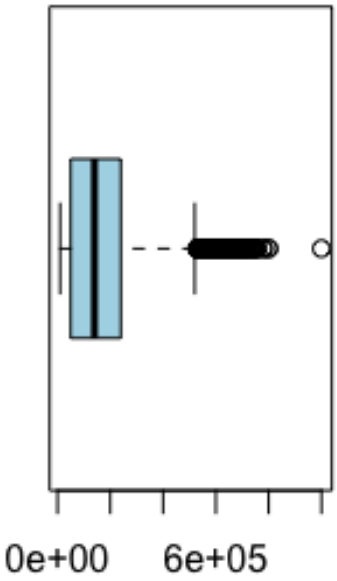
```
cols <- names(df_quant_skewed)

tukey_rule <- function(data, col){
  Q1 <- quantile(data[[col]], 0.25)
  Q3 <- quantile(data[[col]], 0.75)
  IQR <- Q3 - Q1
  upper_lim <- quantile(data[[col]], 0.5) + 2 * IQR
  lower_lim <- quantile(data[[col]], 0.5) - 2 * IQR
  outliers <- which(data[[col]] < lower_lim | data[[col]] >= upper_lim)
  return(outliers)
}

# Identify outliers
for (i in cols) {
  outliers_Tukey <- tukey_rule(df_quant_skewed, i)
  cat("Number of outliers in column", i, "based on Tukey's method:",
length(outliers_Tukey), "\n")
}

## Number of outliers in column LIMIT_BAL based on Tukey's method: 187
## Number of outliers in column AGE based on Tukey's method: 339
## Number of outliers in column BILL_AMT1 based on Tukey's method: 1739
```

```
## Number of outliers in column BILL_AMT2 based on Tukey's method: 886
## Number of outliers in column BILL_AMT3 based on Tukey's method: 1862
## Number of outliers in column BILL_AMT4 based on Tukey's method: 2043
## Number of outliers in column BILL_AMT5 based on Tukey's method: 1467
## Number of outliers in column BILL_AMT6 based on Tukey's method: 2982
## Number of outliers in column PAY_AMT1 based on Tukey's method: 5921
## Number of outliers in column PAY_AMT2 based on Tukey's method: 5958
## Number of outliers in column PAY_AMT3 based on Tukey's method: 6113
## Number of outliers in column PAY_AMT4 based on Tukey's method: 6479
## Number of outliers in column PAY_AMT5 based on Tukey's method: 6728
## Number of outliers in column PAY_AMT6 based on Tukey's method: 0

# Winsorize quant features

# Winsorizing a vector means that a predefined quantum of the smallest and/or
the largest values are replaced by less extreme values. Thereby the
substitute values are the most extreme retained values.
#
https://www.rdocumentation.org/packages/DescTools/versions/0.99.48/topics/Win
sorize

cat("Descriptive Statistics Before\n")

## Descriptive Statistics Before

summary(df_quant_skewed)

##    LIMIT_BAL            AGE          BILL_AMT1        BILL_AMT2
##  Min.   :  10000   Min.   :21.00   Min.   : 1.609   Min.   : 1.609
##  1st Qu.:  50000   1st Qu.:28.00   1st Qu.:12.039   1st Qu.:11.195
##  Median : 140000   Median :34.00   Median :12.144   Median :11.418
##  Mean   : 167484   Mean   :35.49   Mean   :12.245   Mean   :11.569
##  3rd Qu.: 240000   3rd Qu.:41.00   3rd Qu.:12.357   3rd Qu.:11.804
##  Max.   :1000000   Max.   :79.00   Max.   :13.938   Max.   :13.868
##    BILL_AMT3        BILL_AMT4        BILL_AMT5        BILL_AMT6
##  Min.   : 1.609   Min.   : 1.609   Min.   : 1.609   Min.   :-339603
##  1st Qu.:11.983   1st Qu.:12.057   1st Qu.:11.328   1st Qu.:   1256
##  Median :12.086   Median :12.150   Median :11.507   Median :  17071
##  Mean   :12.186   Mean   :12.237   Mean   :11.627   Mean   :  38872
##  3rd Qu.:12.290   3rd Qu.:12.322   3rd Qu.:11.787   3rd Qu.:  49198
##  Max.   :14.415   Max.   :13.875   Max.   :13.824   Max.   : 961664
##     PAY_AMT1         PAY_AMT2         PAY_AMT3         PAY_AMT4
##  Min.   : 1.609   Min.   : 1.609   Min.   : 1.609   Min.   : 1.609
##  1st Qu.: 6.913   1st Qu.: 6.731   1st Qu.: 5.979   1st Qu.: 5.707
##  Median : 7.652   Median : 7.608   Median : 7.498   Median : 7.317
##  Mean   : 6.916   Mean   : 6.857   Mean   : 6.609   Mean   : 6.428
##  3rd Qu.: 8.519   3rd Qu.: 8.518   3rd Qu.: 8.414   3rd Qu.: 8.299
##  Max.   :13.680   Max.   :14.337   Max.   :13.706   Max.   :13.339
##     PAY_AMT5         PAY_AMT6
##  Min.   : 1.609   Min.   : 1.609
##  1st Qu.: 5.551   1st Qu.: 4.810
```

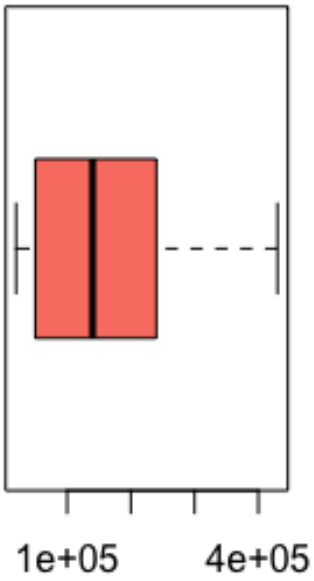
```
##  Median : 7.317   Median : 7.317
##  Mean   : 6.397   Mean   : 6.323
##  3rd Qu.: 8.303   3rd Qu.: 8.295
##  Max.   :12.963   Max.   :13.178

df_quant_winsorized <- df_quant_skewed

for (i in cols) {
  df_quant_winsorized[, i] <- Winsorize(df_quant_winsorized[, i], probs =
c(0.05, 0.95))
}

cat("Descriptive Statistics After\n")

## Descriptive Statistics After

summary(df_quant_winsorized)

##    LIMIT_BAL           AGE         BILL_AMT1       BILL_AMT2
##  Min.   : 20000   Min.   :23.0   Min.   :12.02   Min.   :11.15
##  1st Qu.: 50000   1st Qu.:28.0   1st Qu.:12.04   1st Qu.:11.20
##  Median :140000   Median :34.0   Median :12.14   Median :11.42
##  Mean   :164227   Mean   :35.3   Mean   :12.24   Mean   :11.56
##  3rd Qu.:240000   3rd Qu.:41.0   3rd Qu.:12.36   3rd Qu.:11.80
##  Max.   :430000   Max.   :53.0   Max.   :12.81   Max.   :12.49
##    BILL_AMT3       BILL_AMT4       BILL_AMT5       BILL_AMT6
##  Min.   :11.97   Min.   :12.04   Min.   :11.31   Min.   :     0
##  1st Qu.:11.98   1st Qu.:12.06   1st Qu.:11.33   1st Qu.:  1256
##  Median :12.09   Median :12.15   Median :11.51   Median : 17071
##  Mean   :12.18   Mean   :12.23   Mean   :11.62   Mean   : 35401
##  3rd Qu.:12.29   3rd Qu.:12.32   3rd Qu.:11.79   3rd Qu.: 49198
##  Max.   :12.75   Max.   :12.75   Max.   :12.42   Max.   :161912
##     PAY_AMT1        PAY_AMT2        PAY_AMT3        PAY_AMT4
##  Min.   :1.609   Min.   :1.609   Min.   :1.609   Min.   :1.609
##  1st Qu.:6.913   1st Qu.:6.731   1st Qu.:5.979   1st Qu.:5.707
##  Median :7.652   Median :7.608   Median :7.498   Median :7.317
##  Mean   :6.878   Mean   :6.818   Mean   :6.570   Mean   :6.386
##  3rd Qu.:8.519   3rd Qu.:8.518   3rd Qu.:8.414   3rd Qu.:8.299
##  Max.   :9.822   Max.   :9.853   Max.   :9.775   Max.   :9.682
##     PAY_AMT5        PAY_AMT6
##  Min.   :1.609   Min.   :1.609
##  1st Qu.:5.551   1st Qu.:4.810
##  Median :7.317   Median :7.317
##  Mean   :6.356   Mean   :6.278
##  3rd Qu.:8.303   3rd Qu.:8.295
##  Max.   :9.681   Max.   :9.761

for (i in cols) {
  par(mfrow=c(1, 2))
  boxplot(df_quant_skewed[,i], horizontal=TRUE, main=paste("Origin Boxplot
of", i), col="lightblue")
```

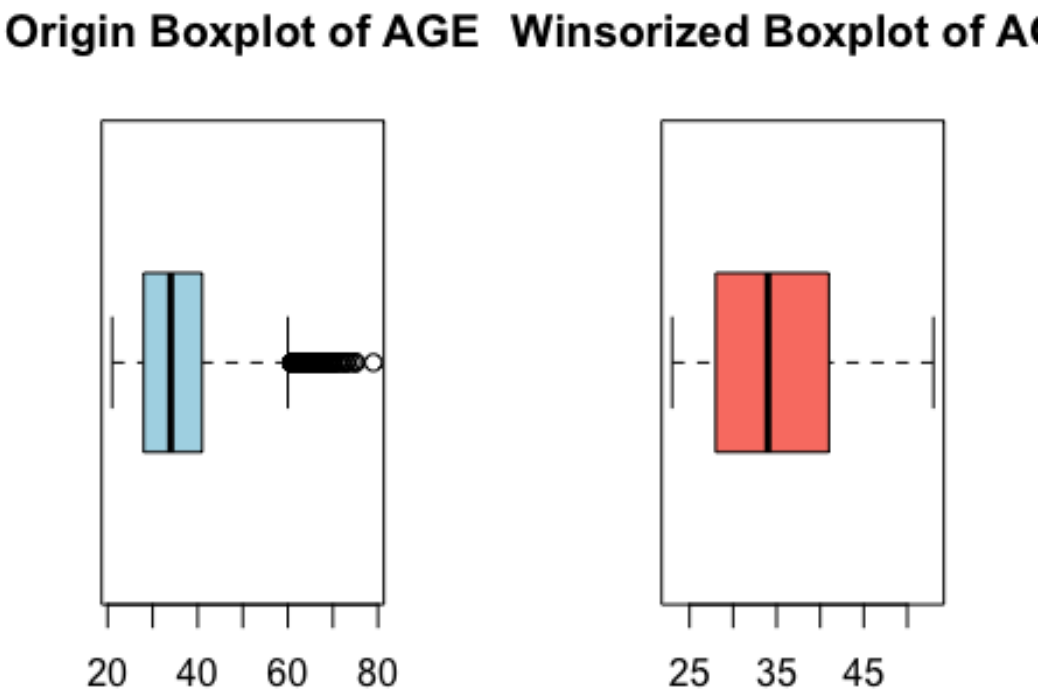
```
boxplot(df_quant_winsorized[,i], horizontal=TRUE, main=paste("Winsorized
Boxplot of", i), col="salmon")
}
```

Origin Boxplot of LIMIT_B

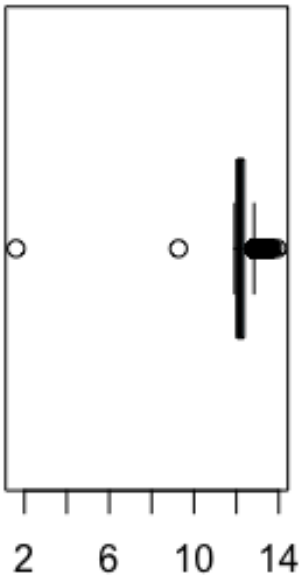


Winsorized Boxplot of LIMIT

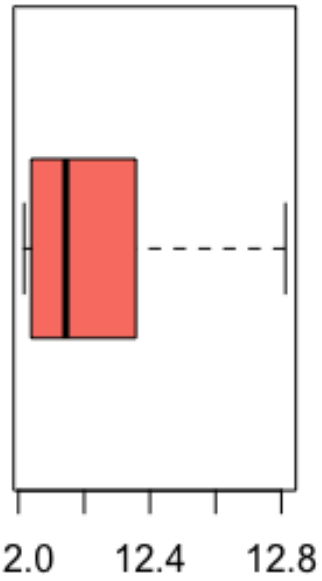



Origin Boxplot of AGE
Winsorized Boxplot of A
20
40
60
80
25
35
45

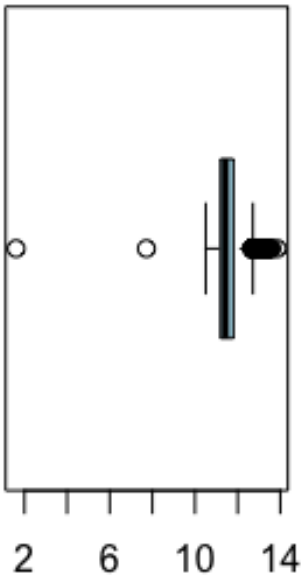
Origin Boxplot of BILL_A



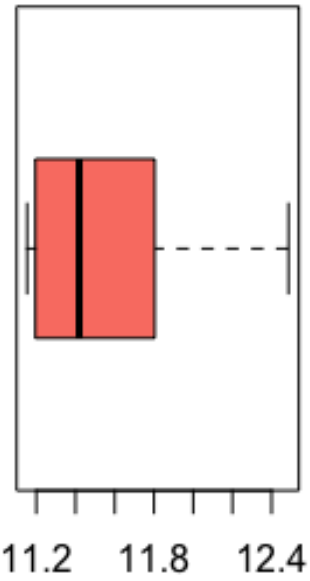
Winsorized Boxplot of BILL_



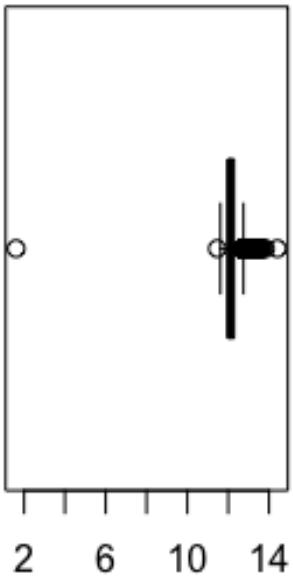
Origin Boxplot of BILL_A



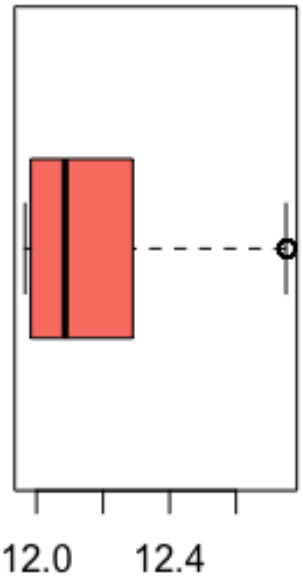
Winsorized Boxplot of BILL_



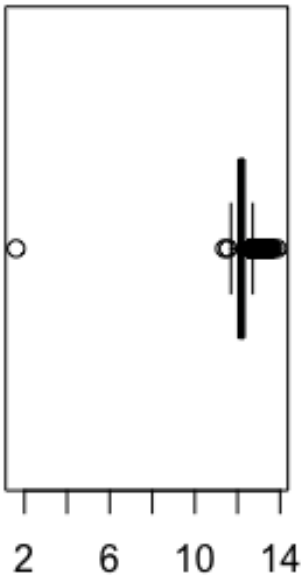
Origin Boxplot of BILL_A



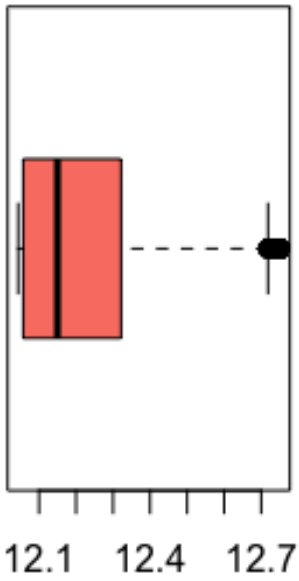
Winsorized Boxplot of BILL_



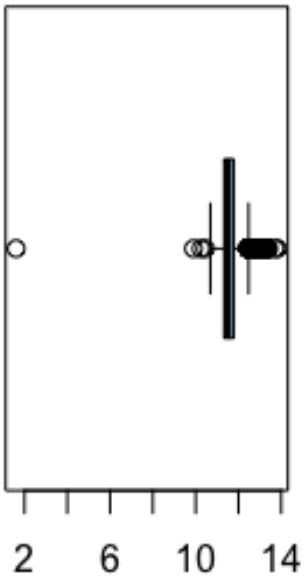
Origin Boxplot of BILL_A



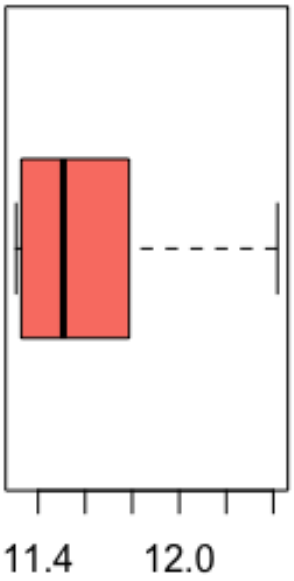
Winsorized Boxplot of BILL_



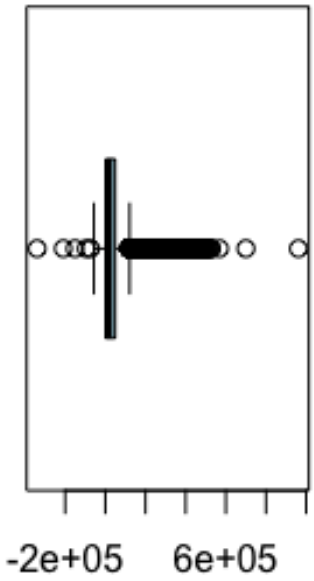
Origin Boxplot of BILL_A



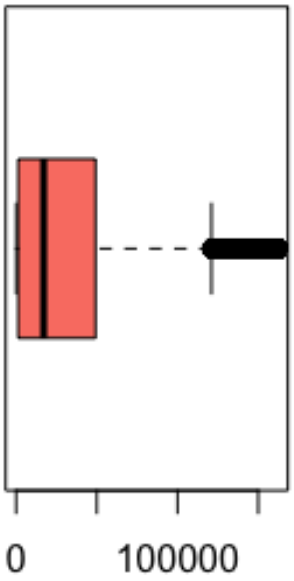
Winsorized Boxplot of BILL_



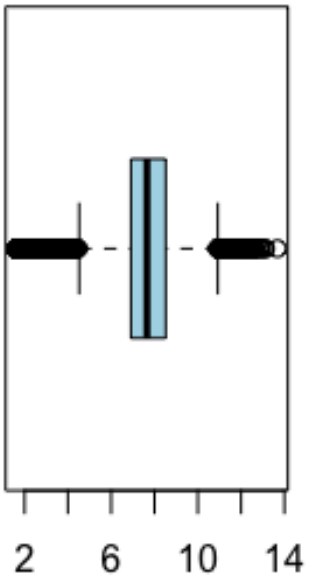
Origin Boxplot of BILL_A



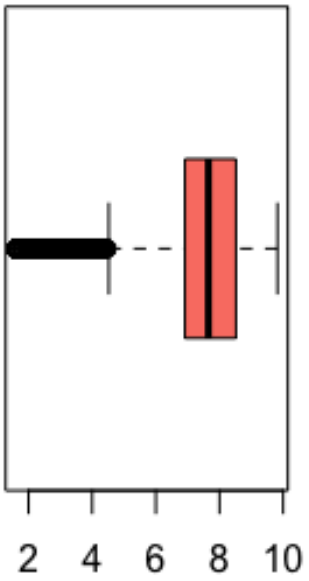
Winsorized Boxplot of BILL_

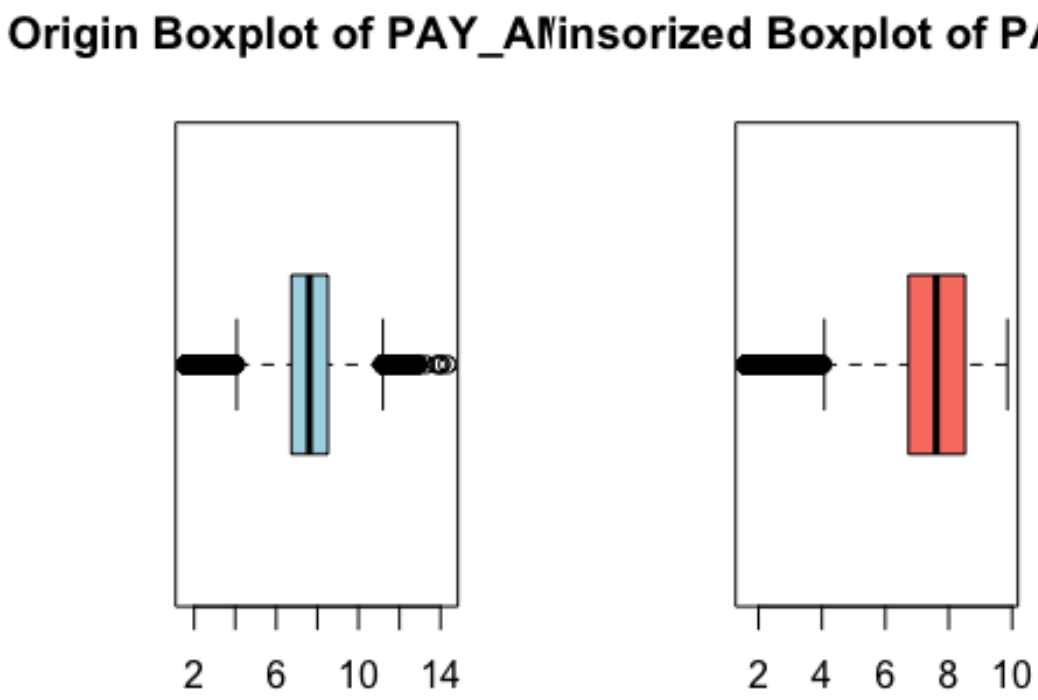


Origin Boxplot of PAY_A

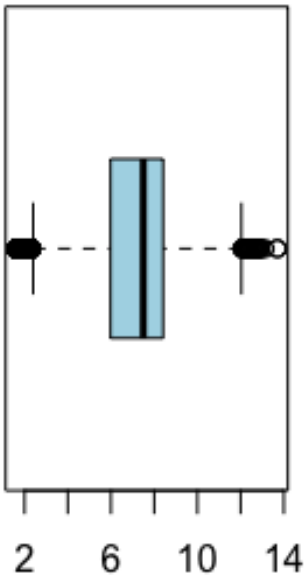


Winsorized Boxplot of PAY_

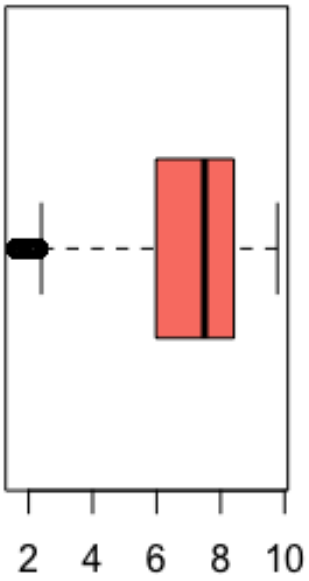



Origin Boxplot of PAY_A
Winsorized Boxplot of PAY_
2
6
10
14
2
4
6
8
10

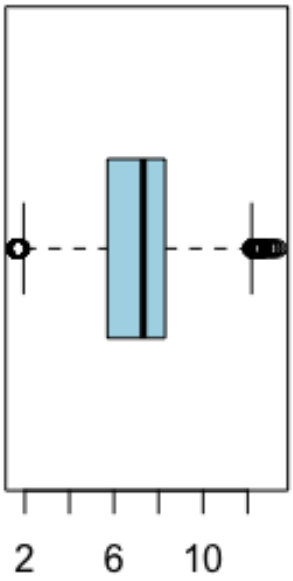
Origin Boxplot of PAY_A



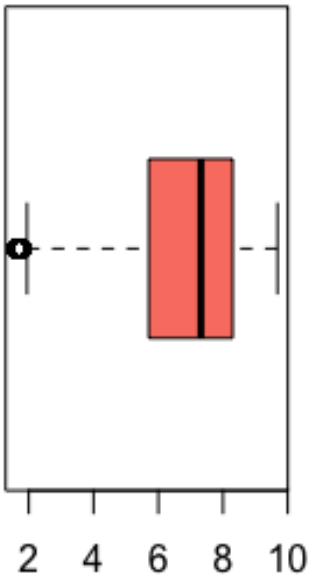
Winsorized Boxplot of PAY_

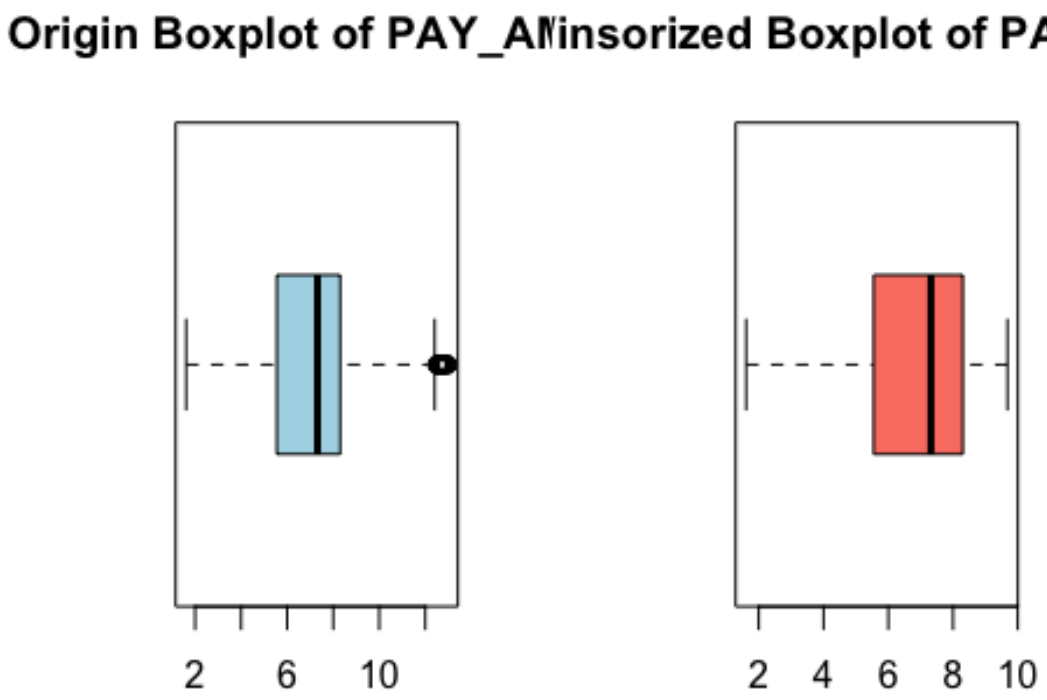


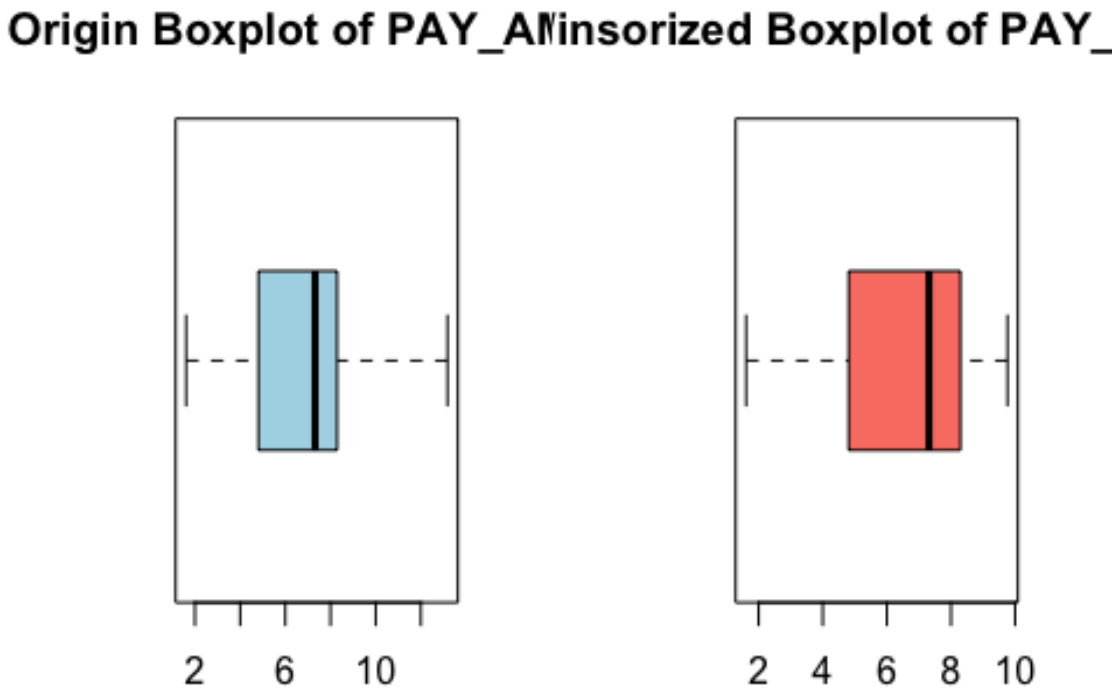
Origin Boxplot of PAY_A



Winsorized Boxplot of PAY_




Origin Boxplot of PAY_A
Winsorized Boxplot of PAY_
2
6
10
2
4
6
8
10




```
# Combine final cleaned df
df_processed <- cbind(df_quant_winsorized, df_cat)
head(df_processed)
```

```
##   LIMIT_BAL AGE BILL_AMT1 BILL_AMT2 BILL_AMT3 BILL_AMT4 BILL_AMT5
BILL_AMT6
## 1     20000  24  12.04060  11.19662  11.97008  12.04358  11.30638
0
## 2    120000  26  12.03331  11.17755  11.98262  12.06265  11.34798
3261
## 3     90000  34  12.17985  11.33630  12.04841  12.12452  11.47509
15549
## 4     50000  37  12.26705  11.67857  12.23835  12.19763  11.61094
29547
## 5     50000  53  12.06797  11.23125  12.17098  12.15974  11.51776
19131
## 6     50000  37  12.34577  11.75077  12.27782  12.15161  11.52246
20024
##   PAY_AMT1 PAY_AMT2 PAY_AMT3 PAY_AMT4 PAY_AMT5 PAY_AMT6 PAY_0 PAY_2 PAY_3
PAY_4
## 1 1.609438 6.542472 1.609438 1.609438 1.609438 1.609438     2     2    -1
-1
## 2 1.609438 6.912743 6.912743 6.912743 1.609438 7.603399    -1     2     0
```

```
0
## 3 7.328437 7.316548 6.912743 6.912743 6.912743 8.518193     0     0     0
0
## 4 7.603399 7.612831 7.094235 7.007601 6.979145 6.912743     0     0     0
0
## 5 7.603399 9.852686 9.210840 9.105535 6.542472 6.527958    -1     0    -1
0
## 6 7.826044 7.506592 6.495266 6.912743 6.912743 6.690842     0     0     0
0
##   PAY_5 PAY_6 DEFAULT SEX.Male EDUCATION.High.School EDUCATION.Other
## 1    -2    -2     Yes        0                    0               0
## 2     0     2     Yes        0                    0               0
## 3     0     0      No        0                    0               0
## 4     0     0      No        0                    0               0
## 5     0     0      No        1                    0               0
## 6     0     0      No        1                    0               0
##   EDUCATION.University EDUCATION.Unknown MARRIAGE.Other MARRIAGE.Single
## 1                    1                 0              0                0
## 2                    1                 0              0                1
## 3                    1                 0              0                1
## 4                    1                 0              0                0
## 5                    1                 0              0                0
## 6                    0                 0              0                1
```

```
names(df_processed)
```

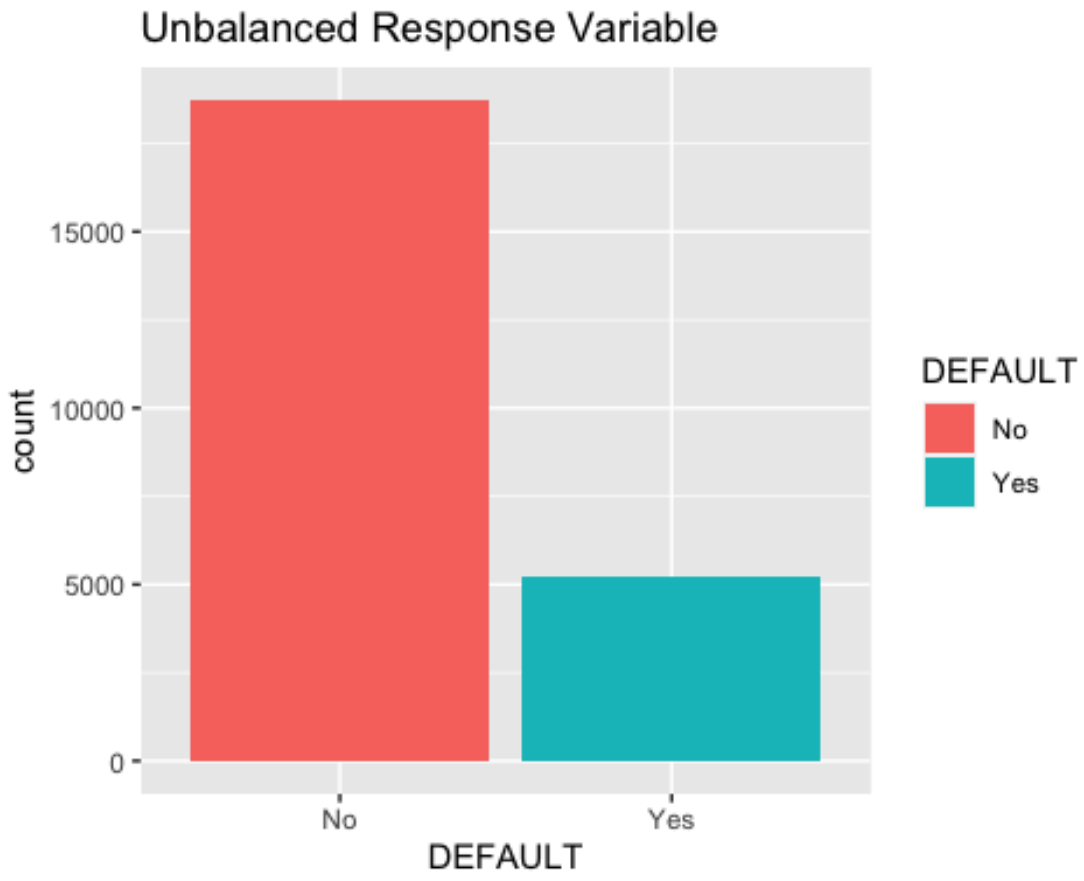
```
##  [1] "LIMIT_BAL"             "AGE"                   "BILL_AMT1"
##  [4] "BILL_AMT2"             "BILL_AMT3"             "BILL_AMT4"
##  [7] "BILL_AMT5"             "BILL_AMT6"             "PAY_AMT1"
## [10] "PAY_AMT2"              "PAY_AMT3"              "PAY_AMT4"
## [13] "PAY_AMT5"              "PAY_AMT6"              "PAY_0"
## [16] "PAY_2"                 "PAY_3"                 "PAY_4"
## [19] "PAY_5"                 "PAY_6"                 "DEFAULT"
## [22] "SEX.Male"              "EDUCATION.High.School" "EDUCATION.Other"
## [25] "EDUCATION.University"  "EDUCATION.Unknown"     "MARRIAGE.Other"
## [28] "MARRIAGE.Single"
```

Split training and testing data sets

```
set.seed(1)   # set a random seed
index <- sample(30000, 6000) # random selection of indices. (20%)

test<-df_processed%>%
  filter(row_number() %in% index)

training<-df_processed%>%setdiff(test)

table(training$DEFAULT)
```

```
##
##    No   Yes
## 18725  5237
```

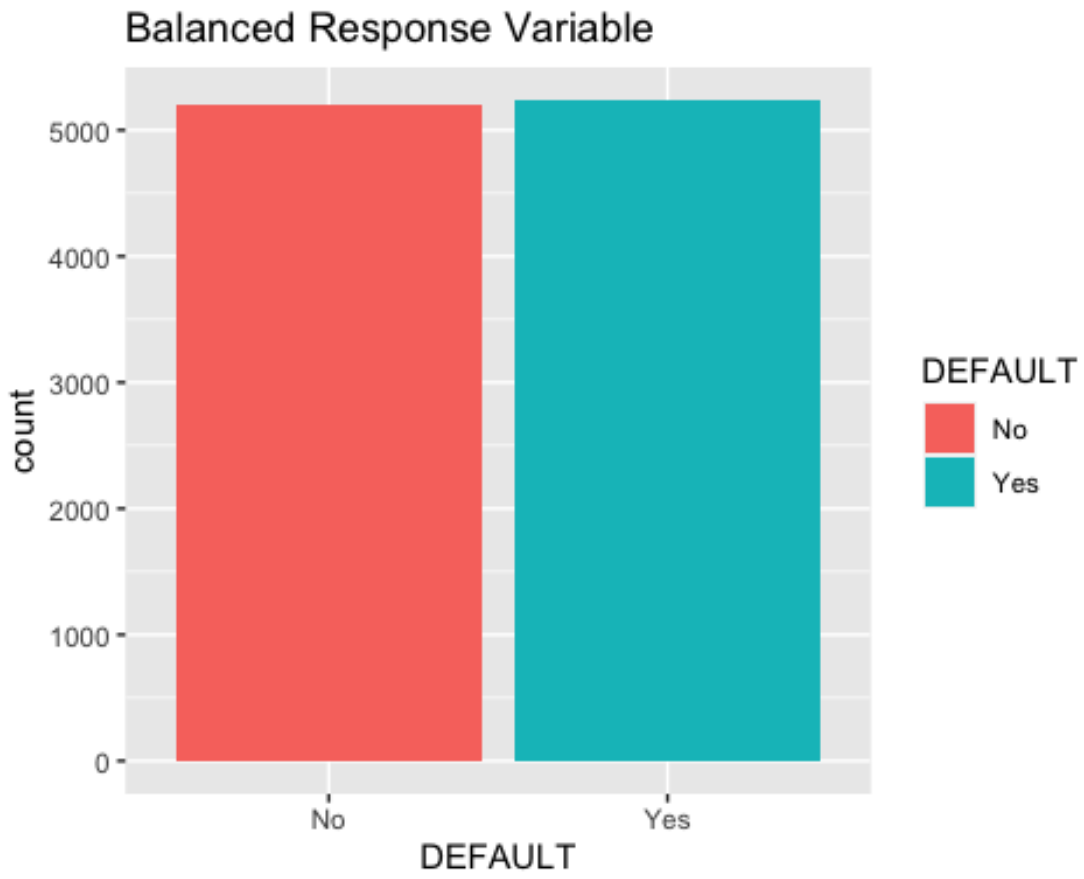
```
status <- ggplot(data=training, aes(x=DEFAULT,fill=DEFAULT)) +
  geom_bar()+
  labs(title = "Unbalanced Response Variable")
status
```



Undersample the Unbalanced Dataset

```
# Undersample Dataset
library(ROSE)

## Loaded ROSE 0.0-4

training_balanced <- ovun.sample(DEFAULT~.,data = training,
                                 p=0.5,seed = 1,method = "under")$data
table(training_balanced$DEFAULT)

##
##   No  Yes
## 5195 5237

table(test$DEFAULT)

##
##   No  Yes
## 4609 1391
```

```
status <- ggplot(data=training_balanced, aes(x=DEFAULT,fill=DEFAULT)) +
  geom_bar()+
  labs(title = "Balanced Response Variable")
status
```



###

True Positive Rate Checking (Rational Behind Undersample)

```
# Balanced Dataset
logit_model_balanced<-glm(DEFAULT~.,
                 family="binomial",
                 data=training_balanced)

test$logit_pred_prob_balanced<-
predict(logit_model_balanced,test,type="response")

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type
== :
## prediction from a rank-deficient fit may be misleading

test$logit_pred_class_balanced <-
ifelse(test$logit_pred_prob_balanced>0.5,"Yes","No")
test <- test%>%
   mutate(
          logit_pred_class_balanced =as.factor(logit_pred_class_balanced)
          )
```

```
confusionMatrix(test$logit_pred_class_balanced, test$DEFAULT, positive =
"Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  3769  524
##        Yes  840  867
##
##                Accuracy : 0.7727
##                  95% CI : (0.7618, 0.7832)
##     No Information Rate : 0.7682
##     P-Value [Acc > NIR] : 0.209
##
##                   Kappa : 0.4086
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.6233
##             Specificity : 0.8177
##          Pos Pred Value : 0.5079
##          Neg Pred Value : 0.8779
##              Prevalence : 0.2318
##          Detection Rate : 0.1445
##    Detection Prevalence : 0.2845
##       Balanced Accuracy : 0.7205
##
##        'Positive' Class : Yes
##
```

```
867/(524+867)
```

```
## [1] 0.6232926
```

```
# Unbalanced Dataset
logit_model_unbalanced<-glm(DEFAULT~.,  # generalized linear models
                 family="binomial",               # specifying error
distribution
                 data=training)

test$logit_pred_prob_unbalanced<-
predict(logit_model_unbalanced,test,type="response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type
== :
## prediction from a rank-deficient fit may be misleading
```

```
test$logit_pred_class_unbalanced<-
ifelse(test$logit_pred_prob_unbalanced>0.5,"Yes","No")
test <- test%>%
```

```
   mutate(
          logit_pred_class_unbalanced =as.factor(logit_pred_class_unbalanced)
          )
confusionMatrix(test$logit_pred_class_unbalanced, test$DEFAULT,positive =
"Yes")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  4393  887
##        Yes  216  504
##
##                Accuracy : 0.8162
##                  95% CI : (0.8061, 0.8259)
##     No Information Rate : 0.7682
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3793
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.3623
##             Specificity : 0.9531
##          Pos Pred Value : 0.7000
##          Neg Pred Value : 0.8320
##              Prevalence : 0.2318
##          Detection Rate : 0.0840
##    Detection Prevalence : 0.1200
##       Balanced Accuracy : 0.6577
##
##        'Positive' Class : Yes
##
```

```
504/(887+504)
```

```
## [1] 0.3623293
```

```
# The TPR has dropped by much without undersampling
```