METHODE DES TRAPEZES

pour le calcul de láire sous la courbe ROC

présenté par AMOUSSOU KOKOU

amoussoukokou96@gmail.com

Ingénieur des Travaux Statistiques - Elève Ingénieur Statisticien économiste

2022-06-27 13:31:34

Table des matières

J'avais évoqué dans un *précédent post* la méthode des trapèzes pour calculer l'aire sous la courbe ROC. Je n'ai toujours pas trouvé ma source où j'avais trouvé les codes mais je vous l'explique dans les lignes à suivre. Vous y trouverez également un petit aperçu de la force de R en matière de graphique.

1 Méthode des trapèzes. . .

Considérons une courbe de cette forme :

```
library(ggpubr)
x = seq(0, 1, by = 0.01)
fun = function(x) 5 + log(x + 0.01)
f = ggplot(data.frame(x = x, y = fun(x)), aes(x = x, y = y)) +
    geom_line() + theme_light()
f
```

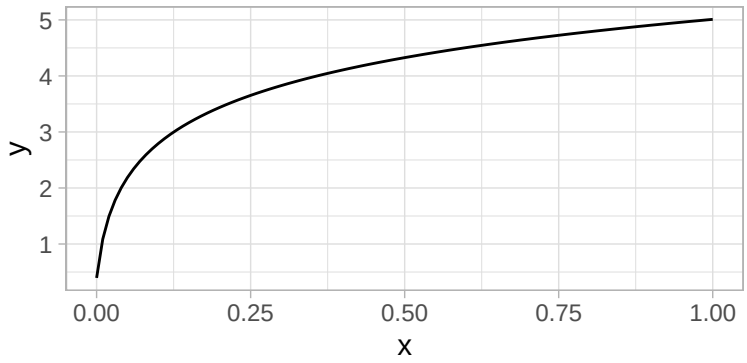

FIGURE 1 – Allure de la fonction $f(x) = 5 + log(x + 0.01)$

C'est la fonction $f(x) = 5 + log(x + 0.01)$ qu'on définit sur $[0, 1]$. J'ai ajouté 0.01 à l'argument du log pour éviter les erreurs liées à la non définition du log en 0. Le problème : **quelle est l'aire sous la courbe ?**

On a bien le droit de faire une intégration : $\int_0^1 5 + \log(x + 0.01) = [5x + (x + 0.01)\log(x + 0.01) - x]_0^1 \sim 4.0561015$. Mais, nous voulons approcher cette valeur par la méthode des trapèzes. D'ailleurs, c'est facile ici puisque la courbe est construite à partir d'une fonction ayant une forme mathématique bien connue. Mais, il peut exister des cas où il est moins aisé de

formaliser la courbe. On dispose que d'un ensemble de couples de points pour la construire. C'est justement le cas pour la courbe de ROC.

L'idée est de subdiviser la surface en de trapèzes dont on sait bien calculer l'aire ($\frac{(Grande\ Base+petite\ base)*Hauteur}{2}$). Ensuite, il suffit de sommer toutes les aires. Notons que plus la subdivisions est fines (donc les trapèzes sont petits), mieux on approche la valeur.

```
p = seq(0, 1, by = 0.25/2) # représente la subdivision considérée pour l'axe des x
pt = fun(p) # Les images de ces subdivisions
# Gymnastique pour construire les trapèzes
i = c(1,2,2,1)
j = c(2, 1)
l = fun(0) # les y minimum
dpoly = data.frame(id = rep(1:8, each = 4),
                   x = c(p[i], p[i+1], p[i+2], p[i+3], p[i+4], p[i+5], p[i+6], p[i+7]),
                   y = c(c(l, l, pt[j]), c(l, l, pt[j+1]), c(l, l, pt[j+2]),
                         c(l, l, pt[j+3]), c(l, l, pt[j+4]), c(l, l, pt[j+5]),
                         c(l, l, pt[j+6]), c(l, l, pt[j+7])
                         )
                   )
# Ajout des trapèzes au graph
f + annotate("polygon", x = dpoly$x, group=dpoly$id,y = dpoly$y, fill=dpoly$id, alpha=0.2)
```

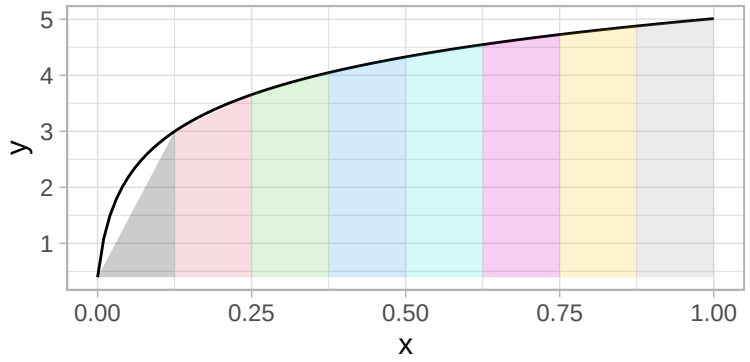

FIGURE 2 – Subdivision de la surface

Il est clair qu'on perd quelques surfaces, mais ça reste une valeur approchée. Dans le premier trapèze (qui est en réalité un triangle), la perte est grande. Normalement, la réduire serait encore mieux. On aura donc pour ce trapèze $\mathcal{A}_1 \sim (2.92 + 0) * 0.125/2 = 0.187345$.

Alors, La somme des aires de toutes ces trapèzes est calculée de la manière suivante :

- **p** est la subdividion générée de l'axe des abscisses,
- **pt** est l'image de la subdividion générée de l'axe des abscisses,
- **height** représente la moitié de la somme des bases. Pour chaque point du **pt** (sauf le premier), le point représente la grande base et son précédent représente la petite base.
- **width** est la largeur (qui représente la hauteur dans la formule) du trapèze. La fonction **diff** fait une différence première d'une série donnée, donc calcule les écarts (qui valent ici tous 0.125).

```
height = (pt[-1] + pt[-length(pt)])/2 # (GB+pb)/2
width = diff(p)  # h
sum(height*width)
```

```
## [1] 3.984281
```

Le résultat est la valeur approchée de notre problème en considérant une subdivision d'écart 0.125. Plus précisément, si on fait une subdivisions d'écarts 0.01, on obtient la valeur suivante qui approche mieux la valeur réelle.

```
x = seq(0, 1, by = 0.01)
y = fun(x)
height = (y[-1]+y[-length(y)])/2
width = diff(x)
sum(height*width)
```

```
## [1] 4.055299
```

2 Application à la courbe ROC

On calcule le fonction `data_sensp` (voir *le post précédent*) et ensuite la courbe ROC.

```
# 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)
}

# 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),")"))
}
```

Prenons l'exemple précédemment utilisé

```
# La fonction mode
library(FactoMineR)
data(hobbies)
d = hobbies
mod = function(arg){return(names(table(arg)[max(table(arg)) == table(arg)]))}

# 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)
```

```r
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
  }
}
```

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

On obtient donc :

```r
library(cowplot)
forme = function(x, dig = 4) {format(round(x, dig),
                                     scientific = F,
                                     nsmall = dig)}
df = forme(data_sensp(reg, n_seuil = 11))
names(df) = c("Probability cutoff", "sensitivity", "specificity", "1-specificity")
plot_grid(
  ggtexttable(df, theme = ttheme("light", 7)),
  lroc(reg, 11)+theme_light(), nrow = 1
)
```

	Probability cutoff	sensitivity	specificity	1−specificity
1	0.0000	1.0000	0.0000	1.0000
2	0.1000	0.9995	0.0236	0.9764
3	0.2000	0.9908	0.1411	0.8589
4	0.3000	0.9695	0.2971	0.7029
5	0.4000	0.9281	0.4447	0.5553
6	0.5000	0.8702	0.5901	0.4099
7	0.6000	0.7956	0.7000	0.3000
8	0.7000	0.6925	0.8020	0.1980
9	0.8000	0.5570	0.8847	0.1153
10	0.9000	0.3730	0.9597	0.0403
11	1.0000	0.0000	1.0000	0.0000

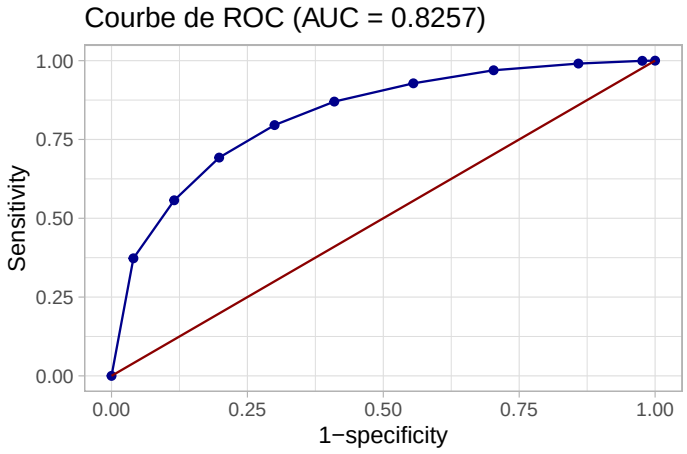

FIGURE 3 – Données | ROC curve and AUC

Et c'est propre!!!