490rt

RyanYip

2018 蹇 11 鏈<88>1 鏃

```
library(ggplot2)
library(dplyr)

##
## 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(funModeling)

## Loading required package: Hmisc

## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:dplyr':
##
##     combine, src, summarize

## The following objects are masked from 'package:base':
##
##     format.pval, round.POSIXt, trunc.POSIXt, units

## funModeling v.1.6.5 :)
## Examples and tutorials at livebook.datascienceheroes.com

library(Hmisc)
```

Load data

```
data = read.csv("./rt.csv",header = TRUE)
basic_eda <- function(data)
```

```
{
  glimpse(data)
  df_status(data)
  freq(data)
  profiling_num(data)
  plot_num(data)
  describe(data)
}
```

Data preprocess

```
summary(data)
```

```
##        ID         Record           RT              Time        
##  Min.   : 1   Min.   :1.00   Min.   :0.0970          : 25  
##  1st Qu.:10   1st Qu.:2.75   1st Qu.:0.3470   23:30  :  7  
##  Median :20   Median :4.50   Median :0.3850   0:30   :  4  
##  Mean   :20   Mean   :4.50   Mean   :0.3976   9:30   :  4  
##  3rd Qu.:30   3rd Qu.:6.25   3rd Qu.:0.4320   11:00  :  3  
##  Max.   :39   Max.   :8.00   Max.   :0.8490   11:30  :  3  
##                              NA's   :10       (Other):266  
##    Stimulate        Fatigue          Hunger       busyOrlight     
##  Min.   :0.000   Min.   :1.000   Min.   :1.000   Min.   :0.0000  
##  1st Qu.:0.000   1st Qu.:3.000   1st Qu.:4.000   1st Qu.:0.0000  
##  Median :0.000   Median :3.500   Median :5.000   Median :1.0000  
##  Mean   :0.122   Mean   :3.629   Mean   :4.745   Mean   :0.5065  
##  3rd Qu.:0.000   3rd Qu.:5.000   3rd Qu.:6.000   3rd Qu.:1.0000  
##  Max.   :1.000   Max.   :7.000   Max.   :9.000   Max.   :1.0000  
##  NA's   :17      NA's   :18      NA's   :18      NA's   :4       
##     illness           Sleep        Protocol      MEQ          
##  Min.   :0.00000   Min.   : 4.000     : 30   Min.   :-1.0000  
##  1st Qu.:0.00000   1st Qu.: 7.000   ?0:  1   1st Qu.:-1.0000  
##  Median :0.00000   Median : 8.000   ?1:  1   Median : 0.0000  
##  Mean   :0.09247   Mean   : 7.898   0 :  4   Mean   :-0.3733  
##  3rd Qu.:0.00000   3rd Qu.: 8.500   1 :276   3rd Qu.: 0.0000  
##  Max.   :1.00000   Max.   :12.000            Max.   : 1.0000  
##  NA's   :20        NA's   :37                NA's   :12
```

According to the summary of data,it exists NA value.It is a point need to handle.

```
sprintf("Total NA value is :%d",sum(is.na(data)),"\n")
```

```
## [1] "Total NA value is :136"
```

```
apply(data,2,function(x){return(sum(is.na(x)))})  # Na  in each varible
```

```
##         ID     Record         RT       Time   Stimulate     Fatigue 
##          0          0         10          0         17         18 
##     Hunger busyOrlight     illness      Sleep    Protocol        MEQ 
##
##         18          4         20         37          0         12
```

```
data=na.omit(data)
```

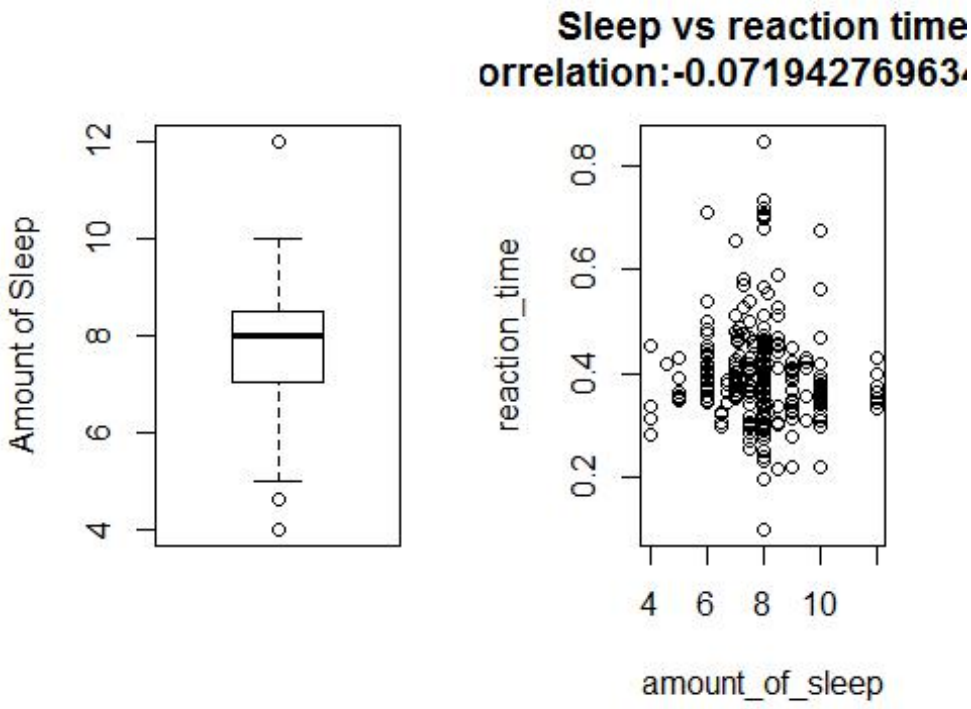
The method processing the NA value I used is removing the data only.

Variable explore

According to common sense, the sleep ,stimulant,Fatigue ,Hunger,nusyOrnight(0 to represent light day and 1 for busy day),illness,Protocol and MEQ has a great influence on people's reaction time.So It is necessary to explore these variable and based on these message from the explore,it is helpful for us to do next anlysis and create model. Explore method mainly includes scatter plot, boxplot, barplot, as well as frequency plot to explore variable in data.

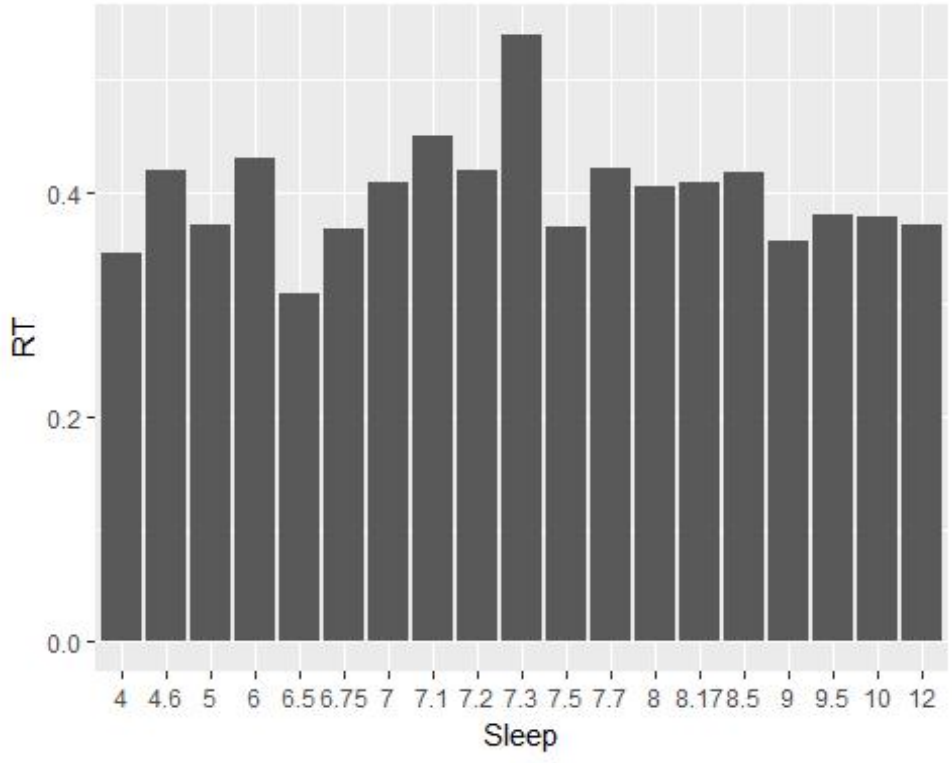
1. Sleep

```
amount_of_sleep = data$Sleep
reaction_time = data$RT
sprintf("the mean of sleep:%f",mean(data$Sleep, na.rm= T))

## [1] "the mean of sleep:7.890076"

summary(data$Sleep)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   7.075   8.000   7.890   8.500  12.000

par(mfrow = c(1,2))
boxplot((data$Sleep), ylab = "Amount of Sleep")
plot(amount_of_sleep, reaction_time,main = paste0("Sleep vs reaction ti
me\n","correlation:",cor(amount_of_sleep,reaction_time)))
```



The Boxplot of sleep tells us that,sleep distributes centerly around 8 hours,it has few outliers and ignore them.The plot of sleep vs reaction time ,is is strange that they have lower correlation and plot tell us the same reslut.According these graph,I probably know the sleep distribution and correlation with RT.

```
af <- na.omit(data)
af$Sleep <- as.factor(af$Sleep)
ggplot(summarise(group_by(af, Sleep), RT = mean(RT)),aes(x= Sleep, y = R
T)) + geom_bar(stat = "identity", position = position_stack())
```



The above plot is Barplot group by sleep on mean value.We can know that the mean of sleep distributes evenly on each time point,But around 7.3 is highest.It is consistent with previous plot.

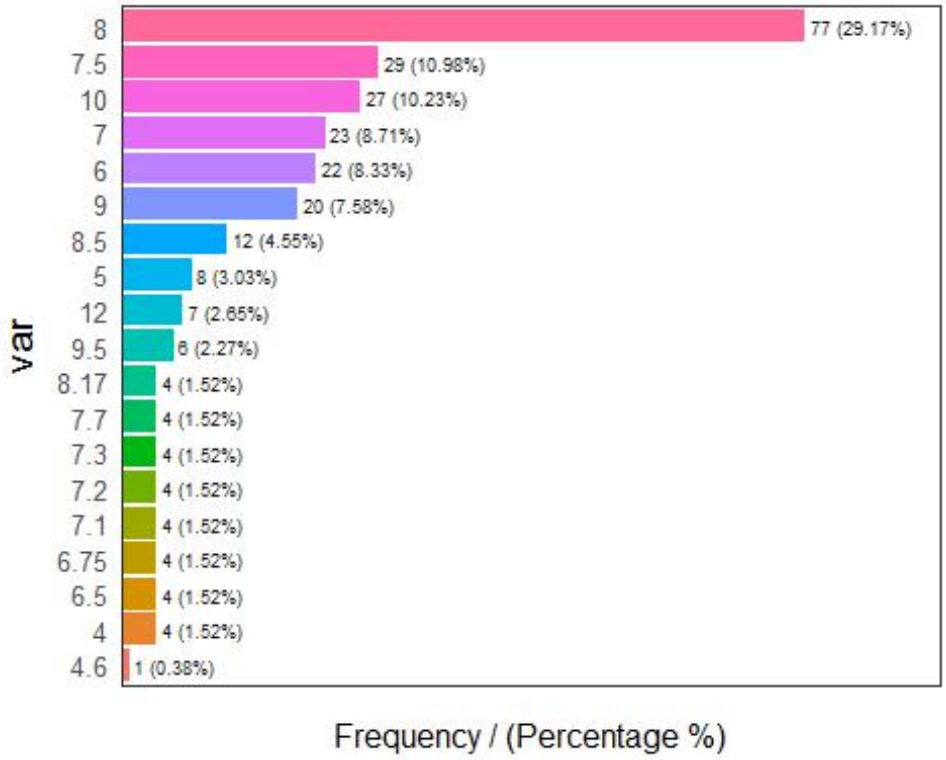
```
summarise(group_by(af, Sleep), mean(RT))
```

```
## # A tibble: 19 x 2
##    Sleep `mean(RT)`
##    <fctr>      <dbl>
##  1      4  0.3447500
##  2    4.6  0.4190000
##  3      5  0.3702500
##  4      6  0.4309091
##  5    6.5  0.3100000
##  6   6.75  0.3667500
##  7      7  0.4087826
##  8    7.1  0.4502500
##  9    7.2  0.4200000
## 10    7.3  0.5400000
## 11    7.5  0.3691034
## 12    7.7  0.4210000
## 13      8  0.4052987
## 14   8.17  0.4090000
## 15    8.5  0.4178333
## 16      9  0.3561000
## 17    9.5  0.3803333
```

```
## 18     10  0.3778889
## 19     12  0.3711429
```

```
freq(data$Sleep)
```

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



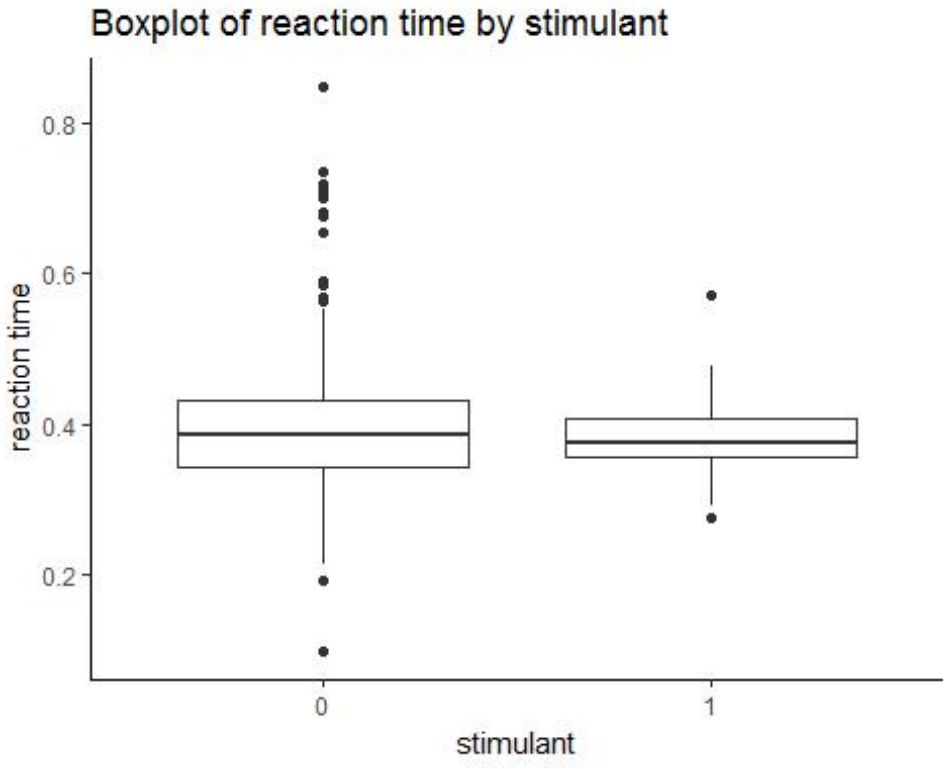

```
##     var frequency percentage cumulative_perc
## 1     8        77      29.17           29.17
## 2   7.5        29      10.98           40.15
## 3    10        27      10.23           50.38
## 4     7        23       8.71           59.09
## 5     6        22       8.33           67.42
## 6     9        20       7.58           75.00
## 7   8.5        12       4.55           79.55
## 8     5         8       3.03           82.58
## 9    12         7       2.65           85.23
## 10  9.5         6       2.27           87.50
## 11    4         4       1.52           89.02
## 12  6.5         4       1.52           90.54
## 13 6.75         4       1.52           92.06
## 14  7.1         4       1.52           93.58
## 15  7.2         4       1.52           95.10
## 16  7.3         4       1.52           96.62
## 17  7.7         4       1.52           98.14
## 18 8.17         4       1.52           99.66
## 19  4.6         1       0.38          100.00
```

The graph show the mean and frequency of each sleep time,** freq function ** is only visual on ** suammrise **,what they tell us is the same.They both tells us that 8 hour sleep has the biggest proportion in total sleep,which is consistent with boxplot.

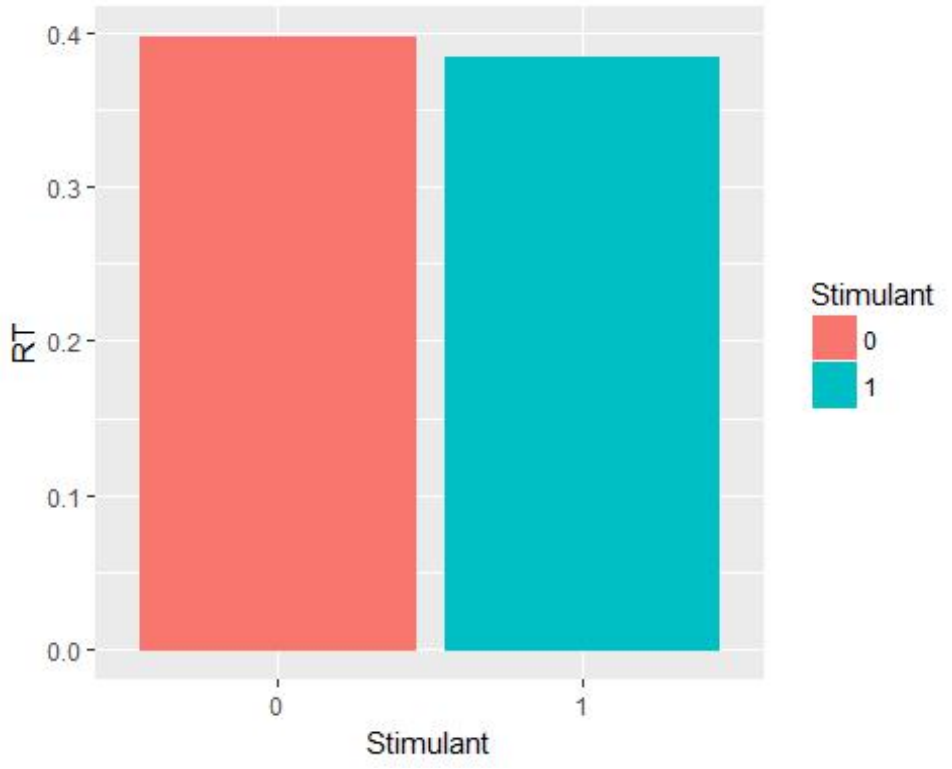
2.stimulant

For stimulant variable, I have the two-side boxplot to compare the mean reaction time value. I did not take out the missing data because they are not useless in this case. I treat them like a reference group. And in frequency plot, it is obvious that most of the people did not use stimulant during the test.

```
#stimulant = data$Stimulant
ggplot(data,aes(x = factor(Stimulate),y = RT)) +
  theme_classic()  +
  geom_boxplot() +
  labs(title = "Boxplot of reaction time by stimulant",
       x = "stimulant",
       y = "reaction time")
```




```
af$Stimulant <- as.factor(af$Stimulate)
ggplot(summarise(group_by(af, Stimulant), RT = mean(RT)),aes(x= Stimula
nt, y = RT)) + geom_bar(stat = "identity", position = position_stack(),a
es(color = Stimulant, fill = Stimulant))
```

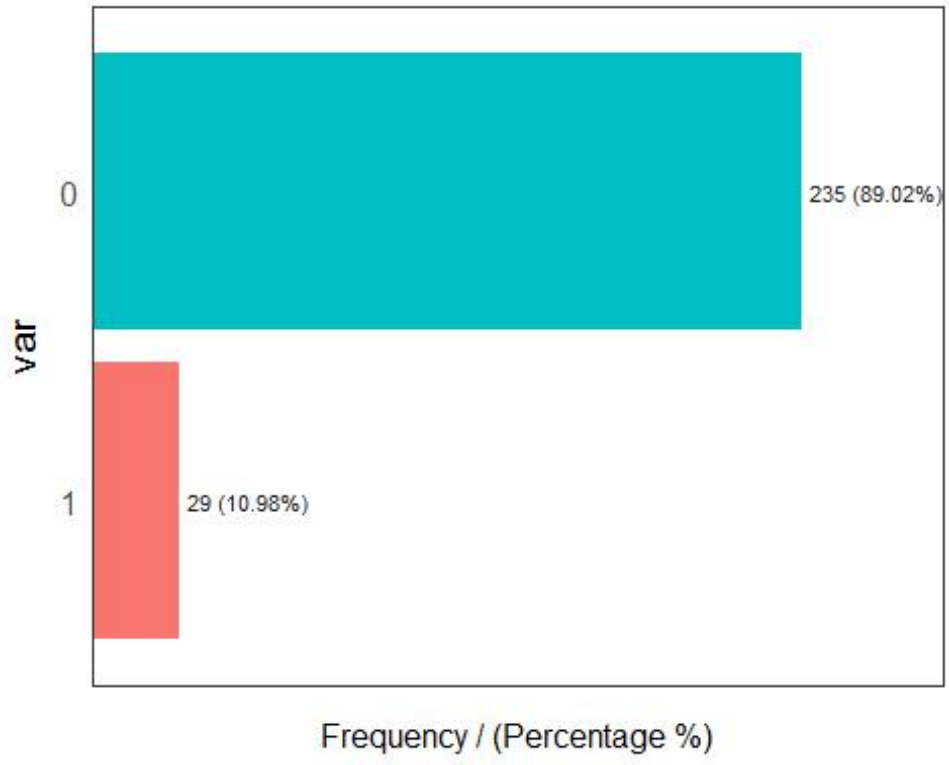


The Boxplot and Barplot both tell us that,the reaction time on using or not using stimulant is almost the same,they have the almost same effect .

```
summarise(group_by(af, Stimulant), mean(RT))
```

```
## # A tibble: 2 x 2
##   Stimulant `mean(RT)`
##      <fctr>      <dbl>
## 1         0  0.3970894
## 2         1  0.3836207
```

```
freq(af$Stimulate)
```




```
##   var frequency percentage cumulative_perc
## 1   0       235      89.02          89.02
## 2   1        29      10.98         100.00
```

In frequency plot, it is obvious that most of the people did not use stimulant during the test,the proportion of not use stimulant is abount ** 89% **,we can learn the distrubution of this variable

3.Fatigue

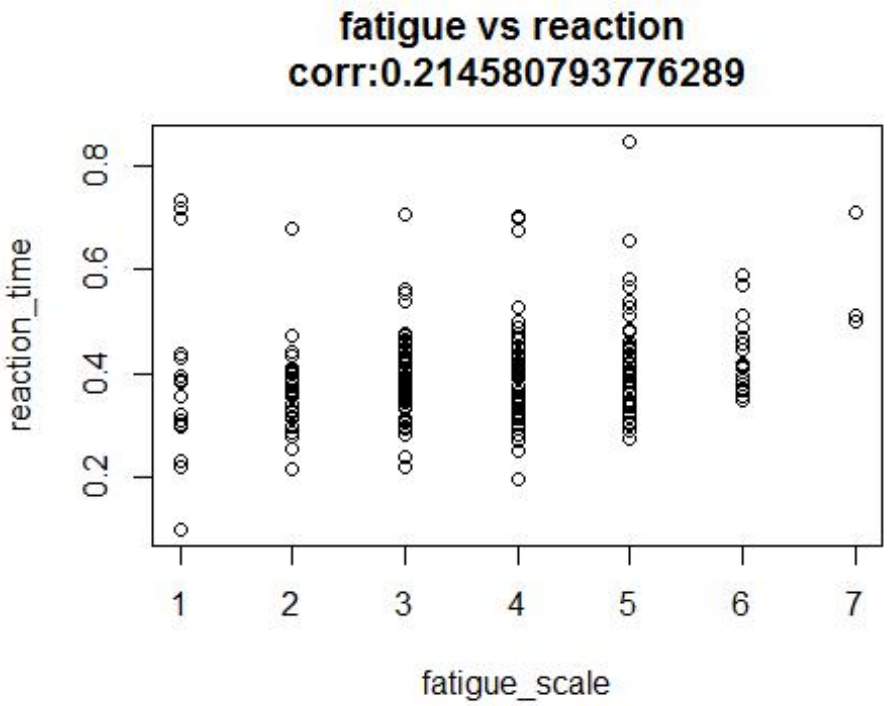
```
fatigue_scale = data$Fatigue
sprintf("the mean of fatigue:%f",mean(data$Fatigue, na.rm= T))
```

```
## [1] "the mean of fatigue:3.602273"
```

```
summary(data$Fatigue)
```

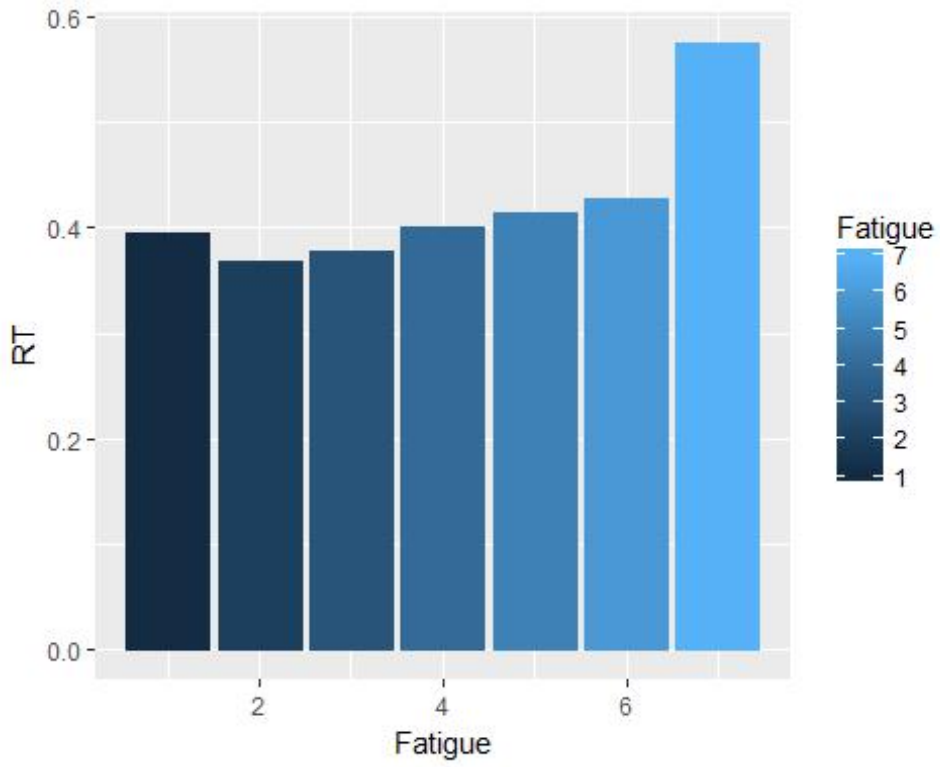
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   3.000   3.602   5.000   7.000
```

```
plot(fatigue_scale, reaction_time,main=paste0("fatigue vs reaction \n",
"corr:",cor(fatigue_scale,reaction_time)))
```

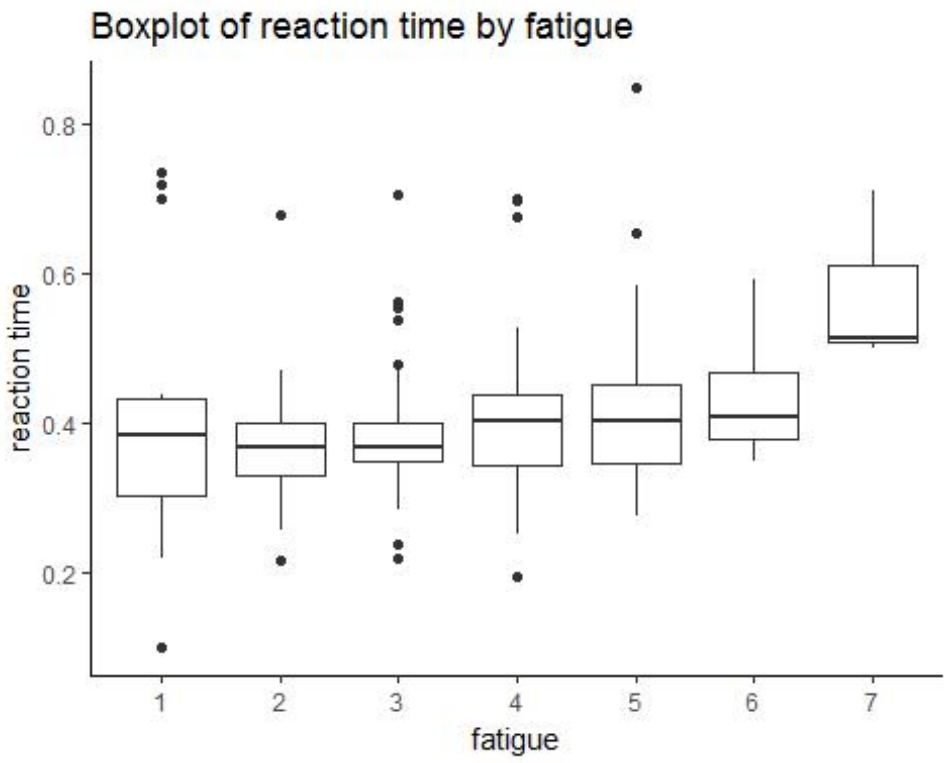


The plot show the distribution of fatigue with reaction time.They have not apparent relationship what the plo show,and the correlation between them is only 0.21,it is low.Summary function just tells us the summary of this variable .

```
af$Sleep <- as.factor(af$Fatigue)
ggplot(summarise(group_by(af, Fatigue), RT = mean(RT)),aes(x= Fatigue,
y = RT)) + geom_bar(stat = "identity", position = position_stack(), aes
(color = Fatigue, fill = Fatigue))
```




```
ggplot(data,aes(x = factor(fatigue_scale),y = reaction_time)) +
  theme_classic()  +
  geom_boxplot() +
  labs(title = "Boxplot of reaction time by fatigue",
       x = "fatigue",
       y = "reaction time")
```

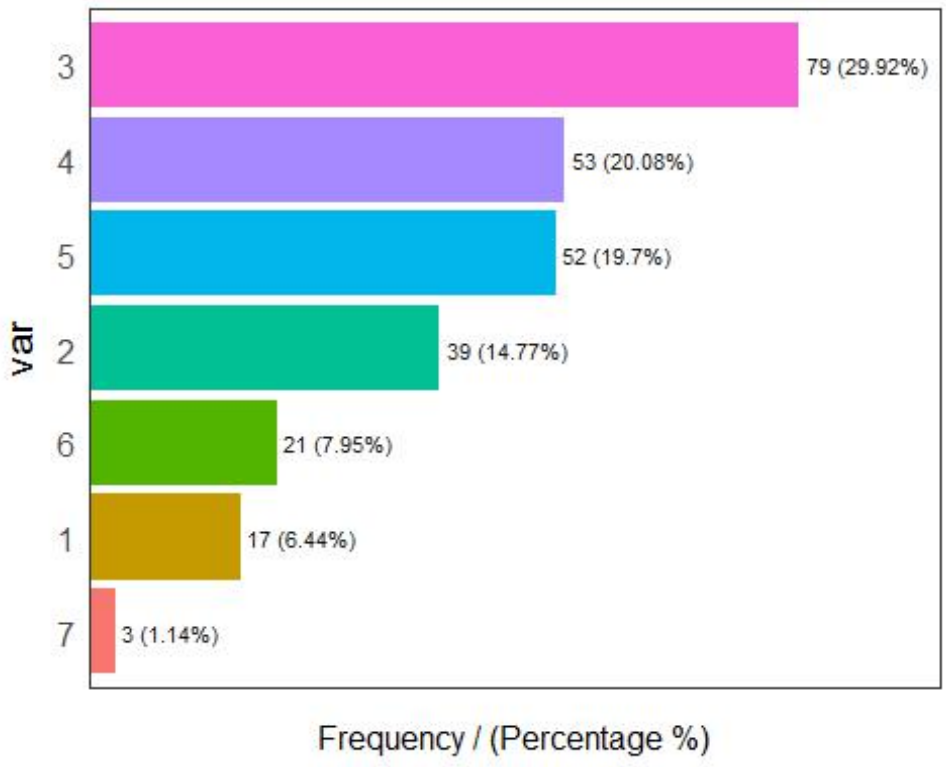


What the message we get from the barplot and boxplot are same.According to above graph,the reaction time is highest when fatigue is 7,and is has a growth trend when fatigue increase,which indicates reacte more slow.It is important.

```
summarise(group_by(af, Fatigue), mean(RT))
```

```
## # A tibble: 7 x 2
##   Fatigue `mean(RT)`
##     <int>      <dbl>
## 1       1  0.3942941
## 2       2  0.3686410
## 3       3  0.3779241
## 4       4  0.4003774
## 5       5  0.4150192
## 6       6  0.4275238
## 7       7  0.5753333
```

```
freq(data$Fatigue)
```

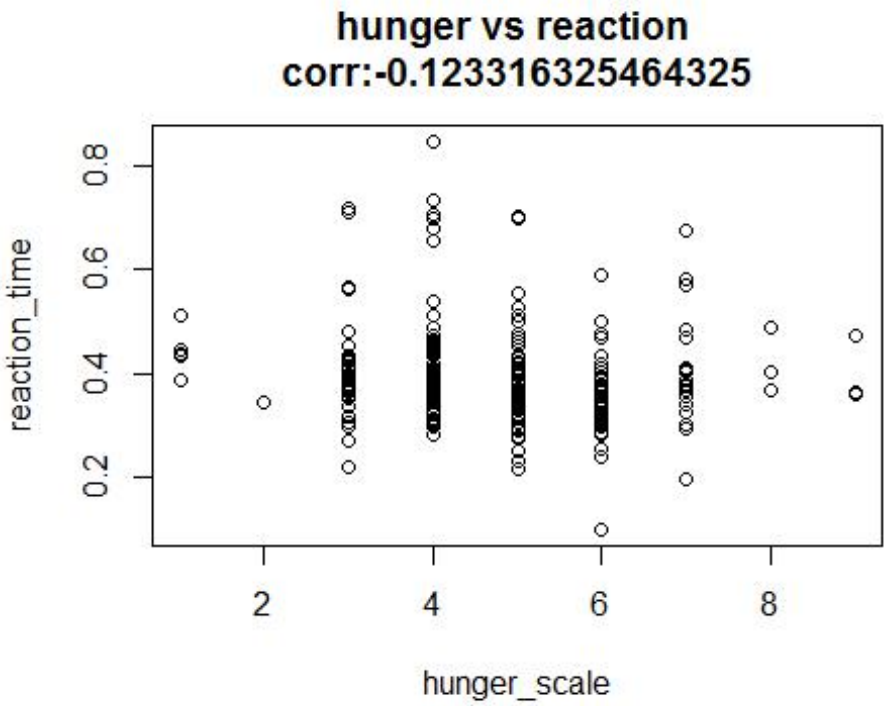



```
##   var frequency percentage cumulative_perc
## 1   3        79      29.92           29.92
## 2   4        53      20.08           50.00
## 3   5        52      19.70           69.70
## 4   2        39      14.77           84.47
## 5   6        21       7.95           92.42
## 6   1        17       6.44           98.86
## 7   7         3       1.14          100.00
```

what the summary of mean RT group by fatigue show is same as the barplot and boxplot.Accordint to the frequency plot,the proportion on fatigue=3 is highest,about 29%.And the proportion between them is not uniform,it has diiference.

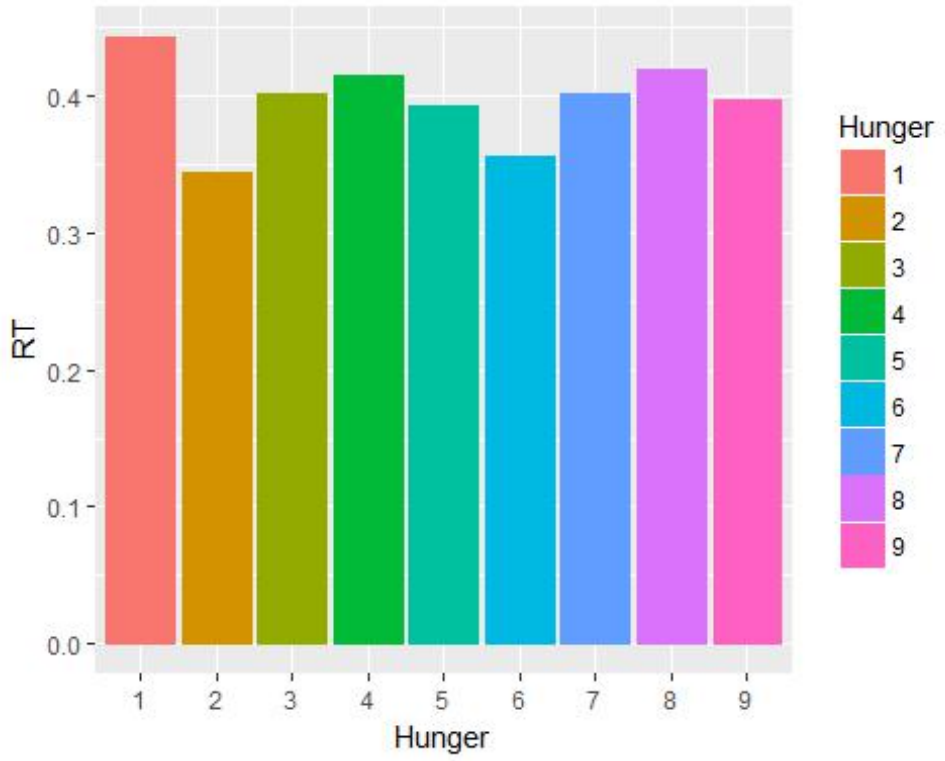
4.Hunger

```
hunger_scale = data$Hunger
sprintf("the mean of hunger:%f",mean(data$Hunger, na.rm= T))

## [1] "the mean of hunger:4.757576"

summary(data$Hunger)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   4.000   5.000   4.758   6.000   9.000

plot(hunger_scale, reaction_time,main=paste0("hunger vs reaction \n","c
orr:",cor(hunger_scale,reaction_time)))
```

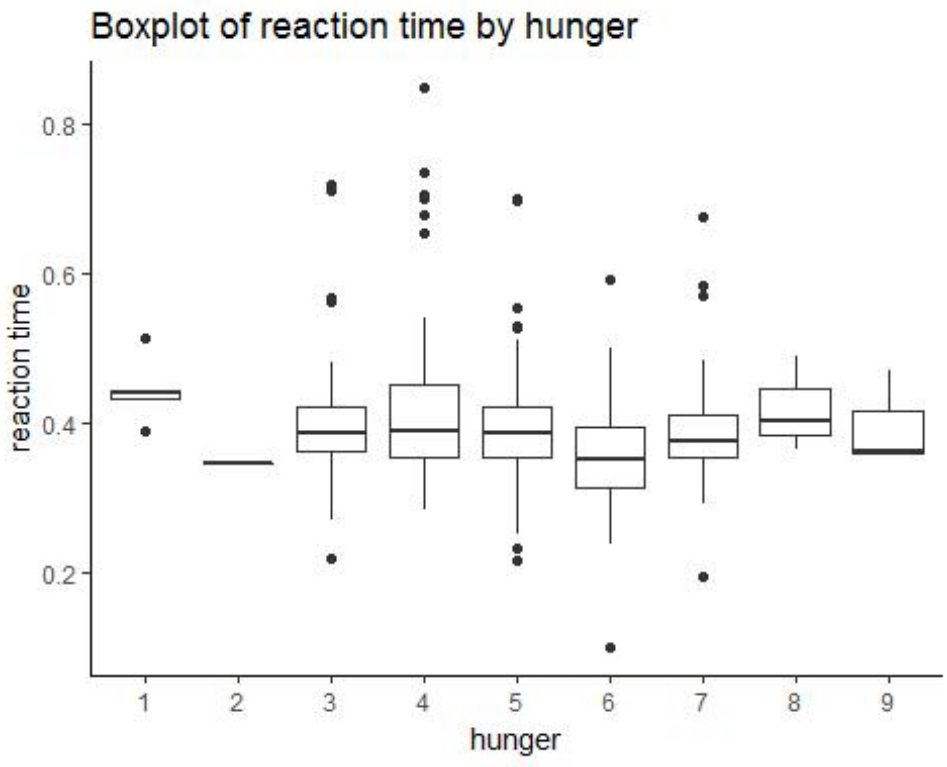


The plot between hunger and reaction time shows,reaction time change apparently when hunger change,and the corrlation is also lower.It presents that most of the people test their reaction time near a hunger level of 4. When hunger level reaches 6, it is more likely that the person would have the fastest reaction speed.

```
af$Hunger <- as.factor(af$Hunger)
af$Record<-as.factor(af$Record)
ggplot(summarise(group_by(af, Hunger), RT = mean(RT)),aes(x= Hunger, y =
 RT)) + geom_bar(stat = "identity", position = position_stack(), aes(col
or = Hunger, fill = Hunger))
```




```r
ggplot(data,aes(x = factor(hunger_scale),y = reaction_time)) +
  theme_classic()  +
  geom_boxplot() +
  labs(title = "Boxplot of reaction time by hunger",
       x = "hunger",
       y = "reaction time")
```

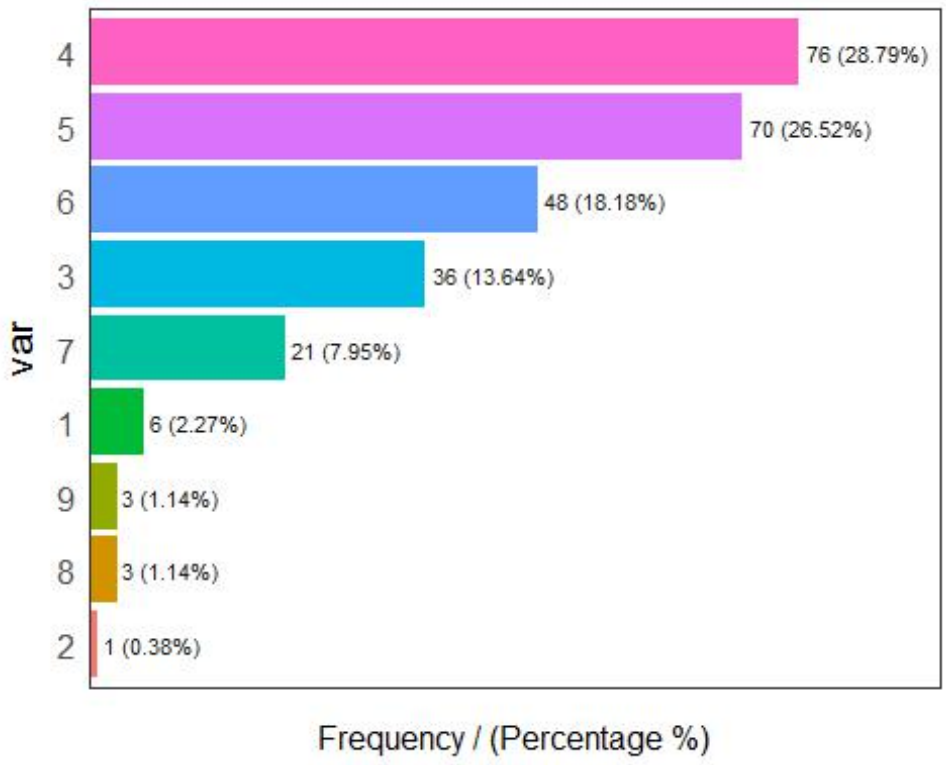


The boxplot and barplot show reaction-time's trend on each mean of hunger is gentle,and when hunger=1,reaction time is the most highest.

```
summarise(group_by(af, Hunger), mean(RT))
```

```
## # A tibble: 9 x 2
##   Hunger `mean(RT)`
##   <fctr>      <dbl>
## 1      1  0.4430000
## 2      2  0.3440000
## 3      3  0.4015833
## 4      4  0.4152895
## 5      5  0.3926000
## 6      6  0.3553125
## 7      7  0.4016667
## 8      8  0.4190000
## 9      9  0.3970000
```

```
freq(data$Hunger)
```



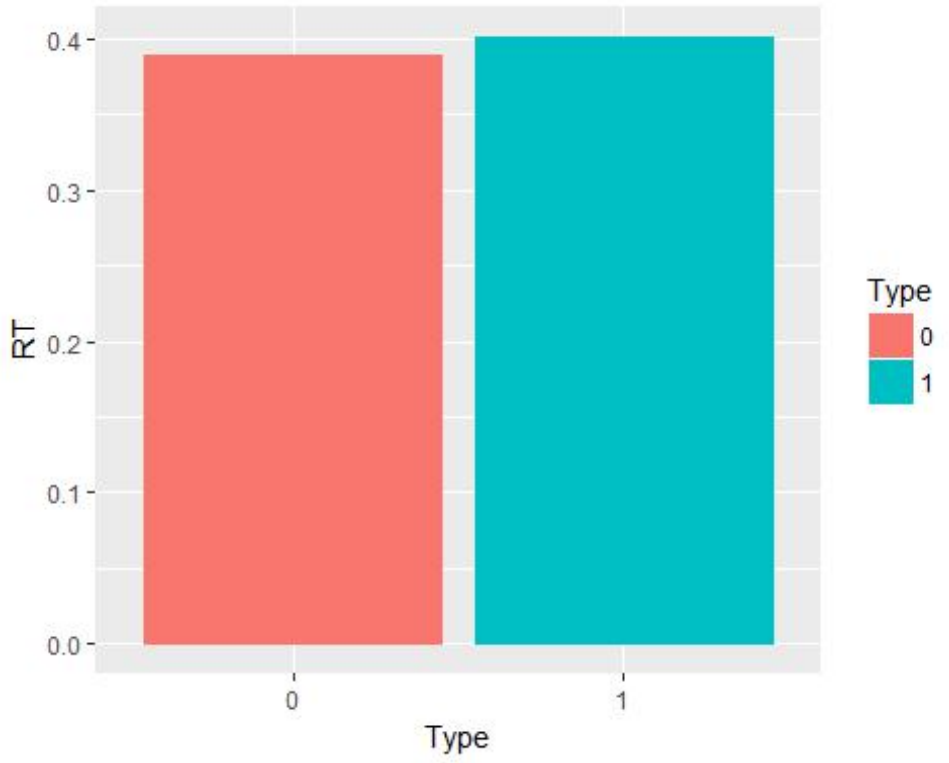

```
##   var frequency percentage cumulative_perc
## 1   4        76      28.79           28.79
## 2   5        70      26.52           55.31
## 3   6        48      18.18           73.49
## 4   3        36      13.64           87.13
## 5   7        21       7.95           95.08
## 6   1         6       2.27           97.35
## 7   8         3       1.14           98.49
## 8   9         3       1.14           99.63
## 9   2         1       0.38          100.00
```

Frequency plot shows us that,the proportion of hunger=4 and hunger=5 is almost 50%,and hunger=4 is the most hightest.hunger=2 is 0.38%,it is almost equal to zero.so Hunger distibutes imbalance.

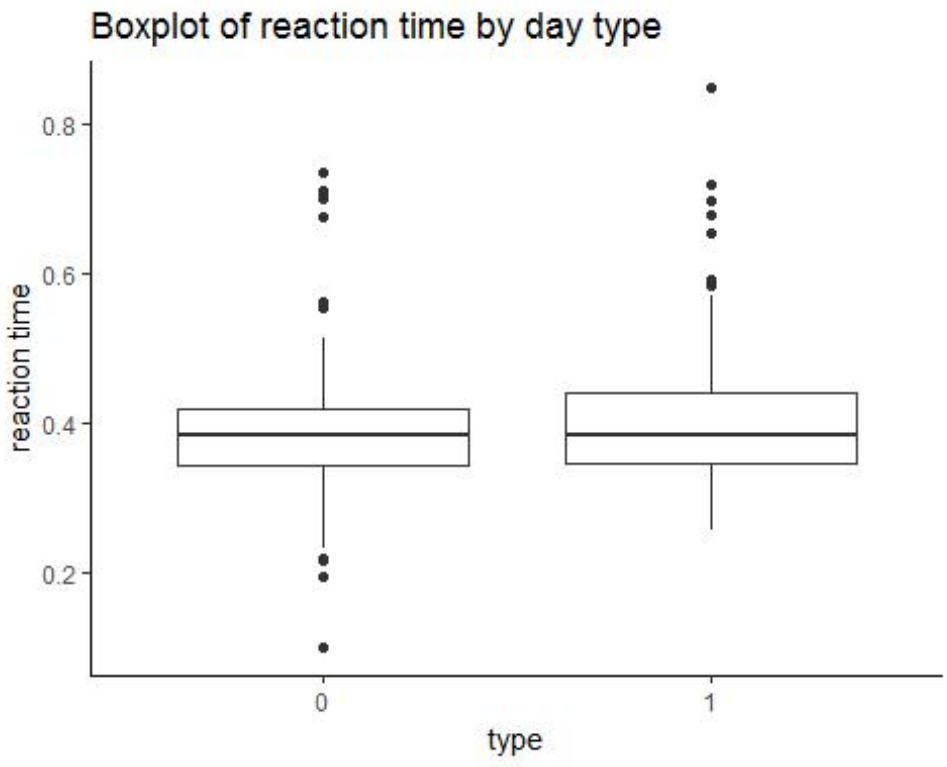
5.busyOrlight

I used number 0 to represent light day and 1 for busy day. According to the plots below, it is easy to find that normally people react faster in light days.

```
day_type = data$busyOrlight
af$Type <- as.factor(af$busyOrlight)
ggplot(summarise(group_by(af, Type), RT = mean(RT)),aes(x= Type, y = RT))
 + geom_bar(stat = "identity", position = position_stack(), aes(color =
Type, fill = Type))
```




```r
ggplot(data,aes(x = factor(day_type),y = reaction_time)) +
  theme_classic()  +
  geom_boxplot() +
  labs(title = "Boxplot of reaction time by day type",
       x = "type",
       y = "reaction time")
```

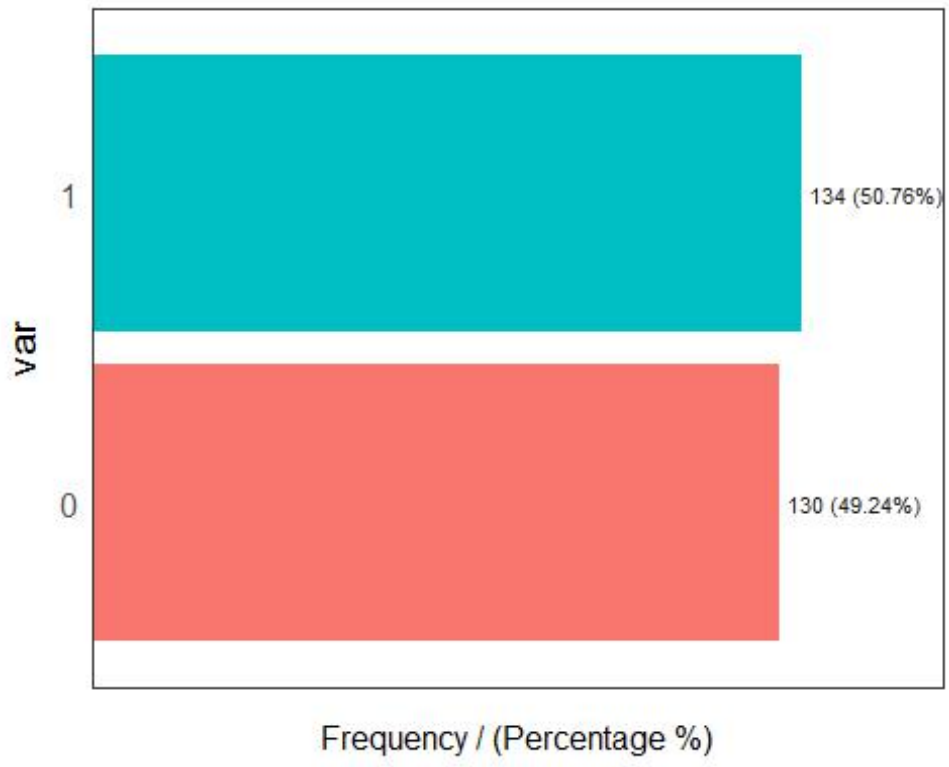


Barplot and boxplot tell us that,reaction time is smaller in light day,which indicates busy to reduce the reaction.It is consistent with common sense.

```
summarise(group_by(af, Type), mean(RT))
```

```
## # A tibble: 2 x 2
##     Type `mean(RT)`
##   <fctr>      <dbl>
## 1      0  0.3896308
## 2      1  0.4014104
```

```
freq(data$busyOrlight)
```



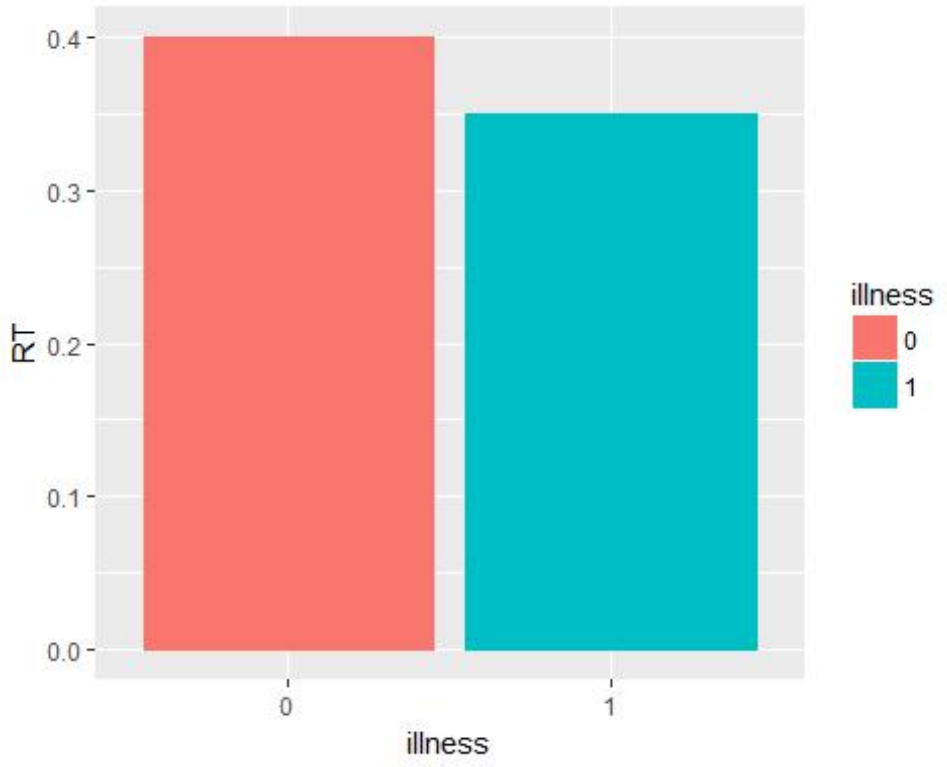

```
##   var frequency percentage cumulative_perc
## 1   1       134      50.76           50.76
## 2   0       130      49.24          100.00
```

The sample proportion of light day and busy data in data is almost the same,about 50%,it distribute uniformally.

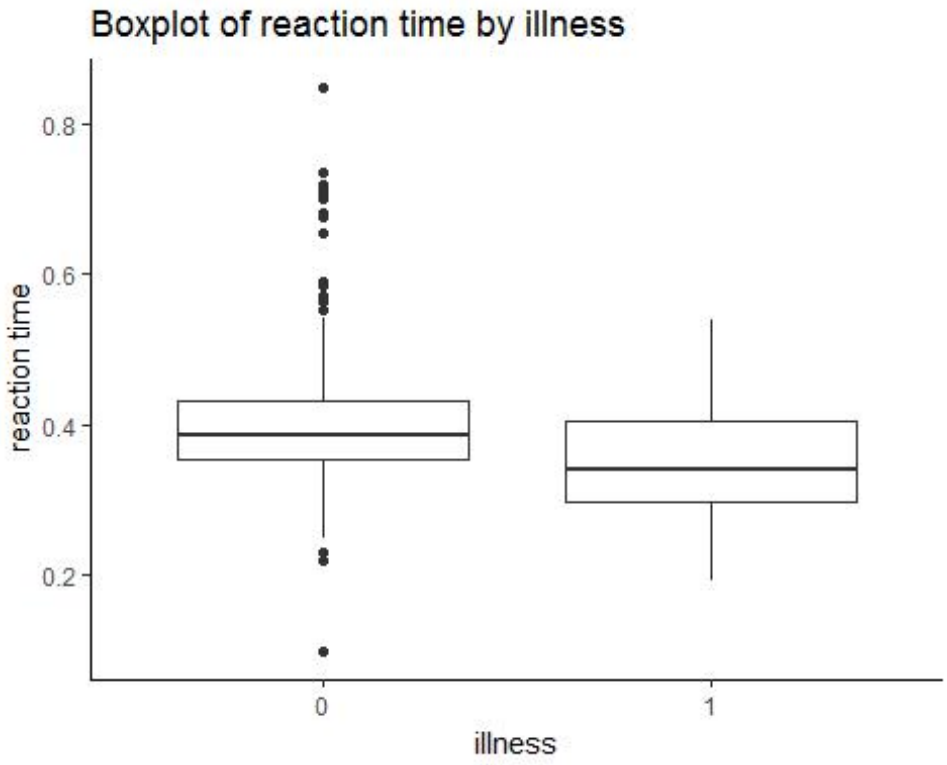
6.illness

illness=1,represnet normal,illness=0,represent sick.

```
af$illness <- as.factor(af$illness)
ggplot(summarise(group_by(af, illness), RT = mean(RT)),aes(x= illness,
y = RT)) + geom_bar(stat = "identity", position = position_stack(), aes
(color = illness, fill = illness))
```




```
ggplot(data,aes(x = factor(illness),y = reaction_time)) +
  theme_classic()  +
  geom_boxplot() +
  labs(title = "Boxplot of reaction time by illness",
       x = "illness",
       y = "reaction time")
```

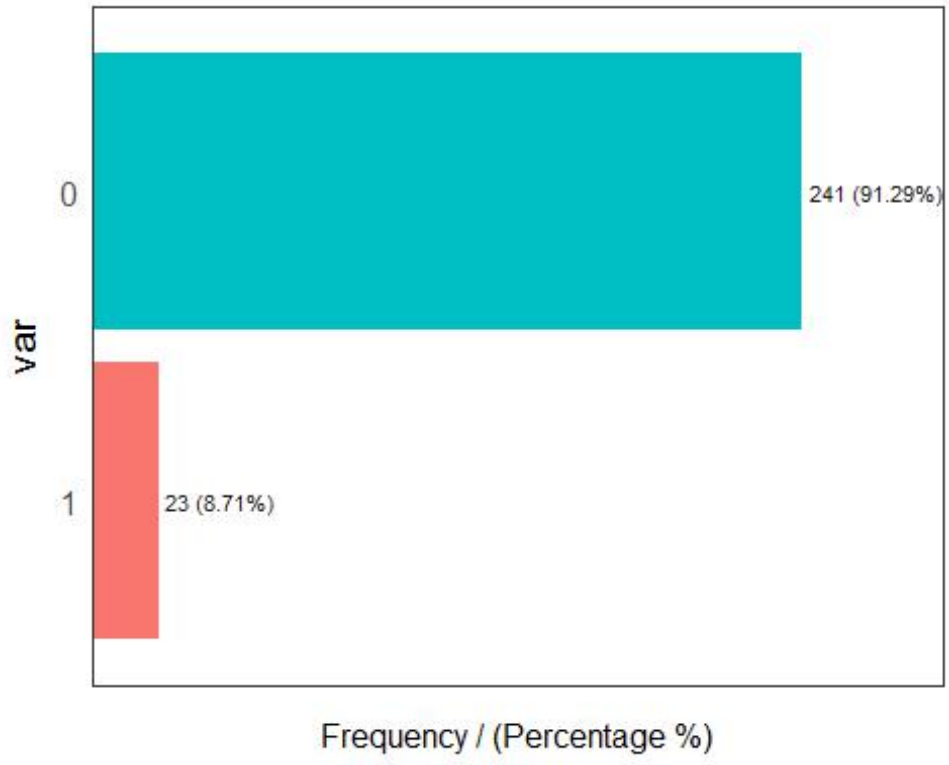


Accordind to the barplot and boxplot,the reaction time on illness=1 is lower than illness=0,the mean between them is the same,which also say,when people are sick ,they reacte slowly,it is consistent with common sense.

```
summarise(group_by(af, illness), mean(RT))
```

```
## # A tibble: 2 x 2
##   illness `mean(RT)`
##    <fctr>      <dbl>
## 1       0  0.4000041
## 2       1  0.3495652
```

```
freq(data$illness)
```



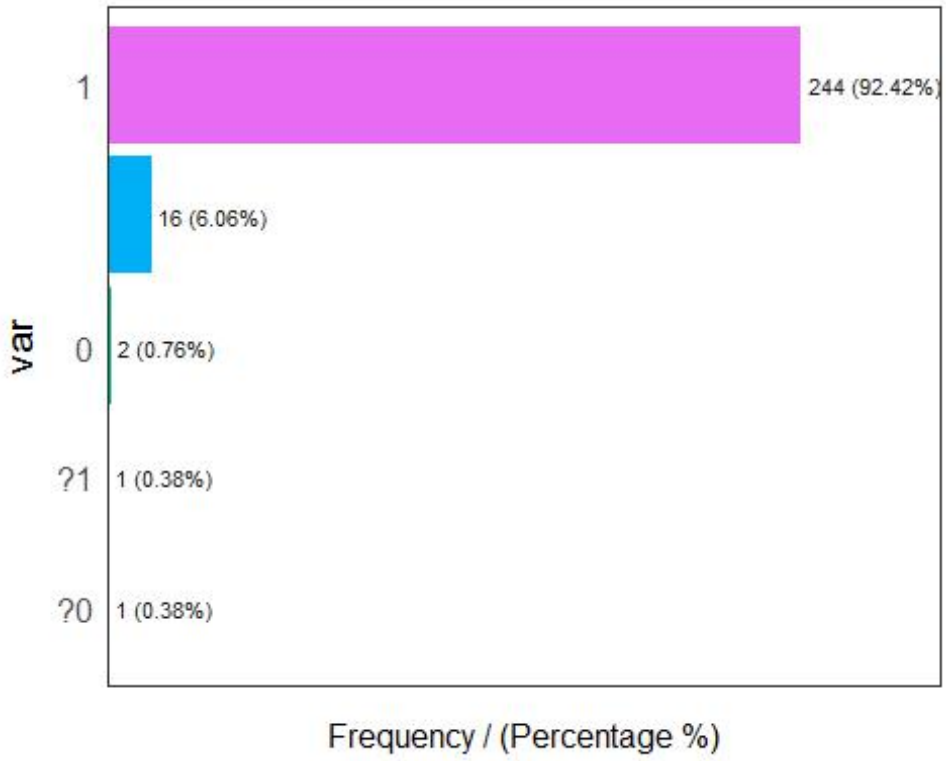

```
##   var frequency percentage cumulative_perc
## 1   0       241      91.29           91.29
## 2   1        23       8.71          100.00
```

The sample proportion of illness=1 is almostly 91%,it tell us that the sample record is not inbalanced .It may be effect the survey?

7.Protocol

Most of the people record themself following the protocol when testing reaction time.(protocal=1,represent comply with)

```
protocol = data$Protocol
summary(data$Protocol)

##      ?0  ?1   0   1
##  16   1   1   2 244

freq(data$Protocol)
```



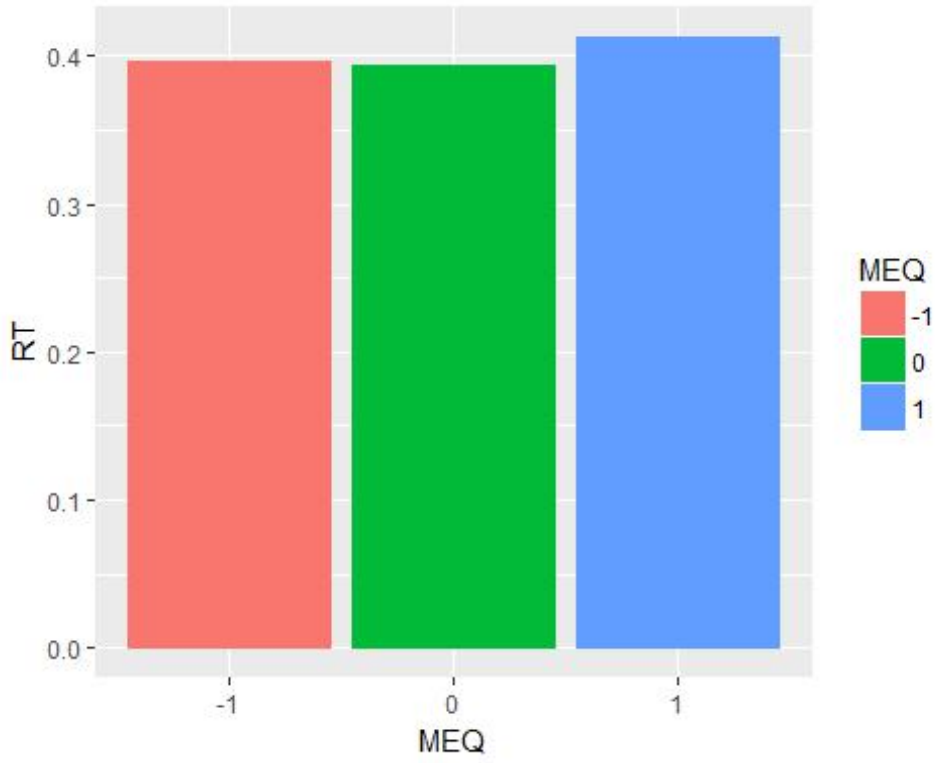

```
##   var frequency percentage cumulative_perc
## 1   1       244      92.42           92.42
## 2            16       6.06           98.48
## 3   0         2       0.76           99.24
## 4  ?0         1       0.38           99.62
## 5  ?1         1       0.38          100.00
```

Accorind to the summary of protocol and frequency plot,the sample proportion of protolcol=1 is almost 92%,and the missing record is about 6%, and unclear record has each one.These indicates that this protocol reocord is not good enough.It need to be processed with other method remove it directly.

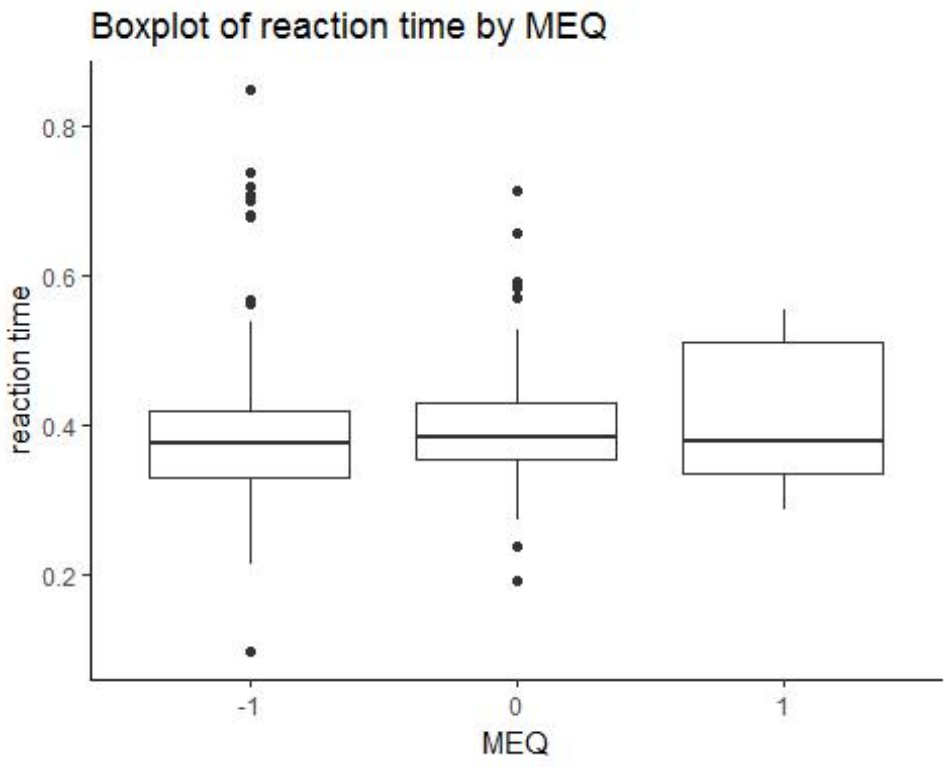
8.MEQ

(night:-1 normal:0 morning:1)

```
af$MEQ <- as.factor(af$MEQ)
ggplot(summarise(group_by(af, MEQ), RT = mean(RT)),aes(x= MEQ, y = RT))
+ geom_bar(stat = "identity", position = position_stack(), aes(color = M
EQ, fill = MEQ))
```




```
ggplot(data,aes(x = factor(MEQ),y = reaction_time)) +
  theme_classic()  +
  geom_boxplot() +
  labs(title = "Boxplot of reaction time by MEQ",
       x = "MEQ",
       y = "reaction time")
```

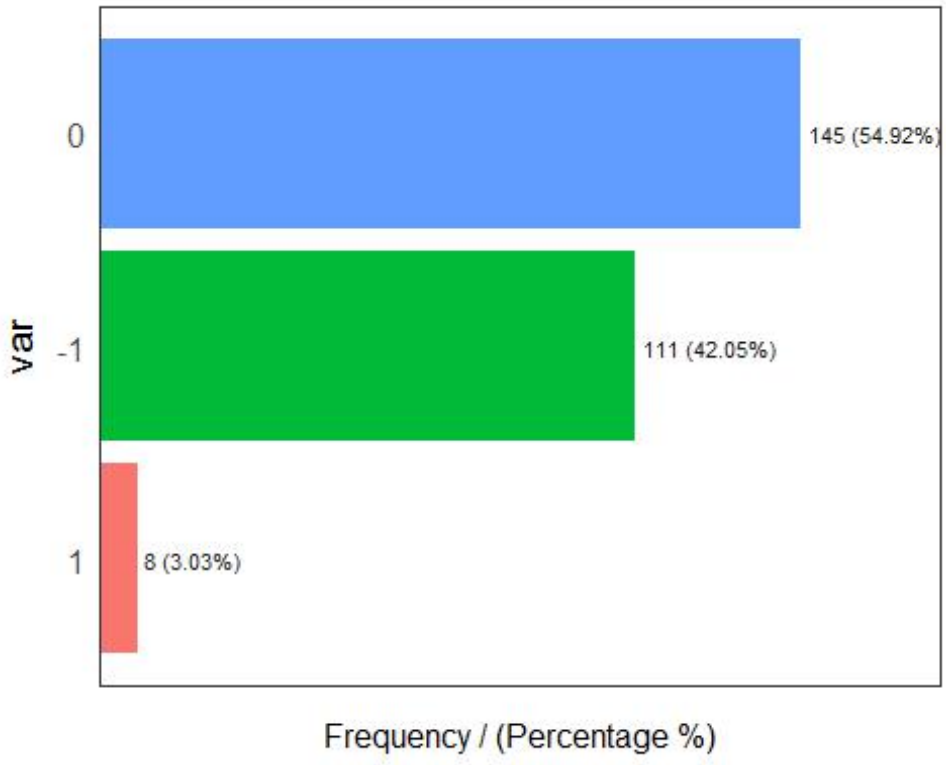


Above plot show that, reaction on night(MEQ=-1) time is slightly shorter than the other two kinds.Normal(MEQ=1) is the most highest.

```
summarise(group_by(af, illness), mean(RT))
```

```
## # A tibble: 2 x 2
##   illness `mean(RT)`
##    <fctr>      <dbl>
## 1       0  0.4000041
## 2       1  0.3495652
```

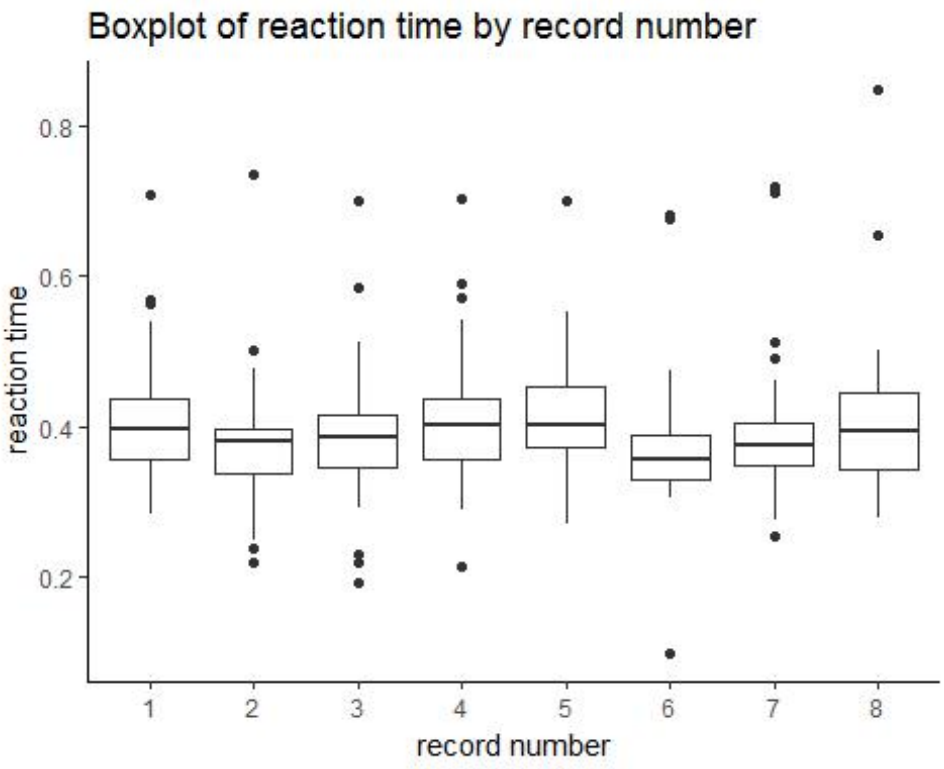
```
freq(data$MEQ)
```




```
##   var frequency percentage cumulative_perc
## 1   0       145      54.92           54.92
## 2  -1       111      42.05           96.97
## 3   1         8       3.03          100.00
```

9.Record

```
record = data$Record
ggplot(data,aes(x = factor(record),y = reaction_time)) +
  theme_classic()  +
  geom_boxplot() +
  labs(title = "Boxplot of reaction time by record number",
       x = "record number",
       y = "reaction time")
```



Here is the boxplot of reaction time for 8 trials, the first four tests are from the first day, and the other four are from the second day people choose. There is a pattern that the middle two tests have shorter reaction time. And the mean reaction time accross them some differ slightly.It is interesting for us.

Explore reaction time pattern,fit mixed model

This step,we will use the variable which are explored in the first step to fit the mixed model.It is also a process of exploration for reaction time pattern.Try to find the best mixed model bellow.

```
library(nlme)
```

```
##
## Attaching package: 'nlme'
```

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

```
library(lme4)
```

```
## Warning: package 'lme4' was built under R version 3.4.2
```

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

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

##
## Attaching package: 'lme4'

## The following object is masked from 'package:nlme':
##
##     lmList
```

As previously explored, variable Protocol has a lot of unclear record,and here,I handle it:

- unknow record-> 2
- ?0->0
- ?1->1

```
protocol=as.character(af$Protocol)
protocol=ifelse(protocol=="",2,ifelse(protocol=="?0",0,ifelse(protocol=
="?1",1,ifelse(protocol=="0",0,1))))
af$Protocol=as.factor(as.integer(protocol))
#af$ID=as.factor(af$ID)
af$Fatigue=as.factor(af$Fatigue)
predictor=c("ID","Record","Stimulant","Fatigue","Hunger","Type","illnes
s","Sleep","MEQ","Protocol")
```

step 1) Fit All model

Use "ID,Record","Stimulant","Fatigue","Hunger","Type","illness","Sleep","MEQ","Protocol " as fixed predictor,the "Record" and "ID" as random effect variable.

Fit full models

We know that ,ID means sample,and the record is repeat observation many times in each ID,So record is random effect.

```
model.full=lmer(RT~Stimulant+Fatigue+Hunger+Type+illness+Sleep+MEQ+Prot
ocol+(1|Record)+(1|ID),data = af,method="ML") #fit a linear model with a
 varying-intercept group effect using the variable Record.

## fixed-effect model matrix is rank deficient so dropping 6 columns / c
oefficients

AIC(model.full)

## [1] -552.3867

anova(model.full)

## Analysis of Variance Table
##           Df   Sum Sq   Mean Sq F value
## Stimulant  1 0.009620 0.0096202  3.5734
```

```
## Fatigue    6 0.162758 0.0271263 10.0761
## Hunger     8 0.032910 0.0041138  1.5281
## Type       1 0.004284 0.0042836  1.5912
## illness    1 0.017380 0.0173804  6.4560
## MEQ        2 0.001467 0.0007335  0.2725
## Protocol   2 0.010905 0.0054523  2.0253
```

Variable selection

Consider the collinearity across predictors is the main reason effect the model,so variable selection is neccessary,and it is a way to avoid this problem.

when applying this function to my reduced model, I got vif values for each of the variables. When vif>5 for a predictor, it probably should be removed. In case multiple variables have a vif>5, I first remove the predictor with the highest vif, then re-run lmer en vif.mer. I remove again the predictor with highest vif (if one or more predictors have still a vif>5), and I repeat this until none of the remaining predictors has a vif>5.

```
vif.mer <- function (fit) {
    ## adapted from rms::vif

    v <- vcov(fit)
    nam <- names(fixef(fit))

    ## exclude intercepts
    ns <- sum(1 * (nam == "Intercept" | nam == "(Intercept)"))
    if (ns > 0) {
        v <- v[-(1:ns), -(1:ns), drop = FALSE]
        nam <- nam[-(1:ns)]
    }

    d <- diag(v)^0.5
    v <- diag(solve(v/(d %o% d)))
    names(v) <- nam
    v
}

kappa.mer <- function (fit,
                       scale = TRUE, center = FALSE,
                       add.intercept = TRUE,
                       exact = FALSE) {
    X <- fit@pp$X
    nam <- names(fixef(fit))

    ## exclude intercepts
    nrp <- sum(1 * (nam == "(Intercept)"))
    if (nrp > 0) {
        X <- X[, -(1:nrp), drop = FALSE]
```

```
        nam <- nam[-(1:nrp)]
    }

    if (add.intercept) {
        X <- cbind(rep(1), scale(X, scale = scale, center = center))
        kappa(X, exact = exact)
    } else {
        kappa(scale(X, scale = scale, center = scale), exact = exact)
    }
}

colldiag.mer <- function (fit,
                          scale = TRUE, center = FALSE,
                          add.intercept = TRUE) {
    ## adapted from perturb::colldiag, method in Belsley, Kuh, and
    ## Welsch (1980).  look for a high condition index (> 30) with
    ## more than one high variance propotion.  see ?colldiag for more
    ## tips.
    result <- NULL
    if (center)
        add.intercept <- FALSE
    if (is.matrix(fit) || is.data.frame(fit)) {
        X <- as.matrix(fit)
        nms <- colnames(fit)
    }
    else if (class(fit) == "mer") {
        nms <- names(fixef(fit))
        X <- fit@X
        if (any(grepl("(Intercept)", nms))) {
            add.intercept <- FALSE
        }
    }
    X <- X[!is.na(apply(X, 1, all)), ]

    if (add.intercept) {
        X <- cbind(1, X)
        colnames(X)[1] <- "(Intercept)"
    }
    X <- scale(X, scale = scale, center = center)

    svdX <- svd(X)
    svdX$d
    condindx <- max(svdX$d)/svdX$d
    dim(condindx) <- c(length(condindx), 1)

    Phi = svdX$v %*% diag(1/svdX$d)
    Phi <- t(Phi^2)
    pi <- prop.table(Phi, 2)
    colnames(condindx) <- "cond.index"
```

```
    if (!is.null(nms)) {
        rownames(condindx) <- nms
        colnames(pi) <- nms
        rownames(pi) <- nms
    } else {
        rownames(condindx) <- 1:length(condindx)
        colnames(pi) <- 1:ncol(pi)
        rownames(pi) <- 1:nrow(pi)
    }

    result <- data.frame(cbind(condindx, pi))
    zapsmall(result)
}

maxcorr.mer <- function (fit,
                         exclude.intercept = TRUE) {
    so <- summary(fit)
    corF <- so@vcov@factors$correlation
    nam <- names(fixef(fit))

    ## exclude intercepts
    ns <- sum(1 * (nam == "Intercept" | nam == "(Intercept)"))
    if (ns > 0 & exclude.intercept) {
        corF <- corF[-(1:ns), -(1:ns), drop = FALSE]
        nam <- nam[-(1:ns)]
    }
    corF[!lower.tri(corF)] <- 0
    maxCor <- max(corF)
    minCor <- min(corF)
    if (abs(maxCor) > abs(minCor)) {
        zapsmall(maxCor)
    } else {
        zapsmall(minCor)
    }
}
```

Step 1

- run vif.mer function for full model,remove the variable if this variable's vif >5.0

```
vif.step.1=vif.mer(model.full)
names(vif.step.1)[which(vif.step.1>5.0)]

## [1] "Hunger3" "Hunger4" "Hunger5" "Hunger6" "Hunger7"
```

According the result got from step vif process,consider remove variable protocol and Hunger from full model,and re-run new mixwd model

Step 2

- re-run model from Step,remove Protocol and Hunger

```
model.reduce.1=lmer(RT~Stimulant+Fatigue+Type+illness+Sleep+MEQ+Protocol+(1|Record)+(1| ID),data = af,method="ML")

## fixed-effect model matrix is rank deficient so dropping 6 columns / coefficients
```

Step 3

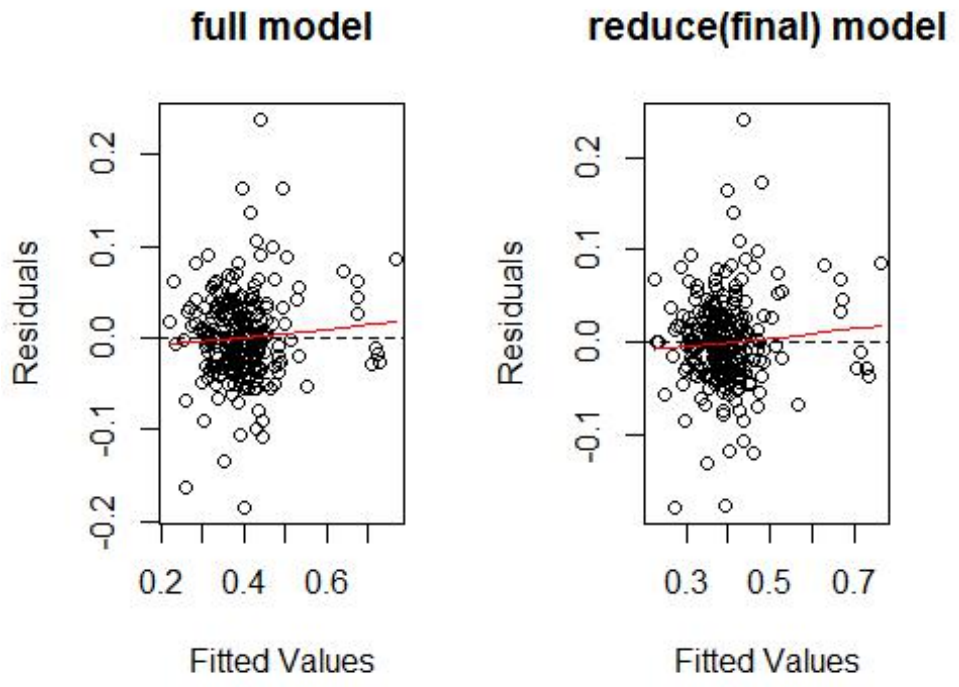
- run vif.mer function for full model,remove the variable if this variable's vif >5.0 from step 2

```
vif.step.2=vif.mer(model.reduce.1)
names(vif.step.2)[which(vif.step.2>5.0)]

## character(0)
```

In the Step 3,run vif process,and the result tell us that,there is not variable's vif >5.0,so we stop re-run model and do not remove any variable from model got from step 2,it is the final model.

Plot resudual vs fiited value

```
par(mfrow=c(1,2))
plot(fitted(model.full), residuals(model.full), xlab = "Fitted Values",
 ylab = "Residuals",main="full model")
abline(h = 0, lty = 2)
lines(smooth.spline(fitted(model.full), residuals(model.full)),col="red")



plot(fitted(model.reduce.1), residuals(model.reduce.1), xlab = "Fitted Values", ylab = "Residuals",main="reduce(final) model")
abline(h = 0, lty = 2)
lines(smooth.spline(fitted(model.reduce.1), residuals(model.reduce.1)), col="red")
```



The two graph of fiited value vs residual is similair,ans residual is almost around 0,which indicates the two model perform similialy and perform not bad.

Compare reduce model and full model

```
logLik(model.full)

## 'log Lik.' 301.1934 (df=25)

logLik(model.reduce.1)

## 'log Lik.' 319.1131 (df=17)

sprintf("The R square of full model is :%f",cor(af$RT,fitted(model.full))
^2)

## [1] "The R square of full model is :0.763350"

sprintf("The R square of reduce  model is :%f",cor(af$RT,fitted(model.re
duce.1))^2)

## [1] "The R square of reduce  model is :0.751778"
```

The likelihood of full model is 301.193366,and reduce model is 319.1130786., the higher likelihood value ,the better model.And the R square of them has almost no difference,but reeduce model has few coefficients.

Explain the model

```
print(lme4::fixef(model.reduce.1))  # print fixed effect coefficients
```

```
##  (Intercept)   Stimulant1     Fatigue2     Fatigue3     Fatigue4
##  0.410915132 -0.012594079  0.034450824  0.050893967  0.063700427
##     Fatigue5     Fatigue6     Fatigue7        Type1     illness1
##  0.090204197  0.092075670  0.148980036  0.005479846 -0.065998428
##         MEQ0         MEQ1    Protocol1    Protocol2
## -0.015022736  0.011525765 -0.059485648 -0.120272037
```

According to the data,wo know that

- Stimulant: Stimulant=1 means use stimulate,and 0 means do not use.
- Fatigue :The higher the value, the higher degree of fatigue
- Type(busy or ligtht): Type=1,mean busy,0 mean light
- illness: 0 mean health,1 mean sick
- MEQ: 0 mean normal,1 mean morning
- Protocol: 0:not comply,1:comply,2:unknow

Accoring to the fiexed effect and random effect coefficients of print,the coefficient of Stimulant=1 is negative,which mean when using stimulate has negative effect on reaction,will reacte slowly;When Fatigue vary from 2 to 7,cofficient is from small to large,which mean more fatigue,reacte fastser,I think it is consistent with what we know(common sense);The coefficient of Type=1 is positive,which mean busy is helpful to reacte quickly.When sick(coef of illness=1 is negative),reaction time will become longer.moring questionnare will reacte faster.And when comply with protocol,is has positive effect on reaction time,which mean will reacte faster,otherwise.

In a word,the variable I seleced in the final model is reasonable,and they actualli can explain model well.So I use the reduce model as final model.