Projet

Nadia GHERNAOUT
Philippine RENAUDIN

1 Introduction

Le but est d'affecter un *score* à un individu par rapport à son comportement en comparaison au comportement général d'un panel. Ce score sert ensuite de critère de classification : l'individu sera affecté à un groupe en fonction de celui-ci.

Exemples d'applications :

1. risque médical : prédire si le patient présente des prédispositions à développer une certaine maladie
2. risque financier : prédire la bonne ou mauvaise santé d'une entreprise

Un score peut également être définit comme un outil statistique ou probabiliste de détection de risque. Plusieurs méthodes permettent de construire un score :

— La régression logistique
— L'analyse factorielle discriminante

2 Régression logistique

2.1 Introduction

La Régression logistique consiste à expliquer une variable Y (variable cible), par une ou plusieurs variables explicatives.

Dans ce projet on se concentre au cas où la variable à expliquer est binaire.
Quand le nombre de modalités de la variable à expliquer est supérieur à 2 on parle de régression logistique *polytomique* (scrutin a plus de deux candidats, degrés de satisfaction pour un produit, mention a un examen...)
Les avantages de cette méthode sont qu'il n'y a pas besoin d'hypothèses de multinormalité. Dans la régression logistique

$$\mathbb{P}(G1|x) = \mathbb{P}(Y = 1|X = x) = \pi(x) = \frac{e^{\beta_0+\beta_1 x_1+\cdots+\beta_p x_p}}{1 + e^{\beta_0+\beta_1 x_1+\cdots+\beta_p x_p}}$$

On note : $\beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}$ et $x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{pmatrix}$

$$\pi(x) = \frac{e^{\beta_0+\beta x}}{1 + e^{\beta_0+\beta x}}$$

La fonction $\pi(x)$ est appelée *fonction logistique*. Sa représentation graphique est une sigmoïde en fonction des modalités de x. La fonction $\pi(x)$ est comprise dans $[0, 1]$, elle convient donc à une probabilité et donne souvent une bonne représentation des phénomènes.

On cherche à écrire l'espérance conditionnelle de la variable à expliquer Y comme combinaison linéaire de variables à expliquer X. On veut modéliser l'espérance conditionnelle $\mathbb{E}[Y|X = x]$.
On cherche la valeur moyenne de Y pour toute valeurs de X.

$$logit(\pi(x)) = \ln\left(\frac{\pi(x)}{1-\pi(x)}\right) = \beta_0 + \beta_1 + \cdots + \beta_p x_p$$

Utilisée avec la fonction de logarithme néperien, logit est la réciproqie de $f(x) = \frac{1}{1+e^{-x}}$ qui est utilisée pour linéariser les fonctions logistiques.

```
#librairies
library(ROCR)
library(ggplot2)
library(GGally)
library(forestmodel)
library(effects)
library(ggeffects)
library(boot)
library(corrplot)
library(dplyr)
library(devtools)
library(caTools)

#fichier
diabete = read.table('/Users/Nadia/Documents/Maths/DATA_SCIENCES/PROJET_SCORING/
    diabetes.csv', header = TRUE, sep= ',')

attach(diabete)

#Colonnes supp
Breaks_age = c(min(Age),24,29,41, max(Age)
Breaks_preg = c(0,1,3,6, max(Pregnancies))

diabete$Age_classe = cut(Age, breaks = Breaks_age, include.lowest = TRUE)
diabete$Pregnancies_classe = cut(Pregnancies, breaks = Breaks_preg, include.lowest
    = TRUE)
diabete$sqrtInsulin = (sqrt(Insulin)-1)*2
  #boxcox
# diabete$logInsulin = log(Insulin) pose probleme, les valeurs nulles

diabete
summary(diabete)

diabete[,9] <- factor(diabete[,9])
str(diabete)    # 0: pas de diabete, 1: diabete


db_cor <- round(cor(diabete[1:8]),1)

corrplot(db_cor)   #pas beaucoup de correlation entre les variables
N = 1000
mod_ret = rep(0,N)

scores_A = rep(0,10)


for (k in 1:N){
  sample = sample.split(Outcome, SplitRatio = 0.8)
```

```
47  train = subset(diabete, sample == TRUE)
    test = subset(diabete, sample == FALSE)
49
    modele_1 = glm(Outcome ~ . - Age_classe - sqrtInsulin - Pregnancies_classe ,
       data = train, family = binomial(link = "logit"))
51  modele_2 = glm(Outcome ~ . -Insulin - Age_classe - Pregnancies_classe , data =
       train, family=binomial(link="logit"))
    modele_3 = glm(Outcome ~ Pregnancies + Glucose + BMI + DiabetesPedigreeFunction
       + BloodPressure + Age, data = train, family = binomial(link = "logit"))
53  modele_4 = glm(Outcome ~  Glucose + BMI + DiabetesPedigreeFunction , data =
       train, family = binomial(link = "logit"))
    modele_5 = glm(Outcome ~  Glucose + BMI + DiabetesPedigreeFunction +
       BloodPressure + Pregnancies, data = train, family = binomial(link = "logit")
       )
55  modele_6 = glm(Outcome ~  Glucose + BMI + DiabetesPedigreeFunction +
       BloodPressure + Age, data = train, family = binomial(link = "logit"))
    modele_7 = glm(Outcome ~  . - Age - sqrtInsulin - Pregnancies_classe , data =
       train, family = binomial(link = "logit"))  #modele complet avec les classes
       d' ge
57  modele_8 = glm(Outcome ~  Glucose + BMI + DiabetesPedigreeFunction +
       BloodPressure + Age_classe, data = train, family = binomial(link = "logit"))
    modele_9 = glm(Outcome ~  Glucose + BMI + DiabetesPedigreeFunction + Age_classe,
        data = train, family = binomial(link = "logit"))
59  modele_10 = glm(Outcome ~  Glucose + BMI + DiabetesPedigreeFunction +
       BloodPressure + Pregnancies_classe, data = train, family = binomial(link = "
       logit"))

61  liste_modeles = list(modele_1, modele_2, modele_3, modele_4, modele_5, modele_6,
        modele_7, modele_8, modele_9, modele_10)
    n = length(liste_modeles)
63  erreur = 1
    j = 0           #num ro du mod le retenu
65  A = matrix(0, nrow = 2, ncol=n)
    A[1,]= 1:n
67
    for (i in 1:n){
69    outcome.pred = predict(liste_modeles[[i]], newdata=test, type="response")  #
         rend un score
      erreur_pred = prop.table(table(outcome.pred>0.3, test$Outcome))[3]   #rend le
         taux de faux negatifs
71    A[2,i] = erreur_pred
#     if (erreur_pred<erreur){  #on cherche    minimiser les faux n gatifs
73    #  erreur<-erreur_pred
      #  j = as.integer(i) #modele i retenu
75     # }
    }
77  A_tri = A[,order(A[2,], decreasing = FALSE)]
    for (i in 1:n){
79    scores_A[A_tri[1,i]] = scores_A[A_tri[1,i]] + (11-i)}
 #mod_ret[k]=j
81 }
   scores_A
83 #attribuer des scores: matrice deux lignes, une avec les modeles une avec les
      scores

85 res = as.data.frame(prop.table(table(mod_ret))); res
   max_occ = which.max(res$Freq)
87 cat("Le mod le    retenir est le mod le", max_occ)
```

```
"Mod le final"

Reg_fin = glm(Outcome ~  Glucose + BMI + DiabetesPedigreeFunction + BloodPressure
   + Age_classe, data = diabete, family = binomial(link = "logit"))
outcome_pred = rep(0,dim(diabete)[1])     #vecteur de 0

for (i in 1: dim(diabete)[1]){
  if (Reg_fin$fitted.values[i] >=0.3){
    outcome_pred[i] = 1
  }   #ce qu'on a pr dit
}
score_pred = filter(data.frame(Reg_fin$fitted.values, outcome_pred), outcome_pred
   == "1")  #score predit, ceuw qu'on a predit comme  tant  malades
score = filter(data.frame(Reg_fin$fitted.values, diabete$Outcome), diabete$Outcome
    == "1")  #vrai score, ceux qui sont
score_1 = filter(data.frame(Reg_fin$fitted.values, diabete$Outcome), diabete$
   Outcome == "1")  #vrai score, ceux qui sont
score_0 = filter(data.frame(Reg_fin$fitted.values, diabete$Outcome), diabete$
   Outcome == "0")  #vrai score, ceux qui sont


hgA = hist(score_pred$Reg_fin.fitted.values, breaks=10, plot=F)
hgB1 = hist(score_1$Reg_fin.fitted.values, breaks=20, plot=F)
hgB2 = hist(score_0$Reg_fin.fitted.values, breaks=20, plot=F)


col_1 = rgb(1,0,0,0.5)
col_2 = rgb(0,0,1,0.5)

plot(hgB1, col= col_1, freq=FALSE, xlim=c(0,1), ylim= c(0,3))
plot(hgB2, col=col_2, freq=FALSE, xlim=c(0,1), add=T)
lines(density(score_0$Reg_fin.fitted.values), lwd=1.5)
lines(density(score_1$Reg_fin.fitted.values), lwd=1.5)
```

3 Scoring