S5_4_2_GLM_TP

Table of contents

1 Consigne

1. Construire une table descriptive de cet échantillon
2. Proposer un ou plusieurs modèles de régression logistique pour étudier les facteurs associés à la démence et rapportez les résultats sous forme de table ou figure pouvant apparaître dans un article scientifique.

Base de données de 373 patients atteints ou non d’une démence de type Alzheimer.

La colonne “Group” contient cette information sous forme d’une chaîne de caractères.

Les autres variables importantes sont :

- Age : contient l’âge en année
- M.F. : contient le sexe (M/F)
- EDUC : contient le niveau d’éducation en année
- SES : contient le statut socio-économique (1 à 5)
- MMSE : contient le score au Mini-Mental State Examination (0 à 30)
- CDR : contient le Clinical Dementia Rating (0 à 5)
- eTIV : contient le volume intracrânien estimé en mL
- nWBV : contient la proportion du volume intracrânien occupée par le cerveau

2 Question 1 = Table descriptive

```
head(alzh)
```

```
        Group M.F Age EDUC SES MMSE CDR eTIV  nWBV   ASF
1 Nondemented   M  87   14   2   27 0.0 1987 0.696 0.883
2 Nondemented   M  88   14   2   30 0.0 2004 0.681 0.876
3    Demented   M  75   12  NA   23 0.5 1678 0.736 1.046
4    Demented   M  76   12  NA   28 0.5 1738 0.713 1.010
5    Demented   M  80   12  NA   22 0.5 1698 0.701 1.034
6 Nondemented   F  88   18   3   28 0.0 1215 0.710 1.444
```

```
levels(alzh$Group)
```

```
NULL
```

```
str(alzh)
```

```
'data.frame':   373 obs. of  10 variables:
 $ Group: chr  "Nondemented" "Nondemented" "Demented" "Demented" ...
 $ M.F  : chr  "M" "M" "M" "M" ...
 $ Age  : int  87 88 75 76 80 88 90 80 83 85 ...
 $ EDUC : int  14 14 12 12 12 18 18 12 12 12 ...
 $ SES  : int  2 2 NA NA NA 3 3 4 4 4 ...
 $ MMSE : int  27 30 23 28 22 28 27 28 29 30 ...
 $ CDR  : num  0 0 0.5 0.5 0.5 0 0 0 0.5 0 ...
 $ eTIV : int  1987 2004 1678 1738 1698 1215 1200 1689 1701 1699 ...
 $ nWBV : num  0.696 0.681 0.736 0.713 0.701 0.71 0.718 0.712 0.711 0.705 ...
 $ ASF  : num  0.883 0.876 1.046 1.01 1.034 ...
```

```
unique(alzh$Group)
```

```
[1] "Nondemented" "Demented"    "Converted"
```

```
table(alzh$Group)
```

```
  Converted    Demented Nondemented
         37         146         190
```

2.A Recode de la variable “Group”

On recode la variable “Group” pour qu’elle devienne binaire et prenne la valeur 1 si le patient est atteint de démence et 0 sinon.

```
alzh = read.csv("~/Documents/Projets/M2biostatistiques/Cours/alzheimer.csv")
alzh$Group = 1*(alzh$Group %in% c("Demented","Converted"))
table(alzh$Group)
```

```
  0   1
190 183
```

Syntaxe :

- `1*(condition)` : crée une variable binaire qui prend la valeur 1 si la condition est vraie et 0 sinon. (conversion logique en numérique)
- `condition` : `df$var %in% c("val1","val2")` vérifie si les valeurs de la variable `var` dans le dataframe `df` appartiennent au vecteur `c("val1","val2")`. Contient TRUE ou FALSE.
- l'opérateur `%in%` est utilisé pour vérifier l'appartenance d'une valeur à un ensemble de valeurs.

On aurait pu faire :

```
alzh = read.csv("~/Documents/Projets/M2biostatistiques/Cours/alzheimer.csv")
condition_validation <- c("Demented","Converted")
alzh$Group = 1*(alzh$Group %in% condition_validation)
table(alzh$Group)
```

```
  0   1
190 183
```

ou utiliser la fonction `ifelse()` avec :

- `ifelse(test, yes, no)` : retourne `yes` si `test` est vrai et `no` sinon.

```
alzh = read.csv("~/Documents/Projets/M2biostatistiques/Cours/alzheimer.csv")
alzh$Group = ifelse(alzh$Group %in% c("Demented","Converted"), 1, 0)
table(alzh$Group)
```

```
  0   1
190 183
```

2.B Construction du tableau

2.B.1 Version nullach !!!

```
median(alzh$Age)
```

```
[1] 77
```

```r
quantile(alzh$Age, probs = c(0.25, 0.75))
```

```
25% 75%
 71  82
```

```r
median(alzh$Age[alzh$Group==1])
```

```
[1] 76
```

```r
mm = median(alzh$Age[alzh$Group==0])
```

Si on veut afficher entre tirets

```r
iqr = paste(quantile(alzh$Age, probs = c(0.25, 0.75)), collapse = " - ")
```

Si on veut faire stylé :

```r
paste(mm," (", iqr, ")", sep = "")
```

```
[1] "77 (71 - 82)"
```

Et mettre ça dans une fonction :

- function(gr,v) : gr = groupe (0 ou 1), v = variable (Age, EDUC, etc..)
- median(alzh$Age[alzh$Group==gr]) : calcule la médiane de la variable Age pour le groupe gr
- quantile(alzh$Age, probs = c(0.25, 0.75)) : calcule le 1er et 3ème quartile de la variable Age
- paste(..., collapse = " - ") : concatène les deux quartiles avec un tiret entre eux
- paste(mm," (", iqr,")", sep ="") : crée une chaîne de caractères avec la médiane et l'IQR formatée

```r
#ajouter la fonction à l'environnement
report_quanti = function(v, gr) {
    x = alzh[[v]][alzh$Group == gr] #groupe = gr car on veut séparer les deux groupes
    mm  = median(x, na.rm = TRUE)
    iqr = paste(quantile(x, probs = c(0.25, 0.75), na.rm = TRUE),
                collapse = " - ")
    paste(mm, " (", iqr, ")", sep = "")
}
# il faut mettre v entre double crochet car on veut accéder à la variable dont le nom est stock
# si on met qu'un seul crochet, R cherche une variable nommée "v" dans le dataframe alzh
```

Tip

Pourquoi il faut mettre un double crochets ?
Les dataframe se comportent à la fois comme des listes, à la fois comme des matrices

```
class(alzh)
```

```
[1] "data.frame"
```

comportement de matrice :

```
#avec un seul crochet, accès par nom de colonne ou par indice
alzh[1,3]
```

```
[1] 87
```

comportement de liste :

```
#avec double crochet, accès du contenu
alzh[[1]] #contenu de la première variable
```

```
  [1] 0 0 1 1 1 0 0 0 0 0 1 1 1 0 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1
 [38] 1 1 1 1 0 0 1 1 1 1 0 0 0 0 1 1 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 0 1 1
 [75] 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 0 0 0 1 1 0 0 0 0
[112] 0 0 0 1 1 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 0
[149] 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 1 1
[186] 1 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1
[223] 1 0 0 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 0 0 0 1 1 0 0 0 0 1 1 0
[260] 0 0 1 1 1 1 1 0 0 0 0 0 1 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 0 0 0 0 0 0 0 1
[297] 1 1 1 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 1 1 0 0 1 1 1 1 0 0 0 1 1 1 1 1 1
[334] 0 0 0 0 0 1 1 0 0 0 1 1 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1
[371] 0 0 0
```

Pour appliquer à toutes les variables quantitatives de la base : Age, EDUC, SES, MMSE, CDR, eTIV, nWBV

```
vars = c("Age","EDUC","SES","MMSE","CDR","eTIV","nWBV")
```

Utilisation de la fonction `sapply()` pour appliquer `report_quanti` à chaque variable dans `vars` pour les deux groupes (0 et 1)

- Syntaxe : 2 arguments : `sapply(X, FUN)`
  - `X` : vecteur d'éléments à itérer (ici `vars`)
  - `FUN` : fonction à appliquer (ici une fonction anonyme qui prend `v` et appelle `report_quanti` pour les deux groupes)

```
sapply(vars, report_quanti, gr=0) #groupe 0 (il faut le donner parce que la fonction report_qua
```

```
                      Age                       EDUC
            "77 (71 - 82)"             "16 (13 - 18)"
                      SES                       MMSE
               "2 (2 - 3)"             "29 (29 - 30)"
                      CDR                       eTIV
               "0 (0 - 0)" "1474.5 (1358.25 - 1634.75)"
                     nWBV
 "0.739 (0.71725 - 0.769)"
```

```
sapply(vars, report_quanti, gr=1) #groupe 1
```

```
                   Age                     EDUC                      SES
        "76 (72 - 82)"        "14 (12 - 16)"           "3 (1 - 4)"
                  MMSE                      CDR                     eTIV
        "27 (23 - 29)"     "0.5 (0.5 - 0.5)"  "1463 (1357 - 1568.5)"
                  nWBV
"0.713 (0.695 - 0.738)"
```

2.C Autres façons de faire

2.C.1 tableone

```
library(tableone)
vars = c("Age","M.F","EDUC","SES","MMSE","CDR","eTIV","nWBV")
table1 = CreateTableOne(vars = vars, strata = "Group", data = alzh, factorVars = c("M.F.","SES"
```

```
Warning in ModuleReturnVarsExist(factorVars, data): The data frame does not
have: M.F.  Dropped
```

```
print(table1, showAllLevels = TRUE, formatOptions = list(big.mark = ","))
```

```
                  Stratified by Group
                   level 0                 1                 p      test
  n                              190               183
  Age (mean (SD))            77.06 (8.10)      76.97 (7.16)    0.909
  M.F (%)          F           129 (67.9)         84 (45.9)   <0.001
                   M            61 (32.1)         99 (54.1)
  EDUC (mean (SD))           15.14 (2.74)      14.03 (2.91)   <0.001
  SES (%)          1            41 (21.6)         47 (28.7)    0.004
                   2            71 (37.4)         32 (19.5)
                   3            42 (22.1)         40 (24.4)
                   4            34 (17.9)         40 (24.4)
```

```
                  5            2 ( 1.1)          5 ( 3.0)
  MMSE (mean (SD))          29.23 (0.88)      25.36 (4.40)   <0.001
  CDR (%)          0          188 (98.9)         18 ( 9.8)   <0.001
                   0.5          2 ( 1.1)        121 (66.1)
                   1            0 ( 0.0)         41 (22.4)
                   2            0 ( 0.0)          3 ( 1.6)
  eTIV (mean (SD))       1,495.50 (184.89) 1,480.48 (166.73)  0.411
  nWBV (mean (SD))           0.74 (0.04)       0.72 (0.03)   <0.001
```

2.C.2 gtsummary : je vois pas pourquoi on utilise pas ce banger !

```
library(gtsummary)
```

```
Attaching package: 'gtsummary'

The following object is masked from 'package:ape':

    where
```

```
colnames(alzh)
```

```
 [1] "Group" "M.F"   "Age"   "EDUC"  "SES"   "MMSE"  "CDR"   "eTIV"  "nWBV"
[10] "ASF"
```

```
cols_to_include = c("Group", "Age","M.F","EDUC","SES","MMSE","CDR","eTIV","nWBV")
alzh$Group = factor(alzh$Group, labels = c("Non-Demented","Demented"))

table2 = alzh %>%
    select(all_of(cols_to_include)) %>%
    tbl_summary(
        by = Group,
        # Change 'labels' to 'label' below
        label = list(
        Age ~ "Age (years)",
        M.F ~ "Sex",
        EDUC ~ "Education (years)",
        SES ~ "Socioeconomic Status",
        MMSE ~ "Mini-Mental State Examination",
        CDR ~ "Clinical Dementia Rating",
        eTIV ~ "Estimated Total Intracranial Volume",
        nWBV ~ "Normalized Whole Brain Volume"
        ),
        statistic = list(all_continuous() ~ "{median} ({p25} - {p75})",
                         all_categorical() ~ "{n} / {N} ({p}%)"),
        digits = all_continuous() ~ 2
```

```
    ) %>%
    modify_header(label = "**Variable**") %>%
    bold_labels() %>%
    add_n() %>%
    add_p() %>%
    modify_caption("**Table 1. Descriptive Statistics by Dementia Status**") %>%
    as_kable()

# View the table
table2
```

Table 1: **Table 1. Descriptive Statistics by Dementia Status**

Variable	**N**	**Non-Demented** N = 190	**Demented** N = 183	**p-value**
Age (years)	373	77.00 (71.00 - 82.00)	76.00 (72.00 - 82.00)	0.9
Sex	373			<0.001
F		129 / 190 (68%)	84 / 183 (46%)	
M		61 / 190 (32%)	99 / 183 (54%)	
Education (years)	373	16.00 (13.00 - 18.00)	14.00 (12.00 - 16.00)	<0.001
Socioeconomic Status	354			0.003
1		41 / 190 (22%)	47 / 164 (29%)	
2		71 / 190 (37%)	32 / 164 (20%)	
3		42 / 190 (22%)	40 / 164 (24%)	
4		34 / 190 (18%)	40 / 164 (24%)	
5		2 / 190 (1.1%)	5 / 164 (3.0%)	
Unknown		0	19	
Mini-Mental State Examination	371	29.00 (29.00 - 30.00)	27.00 (23.00 - 29.00)	<0.001
Unknown		0	2	
Clinical Dementia Rating	373			<0.001
0		188 / 190 (99%)	18 / 183 (9.8%)	
0.5		2 / 190 (1.1%)	121 / 183 (66%)	
1		0 / 190 (0%)	41 / 183 (22%)	
2		0 / 190 (0%)	3 / 183 (1.6%)	
Estimated Total Intracranial Volume	373	1,474.50 (1,358.00 - 1,636.00)	1,463.00 (1,357.00 - 1,569.00)	0.5
Normalized Whole Brain Volume	373	0.74 (0.72 - 0.77)	0.71 (0.70 - 0.74)	<0.001