Practical session - Modèles de Régression Linéaire

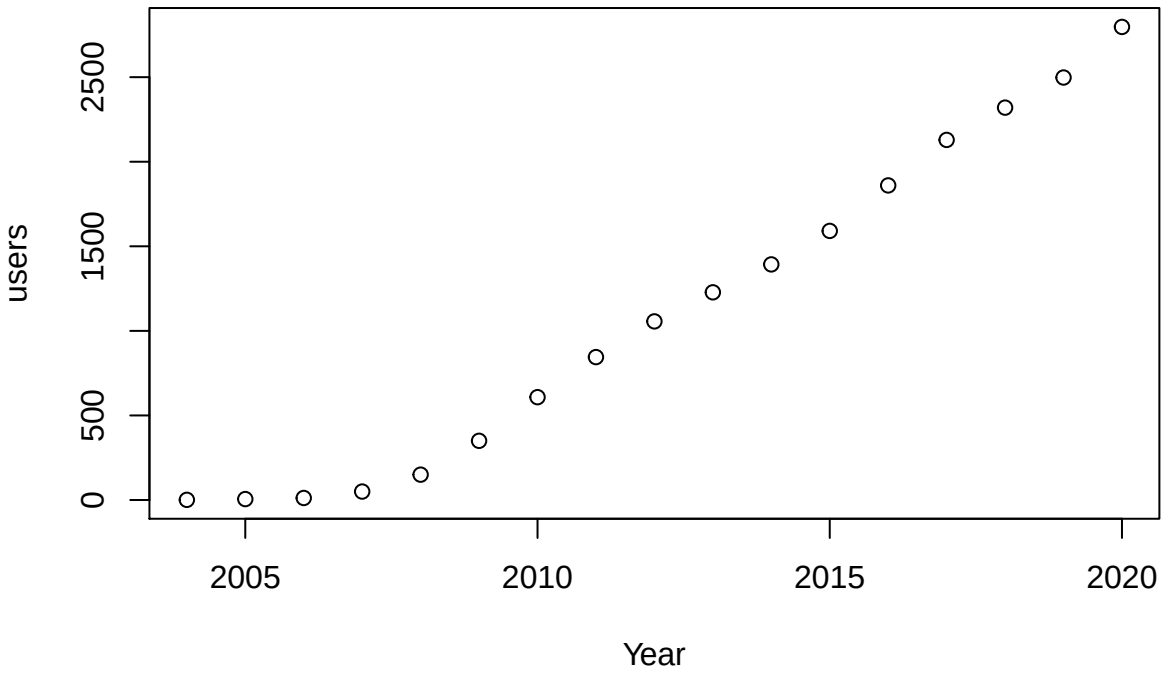
MECH Bealy et CHHEANG Vinha

27/09/2021

IV . Application: GAFAM or BATX data set

Application: Facebook data set

```
tab <- data.frame(read.table("./facebook_users.csv", header = TRUE, sep = ";"))
plot(tab)
```



We observe that our datas are nearly linear function. In order to get the best model, we just need to verify directly base on the data seems to be good enough to estimate with the up coming year.

```
mod1 = lm(users ~ Year, data = tab)
summary(mod1)
```

```
##
## Call:
## lm(formula = users ~ Year, data = tab)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -210.65  -93.75  -55.38   79.20  391.08
##
## Coefficients:
```

```
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.765e+05  1.615e+04  -23.31 3.40e-13 ***
## Year         1.877e+02  8.028e+00   23.38 3.25e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 162.2 on 15 degrees of freedom
## Multiple R-squared:  0.9733, Adjusted R-squared:  0.9715
## F-statistic: 546.5 on 1 and 15 DF,  p-value: 3.254e-13
```

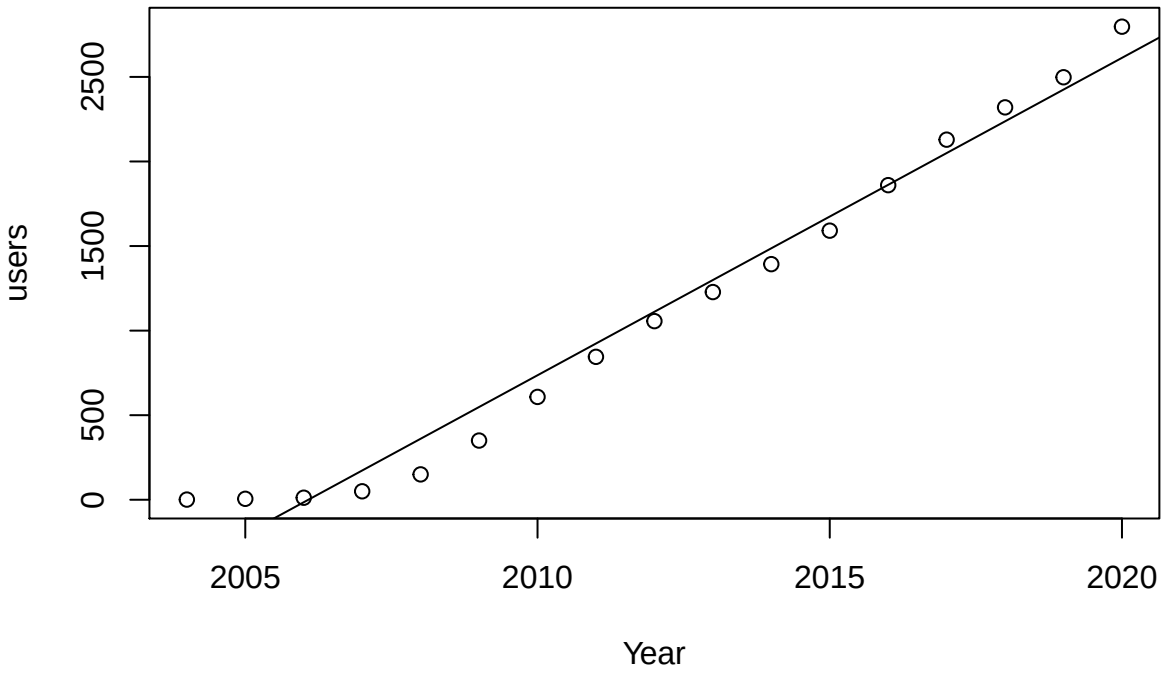
These codes below write to test statistic at level alpha = 0.001 of `X2` where `X2` refers to "Year".

```
Y = as.matrix(tab$users)
X1 = rep(1, 17)
X2 = tab$Year
X = matrix(data = c(X1,X2), ncol = 2)
beta_est = as.matrix(mod1$coefficients)
Y_est = X%*%beta_est

#test
n <- length(Y_est)
p <- length(beta_est)
eps <- sum((Y - Y_est)**2)
sigma2 <- eps/(n-p)
for(k in 2:p)
{
  if((beta_est[k]/sqrt(sigma2*(t(X)%*%X)[k,k]))> qt(1-0.001/2, df = n - p))
  {
    print(paste("Model does not depend on", paste("X",k)))
  }
  else
  {
    print(paste("Model depends on", paste("X",k)))
  }
}
```

```
## [1] "Model depends on X 2"
```

```
plot(tab)
abline(mod1$coefficients)
```




```
predict(mod1, data.frame(Year = 2020))
```

```
##        1
## 2612.843
```

According to our linear line, number of Facebook users estimated is more than 2500 millions users. (R2=0.9733)

V. Real estate data

```
tab1 = read.csv("./housedata.csv")
names(tab1)
```

```
##  [1] "id"            "date"          "price"         "bedrooms"
##  [5] "bathrooms"     "sqft_living"   "sqft_lot"      "floors"
##  [9] "waterfront"    "view"          "condition"     "grade"
## [13] "sqft_above"    "sqft_basement" "yr_built"      "yr_renovated"
## [17] "zipcode"       "lat"           "long"          "sqft_living15"
## [21] "sqft_lot15"
```

```
# dimension of data
dim(tab1)
```

```
## [1] 21613    21
```

```
Y = matrix(tab1[,3], nrow = 21613, ncol = 1 )
Y = tab1[ , 3]

X1 = tab1[ , 1]
X2 = tab1[ , 2]
X3 = tab1[ , 4]
X4 = tab1[ , 5]
X5 = tab1[ , 6]
X6 = tab1[ , 7]
X7 = tab1[ , 8]
X8 = tab1[ , 9]
```

```
X9 = tab1[ , 10]
X10 = tab1[ , 11]
X11 = tab1[ , 12]
X12 = tab1[ , 13]
X13 = tab1[ , 14]
X14 = tab1[ , 15]
X15 = tab1[ , 16]
X16 = tab1[ , 17]
X17 = tab1[ , 18]
X18 = tab1[ , 19]
X19 = tab1[ , 20]
X20 = tab1[ , 21]
```

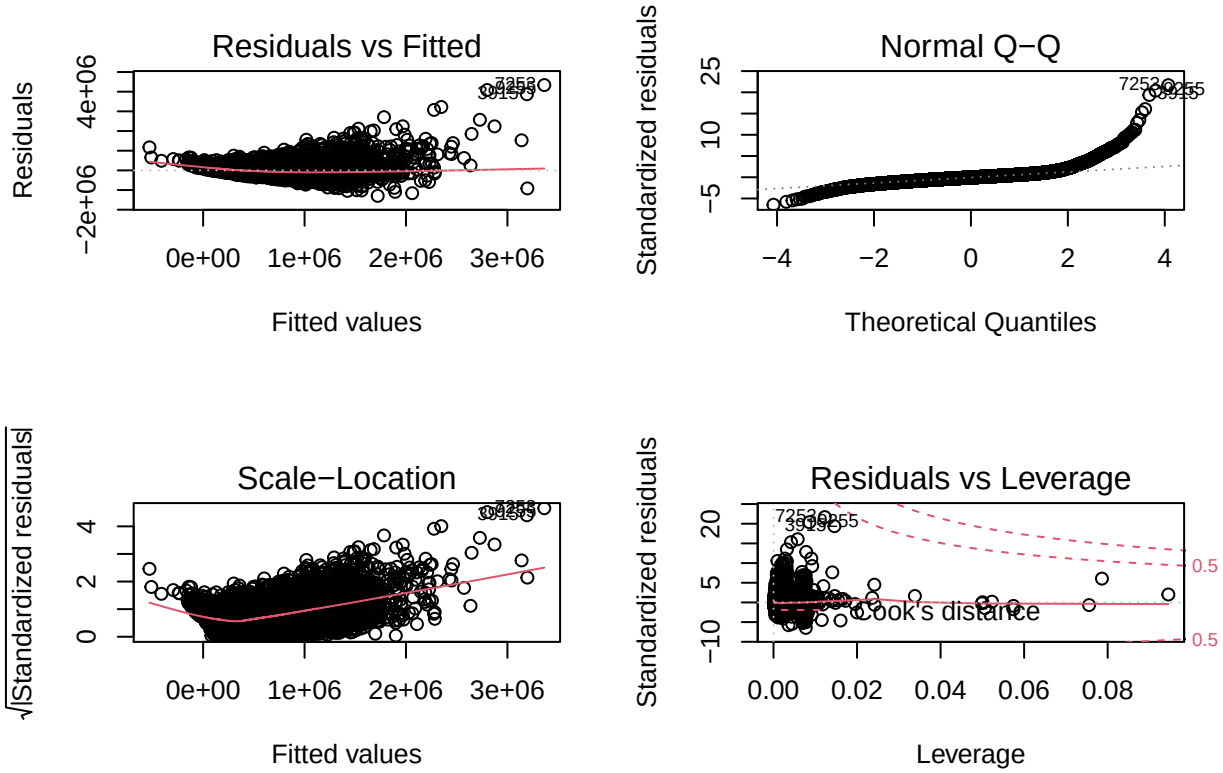
We don't include `X2` because data type of `X2` is not a number.

```
df <- data.frame(Y,X1,X3,X4,X5,X6,X7,X8,X9,X10,X11,X12,X14,X15,X16,X17,X18,X19,X20)
```

```
model = lm(Y~. , data = df)
summary(model)
```

```
##
## Call:
## lm(formula = Y ~ ., data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1296043   -99532    -9383    77195  4336404
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.939e+06  2.933e+06   2.366  0.01798 *
## X1          -1.289e-06  4.825e-07  -2.671  0.00757 **
## X3          -3.576e+04  1.892e+03 -18.907  < 2e-16 ***
## X4           4.116e+04  3.253e+03  12.651  < 2e-16 ***
## X5           1.501e+02  4.385e+00  34.233  < 2e-16 ***
## X6           1.218e-01  4.798e-02   2.539  0.01112 *
## X7           6.765e+03  3.595e+03   1.881  0.05992 .
## X8           5.827e+05  1.736e+04  33.567  < 2e-16 ***
## X9           5.303e+04  2.141e+03  24.774  < 2e-16 ***
## X10          2.625e+04  2.352e+03  11.164  < 2e-16 ***
## X11          9.601e+04  2.153e+03  44.594  < 2e-16 ***
## X12          3.098e+01  4.360e+00   7.106 1.23e-12 ***
## X14         -2.622e+03  7.265e+01 -36.093  < 2e-16 ***
## X15          1.965e+01  3.656e+00   5.374 7.77e-08 ***
## X16         -5.822e+02  3.298e+01 -17.654  < 2e-16 ***
## X17          6.024e+05  1.073e+04  56.115  < 2e-16 ***
## X18         -2.128e+05  1.315e+04 -16.179  < 2e-16 ***
## X19          2.163e+01  3.447e+00   6.273 3.60e-10 ***
## X20         -3.964e-01  7.344e-02  -5.398 6.80e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 201200 on 21594 degrees of freedom
## Multiple R-squared:  0.6998, Adjusted R-squared:  0.6996
## F-statistic:  2797 on 18 and 21594 DF,  p-value: < 2.2e-16
```

```
par(mfrow = c(2,2))
plot(model)
```



The graph on the left at the top shows if residuals have non-linear patterns. The data are simulated in a way that meets the regression assumptions very well → good model.

The graph on the right at the top shows if residuals are normally distributed. It looks like a line → good model.

The graph on the left at bottom shows if residuals are spread equally along the ranges of predictors. It has a horizontal line with equally (randomly) spread points → good model.

There is no influential case for the last graph.