LE LOGIT BINAIRE / COURBE ROC / ETC. AVEC R

Des fonctions de stata dans R

réalisé par AMOUSSOU Kokou

amoussoukokou96@gmail.com

Ingénieur des Travaux Statiqtiques - Elève Ingénieur Statisticiens économiste

2022-06-27 13:23:01

Table des matières

Connaissez-vous le **logit binaire** ? C'est un modèle pour expliquer une variable catégorielle binaire codée 1 pour l'observation d'un phénomène et 0 sinon. On désire connaître la probabilité relative d'appartenance à une catégorie selon des caractéristiques observées. Vous voulez en apprendre ? Veuillez Regarder *Ricco R., Pratique de la Régression Logistique - Régression Logistique Binaire et Polytomique, 2017. https://eric.univ-lyon2.fr/~ricco/cours/cours/pratique_regression_logistique.pdf*. Ce modèle est utilisé dans plusieurs domaines, la médécine par exemple, pour par exemple déterminer à quel point un individu (ayant certaines caractéristiques spécifiques) est susceptible de présenter une maladie par rapport à un autre.

Prenons un exemple : Je vous propose la base `hobbies` du package `FactoMineR` de R. Vous pouvez trouver l'aide sur la base en tapant `help(hobbies)` dans une console R, sachant bien sûr que le package est chargé. Ici, on veut pouvoir expliquer le fait d'avoir pour hobby la lecture `Reading` selon qu'on est femme ou hommes, qu'on est dans une certaine catégorie d'âge,. . .

1 Corrigeons les valeurs manquantes

Si vous avez chargé la base `hobbies`, vous devez vous rendre compte que la variable Profession présente des valeurs manquantes. Cherchons à la corriger :

L'idée est toute simple. Comme, il s'agit d'une variable catégorielle, nous décidons de corriger avec le mode de la distribution. . . mais, rassurez-vous, ce ne sera pas fait de manière "brute". Nous allons nous créer des groupes dans les données dans lesquels nous ferons le travail. Admettons que l'Age et le Sex ont une corrélation avec la variable Profession. Imputons donc la variable `Profession` par le mode selon la catégorie simultanée d'`Age` et `Sex`.

```
# La fonction mode
library(FactoMineR)
data(hobbies)
d = hobbies
```

```r
mod = function(arg){return(names(table(arg)[max(table(arg)) == table(arg)]))}
library(questionr)
avI = freq(d$Profession, total = TRUE)

# Création des groupes
library(dplyr)
tab = d %>% group_by(Age, Sex) %>%
  summarise(
    mode = mod(Profession),
    n = n(),
    n.NA = sum(is.na(Profession)),
    "%" = round(sum(is.na(Profession))*100/n(), 2)
  )

# Imputation par le mode de chaque groupe
library(data.table)
tab = data.table(tab)
for(i in (1:dim(d)[1])){
  if(is.na(d$Profession[i])){
    d$Profession[i] = tab[tab$Age == d$Age[i] & tab$Sex == d$Sex[i]]$mode
  }
}
apI = freq(d$Profession, total = TRUE)

# Affichage
library(ggpubr); library(cowplot)
theme1 = function(taille = 12) ttheme(base_style = "light", base_size = taille)
plot_grid(
  ggtexttable(avI, theme = theme1(8))%>%
    tab_add_title(text = "Avant imputation"),
  ggtexttable(tab, theme = theme1(8), rows = NULL)%>%
    tab_add_title(text = "Les groupes pour l'imputation", size = 8),
  ggtexttable(apI, theme = theme1(8))%>%
    tab_add_title(text = "Après imputation"),
  nrow = 1
)
```

Avant imputation

	n	%	val%
Unskilled worker	792	9.4	11.5
Manual labourer	1161	13.8	16.8
Technician	401	4.8	5.8
Foreman	735	8.7	10.6
Management	1052	12.5	15.2
Employee	2552	30.4	37
Other	212	2.5	3.1
NA	1498	17.8	NA
Total	8403	100	100

Les groupes pour l'imputation

Age	Sex	mode	n	n.NA	%
[15,25]	M	Unskilled worker	401	181	45.14
[15,25]	F	Employee	456	220	48.25
(25,35]	M	Manual labourer	602	58	9.63
(25,35]	F	Employee	700	61	8.71
(35,45]	M	Manual labourer	716	76	10.61
(35,45]	F	Employee	930	115	12.37
(45,55]	M	Manual labourer	833	98	11.76
(45,55]	F	Employee	1004	115	11.45
(55,65]	M	Management	601	89	14.81
(55,65]	F	Employee	656	101	15.40
(65,75]	M	Manual labourer	404	90	22.28
(65,75]	F	Employee	533	134	25.14
(75,85]	M	Manual labourer	197	51	25.89
(75,85]	F	Employee	285	83	29.12
(85,100]	M	Manual labourer	33	11	33.33
(85,100]	F	Employee	52	15	28.85

Après imputation

	n	%	val%
Unskilled worker	973	11.6	11.6
Manual labourer	1545	18.4	18.4
Technician	401	4.8	4.8
Foreman	735	8.7	8.7
Management	1141	13.6	13.6
Employee	3396	40.4	40.4
Other	212	2.5	2.5
Total	8403	100	100

FIGURE 1 – Frequences avant et après imputation

2 Bon, on y va maintenant ! ! !

Le calcul du modèle se fait avec la fonction `glm` sous R.

```
reg = glm(Reading ~ Sex + Age + `Marital status` + Profession + nb.activitees,
          data = d, family = binomial())
```

Je vous présente les résultats de manière plus attrayante :

```
library(forestmodel)
forest_model(model = reg, exponentiate = F,
             format_options = forest_model_format_options(
               colour = "darkblue",
               color = NULL,
               shape = 15,
               text_size = 3,
               point_size = 2,
               banded = TRUE
             )
)
```

Variable		N	Odds ratio		p
Sex	M	3787		Reference	
	F	4616		0.71 (0.57, 0.85)	<0.001
Age	[15,25]	857		Reference	
	(25,35]	1302		0.32 (0.08, 0.56)	0.008
	(35,45]	1646		0.41 (0.16, 0.65)	0.001
	(45,55]	1837		0.61 (0.35, 0.86)	<0.001
	(55,65]	1257		0.68 (0.41, 0.95)	<0.001
	(65,75]	937		1.27 (0.99, 1.56)	<0.001
	(75,85]	482		1.93 (1.60, 2.27)	<0.001
	(85,100]	85		2.20 (1.64, 2.78)	<0.001
Marital status	Single	2140		Reference	
	Married	4333		−0.23 (−0.40, −0.06)	0.008
	Widower	734		−0.40 (−0.67, −0.14)	0.003
	Divorcee	792		−0.06 (−0.30, 0.18)	0.640
	Remarried	404		−0.30 (−0.60, 0.01)	0.053
Profession	Unskilled worker	973		Reference	
	Manual labourer	1545		0.27 (0.08, 0.47)	0.006
	Technician	401		−0.12 (−0.41, 0.16)	0.395
	Foreman	735		0.49 (0.23, 0.75)	<0.001
	Management	1141		0.94 (0.70, 1.19)	<0.001
	Employee	3396		0.30 (0.12, 0.48)	0.001
	Other	212		0.43 (0.05, 0.81)	0.028
nb.activitees		8403		0.47 (0.44, 0.49)	<0.001

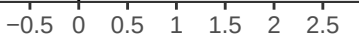

FIGURE 2 – Coefficients du modèle

C'est en réalité les coefficients qui sont présentés ci-dessus. Mais pour le modèle logit, les coefficients ne sont pas directement interprétables, si ce n'est leur signe ou au mieux seulement une comparaison relative à la référence. Pour quantifier la comparaison, on pourrait donc passer par les effets marginaux ou encore les odds ratio (rapport de côte), qui ne sont que l'exponentiel des coefficients. On obtient :

```
library(forestmodel)
forest_model(
  model = reg,
   format_options = forest_model_format_options(
     colour = "darkblue",
     color = NULL,
     shape = 10,
     text_size = 3,
     point_size = 2,
```

```
    banded = TRUE
  )
)
```

Variable		N	Odds ratio	p
Sex	M	3787	Reference	
	F	4616	2.04 (1.77, 2.34)	<0.001
Age	[15,25]	857	Reference	
	(25,35]	1302	1.38 (1.09, 1.74)	0.008
	(35,45]	1646	1.50 (1.18, 1.92)	0.001
	(45,55]	1837	1.83 (1.42, 2.37)	<0.001
	(55,65]	1257	1.97 (1.50, 2.59)	<0.001
	(65,75]	937	3.57 (2.68, 4.76)	<0.001
	(75,85]	482	6.91 (4.94, 9.69)	<0.001
	(85,100]	85	9.07 (5.16, 16.19)	<0.001
Marital status	Single	2140	Reference	
	Married	4333	0.79 (0.67, 0.94)	0.008
	Widower	734	0.67 (0.51, 0.87)	0.003
	Divorcee	792	0.94 (0.74, 1.20)	0.640
	Remarried	404	0.74 (0.55, 1.01)	0.053
Profession	Unskilled worker	973	Reference	
	Manual labourer	1545	1.31 (1.08, 1.60)	0.006
	Technician	401	0.88 (0.66, 1.18)	0.395
	Foreman	735	1.63 (1.26, 2.11)	<0.001
	Management	1141	2.57 (2.01, 3.29)	<0.001
	Employee	3396	1.35 (1.12, 1.61)	0.001
	Other	212	1.53 (1.05, 2.26)	0.028
nb.activitees		8403	1.60 (1.56, 1.64)	<0.001

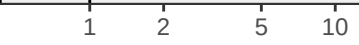

FIGURE 3 – Odds ratio

On dira par exemple que les femmes ont 2 (2.04) fois plus de probabilité que les hommes de faire la lecture. Aussi, plus on possède de hobbies, plus probable serait comprise la lecture. Plus précisément, une unité additionnelle du nombre de hobbies effectués augmente la probabilité de faire la lecture de 1.6 fois. Je vous laisse déduire le reste.

3 Validation et qualité du modèle

En réalité, rigoureusement, on n'avait pas le droit à ce niveau de dire quelque interprétation que ce soit. Il faut prouver que le modèle est de bonne qualité. Vous pouvez mieux vous renseigner sur les tests de validation du modèle binaire avec la référence citée plus haut ou encore sur le net.

3.1 Les courbes de ROC, de sensibilité/spécificité

"La courbe ROC met en relation le taux de vrais positifs TVP (la sensibilité, le rappel) et le taux de faux positifs TFP (TFP = 1 - Spécificité) dans un graphique nuage de points" (Ricco, R. (2017)). Construisons cette courbe sur R. Il y a plusieurs références qui permettent de construire la courbe de ROC sous R. Mais, en explorant certains, je n'ai pas été tout à fait satisfait, si je me réfère à la manière de faire sous stata. Mais, le package `blorr` en propose quand même d'intéressant. Ce package m'a aussi inspiré. La construction de la courbe ROC nécessite la précision d'un seuil (cutoff). Le package `blorr` prend par défaut 0.5 ou alors laisse le choix à l'utilisateur de la mentionner lui-même. Alors qu'ici, nous cherchons à approcher le point d'intersection entre la courbe de sensibilité et de spécificité pour la construction de la courbe ROC par défaut. En spécifiant la bonne valeur du cutoff, on obtient plus d'exactitude quant aux différents résultats.

Ici, je vous propose quelques fonctions qui permettront d'arriver à la courbe de ROC.

— Une fonction `data_sensp` qui recupère le modèle et un nombre de seuils fixé pour retourner une base de données contenant 4 variables : une série (le nombre doit être spécifié en argument) de valeurs de seuils (cutoff) entre 0 et 1, une série des valeurs de sensibilité, une série des valeurs de spécificité et une dernière qui donne l'opposé des spécificités pour chaque seuil.

— Une fonction `cutoff` qui retourne la valeur approximative de l'intersection de la courbe de sensibilité et de spécificité.

— Une fonction `lsens` (comme on le nommerait en stata), qui retourne le graphe de la courbe de spécificité et de spécificité pour chaque cutoff.

— Une fonction `lroc` qui retourne la courbe de ROC du modèle en fonction des cutoff calculés à partir des seuils. Elle retourne aussi l'AUC (Area Under the Curve) à l'aide de la méthode des trapèzes (j'ai trouvé la procédure sur une page web dont je me rappelle plus exactement).

```
# Calculons d'abord les données pour le tracé
library(labelled)
library(ggplot2)
data_sensp = function(modele, n_seuil = 10){
  S = predict(modele, type = "response")
  lsensp1 = lsensp2 = lsensp3 = rep(NA, n_seuil)
  for (s in seq(0, 1, length.out = n_seuil)) {
    ps = (S > s) * 1
    lsensp1[round(1+s*(n_seuil-1))] = s # Probability cutoff
    # sensitivity
    lsensp2[round(1+s*(n_seuil-1))] = sum((ps == 1)*(modele$y == 1))/sum(modele$y == 1)
    # specificity
    lsensp3[round(1+s*(n_seuil-1))] = 1-sum((ps == 1)*(modele$y == 0))/sum(modele$y == 0)
  }
  dsensp = data.frame(lsensp1,lsensp2,lsensp3)
  var_label(dsensp$lsensp1) = "Probability cutoff"
  var_label(dsensp$lsensp2) = "sensitivity"
  var_label(dsensp$lsensp3) = "specificity"
  dsensp$lsensp4 = 1-dsensp$lsensp3
  var_label(dsensp$lsensp4) = "1-specificity"

  return(dsensp)
}

# Le cutoff
cutoff = function(modele, n_seuil = 10){
  dsensp = data_sensp(modele, n_seuil)
  dsensp$find = dsensp$lsensp2-dsensp$lsensp3
  i = 1
  s = seq(1, 0, by = -1/n_seuil)[i]
  cutoff = dsensp$lsensp1[abs(dsensp$find) < s]

  while(mean(cutoff)!=cutoff[1] & i <= n_seuil){
    # La condition mean(cutoff)!=cutoff[1] vérifie s'il y a plusieurs mêmes valeurs
    # Si oui, il suffit de prendre une ou la moyenne
    i = i + 1
    s = seq(1, 0, by = -1/n_seuil)[i]
    cutoff = dsensp$lsensp1[abs(dsensp$find) < s]
    # Et si le vecteur cutoff est vide d'un coup ? on prend le vecteur précédent
    if (length(cutoff)==0){
      i = i-1
      s = seq(1, 0, by = -1/n_seuil)[i]
      cutoff = dsensp$lsensp1[abs(dsensp$find) < s]
      break
    }
  }
  return(mean(cutoff))
}

# La courbe de sensibilté / spécificité
lsens = function(modele, n_seuil = 10, titre = "Courbes Sensitivity/Specificity"){
  dsensp = data_sensp(modele, n_seuil)
  ggplot(dsensp, aes(x=lsensp1,y=lsensp2))+geom_line(col = "darkblue")+
    geom_line(y=dsensp$lsensp3,col='darkred')+
    labs(x = "Probability cutoff", y = "Sensitivity/Specificity")+
    ggtitle(titre)
}
```

```r
# La courbe de roc
lroc = function(modele, n_seuil = 10, titre = "Courbe de ROC"){
  dsensp = data_sensp(modele, n_seuil)
  # calcul de l'aire sous la courbe
  height = (dsensp$lsensp2[-1]+dsensp$lsensp2[-length(dsensp$lsensp2)])/2
  width = -diff(dsensp$lsensp4)
  auc = sum(height*width)
  #la courbe
  message("Aire sous la courbe : ",auc)
  ggplot(dsensp, aes(x=lsensp4, y=lsensp2))+
    geom_line(col = 'darkblue') + geom_point(col = 'darkblue')+
    geom_line(y=sort(dsensp$lsensp4), col='darkred')+
    labs(x = "1-specificity", y = "Sensitivity")+
    ggtitle(paste0(titre, " (AUC = ", round(auc, 4),")"))
}
```

Pour ce modèle, on obtient un cutoff de 0.646 (`cutoff(reg, 501)`). J'ai pris 501 seuils pour plus d'exactitude. Avec moins de seuils, on a : 0.65 (`cutoff(reg, 11)`).

Les courbes :

```r
cowplot::plot_grid(lroc(reg, 11)+theme_light(), lsens(reg, 11)+theme_light())
```

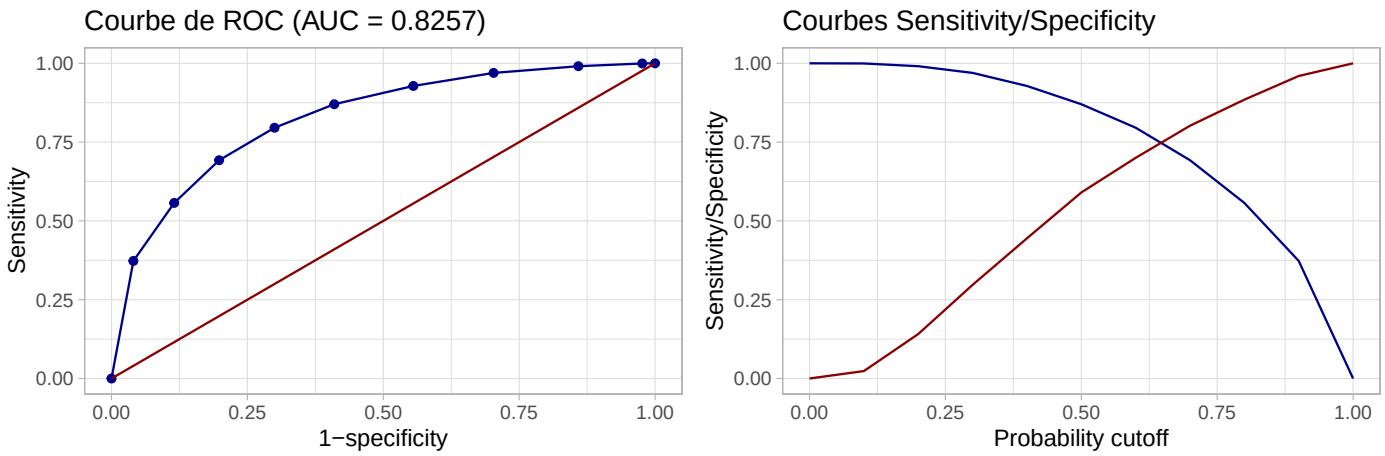

FIGURE 4 – ROC curve , Sensitivity/Specificity curve

3.2 estat_class

Si vous êtes utilisateurs de Stata, et que vous y avez déjà fait le modèle logit, vous connaissez probablement la fonction estat_class. Bon essayons sur R.

```r
estat_class = function(modele, n_seuil = 10, Vcutoff = cutoff(modele, n_seuil), newdata = NULL, stat_digit
                       theme_ = ttheme(base_style = "light", base_size = 9)){
  tab = table(predict(modele, newdata = newdata, type = "response") >= Vcutoff, modele$y)

  Sensitivity = round(tab[2,2]/(tab[2,2] + tab[1,2]), stat_digit) # P(y_chap = 1 | y = 1) ou VP/(VP + FN) o
  Specificity = round(tab[1,1]/(tab[2,1] + tab[1,1]), stat_digit) # P(y_chap = 0 | y = 0) ou VN/(VN + FP) o
  Pos_Pred_Value = round(tab[2,2]/(tab[2,1] + tab[2,2]), stat_digit) # Pr( D| +)
  Neg_Pred_Value = round(tab[1,1]/(tab[1,1] + tab[1,2]), stat_digit) # Pr(~D| -)
  faux_pos_tND = round(tab[2,1]/(tab[1,1] + tab[2,1]), stat_digit) # Pr(+|~D)
  faux_neg_tD = round(tab[1,2]/(tab[1,2] + tab[2,2]), stat_digit) # Pr(-| D)
  faux_pos_cP = round(tab[2,1]/(tab[2,1] + tab[2,2]), stat_digit) # Pr(~D| +)
  faux_neg_cN = round(tab[1,2]/(tab[1,1] + tab[1,2]), stat_digit) # Pr( D| -)
  Prevalence = round(length(modele$y[modele$y==1])/length(modele$y), stat_digit)
  Correctly_classified = round(Sensitivity*Prevalence + Specificity*(1-Prevalence), stat_digit)
  Accuracy = Correctly_classified # Accuracy
  Precision = Pos_Pred_Value
  Recall = Sensitivity

  Positive_Class = 1
```

```
  dfr = data.frame(
    #"Accuracy" = format(Accuracy, nsmall = stat_digit),
    "Sensibilite Pr( +| D)" = format(Sensitivity, nsmall = stat_digit),
    "Specificite Pr( -|~D)" = format(Specificity, nsmall = stat_digit),
    "Positive Predictive Value Pr( D| +)" = format(Pos_Pred_Value, nsmall = stat_digit),
    "Negative Predictive Value Pr(~D| -)" = format(Neg_Pred_Value, nsmall = stat_digit),
    "False + rate for true ~D Pr(+|~D)" = format(faux_pos_tND, nsmall = stat_digit),
    "False - rate for true D Pr(-| D)" = format(faux_neg_tD, nsmall = stat_digit),
    "False + rate for classified + Pr(~D| +)" = format(faux_pos_cP, nsmall = stat_digit),
    "False - rate for classified - Pr( D| -)" = format(faux_neg_cN, nsmall = stat_digit),
    "Correctly classified" = format(Correctly_classified, nsmall = stat_digit),
    "Prevalence" = format(Prevalence, nsmall = stat_digit)
    #,"Precision" = format(Precision, nsmall = stat_digit),
    #"Recall (rappel)" = format(Recall, nsmall = stat_digit),
    #"'Positive' Class" = format(Positive_Class, width = stat_digit + 2)
  )

  names(dfr) = c(
    #"Accuracy",
    "Sensibilite Pr( +| D)", "Specificite Pr( -|~D)",
    "Positive Predictive Value Pr( D| +)", "Negative Predictive Value Pr(~D| -)",
    "False + rate for true ~D Pr(+|~D)", "False - rate for true D Pr(-| D)",
    "False + rate for classified + Pr(~D| +)", "False - rate for classified - Pr( D| -)",
    "Correctly classified", "Prevalence"
    #, "Precision", "Recall (rappel)", "'Positive' Class"
  )

  dfr = data.frame(t(dfr))

  names(dfr) = "Valeur"

  plot_grid(
    ggtexttable(tab, theme = theme_)%>%
      tab_add_title(text = "Matrice de confusion", size = 8)+theme_bw(),
    ggtexttable(dfr, theme = theme_)%>%
      tab_add_title(text = paste0("Classified + if predicted Pr(D) >=",
                                  format(round(Vcutoff, stat_digit), nsmall = stat_digit),
                                  " (Cutoff)"), size = 8)+theme_bw(),
    nrow = 1, rel_widths = c(1,2)
  )
}
```

On obtient :

```
estat_class(reg, n_seuil = 10, theme_ = ttheme(base_style = "light", base_size = 9))
```

Matrice de confusion

	0	1
FALSE	2120	1561
TRUE	637	4085

Classified + if predicted Pr(D) >=0.6667 (Cutoff)

	Valeur
Sensibilite Pr(+| D)	0.7235
Specificite Pr(−|~D)	0.7690
Positive Predictive Value Pr(D| +)	0.8651
Negative Predictive Value Pr(~D| −)	0.5759
False + rate for true ~D Pr(+|~D)	0.2310
False − rate for true D Pr(−| D)	0.2765
False + rate for classified + Pr(~D| +)	0.1349
False − rate for classified − Pr(D| −)	0.4241
Correctly classified	0.7384
Prevalence	0.6719

3.3 Autres indicateurs importants

Avec le package blorr, on peut obtenir d'autres infos importantes sur le modèle.

```
blorr::blr_model_fit_stats(reg)
```

```
##                              Model Fit Statistics
## -------------------------------------------------------------------------------
## Log-Lik Intercept Only:     -5317.610    Log-Lik Full Model:            -3917.228
## Deviance(8383):              7834.456    LR(19):                         2800.765
##                                          Prob > LR:                         0.000
## MCFadden's R2                   0.263    McFadden's Adj R2:                 0.260
## ML (Cox-Snell) R2:              0.283    Cragg-Uhler(Nagelkerke) R2:        0.395
## McKelvey & Zavoina's R2:        0.458    Efron's R2:                        0.306
## Count R2:                       0.778    Adj Count R2:                      0.324
## BIC:                         8015.183    AIC:                            7874.456
## -------------------------------------------------------------------------------
```

3.4 Comparons nos résultats avec ceux du package blorr

La courbe ROC :

```
library(blorr)
library(ggpubr)
plot_grid(
  lroc(reg, n_seuil = 10)+theme_light(),
  blr_roc_curve(blr_gains_table(reg),print_plot = F) + theme_light(),
  nrow = 1
)
```

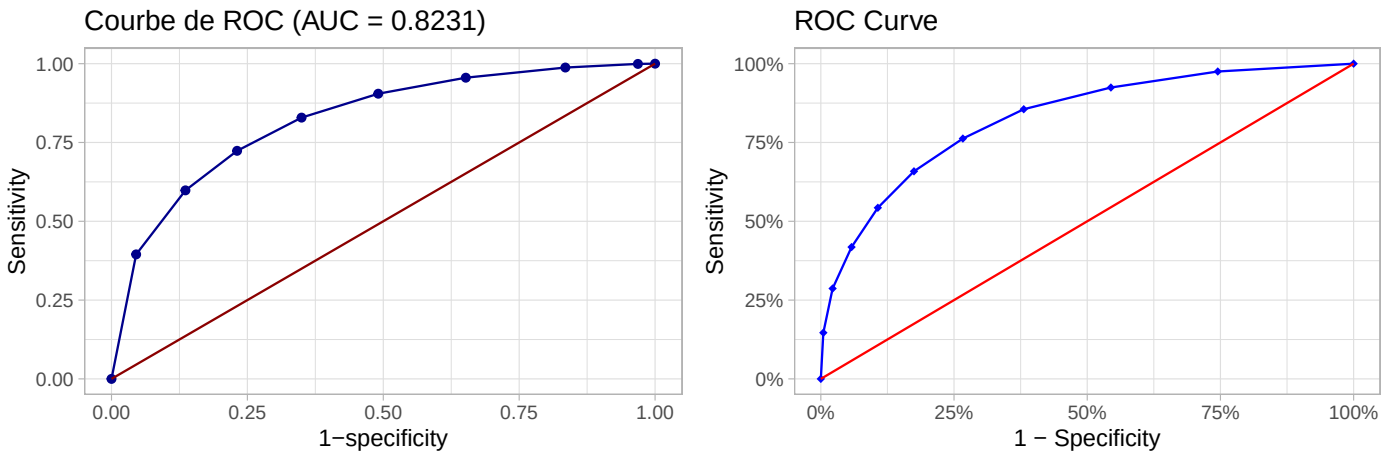

FIGURE 5 – Courbe de ROC (comparaison avec blorr)

En réalité, ce sont deux méthodes de calcul différentes mais qui convergent pratiquement vers la même chose.

La matrice de confusion et ses indicateurs :

```
print("Notre fonction")
```

```
## [1] "Notre fonction"
```

```
estat_class(reg, n_seuil = 10, theme_ = ttheme(base_style = "light", base_size = 9))
```

Matrice de confusion

	0	1
FALSE	2120	1561
TRUE	637	4085

Classified + if predicted Pr(D) >=0.6667 (Cutoff)

	Valeur
Sensibilite Pr(+| D)	0.7235
Specificite Pr(−|~D)	0.7690
Positive Predictive Value Pr(D| +)	0.8651
Negative Predictive Value Pr(~D| −)	0.5759
False + rate for true ~D Pr(+|~D)	0.2310
False − rate for true D Pr(−| D)	0.2765
False + rate for classified + Pr(~D| +)	0.1349
False − rate for classified − Pr(D| −)	0.4241
Correctly classified	0.7384
Prevalence	0.6719

```
print("le package blorr")
```

```
## [1] "le package blorr"
```

```
blr_confusion_matrix(reg)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction    0    1
##          0 1627  733
##          1 1130 4913
##
##
##                  Accuracy : 0.7783
##       No Information Rate : 0.3281
##
```

```
##                   Kappa : 0.4779
##
## McNemars's Test P-Value  : 0.0000
##
##              Sensitivity : 0.8702
##              Specificity : 0.5901
##           Pos Pred Value : 0.8130
##           Neg Pred Value : 0.6894
##               Prevalence : 0.6719
##           Detection Rate : 0.5847
##     Detection Prevalence : 0.7191
##        Balanced Accuracy : 0.7302
##                Precision : 0.8130
##                   Recall : 0.8702
##
##         'Positive' Class : 1
```

Les résultats sont différents, puisque les valeurs de cutoff diffèrent, 0.5 pour `blorr` et 0.6666667 (la valeur à l'intersection) pour notre fonction. Reprenons en imposant une valeur de 0.5 pour les deux.

```
print("Notre fonction")
```

```
## [1] "Notre fonction"
```

```
estat_class(reg, n_seuil = 10, theme_ = ttheme(base_style = "light", base_size = 9), Vcutoff = 0.5)
```

Matrice de confusion

	0	1
FALSE	1627	733
TRUE	1130	4913

Classified + if predicted Pr(D) >=0.5000 (Cutoff)

	Valeur
Sensibilite Pr(+| D)	0.8702
Specificite Pr(−|~D)	0.5901
Positive Predictive Value Pr(D| +)	0.8130
Negative Predictive Value Pr(~D| −)	0.6894
False + rate for true ~D Pr(+|~D)	0.4099
False − rate for true D Pr(−| D)	0.1298
False + rate for classified + Pr(~D| +)	0.1870
False − rate for classified − Pr(D| −)	0.3106
Correctly classified	0.7783
Prevalence	0.6719

```
print("Le package blorr")
```

```
## [1] "Le package blorr"
```

```
blr_confusion_matrix(reg, cutoff = 0.5)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction    0    1
##          0 1627  733
##          1 1130 4913
##
##
##                  Accuracy : 0.7783
##       No Information Rate : 0.3281
##
##                    Kappa : 0.4779
##
```

```
## McNemars's Test P-Value  : 0.0000
##
##              Sensitivity : 0.8702
##              Specificity : 0.5901
##           Pos Pred Value : 0.8130
##           Neg Pred Value : 0.6894
##               Prevalence : 0.6719
##           Detection Rate : 0.5847
##     Detection Prevalence : 0.7191
##        Balanced Accuracy : 0.7302
##                 Precision : 0.8130
##                    Recall : 0.8702
##
##         'Positive' Class : 1
```

Et les résultats sont les mêmes!!!

4 Quelques références

[1] Ricco, R.(2017) Pratique de la Régression Logistique - Régression Logistique Binaire et Polytomique. https://eric.univ-lyon2.fr/~ricco/cours/cours/pratique_regression_logistique.pdf

[2] https://en.wikipedia.org/wiki/Binary_classification

[3] Autres...