PimaIndiansDiabetes-TP

Solène Issartel

TP Noté: Pima Indians Diabetes

```
library(ggplot2)
```

Problématique

1- Question

Peut-on modéliser la présence/absence du diabètes (variable `diabetes`) chez des femmes Amérindiennes de la tribu Pima en fonction de différentes mesures physiologiques et médicales ?

2- Objectif

L'objectif de ce TP est de modéliser voire prédire la présence/absence du diabètes en fonction de différentes mesures physiologiques et médicales qui sont :

- `pregnant` : Nombre de fois enceinte
- `glucose` : La concentration de glucose dans le plasma (test de tolérance au glucose)
- `pressure` : Pression sanguine diastolique (mm Hg)
- `triceps` : Epaisseur du pli cutané du triceps (mm)
- `insulin` : Insuline sérique à 2 heures (mu U/ml)
- `mass` : Indice de masse corporelle (poids en kg/(taille en m)^2)
- `pedigree` : Fonction du pedigree du diabète
- `age` : Age (années)

La variable sur laquelle se baser est la variable `diabetes`, qui est binaire :

- égale à 0 si la personne n'est pas atteinte du diabéte
- égale à 1 si la personne est diabétique

Pour répondre à la problématique, il faut trouver une régression adaptée qui permettra de construire un modèle et déterminer quelles variables ont un effet sur la présence/absence du diabéte.

Analyse des données

```
pima <- read.csv("https://plmlab.math.cnrs.fr/gdurif_teaching/polytech_ig5_regression_tutorial/raw/master/data/PimaIndiansDiabetes.csv", sep = ";", header=TRUE)

head(pima)
```

```
##   pregnant glucose pressure triceps insulin mass pedigree age diabetes
## 1        6     148       72      35       0 33.6    0.627  50        1
## 2        1      85       66      29       0 26.6    0.351  31        0
## 3        8     183       64       0       0 23.3    0.672  32        1
## 4        1      89       66      23      94 28.1    0.167  21        0
## 5        0     137       40      35     168 43.1    2.288  33        1
## 6        5     116       74       0       0 25.6    0.201  30        0
```

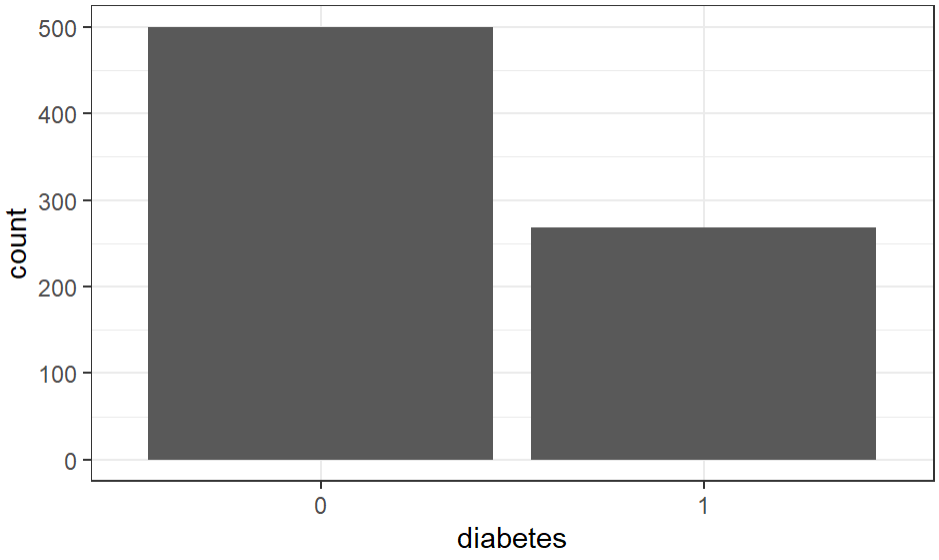
```
str(pima)
```

```
## 'data.frame':    768 obs. of  9 variables:
##  $ pregnant: int  6 1 8 1 0 5 3 10 2 8 ...
##  $ glucose : int  148 85 183 89 137 116 78 115 197 125 ...
##  $ pressure: int  72 66 64 66 40 74 50 0 70 96 ...
##  $ triceps : int  35 29 0 23 35 0 32 0 45 0 ...
##  $ insulin : int  0 0 0 94 168 0 88 0 543 0 ...
##  $ mass    : num  33.6 26.6 23.3 28.1 43.1 25.6 31 35.3 30.5 0 ...
##  $ pedigree: num  0.627 0.351 0.672 0.167 2.288 ...
##  $ age     : int  50 31 32 21 33 30 26 29 53 54 ...
##  $ diabetes: int  1 0 1 0 1 0 1 0 1 1 ...
```

```
# effectif des groupes (important pour vérifier si les groupes sont équilibrés)
table(pima$diabetes)
```

```
##
##   0   1
## 500 268
```

```
ggplot(pima) + geom_bar(aes(as.factor(diabetes))) + xlab("diabetes") + theme_bw()
```



Remarque : On peut voir que l'échantillon des femmes Amérindiennes de la tribu Pima contient des effectifs

très déséquilibrés. En effet, on remarque que les personnes non diabétiques sont deux fois plus présentes que les personnes diabétiques. Pour que cela n'est pas d'impacts, il faut utiliser des probabilités. Cela est directement mis en place par la fonction glm.

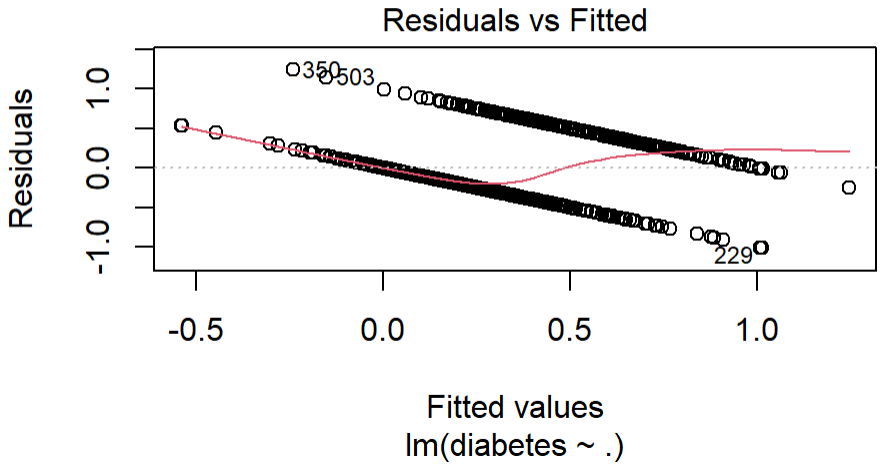
Régression linéaire

Pour répondre à la problématique, nous pouvons tester d'utiliser un modèle de régression linéaire pour voir s'il est adapté.

```
# modèle de régression linéaire
lin_reg <- lm(diabetes ~ ., data = pima)
# R2
summary(lin_reg)$r.squared
```

```
## [1] 0.3032531
```

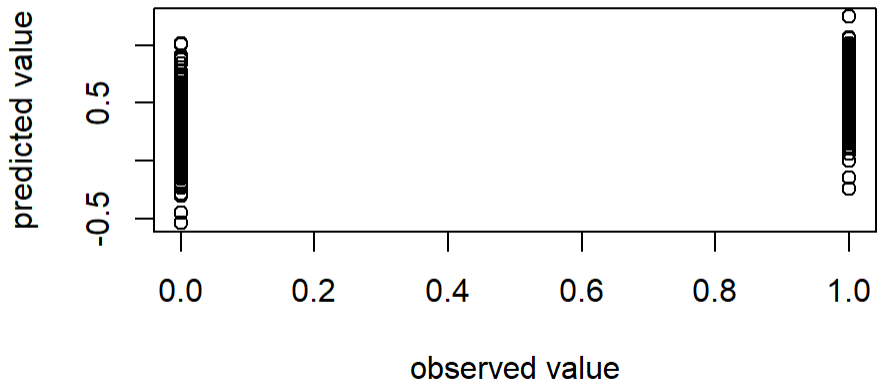
```
plot(lin_reg, which=1)
```




```
# resultat
summary(lin_reg)
```

```
##
## Call:
## lm(formula = diabetes ~ ., data = pima)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.01348 -0.29513 -0.09541  0.32112  1.24160
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.8538943  0.0854850  -9.989  < 2e-16 ***
## pregnant     0.0205919  0.0051300   4.014 6.56e-05 ***
## glucose      0.0059203  0.0005151  11.493  < 2e-16 ***
## pressure    -0.0023319  0.0008116  -2.873  0.00418 **
## triceps      0.0001545  0.0011122   0.139  0.88954
## insulin     -0.0001805  0.0001498  -1.205  0.22857
## mass         0.0132440  0.0020878   6.344 3.85e-10 ***
## pedigree     0.1472374  0.0450539   3.268  0.00113 **
## age          0.0026214  0.0015486   1.693  0.09092 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4002 on 759 degrees of freedom
## Multiple R-squared:  0.3033, Adjusted R-squared:  0.2959
## F-statistic: 41.29 on 8 and 759 DF,  p-value: < 2.2e-16
```

```
# réponse : valeurs prédites vs valeurs observées
plot(pima$diabetes, predict(lin_reg), xlab = "observed value",
     ylab = "predicted value")
```



Résultat : La régression linéaire n'est pas adaptée car la réponse n'est pas continue. En effet, on peut voir qu'elle prend ses valeurs seulement dans $0, 1$.

Régression logistique

```
# modèle de régression logistique
log_reg <- glm(diabetes ~ ., data = pima, family="binomial")
coef(log_reg)
```

```
##   (Intercept)      pregnant       glucose      pressure       triceps
## -8.4046963669  0.1231822984  0.0351637146 -0.0132955469  0.0006189644
##       insulin          mass      pedigree           age
## -0.0011916990  0.0897009700  0.9451797406  0.0148690047
```

```
#Résultats plus détaillés
summary(log_reg)
```

```
##
## Call:
## glm(formula = diabetes ~ ., family = "binomial", data = pima)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5566  -0.7274  -0.4159   0.7267   2.9297
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.4046964  0.7166359 -11.728  < 2e-16 ***
## pregnant     0.1231823  0.0320776   3.840 0.000123 ***
## glucose      0.0351637  0.0037087   9.481  < 2e-16 ***
## pressure    -0.0132955  0.0052336  -2.540 0.011072 *
## triceps      0.0006190  0.0068994   0.090 0.928515
## insulin     -0.0011917  0.0009012  -1.322 0.186065
## mass         0.0897010  0.0150876   5.945 2.76e-09 ***
## pedigree     0.9451797  0.2991475   3.160 0.001580 **
## age          0.0148690  0.0093348   1.593 0.111192
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 993.48  on 767  degrees of freedom
## Residual deviance: 723.45  on 759  degrees of freedom
## AIC: 741.45
##
## Number of Fisher Scoring iterations: 5
```

Résultat :

- `Deviance Residuals` : Ici on remarque que la médiane est proche de 0 (ici `-0.4159`), ce qui signifie que le modèle est assez efficace.

- `Coefficients` : Le résultat montre que les variables `Triceps` , `Insulin` et `Age` ne sont pas statistiquement significatives. En d'autres termes, les autres variables (`pregnant` , `glucose` , `pressure` , `mass`) ont un effet sur la présence ou non du diabète.
- `Null deviance` :

Rappel : Plus la `Null deviance` est faible, plus le modèle nul explique bien la réponse et les prédicteurs n'apportent pas d'information supplémentaires pour expliquer la réponse.

Remarque : On peut voir que la `Null deviance` est de *993.48 à 767 degrés de liberté*, ce qui est très élevé. On peut donc dire que le modèle nul n'explique pas très bien la réponse et les prédicteurs apportent des informations supplémentaires pour expliquer la réponse.

- `Residual Deviance` :

Rappel : Plus la `Residuals deviance` est faible, plus le modèle proposé explique bien la réponse et les prédicteurs sont utiles pour mieux expliquer la réponse.

Remarque : On peut voir qu'en incluant les variables indépendantes, la déviance diminue et passe à *723.45 pour 759 degrés de liberté*, ce qui correspond à un diminution de *27%* par rapport à la déviance nulle, pour une perte de *8* degrés de liberté. Cela prouve que le modèle a bien été ajusté et que les prédicteurs sont utiles pour expliquer la variable réponse.

Prédiction

Valeurs prédites $\hat{y}_i$:

```
# probabilités ajustées
hat_pi <- predict(log_reg, type="response")
# valeurs ajustées (threshold 50%)
hat_y <- as.integer(hat_pi > 0.5)
```

Explications : La variable `hat_pi` est une prédiction qui permet de donner des valeurs aux observation en fonction de l'absence/présence du diabéte. Cependant les valeur données sont continues et comprises entre [0,1]. Pour résoudre cela, la variable `hat_y` permet d'arrondir la valeur : si elle est inférieure à 0,5 elle sera considérée comme un 0, sinon comme un 1.

1- Sélection Backward

Objectif : Partir d'un modèle complet avec toutes les variables et essayer de les retirer une par une pour obtenir le meilleur modèle possible. Le critère AIC permettra de comparer ces modèles (on retire l'AIC le plus petit à chaque étape).

```
back_sel <- step(log_reg, direction="backward")
```

```
## Start:  AIC=741.45
## diabetes ~ pregnant + glucose + pressure + triceps + insulin +
##     mass + pedigree + age
##
##            Df Deviance    AIC
## - triceps   1   723.45 739.45
## - insulin   1   725.19 741.19
## <none>          723.45 741.45
## - age       1   725.97 741.97
## - pressure  1   729.99 745.99
## - pedigree  1   733.78 749.78
## - pregnant  1   738.68 754.68
## - mass      1   764.22 780.22
## - glucose   1   838.37 854.37
##
## Step:  AIC=739.45
## diabetes ~ pregnant + glucose + pressure + insulin + mass + pedigree +
##     age
##
##            Df Deviance    AIC
## <none>          723.45 739.45
## - insulin   1   725.46 739.46
## - age       1   725.97 739.97
## - pressure  1   730.13 744.13
## - pedigree  1   733.92 747.92
## - pregnant  1   738.69 752.69
## - mass      1   768.77 782.77
## - glucose   1   840.87 854.87
```

```
summary(back_sel)
```

```
##
## Call:
## glm(formula = diabetes ~ pregnant + glucose + pressure + insulin +
##     mass + pedigree + age, family = "binomial", data = pima)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.5617  -0.7286  -0.4156   0.7271   2.9297
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.4051362  0.7167033 -11.727  < 2e-16 ***
## pregnant     0.1231724  0.0320688   3.841 0.000123 ***
## glucose      0.0351123  0.0036625   9.587  < 2e-16 ***
## pressure    -0.0132136  0.0051537  -2.564 0.010350 *
## insulin     -0.0011570  0.0008142  -1.421 0.155275
## mass         0.0900886  0.0144619   6.229 4.68e-10 ***
## pedigree     0.9475954  0.2980063   3.180 0.001474 **
## age          0.0147888  0.0092897   1.592 0.111393
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 993.48  on 767  degrees of freedom
## Residual deviance: 723.45  on 760  degrees of freedom
## AIC: 739.45
##
## Number of Fisher Scoring iterations: 5
```

Résultat : A la fin de l'opération, l'AIC est de `739,5` ce qui est mieux que le modèle avec toutes les variables. On sélectionne alors le modèle avec les variables : `pregnant` , `glucose` , `pressure` , `insulin` , `mass` , `pedigree` et `age` .

2- Sélection Forward

Objectif : Partir d'un modèle vide sans variables et essayer d'en ajouter une par une pour obtenir le meilleur modèle possible. Le critère AIC permettra de comparer ces modèles (on retient l'AIC le plus petit à chaque étape).

```
# le modèle de base est le modèle nul (celui avec uniquement un intercept)
log_reg0 <- glm(diabetes ~ 1, data = pima, family="binomial")
# la régression forward part du modèle nul et l'enrichit
forward_sel <- step(log_reg0, direction="forward", scope=list(lower=log_reg0,
upper=~pregnant+glucose+pressure+triceps+insulin+mass+pedigree+age))
```

```
## Start:  AIC=995.48
## diabetes ~ 1
##
##            Df Deviance    AIC
## + glucose   1   808.72 812.72
## + mass      1   920.71 924.71
## + age       1   950.72 954.72
## + pregnant  1   956.21 960.21
## + pedigree  1   970.86 974.86
## + insulin   1   980.81 984.81
## + triceps   1   989.19 993.19
## + pressure  1   990.13 994.13
## <none>          993.48 995.48
##
## Step:  AIC=812.72
## diabetes ~ glucose
##
##            Df Deviance    AIC
## + mass      1   771.40 777.40
## + pregnant  1   784.95 790.95
## + pedigree  1   796.99 802.99
## + age       1   797.36 803.36
## <none>          808.72 812.72
## + triceps   1   807.07 813.07
## + insulin   1   807.77 813.77
## + pressure  1   808.59 814.59
##
## Step:  AIC=777.4
## diabetes ~ glucose + mass
##
##            Df Deviance    AIC
## + pregnant  1   744.12 752.12
## + age       1   755.68 763.68
## + pedigree  1   762.87 770.87
## + insulin   1   767.79 775.79
## + pressure  1   769.07 777.07
## <none>          771.40 777.40
## + triceps   1   770.20 778.20
##
## Step:  AIC=752.12
## diabetes ~ glucose + mass + pregnant
##
##            Df Deviance    AIC
## + pedigree  1   734.31 744.31
## + pressure  1   738.43 748.43
## + age       1   742.10 752.10
## <none>          744.12 752.12
## + insulin   1   742.43 752.43
## + triceps   1   743.60 753.60
##
## Step:  AIC=744.31
```

```
## diabetes ~ glucose + mass + pregnant + pedigree
##
##            Df Deviance    AIC
## + pressure  1   728.56 740.56
## + insulin   1   731.51 743.51
## <none>          734.31 744.31
## + age       1   732.51 744.51
## + triceps   1   733.06 745.06
##
## Step:  AIC=740.56
## diabetes ~ glucose + mass + pregnant + pedigree + pressure
##
##           Df Deviance    AIC
## + age      1   725.46 739.46
## + insulin  1   725.97 739.97
## <none>         728.56 740.56
## + triceps  1   728.00 742.00
##
## Step:  AIC=739.46
## diabetes ~ glucose + mass + pregnant + pedigree + pressure +
##     age
##
##           Df Deviance    AIC
## + insulin  1   723.45 739.45
## <none>         725.46 739.46
## + triceps  1   725.19 741.19
##
## Step:  AIC=739.45
## diabetes ~ glucose + mass + pregnant + pedigree + pressure +
##     age + insulin
##
##           Df Deviance    AIC
## <none>         723.45 739.45
## + triceps  1   723.45 741.45
```

Résultat : A la fin de l'opération, l'AIC est de `739,5` comme pour la sélection Backward. Cela est dû au fait que la sélection Forward trouve les même variables pour son modèle. On sélectionne donc le même modèle avec les variables : `pregnant` , `glucose` , `pressure` , `insulin` , `mass` , `pedigree` et `age` .

3- Matrice de confusion

```
# matrice de confusion avec la fonction `table`
table(hat_y, pima$diabetes)
```

```
##
## hat_y   0   1
##     0 445 112
##     1  55 156
```

Explications : La matrice permet de mettre en évidence les données qui ont bien été classées. On retrouve 445 observations qui sont classées comme non diabétiques et qui ne le sont réellement pas (vrais négatifs), 156 observations qui sont classées comme diabétiques et qui le sont vraiment (vrais positifs). Cependant, il y a 112 observations qui ont été classées comme diabétiques et qui ne le sont pas (faux positifs) et 55 observations ont été classées comme non diabétiques alors qu'elles le sont en réalité (faux négatifs).

```
# matrice de confusion (et plus) avec la fonction `caret::confusionMatrix`
library(caret)
confusionMatrix(data = as.factor(hat_y),
                reference = as.factor(pima$diabetes),
                positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 445 112
##          1  55 156
##
##                Accuracy : 0.7826
##                  95% CI : (0.7517, 0.8112)
##     No Information Rate : 0.651
##     P-Value [Acc > NIR] : 1.373e-15
##
##                   Kappa : 0.4966
##
##  Mcnemar's Test P-Value : 1.468e-05
##
##             Sensitivity : 0.5821
##             Specificity : 0.8900
##          Pos Pred Value : 0.7393
##          Neg Pred Value : 0.7989
##              Prevalence : 0.3490
##          Detection Rate : 0.2031
##    Detection Prevalence : 0.2747
##       Balanced Accuracy : 0.7360
##
##        'Positive' Class : 1
##
```

```
# taux d'erreur (taux de mals classés)
# Objectif : plus proche de 0 possible
sum(hat_y != pima$diabetes)/length(pima$diabetes)
```

```
## [1] 0.2174479
```

```
# accuracy = 1 - taux d'erreur
# Objectif : plus proche de 1 possible
sum(hat_y == pima$diabetes)/length(pima$diabetes)
```

```
## [1] 0.7825521
```

Remarque : On peut voir que les deux métriques sont bonnes.

```
# sensibility (taux vrai positif)
# Objectif : plus élevé possible
# Ici représente : sur toutes les personnes prédites diabétiques, combien le sont-ell
es vraiment?
sum(hat_y == 1 & pima$diabetes == 1)/sum(pima$diabetes == 1)
```

```
## [1] 0.5820896
```

```
# taux faux négatifs (aussi égal à 1 - sensibility)
# Ici représente : le nombre de personnes diabétiques considérées comme des personnes
non diabétiques sur le nombre réel de personnes diabétiques.
sum(hat_y == 0 & pima$diabetes == 1)/sum(pima$diabetes == 1)
```

```
## [1] 0.4179104
```

Remarque : Ces deux indicateurs sont très importants. Tout d'abord la Sensibilité car elle permet de déterminer les personnes réellement diabétiques. Ensuite pour ce qui est du taux de faux négatifs, l'objectif serait d'avoir un taux très faible, ce qui permettrait que la majorité des personnes considérées comme diabétiques le soient vraiment. Cependant on peut voir que les deux indicateurs ne sont pas très bons (Sensibility = 58% ce qui est un peu faible et un taux de faux négatifs = 42% ce qui est un peu élevé).

```
# specificity (taux vrais négatifs)
# Ici représente : sur toutes les personnes prédites non diabétiques, combien le sont
-elles vraiment?
sum(hat_y == 0 & pima$diabetes == 0)/sum(pima$diabetes == 0)
```

```
## [1] 0.89
```

```
# taux faux positifs (aussi égale à 1 - specificity)
# Ici représente : le nombre de personnes non diabétiques considérées comme des perso
nnes diabétiques sur le nombre réel de personnes non diabétiques.
sum(hat_y == 1 & pima$diabetes == 0)/sum(pima$diabetes == 0)
```

```
## [1] 0.11
```

Remarque : On remarque que la Spécificité est très bonne (presque 90%), on peut donc dire que notre modèle prédit bien les personnes non diabétiques qui ne le sont réellement pas. Le taux de faux positifs est très faible

(environ 10%), ce qui est important car cela signifie que les personnes non diabétiques son rarement prédites de diabétiques.

Apprentissage

On va maintenant décomposer les données en deux échantillons :

- un **échantillon d'apprentissage**, utilisé pour apprendre le modèle (correspond à **70%** des données);
- un **échantillon de test**, pour tester les performances en prédiction du modèle (correspond à **30%** des données).

```
# taille échantillon
n <- nrow(pima)
# indices des individus dans l'échantillon d'apprentissage
train_index <- sample(x = 1:n, size = round(0.7 * n), replace = FALSE)
# train et test sets
train_data <- pima[train_index,]
test_data <- pima[-train_index,]
```

Données d'apprentissage

Une fois les données séparées, on réalise une régression logistique sur les données d'apprentissage puis on met en place une sélection backward.

```
# training du modèle
log_reg <- glm(diabetes ~ ., data = train_data, family="binomial")
# sélection de modèle
log_reg <- step(log_reg, direction="backward")
```

```
## Start:  AIC=526.76
## diabetes ~ pregnant + glucose + pressure + triceps + insulin +
##     mass + pedigree + age
##
##            Df Deviance    AIC
## - insulin   1   508.76 524.76
## - triceps   1   508.76 524.76
## <none>          508.76 526.76
## - age       1   512.98 528.98
## - pressure  1   513.89 529.89
## - pregnant  1   514.66 530.66
## - pedigree  1   519.34 535.34
## - mass      1   538.09 554.09
## - glucose   1   582.30 598.30
##
## Step:  AIC=524.76
## diabetes ~ pregnant + glucose + pressure + triceps + mass + pedigree +
##     age
##
##            Df Deviance    AIC
## - triceps   1   508.76 522.76
## <none>          508.76 524.76
## - age       1   513.00 527.00
## - pressure  1   513.89 527.89
## - pregnant  1   514.69 528.69
## - pedigree  1   519.42 533.42
## - mass      1   538.09 552.09
## - glucose   1   591.56 605.56
##
## Step:  AIC=522.76
## diabetes ~ pregnant + glucose + pressure + mass + pedigree +
##     age
##
##            Df Deviance    AIC
## <none>          508.76 522.76
## - age       1   513.05 525.05
## - pressure  1   513.96 525.96
## - pregnant  1   514.71 526.71
## - pedigree  1   519.68 531.68
## - mass      1   541.84 553.84
## - glucose   1   591.56 603.56
```

Données de test

Une fois le modèle déterminé, on peut effectuer les prédiction sur l'échantillon de test.

```
# prediction (sur l'échantillon de test)
hat_pi <- predict(log_reg, newdata = test_data, type = "response")
hat_y <- as.integer(hat_pi > 0.5)
```

Matrice de confusion

```
# matrice de confusion avec la fonction `table`
table(hat_y, test_data$diabetes)
```

```
##
## hat_y   0   1
##     0 138  35
##     1  17  40
```

```
# matrice de confusion (et plus) avec la fonction `caret::confusionMatrix`
library(caret)
confusionMatrix(data = as.factor(hat_y),
                reference = as.factor(test_data$diabetes),
                positive = "1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 138  35
##          1  17  40
##
##                Accuracy : 0.7739
##                  95% CI : (0.7143, 0.8263)
##     No Information Rate : 0.6739
##     P-Value [Acc > NIR] : 0.000564
##
##                   Kappa : 0.4516
##
##  Mcnemar's Test P-Value : 0.018400
##
##             Sensitivity : 0.5333
##             Specificity : 0.8903
##          Pos Pred Value : 0.7018
##          Neg Pred Value : 0.7977
##              Prevalence : 0.3261
##          Detection Rate : 0.1739
##    Detection Prevalence : 0.2478
##       Balanced Accuracy : 0.7118
##
##        'Positive' Class : 1
##
```

Remarque : On remarque que la matrice de confusion est assez similaire à celle réalisée auparavant. En effet, on retrouve une bonne Précision (77%), un très bonne Spécificité (presque 90%), mais une Sensibilité faible (seulement 53%).

Sensibilité et spécificité

Définition de 2 functions permettant de calculer la sensibilité et la spécificité en fonction du seuil de décision utiliser pour prédire $\hat{y}$ à l'aide de $\hat{\pi}$, i.e. $\hat{y} = I_{\{\hat{\pi} > c\}}$ pour un seuil $c \in [0; 1]$:

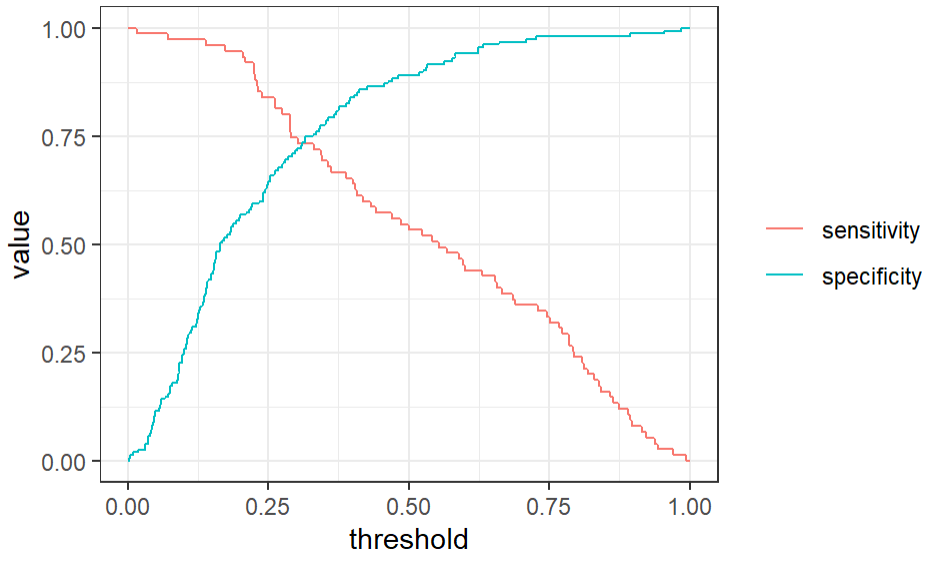
```
# fonction sensiblity en fonction du seuil
sensibility <- function(threshold, hat_pi, df) {
    out <- sum(as.integer(hat_pi > threshold) == 1 &
                   df$diabetes == 1)/sum(df$diabetes == 1)
    return(out)
}
specificity <- function(threshold, hat_pi, df) {
    out <- sum(as.integer(hat_pi > threshold) == 0 &
                   df$diabetes == 0)/sum(df$diabetes == 0)
    return(out)
}
```

Valeurs possibles pour le seuil (comprises entre 0 et 1):

```
threshold <- seq(0, 1, 0.001)
```

Calcul de la sensibilité et de la spécificité en fonction des valeurs possibles pour le seuil:

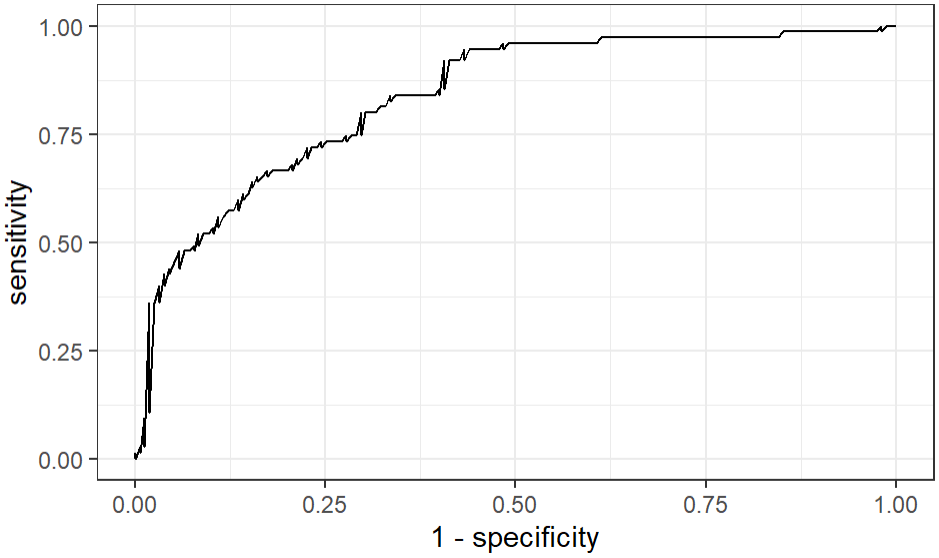
```
library(ggplot2)
sens <- sapply(threshold, sensibility, hat_pi = hat_pi, df = test_data)
spec <- sapply(threshold, specificity, hat_pi = hat_pi, df = test_data)
# sensibility et specificity en fonction du threshold
data2plot <- data.frame(threshold=rep(threshold,2),
                        value=c(sens, spec),
                        tag=rep(c("sensitivity", "specificity"),
                                each = length(threshold)))
ggplot(data2plot, aes(x=threshold, y=value)) +
  geom_line(aes(col=tag)) +
  theme_bw() + theme(legend.title = element_blank())
```



Courbe ROC et AUC

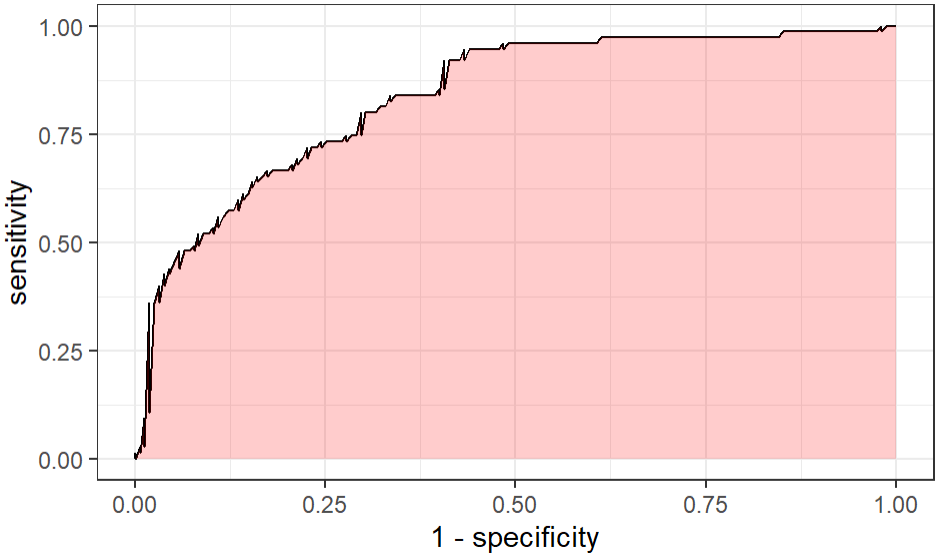
Courbe ROC (`sensibility` en fonction de `1 - specificity`):

```
# courbe roc et auc
data2plot <- data.frame(threshold=threshold,
                        sensitivity=sens,
                        specificity=spec)
ggplot(data2plot, aes(x=1 - specificity, y=sensitivity)) + geom_line() + theme_bw()
```



Rappel : L'AUC est définit comme l'aire sous la courbe ROC. Plus sa valeur est proche de 1, plus le modèle est bien choisit.

```
# auc = area under roc curve
ggplot(data2plot, aes(x=1 - specificity, y=sensitivity)) + geom_line() +
  geom_area(fill="red", alpha=0.2, position = 'identity') + theme_bw()
```



```r
# auc
library(pROC)
auc(test_data$diabetes, hat_pi)
```

```
## Area under the curve: 0.8349
```

Conclusion

On a trouvé un bon modèle pour prédire l'absence/présence du diabète chez les femmes Amérindiennes de la tribu de Pima. En effet, l'accuracy et L'AUC sont tous les deux proches de 1.

Cependant, on aurait préféré avoir plus de données concernant les personnes atteintes du diabéte (`diabetes = 1`) pour que notre modèle puisse mieux prédire les personnes atteintes. Cela aurait aussi amélioré notre échantillon de test pour qu'il soit plus grand et plus diversifié pour pouvoir avoir plus de significativité statistique dans la matrice de confusion et donc dans nos prédictions.