Classification Supervisée

Julien JACQUES

26/09/2018

Les data

En classification supervisée, on cherche à **expliquer et prédire une variable catégorielle** à C modalités, que l'on notera par simplicité $1, \ldots, C$ (mais qui n'ont rien de quantitatif !) correspondant donc à C groupes/classes :

$$Y = (y_1, \ldots, y_n)^t \qquad \text{où } y_i \in \{1, \ldots, C\}$$

à partir de p variables explicatives, que l'on supposera quantitatives :

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix}$$

Pour cela, on dispose d'un échantillon d'apprentissage sur lequel Y est observé : (X, Y)

Il faut alors construire une **règle de classement** r qui permette de prédire la classe de toute nouvelle observation x^* :

$$r : x^* \in \mathbb{R}^p \longrightarrow y^* \in \{1, \ldots, C\}$$

Les méthodes

Il existe de nombreuses méthodes de classification supervisée :

- k plus proches voisins
- analyse (factorielle) discriminante
- régression logistique
- arbres de décision
- forêts aléatoires (random forest)
- réseaux de neuronnes
- SVM
- PLS-DA
- ...

Nous verrons ici la méthode des k **plus proches voisins**, les arbres de décision et les forêts aléatoires

Evaluation de la qualité de prédiction

Pour comparer des méthodes ou choisir des hyper-paramètres, nous devrons être capable d'évaluer la qualité de la régle de classement.

- **piège à éviter** : ne jamais évaluer une méthode sur les données qui ont servies à estimer ses paramètres. Sinon, cela favorisera nécessairement les méthodes les plus complexes, qui pourront alors faire du **sur-apprentissage**.
- il faut donc évaluer les méthodes sur un **échantillon de données indépendantes**. Pour cela plusieurs alternatives

Evaluation de la qualité de prédiction

Pour évaluer une règle de classement nous pouvons :

- séparer l'échantillon total en une partie **apprentissage** pour estimer le modèle et une partie **test** pour l'évaluer (typiquement 2/3 - 1/3)
- lorsqu'on ne dispose pas d'assez de données pour faire cela, on peut réaliser de la **validation croisée**

Validation croisée Leave One Out :

- chaque donnée est tour à tour extraite de l'échantillon d'apprentissage et prédite par le modèle estimé sans cette donnée,
- on évalue alors la qualité de la prédiction,
- et on boucle sur l'ensemble des données de sorte que chacune ait servi une fois de test. L'indicateur final peut-être alors le nombre moyen de bon classements :

$$CV = \frac{1}{n} \sum_{i=1}^{n} 1I_{\hat{y}_{(i)}=y_i}$$

où $\hat{y}_{(i)}$ est la prédiction de la classe du $i$ème individu obtenue sans utiliser cet individu pour estimer les paramètres du modèle.

Validation croisée K-fold :

- l'échantillon total est découpé en K parties,
- chaque partie sert tour à tour de test, le modèle étant estimé sur les $K-1$ autres parties,
- à chaque étape on compte le nombre de bonnes classification,
- le nombre moyen de bon classements est finalement obtenu en sommant l'ensemble des bonnes classifications divisé par la taille d'échantillon.

Classification binaire

Notations :

- $y_i \in \{0, 1\}$,
  - les $y_i = 1$ sont dits *positifs*,
  - les $y_i = 0$ sont dits *négatifs*
- $\hat{y}_i = 1$ alors que $y_i = 1$: vrai positif
- $\hat{y}_i = 1$ alors que $y_i = 0$: faux positif
- $\hat{y}_i = 0$ alors que $y_i = 0$: vrai négatif
- $\hat{y}_i = 0$ alors que $y_i = 1$: faux négatif

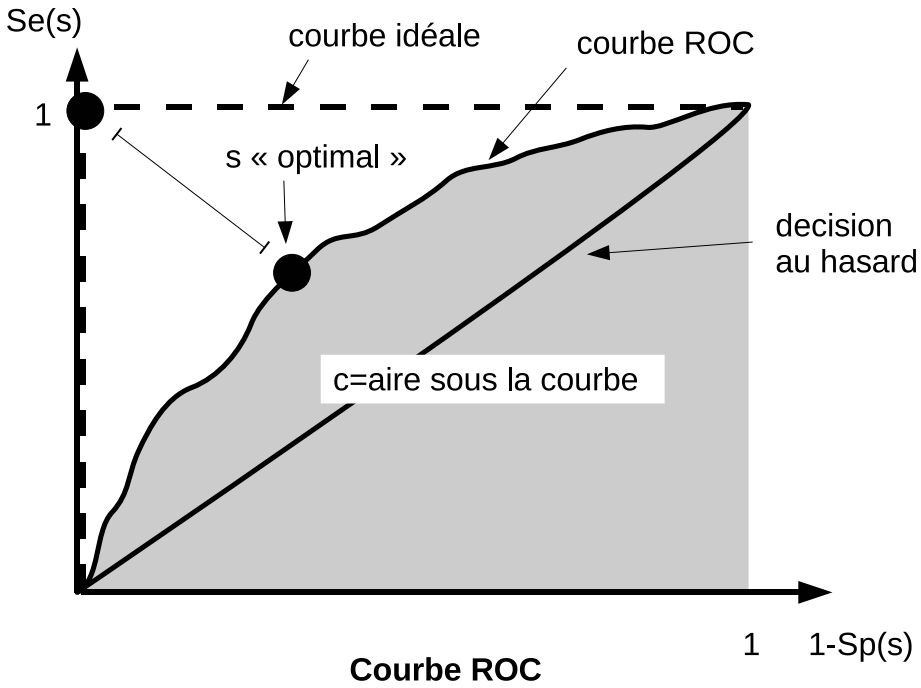
Indicateurs :

- **précision** : nb de vrais positifs / nb de prédictions positives
- **rappel, sensibilité** : nb de vrais positifs / nb de positives
- **spécificité** : nb de vrais négatifs / nb de négatifs
- **F-mesure**

$$F_1 = 2\frac{\text{precision} \times \text{rappel}}{\text{precision} + \text{rappel}}$$

Classification binaire

La courbe ROC représente la “sensibilité” (Se, taux de vrais positifs) en fonction de “1-spécificités” (1-Sp, taux de faux positifs) :



une courbe représente pour un modèle l'évolution de (1-Sp,Se) quand on change le seuil s de probabilité au dessus duquel on décide de prédire $\hat{y} = 1$.

Un indicateur synthétique souvent utilisé est l'**aire sous la courbe ROC** (AUC) qui doit être le plus proche possible de 1

Classification binaire

Sous R, le package **ROCR** permet de calculer tous ces indicateurs, qui sont résumés et complétés ci-dessous :

		True condition			
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$	$\mathbf{F_1}$ **score** $= 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

Figure 1: source Wikipedia

Cas des hyper-paramètres

Dans le cas des méthodes comportants des hyper-paramètres (k de kNN, nombre d'arbres d'une forêt aléatoire, etc. . .), il faut pour pouvoir se comparer honnétement avec des méthodes ne comportant pas d'hyper-paramètres (régression logistique. . .), choisir ces hyper-paramètres sur l'échantillon d'apprentissage.

On découpe alors l'échantillon initial en 3 :

- apprentissage
- test
- validation

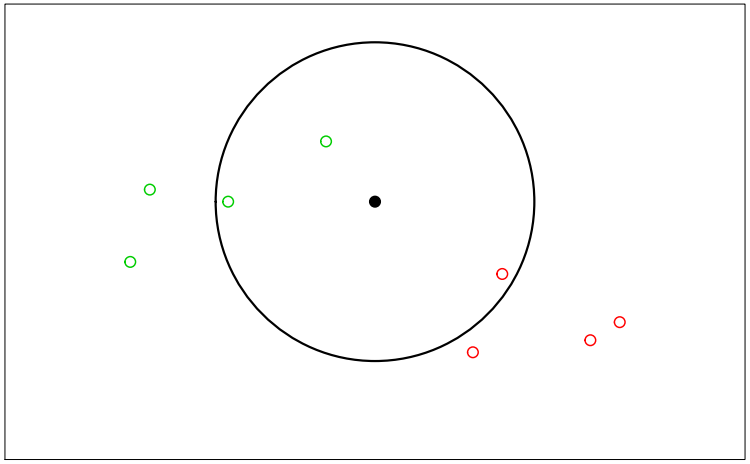
k plus proches voisins

k plus proches voisins

La méthode des k plus proches voisins (kNN pour *k nearest neighbour*) est très simple :

- pour un nouvel x^*, il suffit de rechercher les k voisins les plus proches, au sens d'une certaine distance (euclidienne par exemple)
- on compte le nombre n_c de ces voisins appartenant à la classe c
- on estime alors la probabilité que $y = c$ par :

$$p(Y^* = c) = \frac{n_c}{k}$$



k plus proches voisins

Remarques :

- le choix de k se fera par validation croisée
- plus n est grand, plus on peut se permettre de prendre un k grand
- cette méthode est très simple à mettre en oeuvre mais pas toujours des plus efficaces. En particulier en grande dimension, où l'espace est généralement vide (*les voisins sont tous très éloignés*), cette méthode n'est que peu efficace.

kNN sous R

Chargeons le package class et les données iris, dont on va extraire un échantillon test :

```
library(class)
```

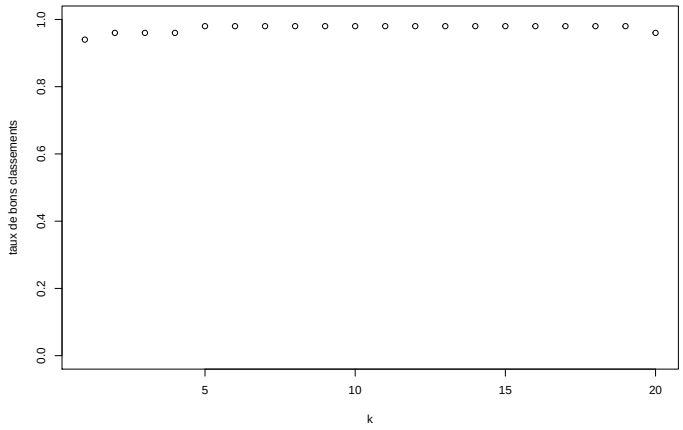
```
## Warning: package 'class' was built under R version 3.5.2
```

```
data(iris)
test_label=sample(1:150,50)
x_app=iris[-test_label,1:4]
x_test=iris[test_label,1:4]
y_app=iris[-test_label,5]
y_test=iris[test_label,5]
```

kNN sous R

Nous allons prédire les labels de l'échantillon test à l'aide de la méthode des kNN pour différente valeur de k (de 1 à 20), et calculer le taux de bons classements

```
tbc=NULL
for (k in 1:20){
  res=knn(x_app,x_test,y_app,k=k)
  tbc[k]=mean(res==y_test)
}
plot(1:20,tbc,ylim=c(0,1),xlab='k',ylab='taux de bons classement
```



Ici la tâche est facile, le classement est très bon quelque soit la valeur de k.