

regression logistique diabète détection

2025-06-18

Exemple R : régression logistique et test de Box–Tidwell sur PimaIndiansDiabetes2

Contexte

- Jeu de données : **PimaIndiansDiabetes2** (package `mlbench`).
- Objectif : prédire la survenue du diabète (`diabetes`, binaire « pos »/« neg ») à partir de mesures physiologiques chez des femmes Pima[^2^].
- Variable à expliquer (Y) : `diabetes`
- Variables explicatives (X) :
 - `glucose` : concentration de glucose plasmatique
 - `pressure` : pression diastolique
 - `triceps` : épaisseur du pli cutané tricipital
 - `insulin` : insulinémie sérique
 - `mass` : indice de masse corporelle
 - `pedigree` : fonction de transmission du diabète
 - `age` : âge en années - `pregnant`: mois de grossesses

hypothèse 1 la variable cible est binaire (0/1 ou 1/2)

visualisation des données

aperçu des données avec `head()`:

```
head(df)
```

```
##   pregnant glucose pressure triceps insulin mass pedigree age diabetes
## 4        1     89       66      23      94 28.1    0.167   21     neg
## 5        0    137       40      35     168 43.1    2.288   33     pos
## 7        3     78       50      32      88 31.0    0.248   26     pos
## 9        2    197       70      45     543 30.5    0.158   53     pos
## 14       1    189       60      23     846 30.1    0.398   59     pos
## 15       5    166       72      19     175 25.8    0.587   51     pos
```

```
levels(PimaIndiansDiabetes2$diabetes)
```

```
## [1] "neg" "pos"
```

oui la variable est bien binaire pour la variable d'intérêt diabète, l'hypothèse 1 est vérifiée (ont 2 résultats possibles “neg” et “pos”)

hypothèse 2 linéarité entre les variables explicatives et la fonction logit de la variable à expliquer:

exemple avec une variable explicative: exemple avec la variable age:

```
# 1) Création des variables explicative adapté au model Box-Tidwell
df$age_bt <- df$age * log(df$age)

# 2) Ajout du terme Box-Tidwell au modèle
mod0 <- glm(diabetes ~ age,           data=df, family=binomial) #modèle de regression basique
mod_bt <- glm(diabetes ~ age + age_bt,
               data = df, family = binomial) #modèle pour vérifier la linéarité de age

# 3) vérification du modèle créer
summary(mod_bt)

##
## Call:
## glm(formula = diabetes ~ age + age_bt, family = binomial, data = df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q      Max
## -1.4417 -0.8054 -0.5196  0.9780  2.1333
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -13.31530   2.64885 -5.027 4.99e-07 ***
## age          1.41810   0.34126  4.155 3.25e-05 ***
## age_bt      -0.29142   0.07389 -3.944 8.02e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 498.10 on 391 degrees of freedom
## Residual deviance: 432.52 on 389 degrees of freedom
## AIC: 438.52
##
## Number of Fisher Scoring iterations: 4

# 4) visualisation directe de la p value:
p_bt <- summary(mod_bt)$coefficients["age_bt", "Pr(>|z|)"]
print(paste("p-value (age_bt) =", round(p_bt, 4)))

## [1] "p-value (age_bt) = 1e-04"
```

Ici la p-value de l'age est inférieur à 0.05 on rejette l'hypothèse H_0 de linéarité du logit, il faut adapter la variable age dans ce cas, si on veux l'ajouter au modèle

```

# 1) Création des termes BT pour plusieurs variables
vars <- c("glucose", "pressure", "triceps", "insulin", "mass", "pedigree", "age")
for (v in vars) {
  df[[paste0(v, "_bt")]] <- df[[v]] * log(df[[v]])
}

# 2) Ajustement du modèle logistique avec tous les termes BT
formule_bt <- paste0("diabetes ~ ",
                      paste(vars, collapse = " + "), " + ",
                      paste(paste0(vars, "_bt"), collapse = " + "))
mod_bt_multi <- glm(as.formula(formule_bt),
                     data = df, family = binomial)

summary(mod_bt_multi)

```

exemple avec plusieurs variables explicatives

```

##
## Call:
## glm(formula = as.formula(formule_bt), family = binomial, data = df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max 
## -2.4686 -0.6508 -0.2860  0.6175  2.7018 
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -17.376498   8.810535 -1.972  0.04858 *  
## glucose      -0.109313   0.258255 -0.423  0.67209    
## pressure     -0.472072   0.380196 -1.242  0.21436    
## triceps      0.215423   0.294365  0.732  0.46428    
## insulin      0.032937   0.020409  1.614  0.10656    
## mass         0.560042   0.641247  0.873  0.38246    
## pedigree     2.074414   0.787171  2.635  0.00841 ** 
## age          1.340740   0.459457  2.918  0.00352 ** 
## glucose_bt   0.024764   0.043869  0.565  0.57241    
## pressure_bt  0.090063   0.072902  1.235  0.21668    
## triceps_bt  -0.046585   0.066541 -0.700  0.48386    
## insulin_bt   -0.005021   0.003072 -1.634  0.10222    
## mass_bt       -0.110755   0.138742 -0.798  0.42471    
## pedigree_bt  -1.557066   0.952943 -1.634  0.10227    
## age_bt        -0.279298   0.099751 -2.800  0.00511 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 498.10  on 391  degrees of freedom
## Residual deviance: 325.77  on 377  degrees of freedom
## AIC: 355.77
##
## Number of Fisher Scoring iterations: 5

```

```

#vérification des VIF, avec modèle glm(simple)
formule <- paste0("diabetes ~ ",
                  paste(vars, collapse=" + "), collapse=" + ") #format d'écriture pour le modèle glm
mod_multi<- glm(as.formula(formule),
                  data = df, family = binomial) #modèle glm
#vérification des vif
vif(mod_multi) #dans ce dataset les vif sont inférieur à 5 on est bon

```

```

## glucose pressure triceps insulin mass pedigree age
## 1.372081 1.190718 1.658114 1.379178 1.823170 1.016410 1.150417

```

Ici la p-value de l'age est toujours inférieur à 0.05 on rejette l'hypothèse H_0 de linéarité du logit, il faut adapté la variable age dans ce cas, si on veux l'ajouter au modèle

adaptation avec la fonction logit

```

#on adapte la variable age qui n'est pas purement linéaire avec logit
df$age2 <- log(df$age) #on la renomme après age2
#donc on refait tout le processus
vars2 <- c("glucose","pressure","triceps","insulin","mass","pedigree","age2")
for (v in vars2) {
  df[[paste0(v,"_bt")]] <- df[[v]] * log(df[[v]])
}

formule_bt2 <- paste0("diabetes ~ ",
                      paste(vars2, collapse=" + "), " + ",
                      paste(paste0(vars2,"_bt"), collapse=" + "))
mod_bt_multi2 <- glm(as.formula(formule_bt2),
                      data = df, family = binomial)

summary(mod_bt_multi2)

```

```

##
## Call:
## glm(formula = as.formula(formule_bt2), family = binomial, data = df)
##
## Deviance Residuals:
##      Min        1Q     Median        3Q       Max
## -2.5050   -0.6424   -0.2787    0.6199    2.7195
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.178e+02  4.367e+01 -2.697  0.00700 **
## glucose     -1.185e-01  2.585e-01 -0.458  0.64659
## pressure    -4.813e-01  3.824e-01 -1.259  0.20816
## triceps     2.288e-01  2.942e-01  0.778  0.43661
## insulin     3.316e-02  2.043e-02  1.623  0.10455
## mass         5.751e-01  6.413e-01  0.897  0.36984
## pedigree    2.085e+00  7.868e-01  2.650  0.00804 **
## age2        6.961e+01  2.802e+01  2.485  0.01297 *
## glucose_bt   2.631e-02  4.391e-02  0.599  0.54903

```

```

## pressure_bt  9.194e-02  7.331e-02   1.254  0.20980
## triceps_bt -4.991e-02  6.648e-02  -0.751  0.45280
## insulin_bt -5.054e-03  3.077e-03  -1.643  0.10044
## mass_bt     -1.140e-01  1.388e-01  -0.822  0.41125
## pedigree_bt -1.568e+00  9.540e-01  -1.643  0.10035
## age2_bt      -2.989e+01  1.239e+01  -2.412  0.01588 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 498.10  on 391  degrees of freedom
## Residual deviance: 325.83  on 377  degrees of freedom
## AIC: 355.83
##
## Number of Fisher Scoring iterations: 5

```

L'âge étant toujours significatif au test de box tidwell malgré la fonction logit, on utilise la librairie mfp qui sélectionne automatiquement la (ou les) meilleures puissances pour adapter notre variable âge

```

# 2) fit d'une logistique avec FP sur age (mettre fp() sur les variables à adapter)
mod_fp <- mfp(
  diabetes ~ fp(age) + glucose + pressure + triceps +
    insulin + mass + pedigree,
  family = binomial, data = df
)

# 3) résumé du modèle
summary(mod_fp)

```

```

##
## Call:
## glm(formula = diabetes ~ glucose + I((age/10)^-2) + pedigree +
##       mass + insulin + triceps + pressure, family = binomial, data = df)
##
## Deviance Residuals:
##       Min        1Q      Median        3Q       Max
## -2.8148  -0.6669  -0.3264   0.6210   2.5414
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.650e+00  1.250e+00 -5.319  1.05e-07 ***
## glucose      3.717e-02  5.827e-03  6.378  1.79e-10 ***
## I((age/10)^-2) -1.206e+01  2.586e+00 -4.662  3.13e-06 ***
## pedigree     1.058e+00  4.255e-01  2.487  0.0129 *
## mass         6.819e-02  2.699e-02  2.526  0.0115 *
## insulin     -5.728e-04  1.332e-03 -0.430  0.6672
## triceps      7.541e-03  1.730e-02  0.436  0.6629
## pressure     -4.438e-03  1.182e-02 -0.376  0.7072
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)

```

```

## Null deviance: 498.10 on 391 degrees of freedom
## Residual deviance: 336.85 on 384 degrees of freedom
## AIC: 352.85
##
## Number of Fisher Scoring iterations: 5

```

on transforme alors la variable âge de cette façon: $(\text{age}/10)^{-2}$ on revérifie dans un modèle de box_tidwell cette adaptation:

```

#on ajoute les variables adapté a notre df pour le verifier ensuite
df$age2 <- (df$age / 10)^(-2)
df$age2_bt <- df$age2 * log(df$age/10)

mod_multi_adapt <- glm(as.formula(formule_bt2),
                         data = df, family = binomial)
summary(mod_multi_adapt)

```

```

##
## Call:
## glm(formula = as.formula(formule_bt2), family = binomial, data = df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max 
## -2.5764 -0.6401 -0.2769  0.6233  2.7284
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -5.591995  8.368156 -0.668  0.50398  
## glucose      -0.127779  0.258697 -0.494  0.62135  
## pressure     -0.496726  0.383945 -1.294  0.19575  
## triceps      0.237325  0.293498  0.809  0.41874  
## insulin      0.033987  0.020413  1.665  0.09592 .  
## mass         0.624351  0.639749  0.976  0.32910  
## pedigree     2.085251  0.786080  2.653  0.00798 ** 
## age2        -26.549137 13.031098 -2.037  0.04161 *  
## glucose_bt   0.027857  0.043951  0.634  0.52620  
## pressure_bt  0.095002  0.073597  1.291  0.19676  
## triceps_bt  -0.052146  0.066322 -0.786  0.43172  
## insulin_bt   -0.005186  0.003075 -1.687  0.09168 .  
## mass_bt      -0.124614  0.138421 -0.900  0.36799  
## pedigree_bt  -1.553122  0.953219 -1.629  0.10324  
## age2_bt      25.058566 21.380979  1.172  0.24120  
## ---      
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ',' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 498.10 on 391 degrees of freedom
## Residual deviance: 326.32 on 377 degrees of freedom
## AIC: 356.32
##
## Number of Fisher Scoring iterations: 5

```

la variables age à bien été adapté au modèle, il n'est plus significatif avec une p-value > 0.05 au test de box_tidwel

hypothèse 3 éviter les multicolinéarités

la meilleur méthodes vérification est la méthodes par le VIF #### Vérification du VIF

```
#vérification des VIF, avec modèle glm(simple)
formule <- paste0("diabète ~ ",
                  paste(vars, collapse=" + "), collapse=" + ") #format d'écriture pour le modèle glm
mod_multi<- glm(as.formula(formule),
                  data = df, family = binomial) #modèle glm
#vérification des vif
vif(mod_multi) #dans ce dataset les vif sont inférieur à 5 on est bon
```

```
## glucose pressure triceps insulin mass pedigree age
## 1.372081 1.190718 1.658114 1.379178 1.823170 1.016410 1.150417
```

les VIF semblent correcte inférieur à 5

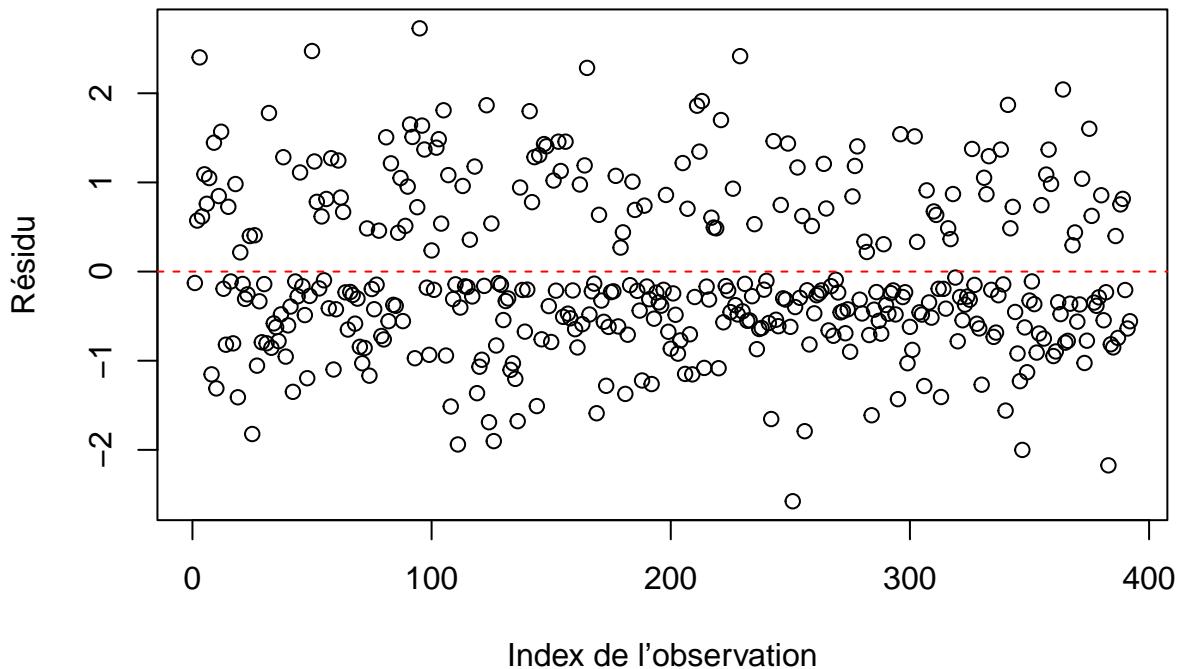
hypothèses 4 l'indépendance des variables explicatives

la meilleure façon de vérifier cette hypothèse reste le graphique des résidus en fonction du temps: dans notre graphique comme il n'y a pas de variable chronologique on se basera sur l'ordre des observations directement

```
#dans le cas où il y'aurait eu une variable date:
# Assure-toi que la variable date est bien au format Date
#df$date <- as.Date(df$date)
# Puis trie le dataframe selon la date (ordre croissant)
#df <- df[order(df$date), ]

# 1. reprendre le modèle mod_multi_adapt
mod_multi_adapt <- mod_multi_adapt #inutile
# 2. Extraire les résidus (Pearson ou de deviance)
résidus <- resid(mod_multi_adapt, type = "deviance") # ou type = "pearson"
# 3. Tracer les résidus en fonction de l'ordre des observations
plot(résidus, type = "p", main = "Résidus en fonction de l'ordre",
      xlab = "Index de l'observation", ylab = "Résidu")
abline(h = 0, col = "red", lty = 2)
```

Résidus en fonction de l'ordre



```
# Compter les positifs et négatifs
n_positifs <- sum(residus > 0)
n_negatifs <- sum(residus < 0)

# Afficher les résultats
cat("Résidus positifs : ", n_positifs, "\n")

## Résidus positifs : 130

cat("Résidus négatifs : ", n_negatifs, "\n")

## Résidus négatifs : 262

total <- length(residus)
prop_positifs <- n_positifs / total
prop_negatifs <- n_negatifs / total

prop_negatifs

## [1] 0.6683673

prop_positifs

## [1] 0.3316327
```

hypothèses 5 les variables abérante et comment les retirer

```
library(ggplot2)
library(gridExtra)

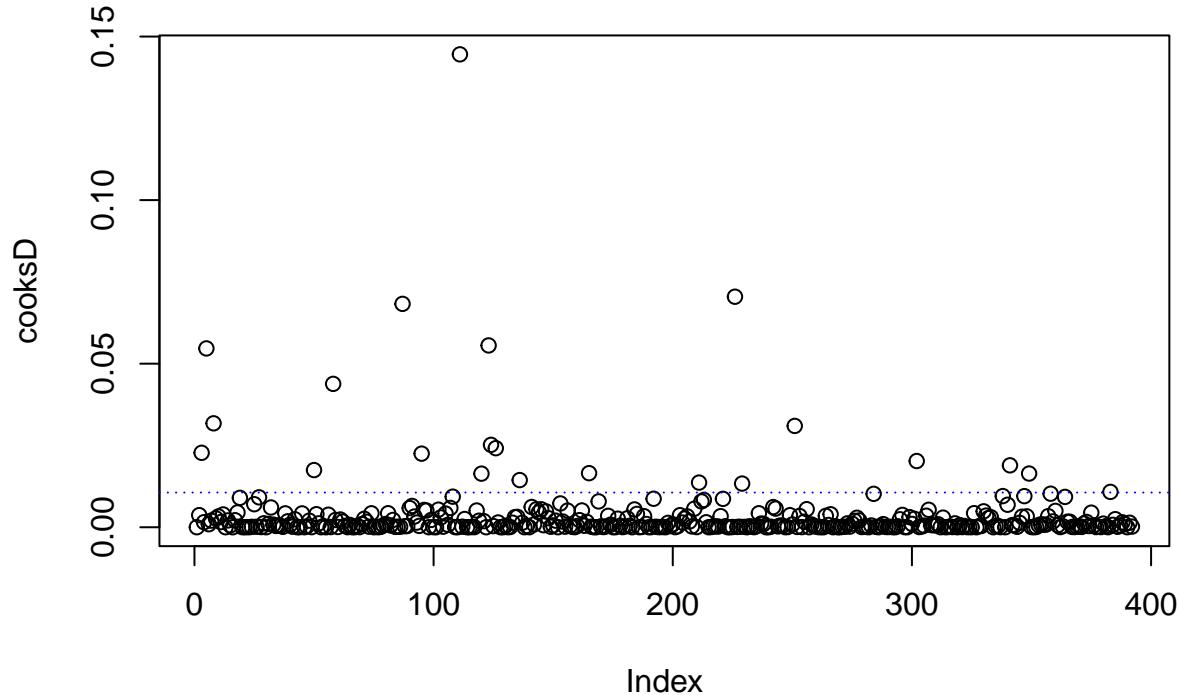
#on repren le modèle multiple avec toutes no variable pour notre regression logisitique
mod_multi_adapt

## 
## Call: glm(formula = as.formula(formule_bt2), family = binomial, data = df)
##
## Coefficients:
## (Intercept)      glucose      pressure      triceps      insulin      mass
## -5.591995     -0.127779     -0.496726     0.237325     0.033987    0.624351
## pedigree       age2      glucose_bt  pressure_bt   triceps_bt  insulin_bt
## 2.085251     -26.549137     0.027857     0.095002    -0.052146   -0.005186
## mass_bt      pedigree_bt      age2_bt
## -0.124614     -1.553122     25.058566
##
## Degrees of Freedom: 391 Total (i.e. Null); 377 Residual
## Null Deviance: 498.1
## Residual Deviance: 326.3 AIC: 356.3

#trouve la Cook's distance pour chaque observation dans le jeu de donnée
cooksD <- cooks.distance(mod_multi_adapt)

# graphique de la distance de Cook with avec une ligne horizontale a 4/n et voir ce qui dépasse
#exceed this thresdhold
n <- nrow(df)
p <- length(coef(mod_multi_adapt)) # nb de paramètres (incluant l'intercept)
plot(cooksD, main = "Cooks Distance for Influential Obs")
abline(h = 4/(n-p), col = "blue", lty = 3) # rajoute la limite (pour p predicteur)
```

Cooks Distance for Influential Obs



```
influentes <- which(cooksD > 4/(n-p))
df[influentes, ] # affiche les lignes influentes
```

	pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age	diabetes
## 7	3	78	50	32	88	31.0	0.248	26	pos
## 14	1	189	60	23	846	30.1	0.398	59	pos
## 19	1	103	30	38	83	43.3	0.183	33	neg
## 110	0	95	85	25	36	37.4	0.247	24	pos
## 126	1	88	30	42	99	55.0	0.496	26	pos
## 178	0	129	110	46	130	67.1	0.319	26	pos
## 198	3	107	62	13	48	22.9	0.678	23	pos
## 229	4	197	70	39	744	36.7	2.329	31	neg
## 248	0	165	90	33	680	52.3	0.427	23	neg
## 255	12	92	62	7	258	27.6	0.926	44	pos
## 259	1	193	50	16	375	25.9	0.655	24	neg
## 261	3	191	68	15	130	30.9	0.299	34	neg
## 287	5	155	84	44	545	38.7	0.619	34	neg
## 329	2	102	86	36	120	45.5	0.127	23	pos
## 415	0	138	60	35	167	34.6	0.534	21	pos
## 446	0	180	78	63	14	59.4	2.420	25	pos
## 449	0	104	64	37	64	33.6	0.510	22	pos
## 488	0	173	78	32	265	46.5	1.159	58	neg
## 585	8	124	76	24	600	28.7	0.687	52	pos
## 660	3	80	82	31	70	34.2	1.292	27	pos
## 674	3	123	100	35	240	57.3	0.880	22	neg

```

## 745      13     153      88      37     140 40.6    1.174   39      neg
##          age_bt glucose_bt pressure_bt triceps_bt insulin_bt mass_bt
## 7      84.71051   339.8233   195.6012  110.90355   394.0056 106.45360
## 14     240.57471   990.6902   245.6607  72.11637   5702.4794 102.47621
## 19     115.38475   477.3771   102.0359  138.22827   366.7638 163.16101
## 110    76.27329   432.6183   377.6254   80.47190   129.0067 135.45048
## 126    84.71051   394.0056   102.0359  156.98212   454.9169 220.40333
## 178    84.71051   626.9158   517.0528  176.11750   632.7795 282.23495
## 198    72.11637   499.9927   255.8823   33.34434   185.8176 71.70304
## 229    106.45360  1040.7911   297.3947  142.87890   4919.3585 132.22191
## 248    72.11637   842.4810   404.9829  115.38475   4435.0231 206.95091
## 255    166.50434  416.0045   255.8823   13.62137   1432.6636 91.57172
## 259    76.27329   1015.6992   195.6012   44.36142   2222.5973 84.28489
## 261    119.89626  1003.1842   286.9265   40.62075   632.7795 106.01037
## 287    119.89626  781.7309   372.1886   166.50434   3433.9283 141.48099
## 329    72.11637   471.7472   383.0739   129.00668   574.4990 173.70591
## 415    63.93497   679.9610   245.6607  124.43718   854.7050 122.61734
## 446    80.47190   934.7322   339.8233  261.01749   36.9468 242.60708
## 449    68.00293   483.0167   266.1685  133.60396   266.1685 118.08808
## 488    235.50569  891.5194   339.8233  110.90355   1478.6284 178.53453
## 585    205.46467  597.7149   329.1357  76.27329   3838.1578 96.34295
## 660    88.98760   350.5621   361.3510  106.45360   297.3947 120.80212
## 674    68.00293   591.8987   460.5170  124.43718   1315.3533 231.96763
## 745    142.87890  769.6570   394.0056  133.60396   691.8299 150.37298
##          pedigree_bt    age2    age2_bt
## 7      -0.34579298 0.14792899 0.14134785
## 14     -0.36667870 0.02872738 0.05098973
## 19     -0.31078325 0.09182736 0.10963475
## 110    -0.34539663 0.17361111 0.15199110
## 126    -0.34778496 0.14792899 0.14134785
## 178    -0.36447797 0.14792899 0.14134785
## 198    -0.26347622 0.18903592 0.15744974
## 229    1.96902741 0.10405827 0.11773175
## 248    -0.36336473 0.18903592 0.15744974
## 255    -0.07119185 0.05165289 0.07652916
## 259    -0.27714363 0.17361111 0.15199110
## 261    -0.36098620 0.08650519 0.10586293
## 287    -0.29690335 0.08650519 0.10586293
## 329    -0.26207316 0.18903592 0.15744974
## 415    -0.33500994 0.22675737 0.16823976
## 446    2.13871745 0.16000000 0.14660652
## 449    -0.34340572 0.20661157 0.16290441
## 488    0.17101922 0.02972652 0.05225499
## 585    -0.25791422 0.03698225 0.06097110
## 660    0.33099930 0.13717421 0.13624853
## 674    -0.11249337 0.20661157 0.16290441
## 745    0.18832923 0.06574622 0.08947906

```

```
nrow(df[influents, ]) #il y a 28 lignes influentes donc trop forte et influent trop le modèle (que l'on
```

```
## [1] 22
```

```
# pour les supprimer, on prend df[-influents, ]
df_clean <- df[-influents, ]
```

df_clean est donc notre nouveau jeu de donnée sans valeur “aberrante” (valeurs trop “rare”)

mise en place du modèle finale

```
model_prefinale <- glm(diabetes ~ glucose + pressure + triceps + insulin + mass + pedigree + age2, data = df)
summary(mod_multi_adapt)
```

```
##
## Call:
## glm(formula = as.formula(formule_bt2), family = binomial, data = df)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.5764  -0.6401  -0.2769   0.6233   2.7284
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -5.591995   8.368156 -0.668  0.50398
## glucose        -0.127779   0.258697 -0.494  0.62135
## pressure       -0.496726   0.383945 -1.294  0.19575
## triceps        0.237325   0.293498  0.809  0.41874
## insulin        0.033987   0.020413  1.665  0.09592 .
## mass            0.624351   0.639749  0.976  0.32910
## pedigree        2.085251   0.786080  2.653  0.00798 **
## age2           -26.549137  13.031098 -2.037  0.04161 *
## glucose_bt     0.027857   0.043951  0.634  0.52620
## pressure_bt    0.095002   0.073597  1.291  0.19676
## triceps_bt    -0.052146   0.066322 -0.786  0.43172
## insulin_bt    -0.005186   0.003075 -1.687  0.09168 .
## mass_bt         -0.124614   0.138421 -0.900  0.36799
## pedigree_bt   -1.553122   0.953219 -1.629  0.10324
## age2_bt        25.058566  21.380979  1.172  0.24120
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 498.10  on 391  degrees of freedom
## Residual deviance: 326.32  on 377  degrees of freedom
## AIC: 356.32
##
## Number of Fisher Scoring iterations: 5
```

On retire les prédicteurs non-significatifs avec une P-value supérieur à 0.05:

```
model_finale <- glm(diabetes ~ glucose + mass + pedigree + age2, data = df, family = binomial)
summary(model_finale)
```

```

## 
## Call:
## glm(formula = diabetes ~ glucose + mass + pedigree + age2, family = binomial,
##      data = df)
## 
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max 
## -2.8796 -0.6549 -0.3303  0.6356  2.5778 
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept) -6.768592  1.028089 -6.584 4.59e-11 ***
## glucose      0.035765  0.005011  7.137 9.54e-13 ***
## mass         0.071212  0.020359  3.498 0.000469 *** 
## pedigree     1.069722  0.421978  2.535 0.011244 *  
## age2        -12.031214  2.442044 -4.927 8.36e-07 *** 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
## Null deviance: 498.10  on 391  degrees of freedom
## Residual deviance: 337.36  on 387  degrees of freedom
## AIC: 347.36
## 
## Number of Fisher Scoring iterations: 5

```

on garde donc le modèle finale “model_finale”

teste de préiction:

```

# 5 dernières lignes de df, enregistrées dans df_derniers
df_derniers <- tail(df, 5)
dernier_clean <- df_derniers[ , -c(1,1)]
dernier_clean2 <- dernier_clean[ , -c(2, 3, 4,7,8,9,10,11,12,13,14,15,17,18)]
# on prédit
(prediction <- predict(model_finale,newdata = dernier_clean2,type = "response"))

##          754          756          761          764          766 
## 0.77661355 0.65935322 0.03678301 0.28218062 0.16110750

```