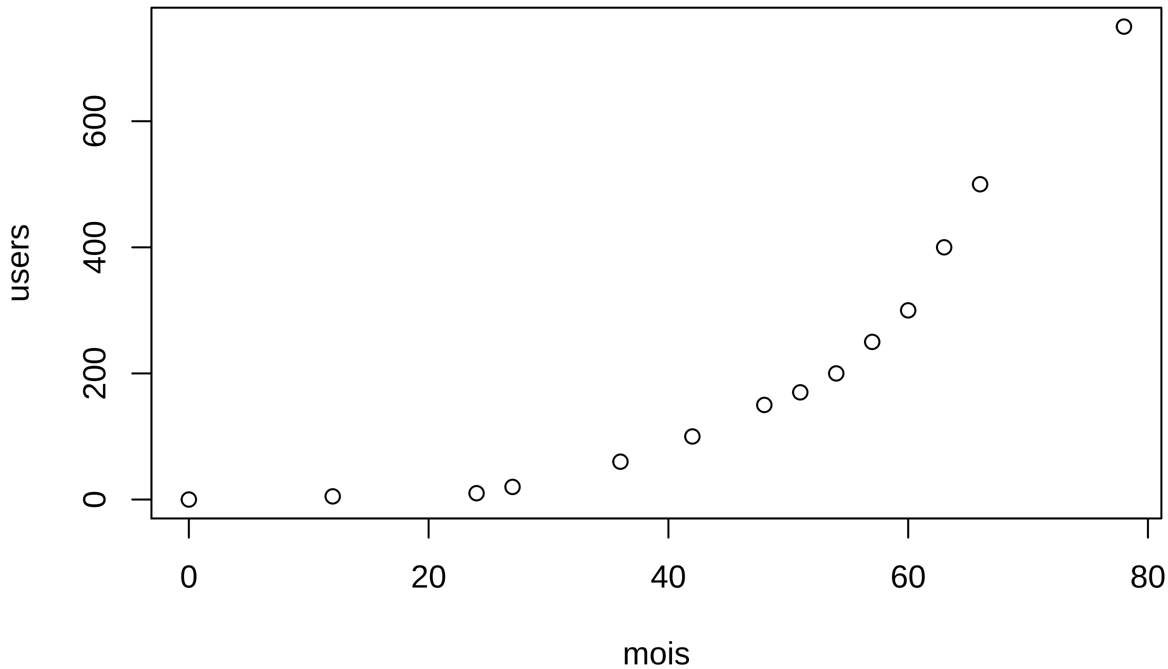
TP1 MRR

Zeyu CHEN and Clement Veyssiere

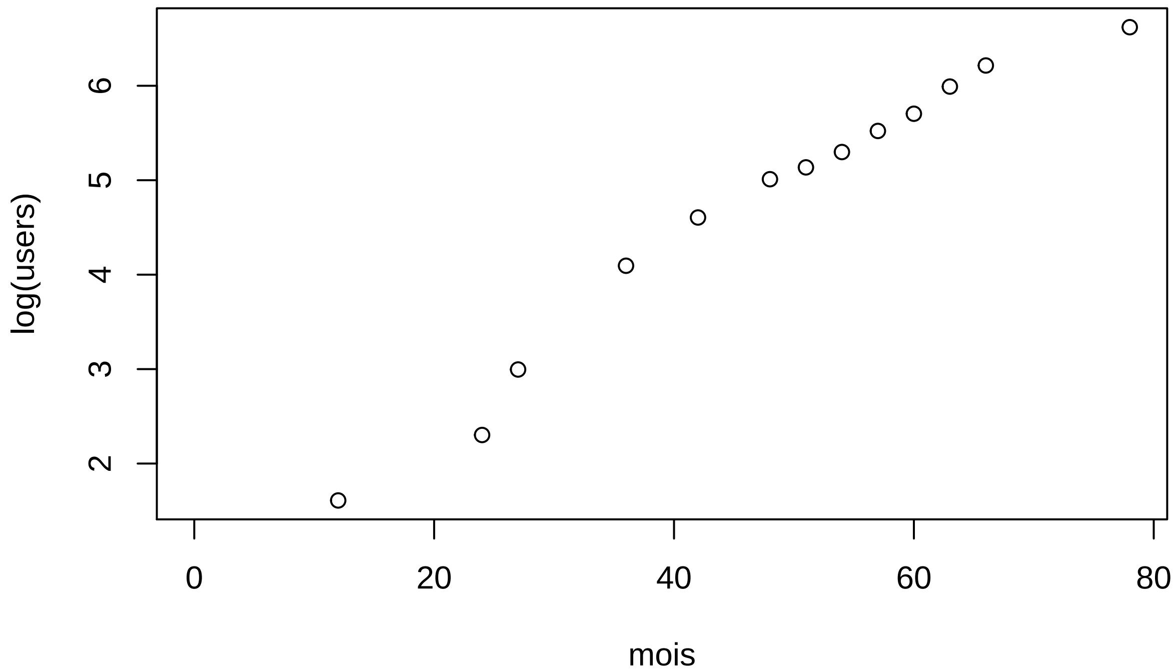
2018/9/27

IV Facebook data set

```r
# read the data
tab <- read.table("data/facebookdata.txt",sep = ";",header = TRUE)

y <- tab$users
plot(tab$mois,y,xlab = "mois",ylab = "users")
```




```r
# as we can see from the plot , the number of uses and mois is not linear. So, we try to
# transform the target variable to make them be linear.
log_y <-log(y)
plot(tab$mois,log_y,xlab = "mois",ylab="log(users)")
```




```r
#tabN[1,1] is INF , we replace it by 0
tabN <- data.frame(log_y,tab$mois)
tabN[1,1] <- 0

#Now we can model
modFacebook <- lm(tabN)
modFacebook
```

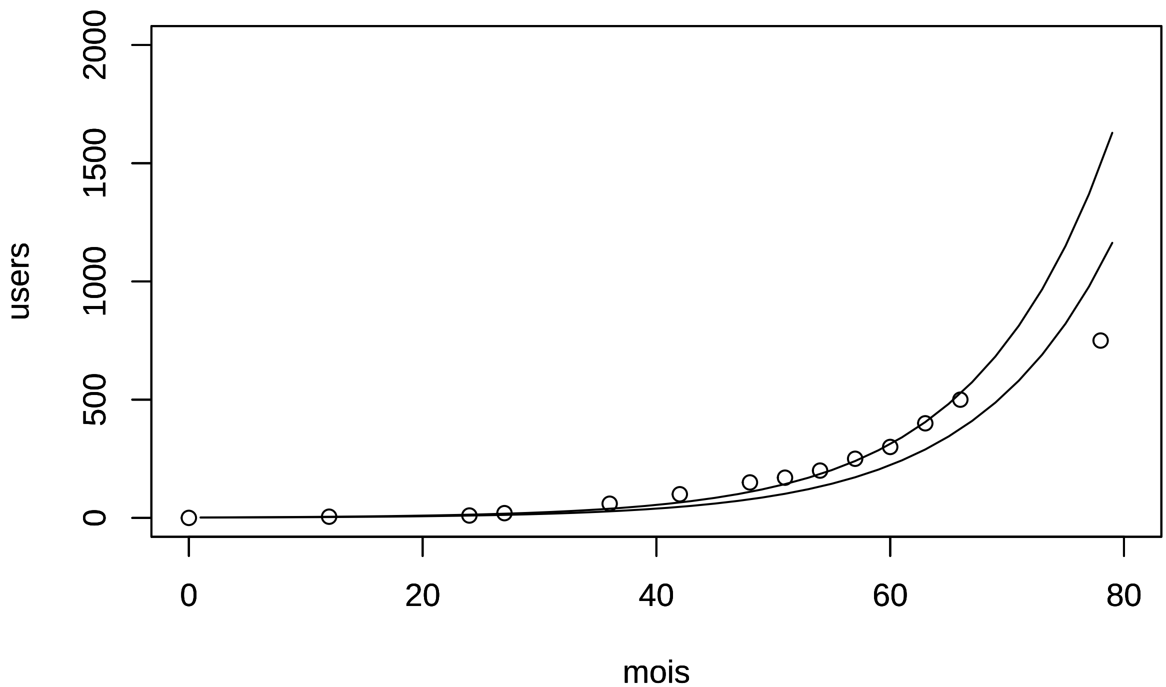
```
##
## Call:
## lm(formula = tabN)
##
## Coefficients:
## (Intercept)     tab.mois
##     0.52612      0.08695
```

```r
summary(modFacebook)
```

```
##
## Call:
## lm(formula = tabN)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.68848 -0.04777  0.03941  0.16172  0.43787
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.526123   0.208876   2.519    0.027 *
## tab.mois    0.086954   0.004264  20.391 1.11e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3387 on 12 degrees of freedom
```

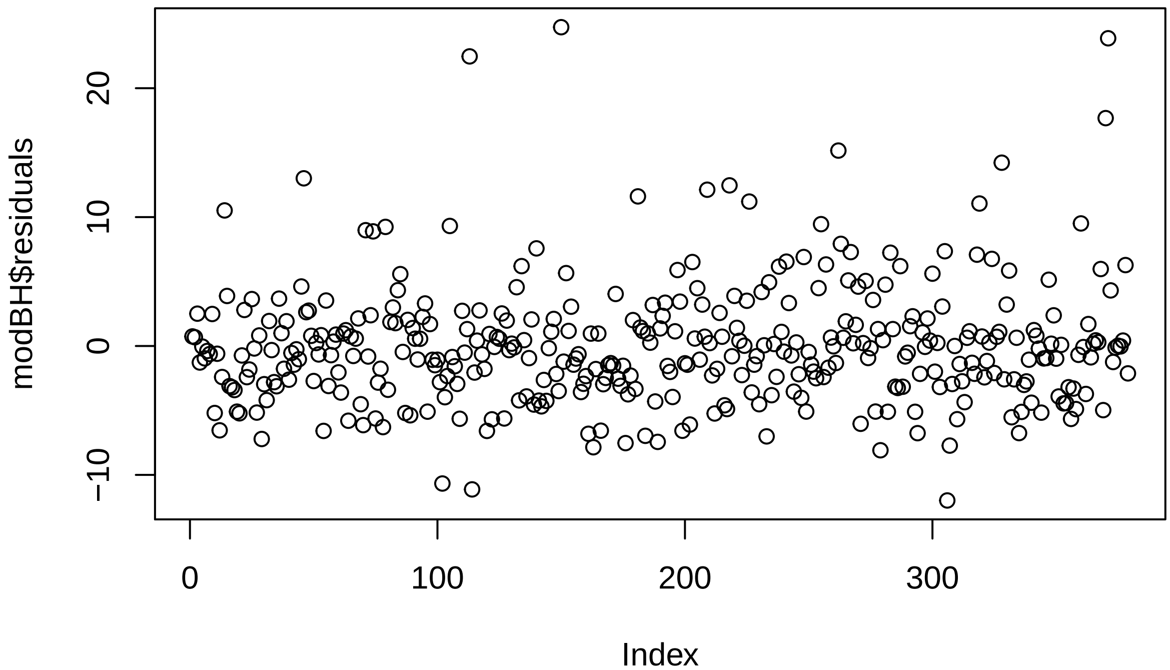
```
## Multiple R-squared:  0.9719, Adjusted R-squared:  0.9696
## F-statistic: 415.8 on 1 and 12 DF,  p-value: 1.113e-10
```

```
newdata <- data.frame(tab.mois = seq(1,80,2))
predictValue <- predict.lm(modFacebook, newdata = newdata)
y <- exp(predictValue)

#we plot the observation and our model together
plot(tab$mois,tab$users,xlab = "mois",ylab = "users",xlim= c(0,80),ylim = c(0,2000))
par(new=TRUE)
plot(seq(1,80,2),y,ylim = c(0,2000),xlim= c(0,80),xlab = "mois",ylab = "users",type='l')
par(new=TRUE)
#Our model fit well in most of time besides the last month.But ,if we can ajuste
# a litte out parametre.We can have a better result.
plot(seq(1,80,2),y/1.4,ylim = c(0,2000),xlim= c(0,80),xlab = "mois",ylab = "users",type='l')
```

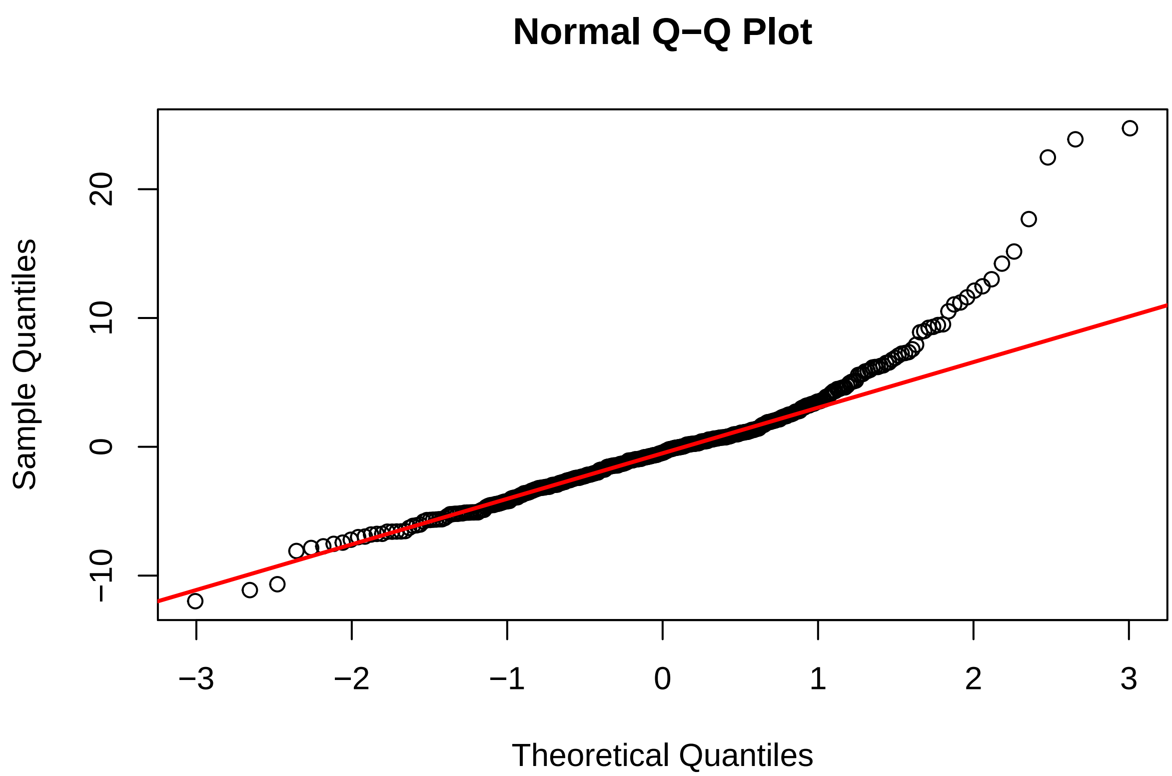


V Boston housing data

```
library(mlbench)
data(BostonHousing)

sub <- sample(nrow(BostonHousing), 0.75 * nrow(BostonHousing))
TabTrain <- BostonHousing[sub,]
TabTest <- BostonHousing[-sub,]
# model with TabTrain
modBH <- lm(medv~.,data=TabTrain)

#plot the residuals ,we can see that most of them are distributed around 0
plot(modBH$residuals)
```




```r
qqnorm(modBH$residuals)
qqline(modBH$residuals,col=2,lwd=2)
```

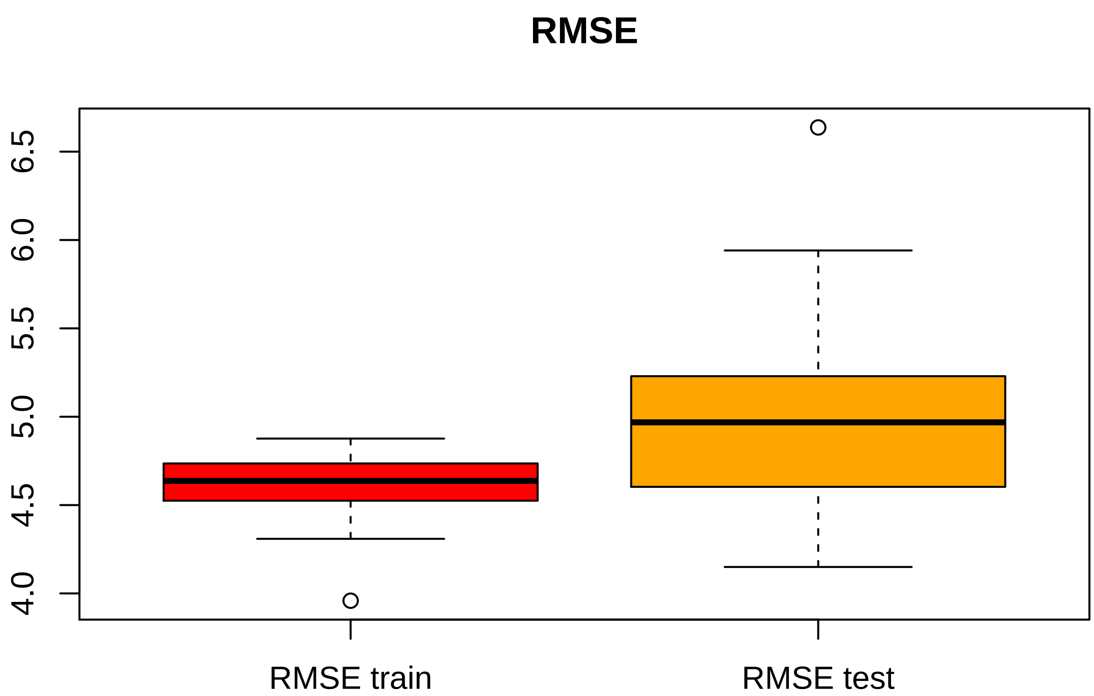



```r
#the shapiro.test shows that the distribution of res follow the law of normal
shapiro.test(modBH$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
```

```
## data:  modBH$residuals
## W = 0.90995, p-value = 3.059e-14
```

```r
# the RMSD vector is used to storing the rmsd calculated by the training data
RMSD <- vector(length = 50)

# the RMSDtest vector is used to storing the rmsd calculated by the test data
RMSDtest <- vector(length = 50)

Rsquaredadj <- vector(length = 50)
#Repeat ten times, each time we split randomly the dataset in to dataframes containing

#respectively 75% of the observation and 25% of the remaining observation.
for(i in 1:50)
{
# partitionning
sub <- sample(nrow(BostonHousing), 0.75 * nrow(BostonHousing))

TabTrain <- BostonHousing[sub,]

TabTest <- BostonHousing[-sub,]

# predict with Test data set
modBH <- lm(medv~.,data=TabTrain)

Rsquaredadj[i] <- summary(modBH)["adj.r.squared"]$adj.r.squared

sumOfSq <- sum((TabTrain$medv - predict(modBH,newdata = TabTrain))^2)
T <- length(TabTrain$medv)
# root mean square deviation
RMSD[i] <-  sqrt(sumOfSq/T)

predict(modBH,newdata = TabTest)
TabTest$medv
# root mean square deviation of test data
sumSqtest <- sum((TabTest$medv - predict(modBH,newdata = TabTest))^2)
Ttest <- length(TabTest$medv)
RMSDtest[i] <-  sqrt(sumSqtest/Ttest)
}

boxplot(RMSD, RMSDtest,
        main = "RMSE",
        names = c("RMSE train", "RMSE test"),
        col = c("red", "orange"))
```




```
#the mean of RMSD
sum(RMSD)/50
```

```
## [1] 4.62068
```

```
#the mean of RMSDtest
sum(RMSDtest)/50
```

```
## [1] 4.943468
```

```
#the mean of R squared adjustef
sum(Rsquaredadj)/50
```

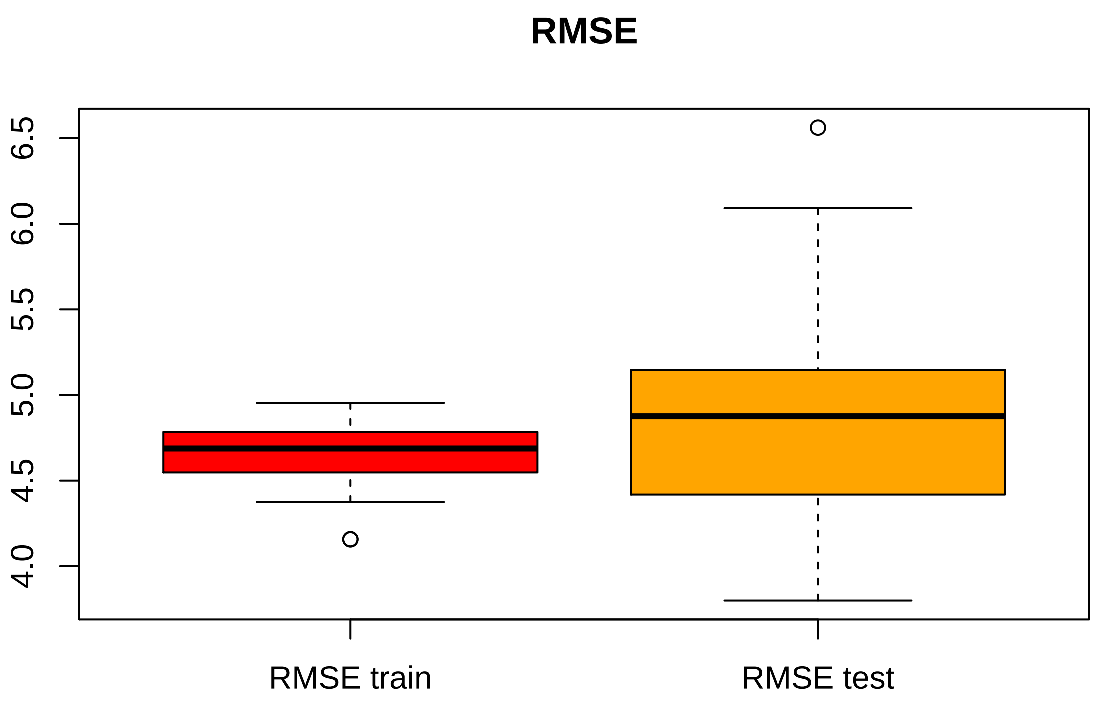
```
## [1] 0.734385
```

```
# As we can see , RMSE of test data is close to RMSE of training data.
```

We also try to use the stepwise selection for find a better model.

```
library(mlbench)
data(BostonHousing)
library(MASS)

# the RMSD vector is used to storing the rmsd calculated by the training data
RMSD <- vector(length = 50)

# the RMSDtest vector is used to storing the rmsd calculated by the test data
RMSDtest <- vector(length = 50)

Rsquaredadj <- vector(length = 50)

for (i in 1:50){
  sub = sample(nrow(BostonHousing),0.75*nrow(BostonHousing)) #split the dataset
```

```
  Tabtrain = BostonHousing[sub,]
  Tabtest = BostonHousing[-sub,]
  fit1 = lm(medv ~ .,Tabtrain)
  fit2 = lm(medv ~ 1,Tabtrain)
  #stepwise selection
  modreg = stepAIC(fit2,direction = "both", scope = list(upper=fit1, lower=fit2),    trace=FALSE)

  Rsquaredadj[i] <- summary(modBH)["adj.r.squared"]$adj.r.squared

  sumsq <- sum((Tabtrain$medv - predict(modreg,Tabtrain))^2)
  Ttrain <- length(Tabtrain$medv)
  RMSD[i] <- sqrt(sumsq/Ttrain)

  sumsqtest <- sum((Tabtest$medv - predict(modreg,Tabtest))^2)
  Ttest <- length(Tabtest$medv)
  RMSDtest[i] <- sqrt(sumsqtest/Ttest)
}

boxplot(RMSD, RMSDtest,
        main = "RMSE",
        names = c("RMSE train", "RMSE test"),
        col = c("red", "orange"))
```




```
#the mean of RMSD
sum(RMSD)/50
```

```
## [1] 4.64932
```

```
#the mean of RMSDtest
sum(RMSDtest)/50
```

```
## [1] 4.868103
```

```r
#the mean of R squared adjustef
sum(Rsquaredadj)/50
```

```
## [1] 0.7332361
```

We can see that this model is better with a smaller RMSDtest and R squared Ajusted