



L'étude du Modèle :

**“ La reformulation d'Oxyde de Carbone (CO)
avec différents types de cigarette”**

Par

“Le Langage R”

Par:
RAIHAN Najib

Encadré par:
Dr. Mohamed ADDAM

Remerciements:

Je tiens à exprimer notre professeur Dr. ADDAM Mohamed pour nous avoir donné la possibilité de travailler sous leur direction et nous aider à acquérir une expérience professionnelle extrêmement enrichissante. Et donnez-nous de précieuses contributions .

Introduction

“ L’analyse des données (aussi appelée analyse exploratoire des données ou AED) est une famille de méthodes statistiques dont les principales caractéristiques sont d’être multidimensionnelles et descriptives. Dans l’acception française, la terminologie « analyse des données » désigne donc un sous-ensemble de ce qui est appelé plus généralement la statistique multivariée. Certaines méthodes, pour la plupart géométriques, aident à faire ressortir les relations pouvant exister entre les différentes données et à en tirer une information statistique qui permet de décrire de façon plus succincte les principales informations contenues dans ces données. D’autres techniques permettent de regrouper les données de façon à faire apparaître clairement ce qui les rend homogènes, et ainsi mieux les connaître. ”

Objectifs

1. L’utilisation du langage R avec l’analyse des données.
2. Appliquer le principe de la Régression Linéaire sur ce modèle.
3. Appliquer l’Analyse des composantes principales (ACP) sur ce modèle.
4. Appliquer les méthodes de classification sur ce modèle.

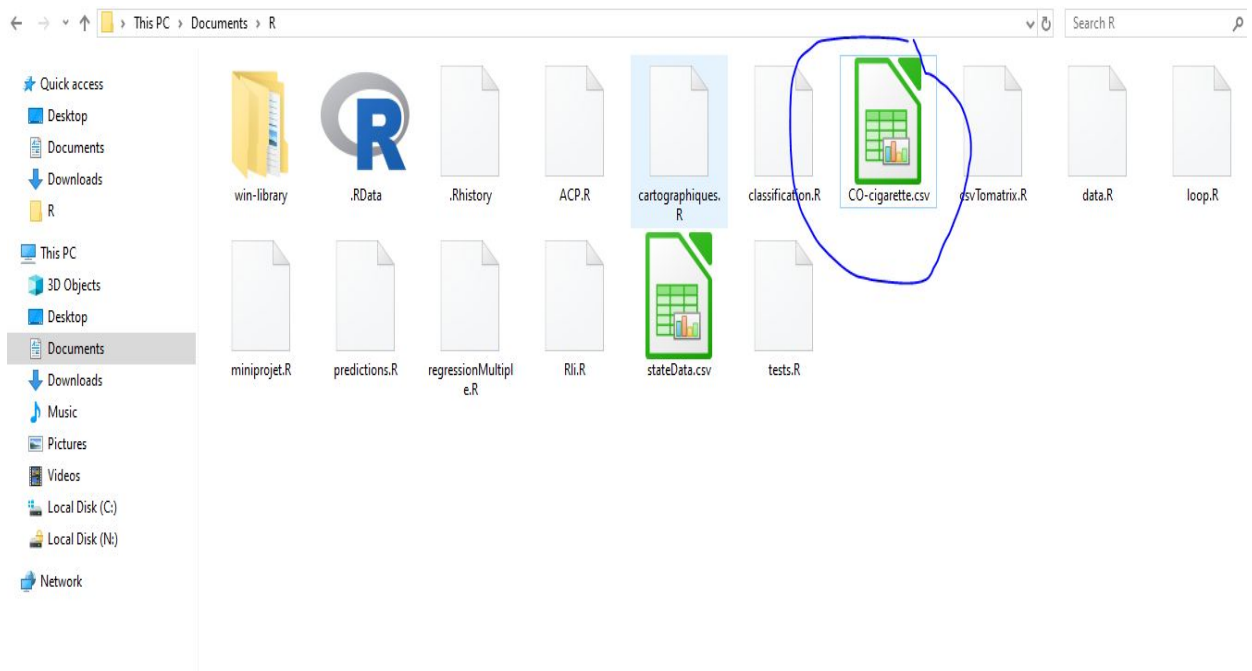
Pourquoi le R ?

R offre des outils puissants de recodage des variables et de formatage des tableaux de données, et permet de lire quasiment tous les formats de fichiers de données utilisés dans le domaine statistique.

Grands Étapes :

- L'importation des données:

Considérons le fichier de données **CO-cigarette.csv** qui regroupe les données sur l'étude de reformulation des CO avec des variables explicatives TAR , Nicotine, WEIGHT ,et des différents types des cigarettes "les individus". Ce fichier comporte 3 variables et 24 observations (individus ou unités statistiques). Il s'agit d'un fichier de type CSV (comma separated values) que l'on peut ouvrir avec un tableur de type Excel ou n'importe quel éditeur de texte. Souvent d'ailleurs, lorsque l'on double-clique sur un fichier portant cette extension (**.csv**), c'est l'application "Openoffice" qui est proposée pour lire ce fichier. Voici à quoi ressemble ce fichier en mode csv :



Liberation Sans 10 B I U A % 00						
A1:A1048576 Cigarette						
	A	B	C	D	E	F
1	Cigarette	TAR	Nicotine	WEIGHT	CO	
2	Alpine	14.1	0.86	985.3	13.6	
3	benson&Hedges	16	1.06	1093.8	16.6	
4	camellights	8	0.67	928	10.2	
5	Carlton	4.1	0.4	946.2	5.4	
6	ChisterField	15	1.04	888.5	15	
7	GoldenLights	8.8	0.76	1026.7	9	
8	kent	12.4	0.95	922.5	12.3	
9	Kool	16.6	1.12	937.2	16.3	
10	L&M	14.9	1.02	885.8	15.4	
11	LarkLights	13.7	1.01	964.3	13	
12	Marlboro	15.1	0.9	931.6	14.4	
13	Merit	7.8	0.57	970.5	10	
14	Multifilter	11.4	0.78	1124	10.2	
15	NewportLights	9	0.74	851.1	9.5	
16	Now	1	0.13	785.1	1.5	
17	OldGold	17	1.26	918.6	18.5	
18	PallMallLight	12.8	1.08	1039.5	12.6	
19	Raleigh	15.8	0.96	957.3	17.5	
20	SalemUltra	4.5	0.42	910.6	4.9	
21	Tareyton	14.5	1.01	1007	15.9	
22	TrueLight	7.3	0.61	980.6	8.5	
23	ViceroyRichLight	8.6	0.69	969.3	10.6	
24	VirginiaSlims	15.2	1.02	949.6	13.9	
25	WinstonLights	12	0.82	1118.4	14.9	
26						
27						
28						
29						
30						

Feuille de calcul sans titre - Feuille 1 (2)

Sheet 1 of 1 Selected: 1,048,576 rows, 1 column Default English (USA)

Pour importer des fichiers CSV sous R, on utilise la commande **read.csv()**, On a effectué ces données à variable **Mydata** :

L'affichage de notre variable: **view(Mydata)**

The screenshot shows the RStudio interface. The top pane displays a data frame with 5 columns: Cigarette, TAR, Nicotine, WEIGHT, and CO. The bottom pane shows the R console output, indicating that the package 'rgeos' was successfully unpacked and MD5 sums were checked. The console also shows the path to the downloaded binary packages and the R code used to read and view the data.

	Cigarette	TAR	Nicotine	WEIGHT	CO
1	Alpine	14.1	0.86	985.3	13.6
2	benson&Hedges	16.0	1.06	1093.8	16.6
3	camellights	8.0	0.67	928.0	10.2
4	Carlton	4.1	0.40	946.2	5.4
5	ChisterField	15.0	1.04	888.5	15.0
6	GoldenLights	8.8	0.76	1026.7	9.0
7	kent	12.4	0.95	922.5	12.3
8	Kool	16.6	1.12	937.2	16.3
9	L&M	14.9	1.02	885.8	15.4
10	LarkLights	13.7	1.01	964.3	13.0
11	Marlboro	15.1	0.90	931.6	14.4
12	Merit	7.8	0.57	970.5	10.0
13	Multifilter	11.4	0.78	1124.0	10.2
14	NewportLights	9.0	0.74	851.1	9.5

```

~/
downloaded 1.8 MB

package 'rgeos' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
C:\Users\NAJIB_RAIHAN\AppData\Local\Temp\Rtmpwary4\downloaded_packages
> #open the file
> Mydata<-read.csv("co-cigarette.csv")
> view(Mydata)
> view(Mydata)
> |

```

On va extraire une matrice **elementDataWithCo** := (TAR|Nicotine|WEIGHT|CO) pour faciliter les calculs des paramètres de position et de dispersion: En utilisant **matrix()**

The screenshot shows the RStudio interface with multiple data frames open in the top pane: dataMatrix, elementsData, elementsDataWithCO, and Mydata. The bottom pane shows the R console output, including the R code used to create the matrix 'elementsDataWithCO' from the 'Mydata' data frame. The console also shows the first few rows of the matrix.

	V1	V2	V3	V4
1	14.1	0.86	985.3	13.6
2	16.0	1.06	1093.8	16.6
3	8.0	0.67	928.0	10.2
4	4.1	0.40	946.2	5.4
5	15.0	1.04	888.5	15.0
6	8.8	0.76	1026.7	9.0
7	12.4	0.95	922.5	12.3
8	16.6	1.12	937.2	16.3
9	14.9	1.02	885.8	15.4
10	13.7	1.01	964.3	13.0

```

~/
[15,] 10.2
[14,] 9.5
[15,] 1.5
[16,] 18.5
[17,] 12.6
[18,] 17.5
[19,] 4.9
[20,] 15.9
[21,] 8.5
[22,] 10.6
[23,] 13.9
[24,] 14.9
> elementsDataWithCO <- matrix(c(Mydata[,2],Mydata[,3],Mydata[,4],Mydata[,5]),nco1 = 4)
> view(elementsDataWithCO)
> view(elementsDataWithCO)

```

NOTE: **X1: TAR** | **X2: Nicotine** | **X3: WEIGHT** | **Y: CO**

1 - Les Moyennes: mean()

```

1 for (i in 1:3) {
2   cat("moyenne de x",i,"est",mean(elementsDatawithCO[,i]),"\n")
3 }
4 cat("moyenne de (CO)Y est",mean(elementsDatawithCO[,4]),"\n")
5 |
6

```

5:1 (Top Level) ↕

Console Terminal Jobs

```

~/
> for (i in 1:3) {
+   cat("moyenne de x",i,"est",mean(elementsDatawithCO[,i]),"\n")
+ }
moyenne de x 1 est 11.48333
moyenne de x 2 est 0.8283333
moyenne de x 3 est 962.1458
> cat("moyenne de (CO)Y est",mean(elementsDatawithCO[,4]),"\n")
moyenne de (CO)Y est 12.07083
> cat("\f")

```

2 - Les Médianes : median()

```

1 for (i in 1:3) {
2   cat("median de x",i,"est",median(elementsDatawithCO[,i]),"\n")
3 }
4 cat("median de (CO)Y est",median(elementsDatawithCO[,4]),"\n")
5 |
6
7

```

5:1 (Top Level) ↕

miniprojet.R* dataMatrix elementsData elementsDataWithCO Mydata

Run Source

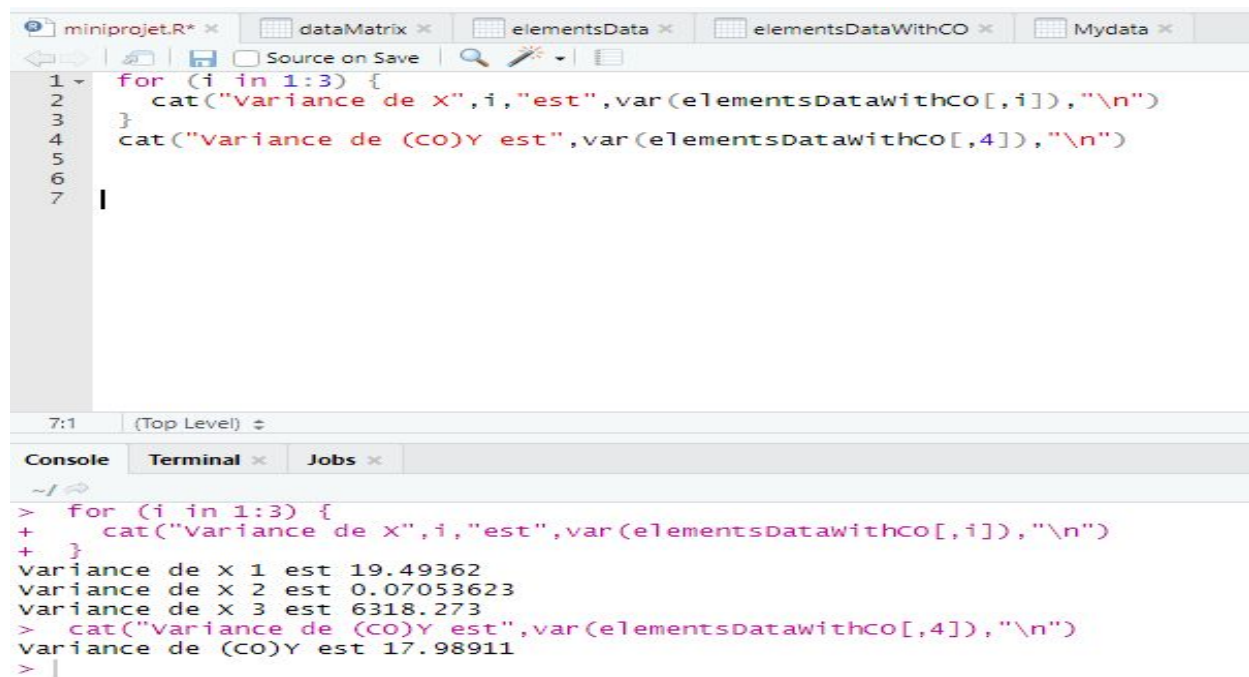
Console Terminal Jobs

```

~/
> for (i in 1:3) {
+   cat("median de x",i,"est",median(elementsDatawithCO[,i]),"\n")
+ }
median de x 1 est 12.6
median de x 2 est 0.88
median de x 3 est 953.45
> cat("median de (CO)Y est",median(elementsDatawithCO[,4]),"\n")
median de (CO)Y est 12.8
> |

```

3 - Variance: var()



```

1 for (i in 1:3) {
2   cat("Variance de x",i,"est",var(elementsDatawithCO[,i]),"\n")
3 }
4 cat("Variance de (CO)Y est",var(elementsDatawithCO[,4]),"\n")
5
6
7 |

```

```

7:1 (Top Level)

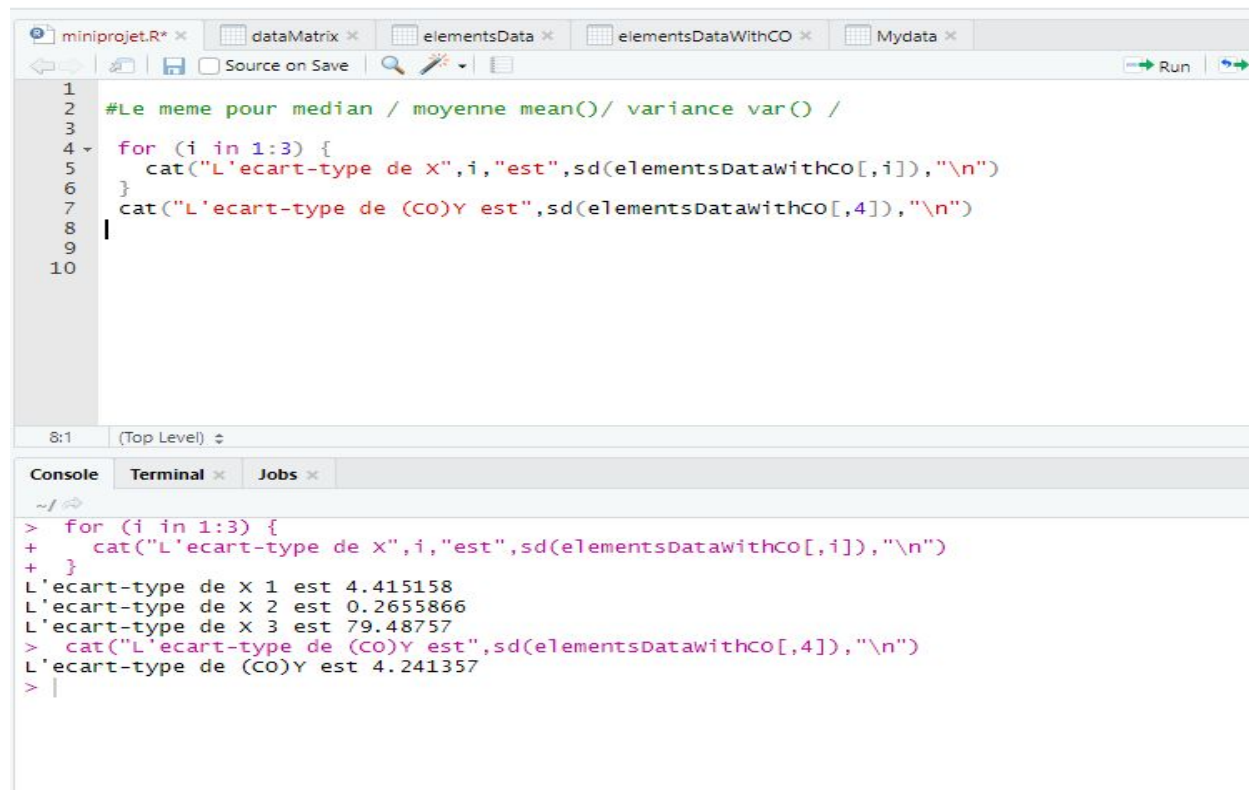
```

```

Console Terminal Jobs
~/
> for (i in 1:3) {
+   cat("Variance de x",i,"est",var(elementsDatawithCO[,i]),"\n")
+ }
variance de x 1 est 19.49362
variance de x 2 est 0.07053623
variance de x 3 est 6318.273
> cat("Variance de (CO)Y est",var(elementsDatawithCO[,4]),"\n")
variance de (CO)Y est 17.98911
> |

```

4 - L'écart-type: sd()



```

1
2 #Le meme pour median / moyenne mean() / variance var() /
3
4 for (i in 1:3) {
5   cat("L'ecart-type de x",i,"est",sd(elementsDatawithCO[,i]),"\n")
6 }
7 cat("L'ecart-type de (CO)Y est",sd(elementsDatawithCO[,4]),"\n")
8
9 |
10

```

```

8:1 (Top Level)

```

```

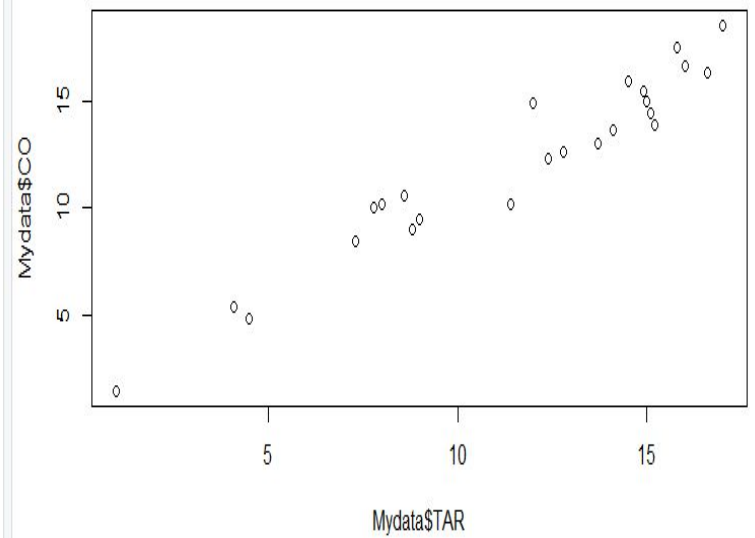
Console Terminal Jobs
~/
> for (i in 1:3) {
+   cat("L'ecart-type de x",i,"est",sd(elementsDatawithCO[,i]),"\n")
+ }
L'ecart-type de x 1 est 4.415158
L'ecart-type de x 2 est 0.2655866
L'ecart-type de x 3 est 79.48757
> cat("L'ecart-type de (CO)Y est",sd(elementsDatawithCO[,4]),"\n")
L'ecart-type de (CO)Y est 4.241357
> |

```


les Nuage des points : `plot(x,y)`

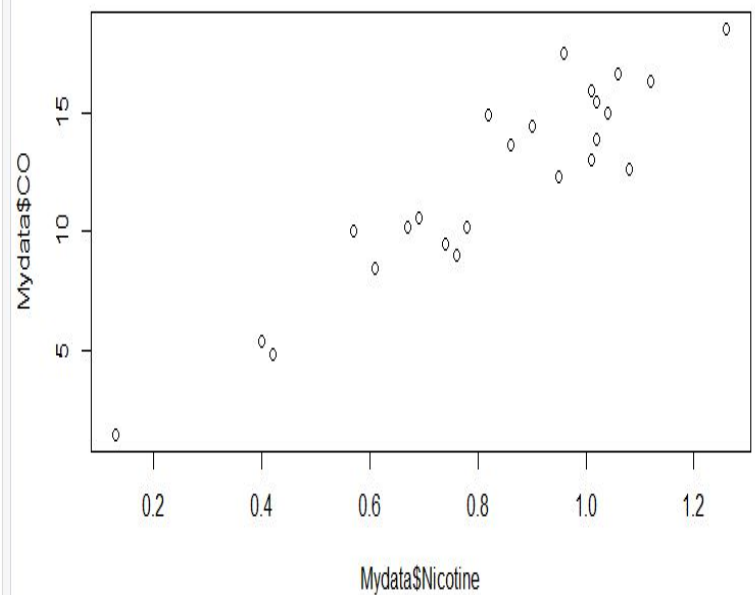
CO = f(TAR) `//plot(Mydata$TAR,Mydata$CO)`

```
> plot(Mydata$TAR,Mydata$CO)  
> |
```



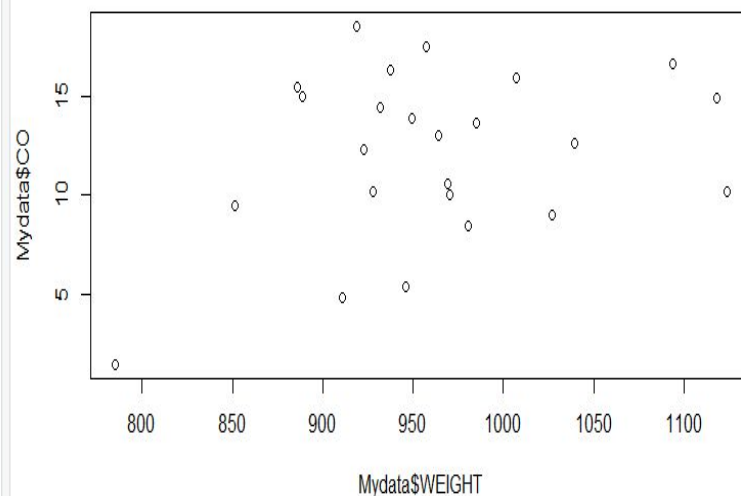
CO = f(Nicotine) `//plot(Mydata$Nicotine,Mydata$CO)`

```
> plot(Mydata$Nicotine,Mydata$CO)  
> |
```



CO = f(WEIGHT) //plot(Mydata\$WEIGHT,Mydata\$CO)

```
> plot(Mydata$WEIGHT,Mydata$CO)
> |
```



- Régression Linéaire simple (Visualisation des données) :

La régression linéaire: Le but de la régression simple (resp. multiple) est d'expliquer une variable Y à l'aide d'une variable X (resp. plusieurs variables X1, ..., Xq). La variable Y est appelée variable dépendante, ou variable à expliquer et les variables Xj (j=1,...,q) sont appelées variables indépendantes, ou variables explicatives.

La régression linéaire simple permet d'évaluer la significativité du lien linéaire entre deux variables. La forme linéaire entre les deux variables (TAR , CO) ou (Nicotine , CO) ou (WEIGHT , CO) est donc présupposée. Autrement dit, on fait l'hypothèse que la forme de la relation entre les variables est linéaire. Néanmoins, il est préférable de vérifier si cette hypothèse est acceptable, ou non, car si ce n'est pas le cas, les résultats de l'analyse n'auront pas de sens.

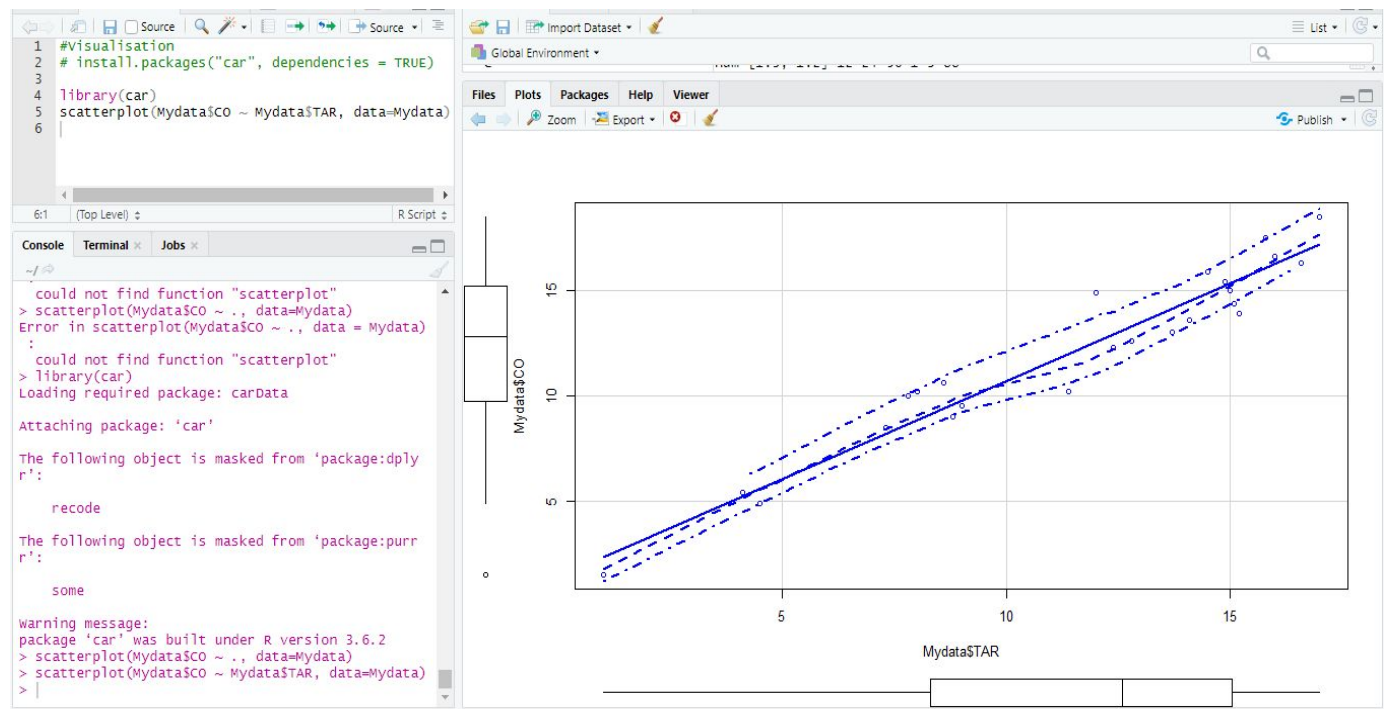
Pour évaluer de façon visuelle, la linéarité entre deux variables, on peut utiliser **scatterplot()** du **package "car"**.

```
>library(car)
```

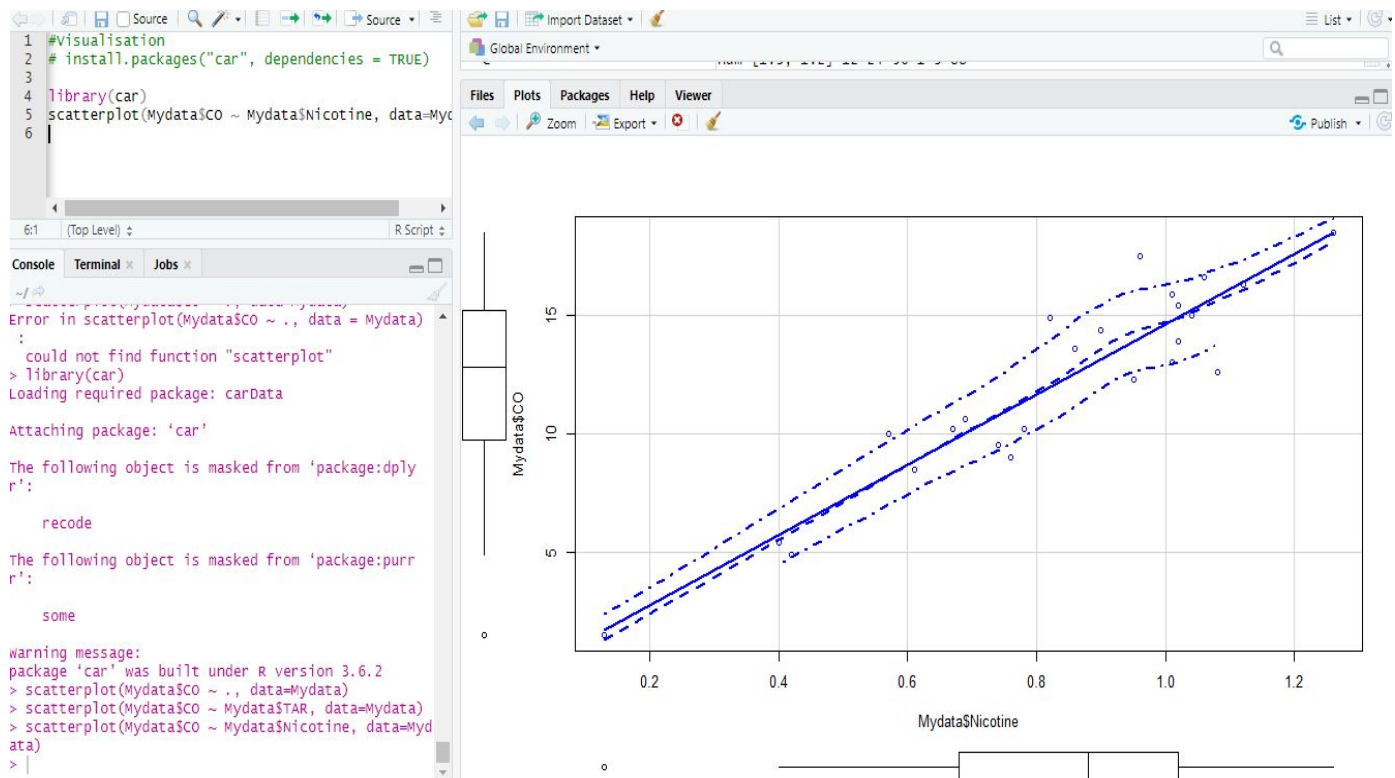
```
>scatterplot(Mydata$CO ~ Mydata$TAR , data = Mydata)
```

Et on fait la même commande pour tous les variables avec changement de Mydata\$TAR avec les autres (Nicotine , WEIGHT).

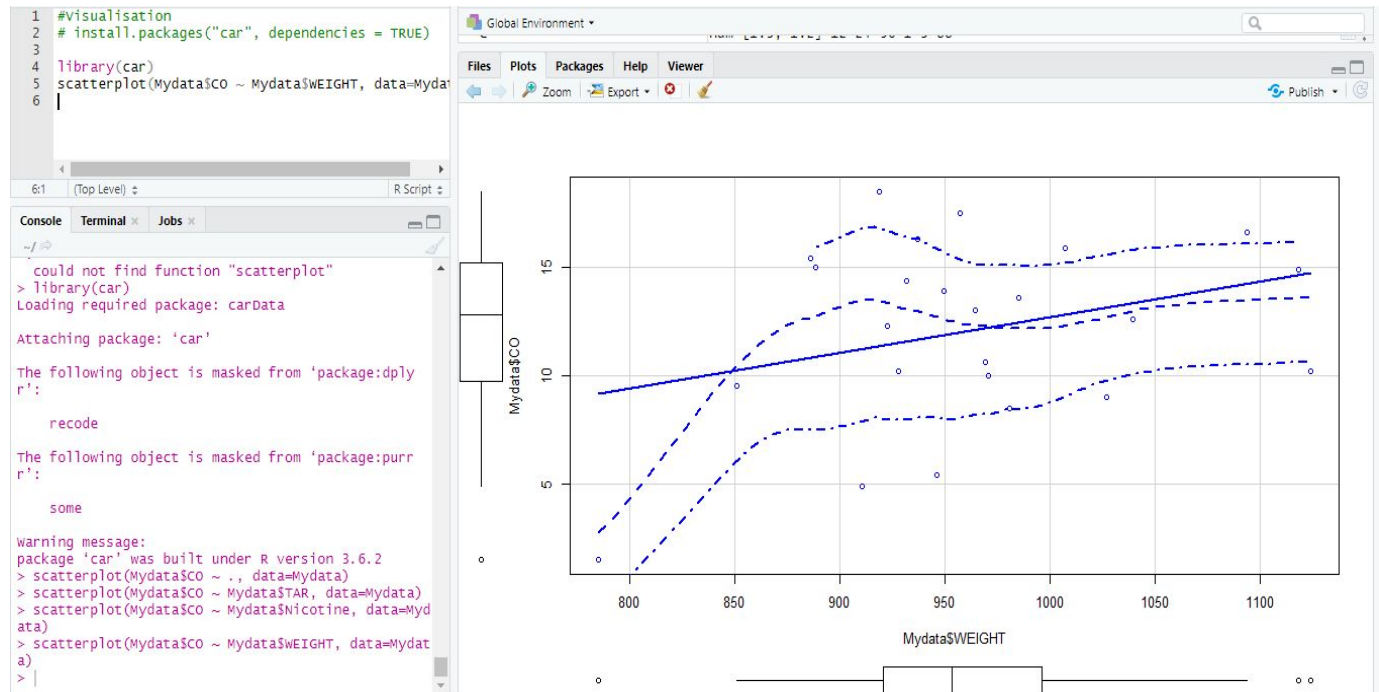
1 - $Y = a_1 \cdot X_1 + b_1$:



2 - $Y = a_2 \cdot X_2 + b_2$:



3 - $Y = a_3 \cdot X_3 + b_3$:



Interprétation:

La ligne en trait plein est la droite de régression linéaire (définie par la **méthode des moindres carrés** MMC) entre les deux variables. La ligne centrale en pointillé est la courbe de régression locale de type lowess. Les deux lignes extérieures représentent un intervalle de confiance de la courbe lowess.

Ici, la droite de régression est comprise dans l'intervalle de confiance de la courbe lowess, surtout pour **TAR (X1)** et **Nicotine(X2)**, l'**hypothèse de linéarité est donc acceptable**.

Evaluation des hypothèses de validité des résultats:

Le test d'évaluation de la significativité du lien linéaire entre les deux variables est valide, si les résidus :

1. sont indépendants
2. sont distribués selon une loi Normale de moyenne 0
3. sont distribués de façon homogènes, c'est à dire, avec une variance constante.

(!!!!!!! Nous allons voir ces trois caractères avec Régression Multiple)

- Régression Linéaire Multiple :

Pour déterminer la droite de régression, on ajuste un modèle linéaire Multiple (resp.simple) aux données, à l'aide de la fonction **lm()** , comme ceci :

```

1 # Multiple Linear Regression
2
3 fit <- lm(Mydata$CO ~ Mydata$STAR + Mydata$Nicotine + Mydata$WEIGHT , data=Mydata)
4
5 summary(fit) # show results|
6

```

5:28 (Top Level) ⚡

Console Terminal x Jobs x

~/

```

> fit <- lm(Mydata$CO ~ Mydata$STAR + Mydata$Nicotine + Mydata$WEIGHT , data=Mydata)
> summary(fit) # show results

Call:
lm(formula = Mydata$CO ~ Mydata$STAR + Mydata$Nicotine + Mydata$WEIGHT,
    data = Mydata)

Residuals:
    Min       1Q   Median       3Q      Max
-2.1082 -0.8046 -0.1194  1.0094  2.0501

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.551174   2.969969  -0.186  0.854641
Mydata$STAR    0.887543   0.195483   4.540  0.000199 ***
Mydata$Nicotine 0.519037   3.252271   0.160  0.874803
Mydata$WEIGHT  0.002079   0.003177   0.654  0.520348
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.16 on 20 degrees of freedom
Multiple R-squared:  0.935,    Adjusted R-squared:  0.9252
F-statistic: 95.86 on 3 and 20 DF,  p-value: 4.85e-12

> view(fit)
> view(fit)

```

Explication:

fit : Notre modèle linéaire

La partie `Residuals` des résultats permet d'évaluer rapidement la normalité des résidus. Lorsque les résidus sont distribués selon une loi Normale, la médiane doit être autour de 0 (c'est le cas ici -0.1194), et les valeurs absolues de Q1 (premier quartile) et Q3 (troisième quartile).

La première ligne de la partie coefficients concerne l'ordonnée à l'origine, alors que la seconde ligne concerne la pente. Concernant les colonnes :

- la première colonne rapporte l'estimation des coefficients des paramètres,
- la seconde colonne l'estimation de leur erreur standard,
- la troisième colonne est la statistique T
- la dernière colonne rapporte la p-value du test évaluant l'égalité à 0 des coefficients. Si la p-value est inférieure au seuil de significativité généralement utilisé de 0.05, alors on conclura que le paramètre est significativement différent de 0.

En général, les coefficients de la pente a vraiment un intérêt. Ici, ils sont positives , Cela signifie que lorsque **TAR** ou **Nicotine** ou **WEIGHT augmente** d'une unité, alors, le **CO augmente** de "coefficient" unités.

$$Y^* = 0.887543 * X1 + 0.519037 * X2 + 0.002079 * X3 - 0.551174$$

Le coefficient de détermination R^2: 0.935

!!!! Cette Équation de régression multiple explique 93% des différences reformulations Co entre les types de cigarette.

R^2 ajustée: 0.9252

Les degres de libertes: (3,24,20)

Régression Multiple Standardisé : lm.beta() de package " QuantPsyc":

```

18
19 # REGRESSION MULTIPLE STANDARISE
20 install.packages("QuantPsyc")
21 library(QuantPsyc)
22 fitStandrized <- lm.beta(fit)
23
24
25
26

```

24:1 (Top Level) R Script

Console Terminal Jobs

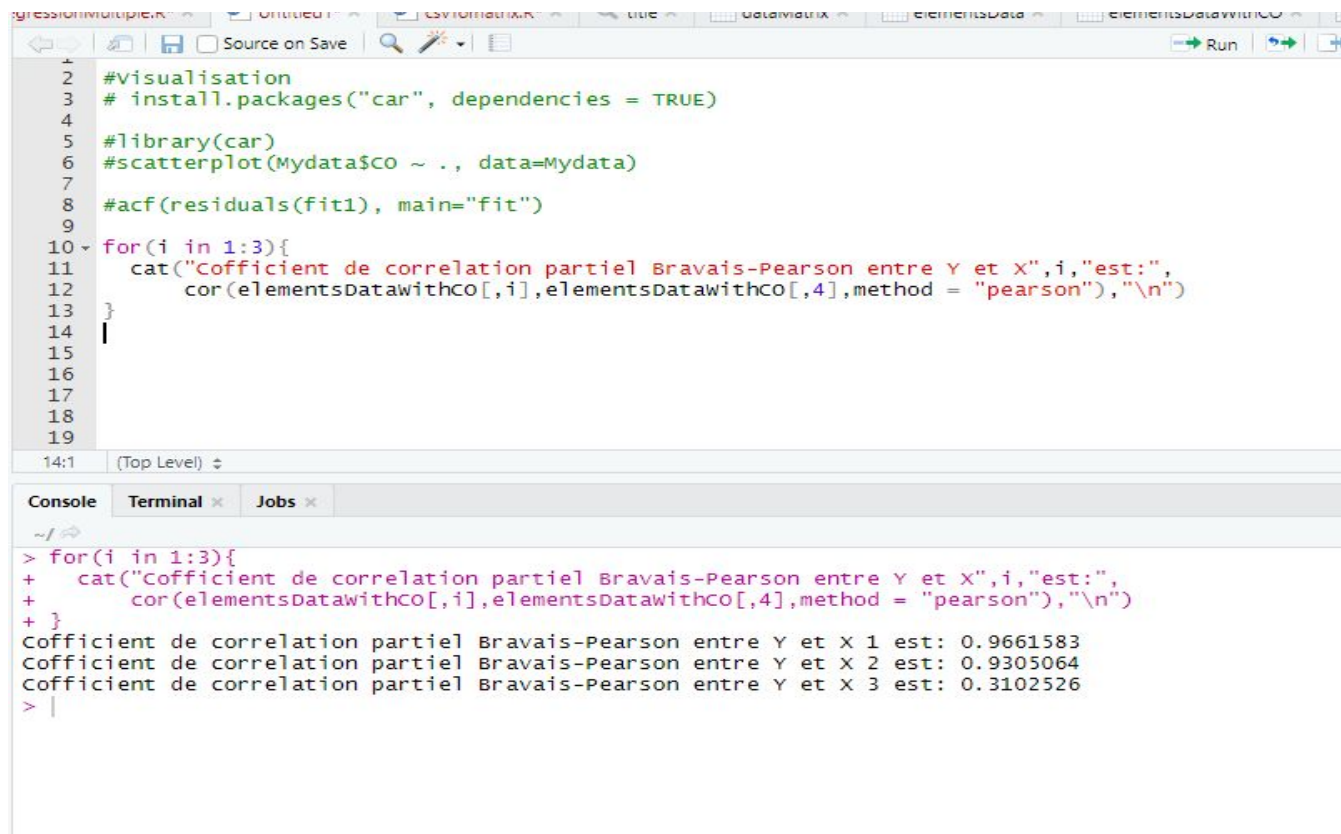
```

> fitstandrized <- lm.beta(fit)
> summary(fitstandrized)
      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.03250 0.03573 0.03896 0.33179 0.48144 0.92391
> fitstandrized
      Mydata$TAR Mydata$Nicotine Mydata$WEIGHT
0.92391291      0.03250119      0.03895905
> |

```

$$Y^* = 0.92391292 \cdot Z1 + 0.0325119 \cdot Z2 + 0.03895905 \cdot Z3$$

Les coefficients de corrélation Pearson (les coefficient de détermination): cor()



```

1 #Visualisation
2 # install.packages("car", dependencies = TRUE)
3
4 #library(car)
5 #scatterplot(Mydata$CO ~ ., data=Mydata)
6
7 #acf(residuals(fit1), main="fit")
8
9
10 for(i in 1:3){
11   cat("Coefficient de corrélation partiel Bravais-Pearson entre Y et X",i,"est:",
12       cor(elementsDatawithCO[,i],elementsDatawithCO[,4],method = "pearson"),"\n")
13 }
14 |
15
16
17
18
19
14:1 (Top Level)

```

```

> for(i in 1:3){
+   cat("Coefficient de corrélation partiel Bravais-Pearson entre Y et X",i,"est:",
+       cor(elementsDatawithCO[,i],elementsDatawithCO[,4],method = "pearson"),"\n")
+ }
Coefficient de corrélation partiel Bravais-Pearson entre Y et X 1 est: 0.9661583
Coefficient de corrélation partiel Bravais-Pearson entre Y et X 2 est: 0.9305064
Coefficient de corrélation partiel Bravais-Pearson entre Y et X 3 est: 0.3102526
> |

```

Interprétation:

Les pourcentages de l'impact de X1 , X2 , X3 sur la santé humain par la reformulation du CO:

X1: 97%

X2: 93%

X3: 31%

La somme carres des ecarts SCeres:

```

14 # for i in 1:30 {
15   # cat("Cofficient de correlation partiel Bravais-Pearson entre Y et X",i,"est:",
16     # cor.test(elementsDatawithCO[,i],elementsDatawithCO[,4],method = "pearson")#, "\n")
17   #}
18
19 # REGRESSION MULTIPLE STANDARISE
20 #install.packages("QuantPsyc")
21 #library(QuantPsyc)
22 #fit.beta <- lm.beta(fit)
23 #coef(fit.beta,standardized=TRUE)
24 #coef(fit)
25 #beta(fit)
26 cat("residus observés :",residuals(fit),"\n")
27 cat("La somme des carres des residus est : ", sum( fit$residuals^2 ),"\n")
28 cat(" le nombre de degres de libertes du residu : ", df.residual(fit))
29
30 |
31
32

```

30:1 (Top Level) R Script

```

> cat("residus observés :",residuals(fit),"\n")
residus observés : -0.8578079 0.1265018 1.37394 0.1376647 -0.1487951 -0.7879866 -0.5651483 -0.411625
4 0.3559528 -1.136991 -0.8544808 1.315003 -2.108248 -0.09007608 -0.535915 1.399358 -0.9308608 1.5396
71 -0.6537279 0.9642092 0.2170171 1.145178 -1.542938 2.050106
> cat("La somme des carres des residus est : ", sum( fit$residuals^2 ),"\n")
La somme des carres des residus est : 26.90371
> cat(" le nombre de degres de libertes du residu : ", df.residual(fit))
 le nombre de degres de libertes du residu : 20
>

```

```

5 #Z
6 #Z
7
8 #test-fisher
9 anova(fit)
10 |
11
12
13

```

10:1 (Top Level) R Script

```

> #test-fisher
> anova(fit)
Analysis of Variance Table

Response: Mydata$CO
          Df Sum Sq Mean Sq F value    Pr(>F)
Mydata$TAR      1 386.22   386.22 287.1124 2.492e-13 ***
Mydata$Nicotine 1  0.55    0.05  0.0375  0.8484
Mydata$Vitamin  1  0.58    0.58  0.4282  0.5203
Residuals      20 26.90    1.35
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
>

```


Matrice de Covariance:

```

7 #les residus
8 #fitted values
9
10 coResidus <- Mydata
11 coResidus$fitted_Ajuste <- fitted(fit)
12 coResidus$residus <- residuals(fit)
13
14
15 #Matrice de variance - covariance
16 vcov(fit)
17 |
18

```

```

> #Matrice de variance - covariance
> vcov(fit)

              (Intercept)    Mydata$STAR Mydata$Nicotine Mydata$WEIGHT
(Intercept)    8.820712996    8.475383e-02   -1.2660204182  -9.031098e-03
Mydata$STAR    0.084753827    3.821349e-02   -0.6080167795   -2.071563e-05
Mydata$Nicotine -1.266020418  -6.080168e-01   10.5772657972   -5.336219e-04
Mydata$WEIGHT  -0.009031098  -2.071563e-05   -0.0005336219   1.009306e-05
> |

```

Evaluation des hypothèses de validité des résultats:

l'hypothèse d'indépendance des résidus && l'hypothèse de normalité des résidus:

Théoriquement:

Le **test de Durbin-Watson** peut être employé pour évaluer la présence d'une autocorrélation pour un lag de valeur 1. L'hypothèse d'indépendance des résidus est rejetée lorsque la p-value du test est inférieure à 0.05.

Le **test de Shapiro-Wilk** peut également être employé pour évaluer la normalité des résidus. L'hypothèse de normalité est rejetée si la p-value est inférieure à 0.05.

```

8 #test-fisher
9 anova(fit)
10
11
12 #Diagnostic plots
13 library(car)
14 influenceIndexPlot(fit)
15 ## Durbin-watson normality test
16 durbinwatsonTest(fit)
17 ## shapiro-wilk normality test
18 shapiro.test(residuals(fit))
19
20
21
22
15:17 (Top Level)

```

```

Console Terminal Jobs
~/f
> durbinwatsonTest(fit)
lag Autocorrelation D-W Statistic p-value
1 -0.4387251 2.693878 0.084
Alternative hypothesis: rho != 0
> shapiro.test(residuals(fit))

shapiro-wilk normality test

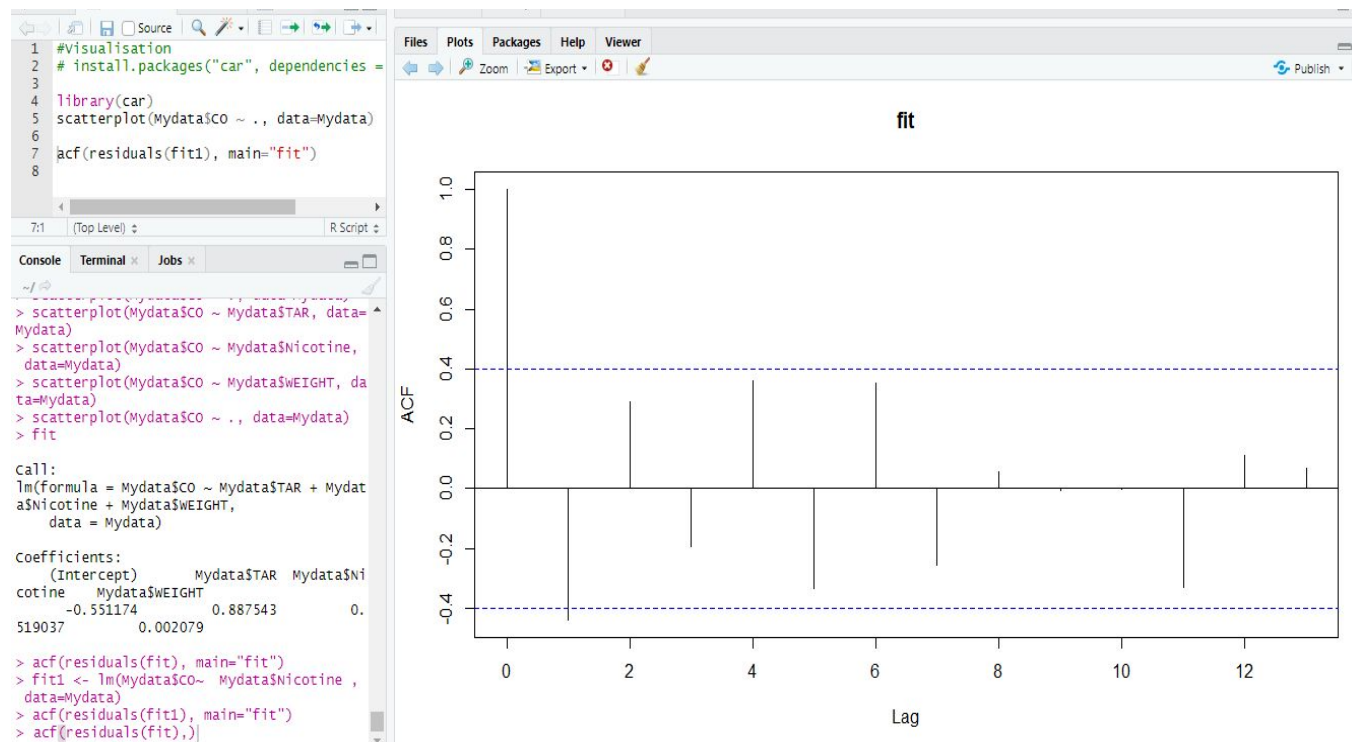
data: residuals(fit)
W = 0.96451, p-value = 0.5354

> Mydata
  cigarette  TAR Nicotine WEIGHT  CO
1    Alpine 14.1  0.86  985.3 13.6

```

Graphiquement:

1-L 'indépendance des résidus: // acf(residuals\$fit)

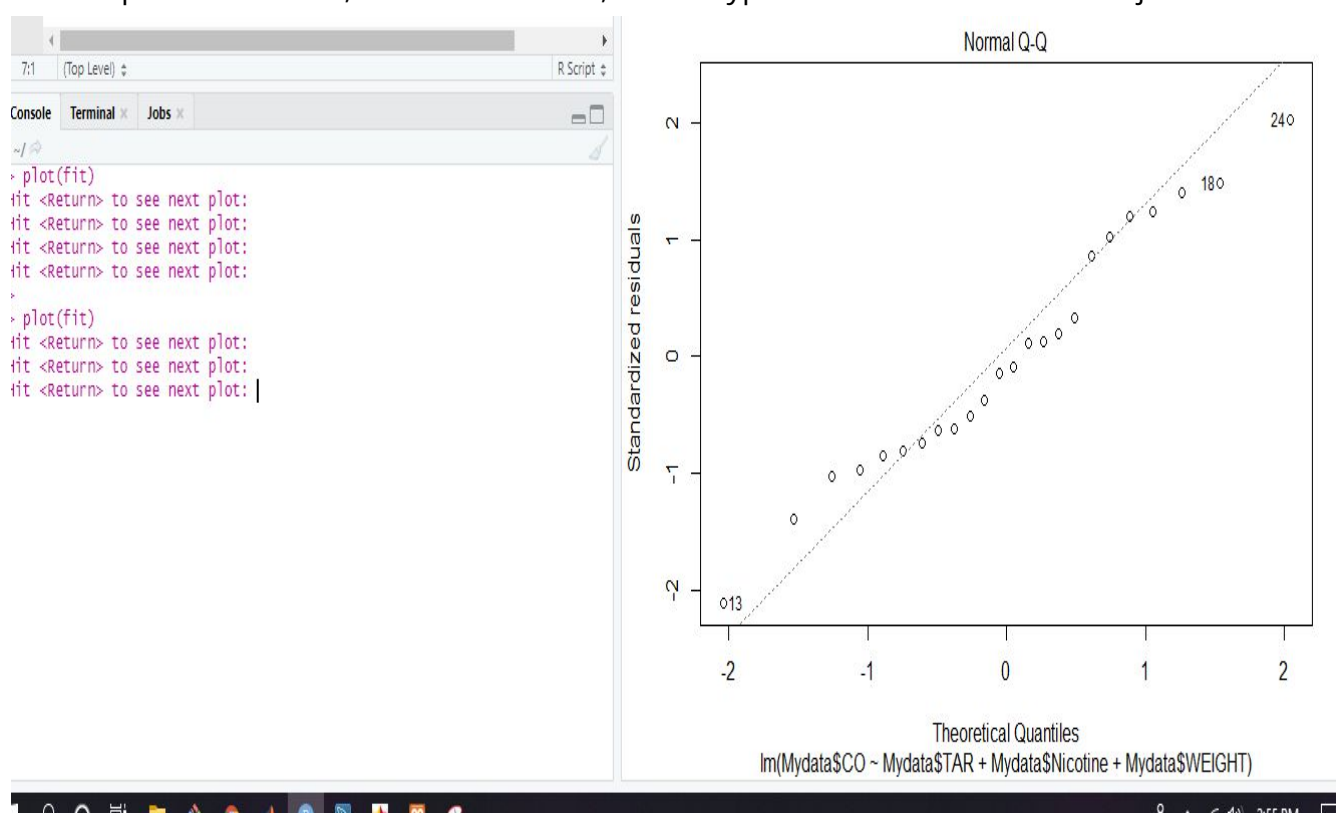


Les pointillés horizontaux: sont les intervalles de confiance du coefficient de corrélation .

Les traits verticaux: représentent les coefficients de corrélation entre les résidus de chaque point

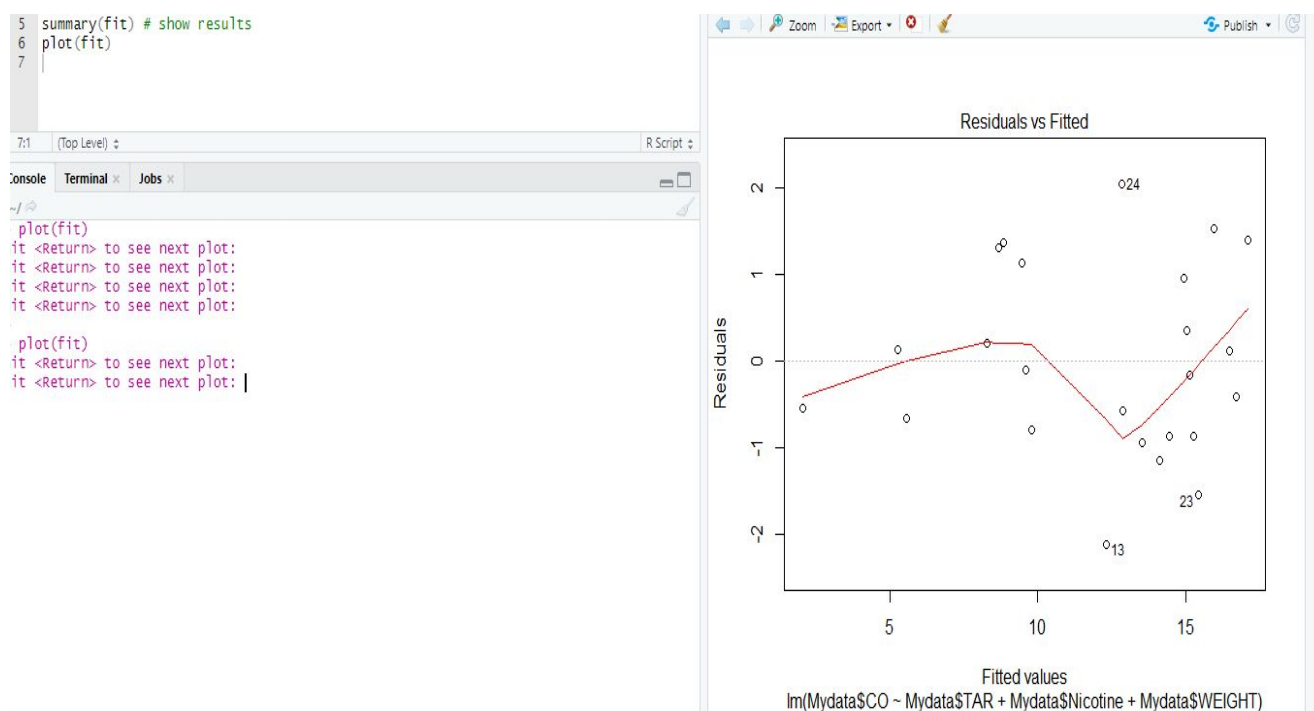
2-Normalité des résidus:

Cette hypothèse peut s'évaluer graphiquement à l'aide d'un QQplot. Si les résidus sont bien distribués le long de la droite figurant sur le plot, alors l'hypothèse de normalité est acceptée. A l'inverse, s'ils s'en écartent, alors l'hypothèse de normalité est rejetée.



3-L'hypothèse de linéarité:

Là encore, cette hypothèse peut se vérifier de façon visuelle, pour cela il faut réaliser un "residuals vs fitted plot". Les "fitted" correspondent aux réponses prédites par le modèle, pour les valeurs observées de la variable prédictive. Si on s'intéresse à la régression linéaire simple entre les variables **X1 X2 X3 Y**, les "fitted" correspondent aux valeurs de **Y(CO)** prédites par le modèle pour les valeurs **X1 X2 X3** présentes dans les données.



Ici, le plot nous montre que lorsque les réponses prédites par le modèle (fitted values) augmentent, les résidus restent globalement uniformément distribués de part et d'autre de 0. Cela montre, qu'en moyenne, la droite de régression, est bien adaptée aux données, et donc que l'hypothèse de linéarité est acceptable.

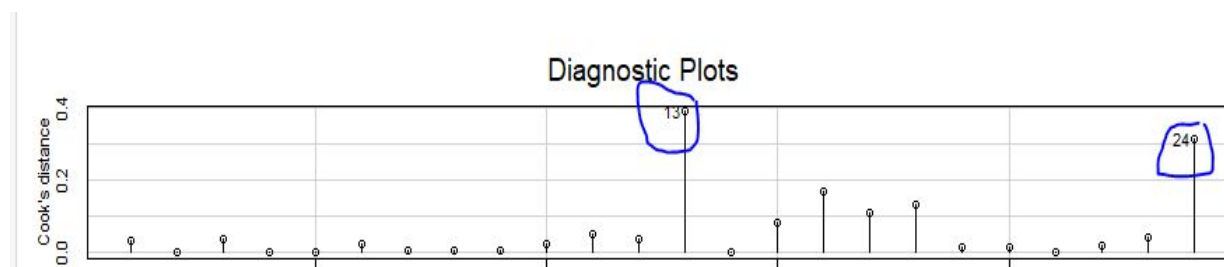
7. Les données influentes:

Avant de conclure définitivement sur la significativité ou pas de la relation linéaire entre la variable réponse et les variable prédictives, il peut être intéressant de rechercher s'il existe des données influentes, et quel est leur impact sur les résultats.

Le **package "car"** dispose d'une fonction très utile pour détecter ces données différentes, il s'agit de la **fonction "influenceIndexPlot"**

```
>library(car)
```

```
>influenceIndexPlot(fit)
```



Le plot des distances de cook met également en évidence une certaine influence de la données 13(Multifilter) , 24(Winston Lights) sur les paramètres du modèle de régression. Par acquis de conscience, on peut refaire tourner le modèle sans ces données 13 et 24, afin visualiser leur impact.

```

20
21
22 ## Breush-Pagan normality test
23 ncvTest(fit)
24
25 ##Intervalles de confiance des coefficients des parametres
26 confint(fit)
27 ## outlier
28 outlierTest(fit)
29
30
31

```

```

> ## Breush-Pagan normality test
> ncvTest(fit)
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 0.446712, Df = 1, p = 0.5039
> ##Intervalles de confiance des coefficients des parametres
> confint(fit)
                2.5 %      97.5 %
(Intercept)  -6.746420032  5.644071503
Mydata$TAR    0.479773673  1.295313202
Mydata$Nicotine -6.265081503  7.303154798
Mydata$WEIGHT -0.004548212  0.008705825
> influenceIndexPlot(fit)
> outlierTest(fit)
No Studentized residuals with Bonferroni p < 0.05
Largest |rstudent|:
      rstudent unadjusted p-value Bonferroni p
13 -2.336418      0.030576      0.73383
>

```

hat-values 0.1 0.3 0.75 0.85 0.95

Studentized residual -2 -1 0 1 2

Cook's distance 0.0 0.2 0.4

p-value < 0.05 :La donnée 13 ne peut donc pas être considérée comme outlier.

Manipulation des composants de la régression:

The screenshot shows the RStudio environment with the following components:

- Editor:** Contains R code for creating a data frame and fitting a model.


```

7 #fitted values
8 #fitted values
9
10 coResidus <- Mydata
11 coResidus$fitted_Ajuste <- fitted(fit)
12 coResidus$residus <- residuals(fit)
13
14
15
      
```
- Console:** Shows the execution of the code, resulting in a data frame named `coResidus`.


```

> coResidus
      
```

	Cigarette	TAR	Nicotine	WEIGHT	CO	fitted_Ajuste	residus
1	Alpine	14.1	0.86	985.3	13.6	14.457808	-0.85780786
2	benson&Hedges	16.0	1.06	1093.8	16.6	16.473498	0.12650177
3	camellights	8.0	0.67	928.0	10.2	8.826060	1.37393969
4	Carlton	4.1	0.40	946.2	5.4	5.262335	0.13766471
5	ChisterField	15.0	1.04	888.5	15.0	15.148795	-0.14879507
6	GoldenLights	8.8	0.76	1026.7	9.0	9.787987	-0.78798657
7	kent	12.4	0.95	922.5	12.3	12.865148	-0.56514826
8	Kool	16.6	1.12	937.2	16.3	16.711625	-0.41162538
9	L&M	14.9	1.02	885.8	15.4	15.044047	0.35595279
10	LarkLights	13.7	1.01	964.3	13.0	14.136991	-1.13699104
11	Marlboro	15.1	0.90	931.6	14.4	15.254481	-0.85448085
12	Merit	7.8	0.57	970.5	10.0	8.684997	1.31500277
13	Multifilter	11.4	0.78	1124.0	10.2	12.308248	-2.10824812
14	NewportLights	9.0	0.74	851.1	9.5	9.590076	-0.09007608
15	Now	1.0	0.13	785.1	1.5	2.035915	-0.53591499
16	oldGold	17.0	1.26	918.6	18.5	17.100642	1.39935792
17	PallMallLight	12.8	1.08	1039.5	12.6	13.530861	-0.93086077
18	Raleigh	15.8	0.96	957.3	17.5	15.960329	1.53967122
19	salemUltra	4.5	0.42	910.6	4.9	5.553728	-0.65372788
20	Tareyton	14.5	1.01	1007.0	15.9	14.935791	0.96420917
21	TrueLight	7.3	0.61	980.6	8.5	8.282983	0.21701707
22	viceroyRichLight	8.6	0.69	969.3	10.6	9.454822	1.14517819
23	virginiaslms	15.2	1.02	949.6	13.9	15.442938	-1.54293811
24	winstonLights	12.0	0.82	1118.4	14.9	12.849894	2.05010567

- Analyse en Composantes Principales :

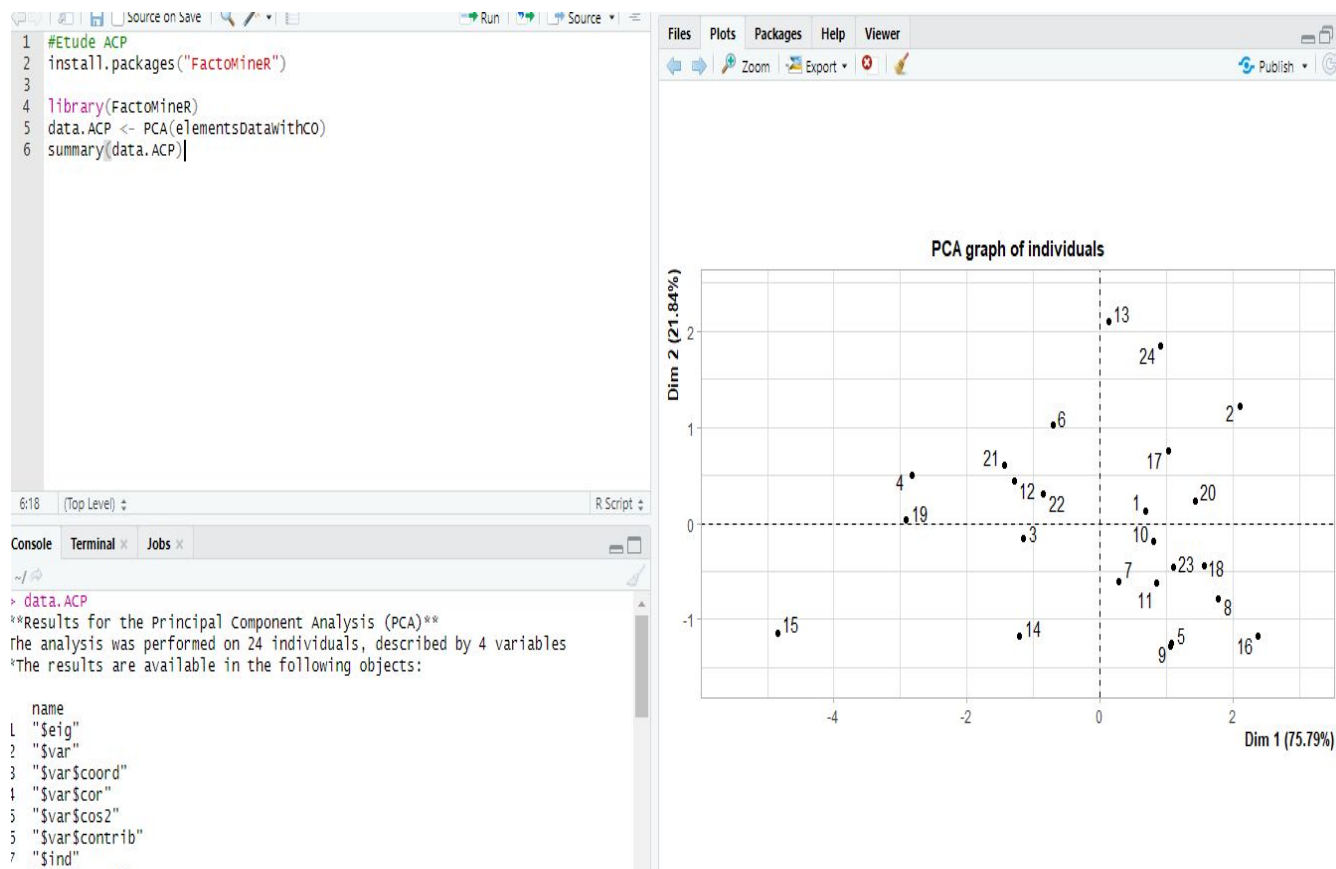
L'ACP permet alors de passer d'un nuage de points (les observations (24)) qui évolue dans un espace à N dimensions (N étant égal au nombre de variables (ou colonnes), à une représentation en deux dimensions, (ou éventuellement plusieurs représentations à deux dimensions).

Elle va ensuite considérer **cet axe : la première composante principale**, et **son perpendiculaire: seconde composante principale**, pour **définir un plan (le premier plan de l'ACP)**. Et pour résumer le nuage des observations, **l'ACP va les projeter sur le plan défini**. Une approche similaire est réalisée avec le nuage des variables.

A la fin de l'ACP, on obtient **deux représentations graphiques, en deux dimensions** : l'une **concernant les observations**:

```
> library(FactoMineR)
```

```
> data.ACP <- PCA(elementsWithDataCo)
```



l'autre les **variables**:

V1:TAR

|V2:Nicotine

|V3:WEIGHT

|V4:CO

```

1 #Etude ACP
2 install.packages("FactoMiner")
3
4 library(FactoMiner)
5 data.ACP <- PCA(elementsDataWithCO)
6 summary(data.ACP)

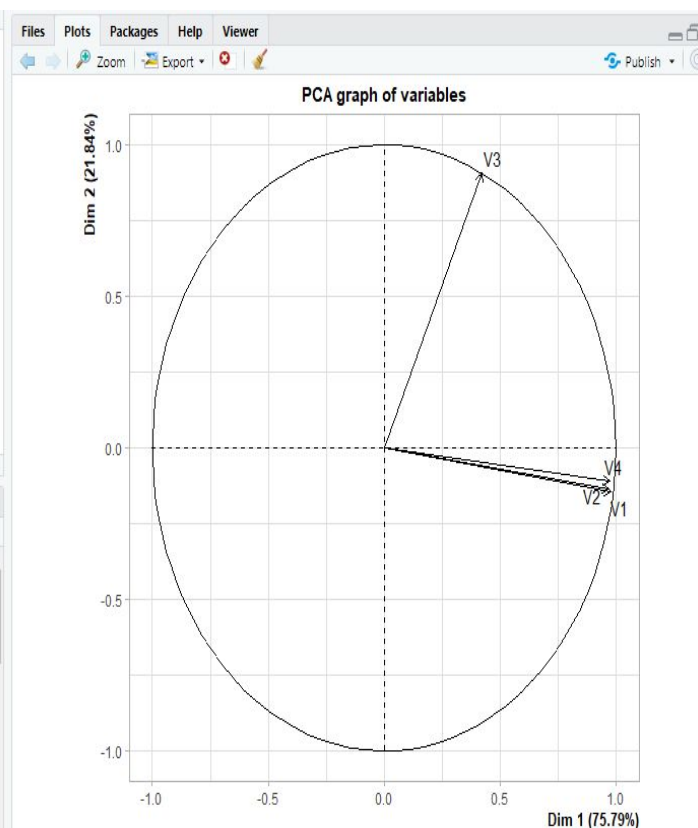
```

```

> data.ACP
**Results for the Principal Component Analysis (PCA)**
The analysis was performed on 24 individuals, described by 4 variables
The results are available in the following objects:

  name
1 "$eig"
2 "$var"
3 "$var$coord"
4 "$var$cor"
5 "$var$cos2"
6 "$var$contrib"
7 "$ind"
8 "$ind$coord"
9 "$ind$cor"

```



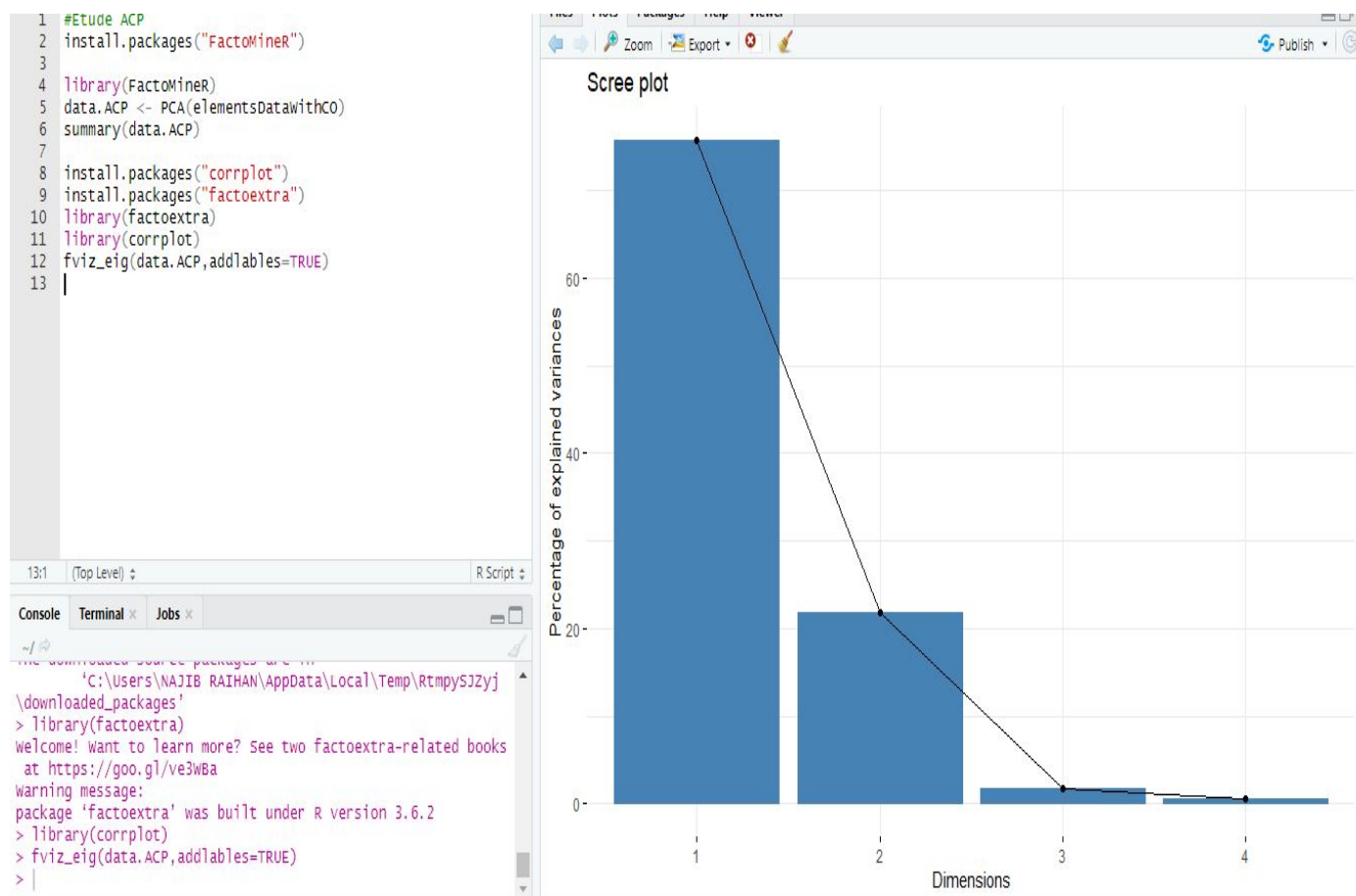
Visualiser le pourcentage d'inertie des axes:

Le package **Factoextra** peut être utilisé pour améliorer les représentations graphiques de base du package **FactoMiner**, et notamment pour explorer la contribution des variables aux axes, ou encore la qualité de leur représentation.

```
>library(Factoextra)
```

```
>library(corrplot)
```

```
>fviz_eig(data.ACP ,addlables =TRUE)
```



Visualiser la qualité de la représentation des observations:

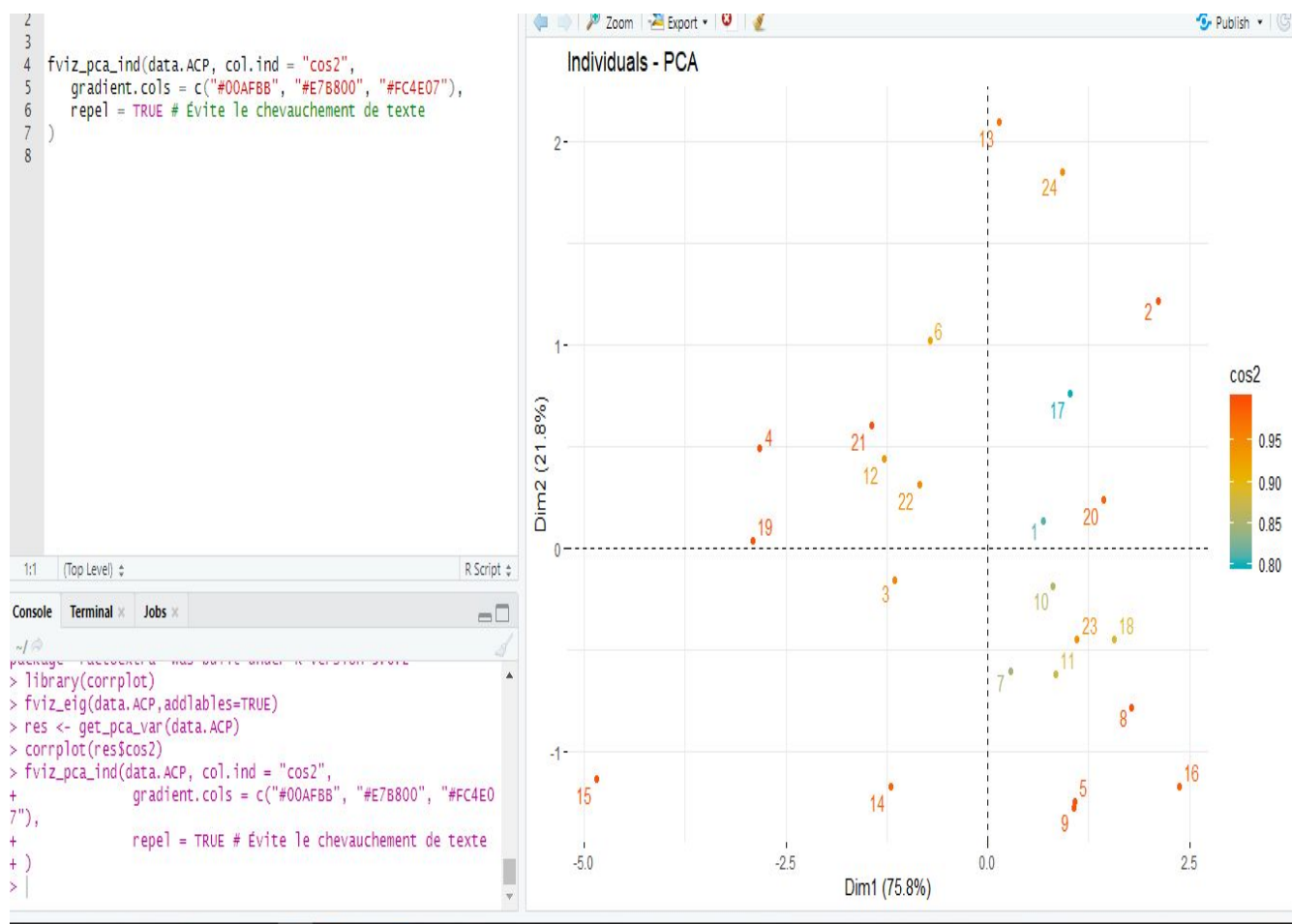
colorer les individus en fonction de leurs valeurs de \cos^2

NOTE: Lorsque l'angle est proche de 0, c'est-à-dire que le cosinus est proche de 1, l'individu(type de cigarette) est bien représenté. Dans le cas inverse, l'angle est proche de 90 ° et le cosinus est proche de 0.

```

>fviz_pca_ind (data.ACP, col.ind = "cos2",
               gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
               repel = TRUE # Évite le chevauchement de texte
               )

```



- Classification sous R pour notre modèle :

Objet: Opérer des regroupements en classes homogènes d'un ensemble d'individus (types Cigarette).

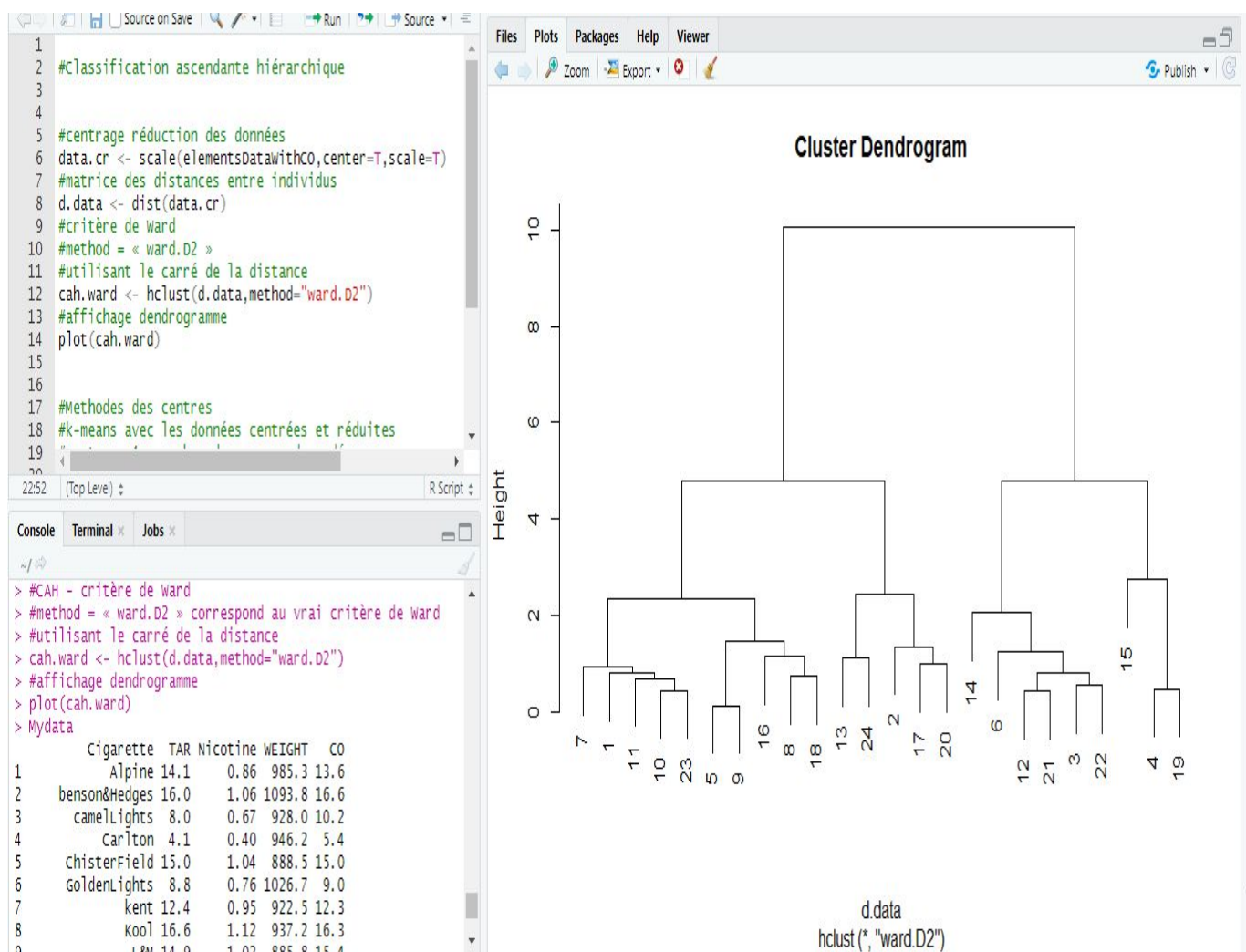
```

1      Cigarette
2      Alpine
3      benson&Hedges
4      camellights
5      Carlton
6      ChisterField
7      GoldenLights
8      kent
9      Kool
10     L&M
11     LarkLights
12     Marlboro
13     Merit
14     Multifilter
15     NewportLights
16     Now
17     OldGold
18     PallMallLight
19     Raleigh
20     SalemUltra
21     Tareyton
22     TrueLight
23     ViceroyRichLight
24     Virginiaslms
25     winstonLights

```

1-Classification hiérarchique "Ward algorithm": suites de partitions emboîtées

```
#Classification ascendante hiérarchique
#centrage réduction des données
>data.cr <- scale(elementsDataWithCO,center=T,scale=T)
#matrice des distances entre individus
>d.data <- dist(data.cr)
#critère de Ward
#method = « ward.D2 »
#utilisant le carré de la distance
>cah.ward <- hclust(d.data,method="ward.D2")
#affichage dendrogramme
>plot(cah.ward)
```



2-Classification non-hiérarchique :Méthode des centres mobiles

```
#k-means avec les données centrées et réduites
#center = nombre de groupes demandés
#nstart = nombre d'essais avec différents individus de départ
groupes.kmeans <- kmeans(data.cr,centers=4,nstart=5)
#affichage des résultats
print(groupes.kmeans)
```

```
25 #k-means avec les données centrées et réduites
26 #center = nombre de groupes demandés
27 #nstart = nombre d'essais avec différents individus de départ
28
29 groupes.kmeans <- kmeans(data.cr,centers=4,nstart=5)
30 #affichage des résultats
31 print(groupes.kmeans)
32
33 #découpage en 4 groupes
34 groupes.cah <- cutree(cah.ward,k=4)
35
```

33:1 (Top Level) ↕

Console Terminal Jobs

```
~/
> groupes.kmeans <- kmeans(data.cr,centers=4,nstart=5)
> #affichage des résultats
> print(groupes.kmeans)
K-means clustering with 4 clusters of sizes 4, 3, 6, 11

Cluster means:
      [,1]      [,2]      [,3]      [,4]
1  0.3548382  0.4016267  1.65785889  0.3546428
2 -1.8761124 -1.9265532 -1.02547486 -1.9186075
3 -0.7323256 -0.5836138 -0.09786646 -0.5746981
4  0.7820853  0.6977123 -0.26980111  0.7077673

Clustering vector:
[1] 4 1 3 2 4 3 4 4 4 4 3 1 3 2 4 1 4 2 4 3 3 4 1

within cluster sum of squares by cluster:
[1] 3.692630 3.888045 3.475678 6.710918
(between_SS / total_SS = 80.7 %)

Available components:
```

Le 4ème group de Cigarettes (Max degré de danger) **18,8,16,9,5,23,10,11,1,7**

Le 3eme groupe de Cigarettes (dangereux) : **20,17,2,24,13**

Le 2eme groupe de Cigarettes (moins degré de danger) : **22,3, 12, 21,6,14**

Le 1er groupe de Cigarettes Lite (Min degré de danger) : **19:SalemUltra 4:Carlton 15: Now**

My Project (Source Code) on Github :



link: <https://github.com/Raihannajib/L-etude-du-Modele-par-langage-R>

References:

Cours de Analyse des données ENSAH (CI-1 | DMI)

<http://www.sthda.com/french/>

<https://statistique-et-logiciel-r.com/>

<http://maths.cnam.fr/IMG/pdf/Classification-2008-2.pdf>

<http://iml.univ-mrs.fr/~reboul/ADD3-MAB.pdf>

http://eric.univ-lyon2.fr/~ricco/cours/didacticiels/R/cah_kmeans_avec_r.pdf