Analyse US Crime

Rémi TANIEL

12/18/2019

Contents

Introduction

On commence par importer nos données grâce au code suivant :

```
data = read.table('us_crime.txt', header = TRUE)
```

On décide de regarder quelle est la taille de nos données :

```
dim(data)
```

```
## [1] 47 14
```

Nous avons donc un jeu de donnée de 47 observations pour 14 variables, on visualise nos données avec la fonction head :

```
head(data)
```

```
##       R Age S  Ed Ex0 Ex1  LF    M   N  NW  U1 U2   W   X
## 1  79.1 151 1  91  58  56 510  950  33 301 108 41 394 261
## 2 163.5 143 0 113 103  95 583 1012  13 102  96 36 557 194
## 3  57.8 142 1  89  45  44 533  969  18 219  94 33 318 250
## 4 196.9 136 0 121 149 141 577  994 157  80 102 39 673 167
## 5 123.4 141 0 121 109 101 591  985  18  30  91 20 578 174
## 6  68.2 121 0 110 118 115 547  964  25  44  84 29 689 126
```

Ainsi que leurs types :

```
str(data)
```

```
## 'data.frame':    47 obs. of  14 variables:
##  $ R  : num  79.1 163.5 57.8 196.9 123.4 ...
##  $ Age: int  151 143 142 136 141 121 127 131 157 140 ...
##  $ S  : int  1 0 1 0 0 0 1 1 1 0 ...
##  $ Ed : int  91 113 89 121 121 110 111 109 90 118 ...
##  $ Ex0: int  58 103 45 149 109 118 82 115 65 71 ...
##  $ Ex1: int  56 95 44 141 101 115 79 109 62 68 ...
##  $ LF : int  510 583 533 577 591 547 519 542 553 632 ...
##  $ M  : int  950 1012 969 994 985 964 982 969 955 1029 ...
##  $ N  : int  33 13 18 157 18 25 4 50 39 7 ...
##  $ NW : int  301 102 219 80 30 44 139 179 286 15 ...
##  $ U1 : int  108 96 94 102 91 84 97 79 81 100 ...
##  $ U2 : int  41 36 33 39 20 29 38 35 28 24 ...
##  $ W  : int  394 557 318 673 578 689 620 472 421 526 ...
##  $ X  : int  261 194 250 167 174 126 168 206 239 174 ...
```

On remarque que la variable S est en réalité un booléen pour savoir si l'état courant est un état du Sud (1) ou un état du Nord (0), nous devons indiquer que cette variable est une variable quantitative :

```
data$S = as.factor(data$S)
```

On peut vérifier cela en réappelant la fonction str :

```
str(data)
```

```
## 'data.frame':    47 obs. of  14 variables:
##  $ R  : num  79.1 163.5 57.8 196.9 123.4 ...
##  $ Age: int  151 143 142 136 141 121 127 131 157 140 ...
##  $ S  : Factor w/ 2 levels "0","1": 2 1 2 1 1 1 2 2 2 1 ...
##  $ Ed : int  91 113 89 121 121 110 111 109 90 118 ...
##  $ Ex0: int  58 103 45 149 109 118 82 115 65 71 ...
##  $ Ex1: int  56 95 44 141 101 115 79 109 62 68 ...
##  $ LF : int  510 583 533 577 591 547 519 542 553 632 ...
##  $ M  : int  950 1012 969 994 985 964 982 969 955 1029 ...
##  $ N  : int  33 13 18 157 18 25 4 50 39 7 ...
##  $ NW : int  301 102 219 80 30 44 139 179 286 15 ...
##  $ U1 : int  108 96 94 102 91 84 97 79 81 100 ...
##  $ U2 : int  41 36 33 39 20 29 38 35 28 24 ...
##  $ W  : int  394 557 318 673 578 689 620 472 421 526 ...
##  $ X  : int  261 194 250 167 174 126 168 206 239 174 ...
```

On cherche à maintenant à vérifier si il y a des valeurs manquantes dans notre jeu de données, on peut appeler la fonction summary :

```
summary(data)
```

```
##        R               Age        S            Ed              Ex0
##  Min.   : 34.20   Min.   :119.0   0:31   Min.   : 87.0   Min.   : 45.0
##  1st Qu.: 65.85   1st Qu.:130.0   1:16   1st Qu.: 97.5   1st Qu.: 62.5
```

```
##  Median : 83.10   Median :136.0          Median :108.0   Median : 78.0
##  Mean   : 90.51   Mean   :138.6          Mean   :105.6   Mean   : 85.0
##  3rd Qu.:105.75   3rd Qu.:146.0          3rd Qu.:114.5   3rd Qu.:104.5
##  Max.   :199.30   Max.   :177.0          Max.   :122.0   Max.   :166.0
##       Ex1              LF              M               N
##  Min.   : 41.00   Min.   :480.0   Min.   : 934.0   Min.   :  3.00
##  1st Qu.: 58.50   1st Qu.:530.5   1st Qu.: 964.5   1st Qu.: 10.00
##  Median : 73.00   Median :560.0   Median : 977.0   Median : 25.00
##  Mean   : 80.23   Mean   :561.2   Mean   : 983.0   Mean   : 36.62
##  3rd Qu.: 97.00   3rd Qu.:593.0   3rd Qu.: 992.0   3rd Qu.: 41.50
##  Max.   :157.00   Max.   :641.0   Max.   :1071.0   Max.   :168.00
##        NW              U1              U2              W
##  Min.   :  2.0   Min.   : 70.00   Min.   :20.00   Min.   :288.0
##  1st Qu.: 24.0   1st Qu.: 80.50   1st Qu.:27.50   1st Qu.:459.5
##  Median : 76.0   Median : 92.00   Median :34.00   Median :537.0
##  Mean   :101.1   Mean   : 95.47   Mean   :33.98   Mean   :525.4
##  3rd Qu.:132.5   3rd Qu.:104.00   3rd Qu.:38.50   3rd Qu.:591.5
##  Max.   :423.0   Max.   :142.00   Max.   :58.00   Max.   :689.0
##        X
##  Min.   :126.0
##  1st Qu.:165.5
##  Median :176.0
##  Mean   :194.0
##  3rd Qu.:227.5
##  Max.   :276.0
```

Régression simple

Nous allons maintenant effectuer une régression simple afin d'expliquer la variable R par rapport à la variable Ed

Création du modèle

On utilise la fonction suivante afin de créer un modèle permettant d'expliquer la variable R par rapport à la variable Ed :

```
model = lm(R~Ed, data = data)
```

Qualité du modèle

On cherche maintenant à vérifier la qualité de notre modèle, pour cela on utilise la fonction summary :

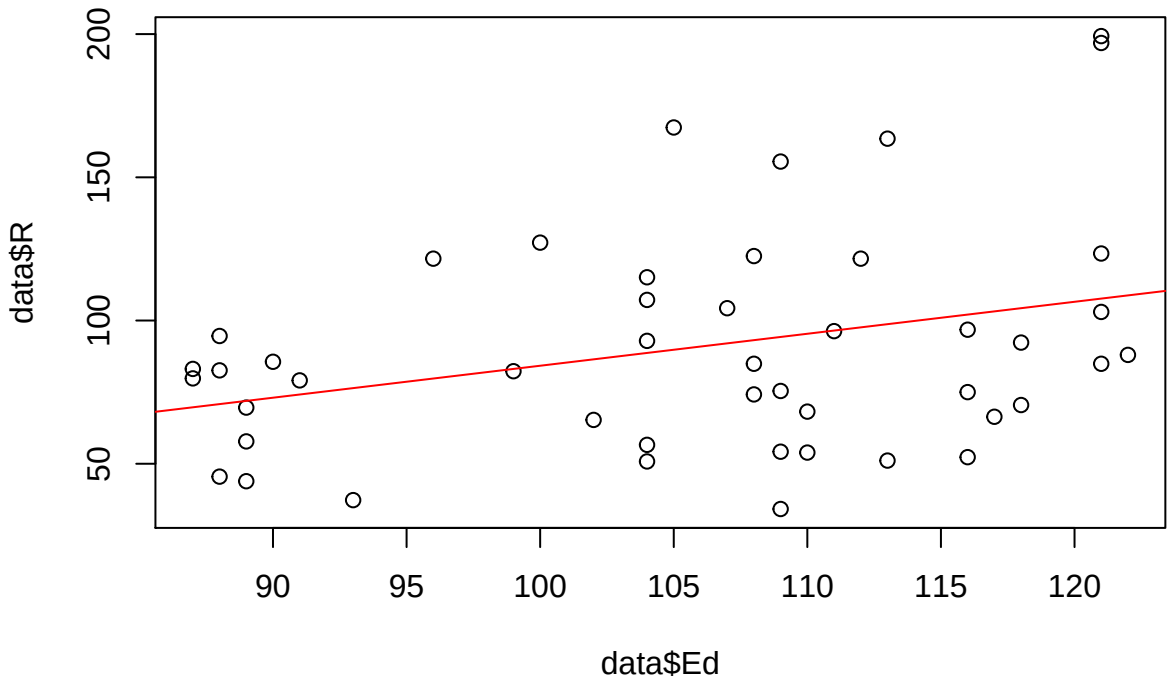
```
summary(model)
```

```
##
## Call:
## lm(formula = R ~ Ed, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -60.061 -27.125  -4.654  17.133  91.646
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -27.3967    51.8104  -0.529   0.5996
## Ed            1.1161     0.4878   2.288   0.0269 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 37.01 on 45 degrees of freedom
## Multiple R-squared:  0.1042, Adjusted R-squared:  0.08432
## F-statistic: 5.236 on 1 and 45 DF,  p-value: 0.02688
```

La r^2 de notre modèle est de 0,08432, c'est à dire que la variable Ed explique 8,432% de l'information portée par la variable R, notre modèle n'est donc pas de très bonne qualité. On peut tracer le graphique suivant pour vérifier ce qu'on a dit précédemment :

```
plot(data$Ed, data$R)
abline(model, col = 'red')
```



Validité du modèle

On décide maintenant de vérifier la validité de notre modèle, pour cela on dispose de trois critères :

- La normalité des résidus

- L'indépendance des résidus
- L'homoscédasticité des résidus

Pour chacun de ces critères nous disposons de tests pour valider le critère ou non, pour que notre modèle soit valide, il faut que les 3 critères soient validés.

Normalité des résidus

La normalité des résidus est vérifié par le test de Shapiro grâce à la fonction :

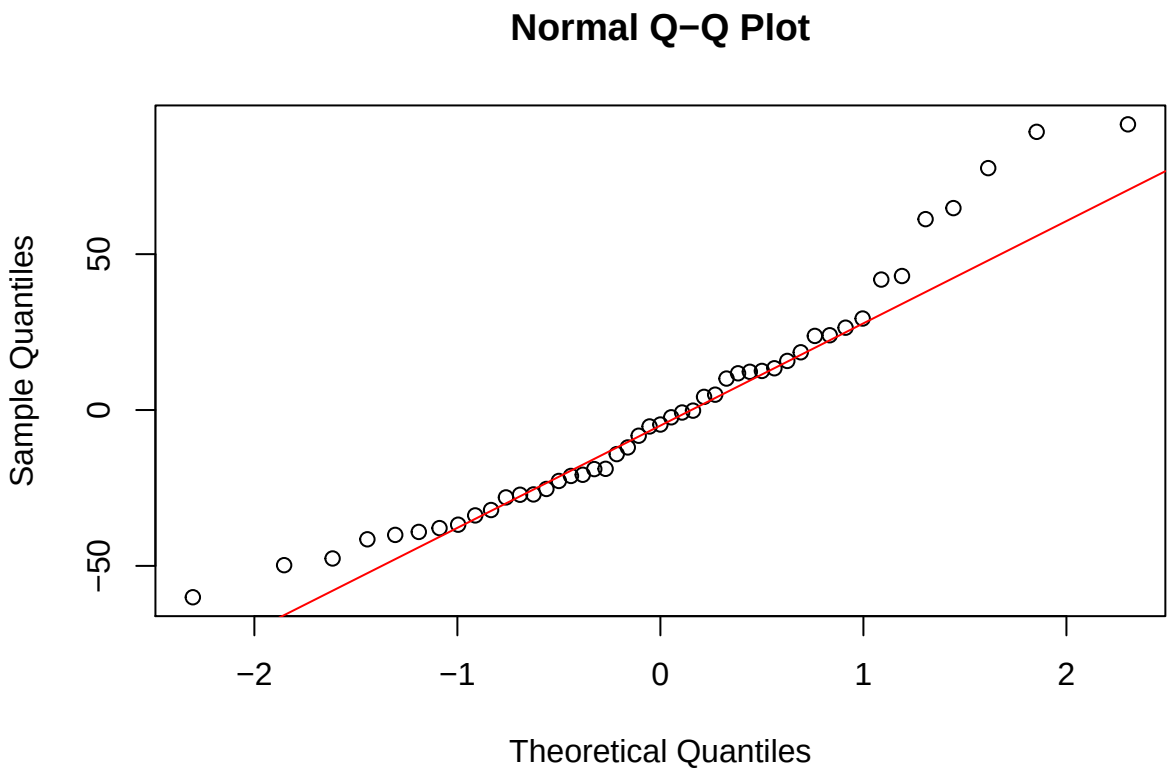
```
shapiro.test(model$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  model$residuals
## W = 0.94361, p-value = 0.02447
```

La p-value étant de 0.02447, elle est donc inférieure à notre seuil d'acceptation qui est de 0.05, donc on rejette l'hypothèse

On peut également tracer le graphique suivant pour vérifier la normalité des résidus :

```
qqnorm(model$residuals)
qqline(model$residuals, col = 'red')
```



Et on remarque que nos résidus ne sont pas normales étant donné que seule une partie de ceux-ci sont proche de la droite

Indépendance des résidus

Pour tester l'indépendance des résidus, on utilise cette fois le test de Durbin-Watson :

```
library(lmtest)
dwtest(model)
```

```
##
##  Durbin-Watson test
##
## data:  model
## DW = 2.4559, p-value = 0.9435
## alternative hypothesis: true autocorrelation is greater than 0
```

Par rapport au test précédent, la p-value est supérieure à 0.05 donc on ne rejette pas l'hypothèse

Homoscédasticité des résidus

Afin de vérifier l'homoscédasticité des résidus on utilise le test de Breusch-Pagan :

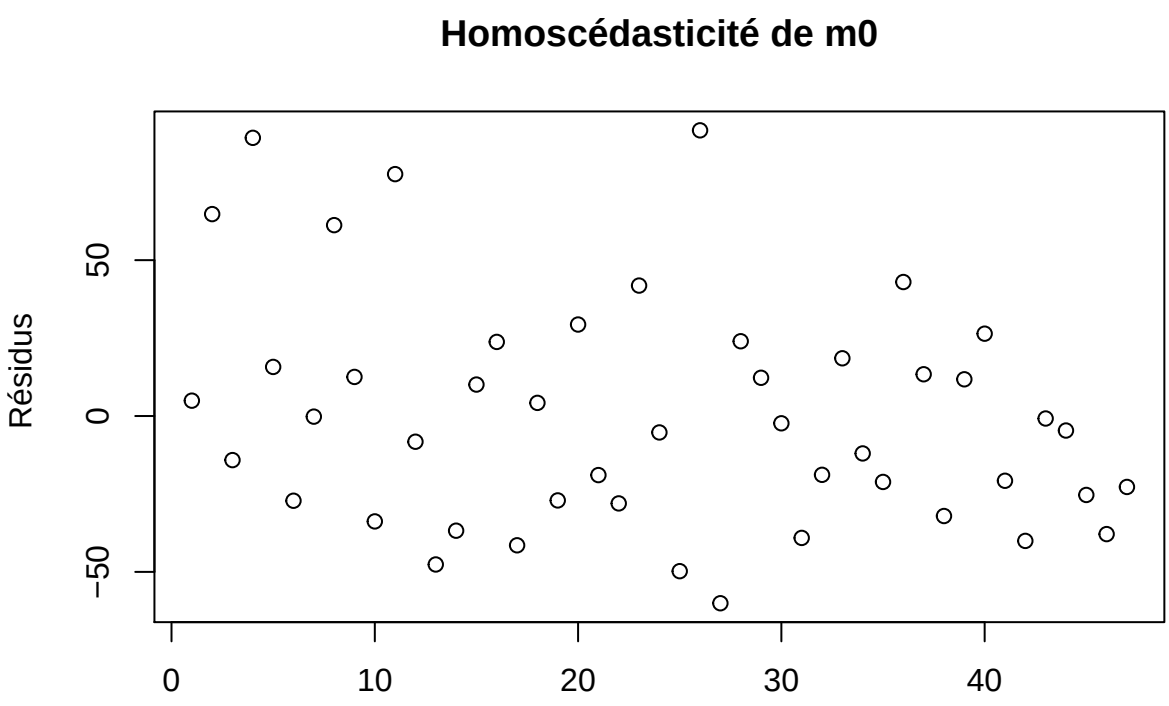
```
bptest(model)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model
## BP = 5.0592, df = 1, p-value = 0.0245
```

Comme pour la normalité des résidus, on rejette l'hypothèse car la p-value est inférieure à 0.05

Ceci est confirmé par le graphique suivant :

```
plot(model$residuals, xlab = "", ylab = "Résidus ", main = "Homoscédasticité de m0")
```


Homoscédasticité de m0
Résidus
50
0
−50
0
10
20
30
40