SY09 Printemps 2015
TP 3
Discrimination, théorie bayésienne de la décision

1 Classifieur euclidien, K plus proches voisins

On souhaite étudier les performances du classifieur euclidien et des K plus proches voisins sur différents jeux de données binaires (constitués d'individus de $\mathbb{R}^p$ répartis dans $g = 2$ classes).

1.1 Programmation

Vous mettrez les codes de vos fonctions en annexe de votre rapport.

Classifieur euclidien

Écrire les fonctions ceuc.app et ceuc.val.

La première fait l'apprentissage des paramètres du classifieur euclidien : elle doit donc prendre en argument d'entrée un tableau individus-variables Xapp (de dimensions napp $\times\, p$) correspondant aux individus d'apprentissage, et le vecteur zapp (de longueur napp) des étiquettes associées à chacun des individus. Elle doit retourner les paramètres estimés du classifieur euclidien, sous la forme d'une matrice mu de dimensions $g \times p$.

```
ceuc.app <- function(Xapp, zapp)
{
...
}
```

La seconde fait le classement d'un tableau individus-variables Xtst (de dimensions ntst $\times\, p$) : elle doit donc prendre en argument d'entrée la matrice mu des paramètres estimés et l'ensemble à évaluer Xtst, et retourner un vecteur (de longueur ntst) d'étiquettes prédites.

```
ceuc.val <- function(mu, Xtst)
{
...
}
```

K plus proches voisins

Écrire les fonctions kppv.app et kppv.val.

La première fait l'apprentissage du nombre « optimal » de voisins K (choisi parmi un ensemble prédéfini de valeurs possibles) à utiliser dans l'algorithme, à partir d'un ensemble de validation Xval (dimensions nval $\times\, p$) et des étiquettes zval associées (longueur nval). Elle doit donc prendre en argument d'entrée le tableau individus-variables Xapp, le vecteur zapp des étiquettes associées, le tableau Xval, le vecteur zval, et un ensemble de valeurs nppv. Elle doit retourner la valeur de K choisie (parmi les valeurs contenues dans nppv), c'est-à-dire celle qui donne les meilleurs résultats sur l'ensemble de validation.

```
kppv.app <- function(Xapp, zapp, Xval, zval, nppv)
{
...
}
```

La seconde fait le classement d'un tableau individus-variables Xtst : elle prend donc en argument Xapp et zapp, le nombre de voisins K, et l'ensemble à évaluer Xtst ; elle retourne donc un vecteur (de longueur ntst) d'étiquettes prédites.

```
kppv.val <- function(Xapp, zapp, K, Xtst)
{
...
}
```

Remarque : il sera judicieux de faire appel à la fonction kppv.val dans la fonction kppv.app.

Test des fonctions

Vous pouvez utiliser le jeu de données Synth1-40 pour tester les fonctions que vous développez. Une fois téléchargé, vous pouvez charger le jeu de données, puis séparer les individus et les étiquettes de manière à former un ensemble d'apprentissage et un ensemble de test (on prendra les 15 premiers individus de chaque classe pour l'apprentissage, et les 5 restants pour le test) :

```
donn <- read.table("Synth1-40.txt", header=F)
X <- donn[,1:2]
z <- donn[,3]

Xapp <- X[c(1:15,21:35),]
zapp <- z[c(1:15,21:35)]
Xtst <- X[c(16:20,36:40),]
ztst <- z[c(16:20,36:40)]
```

Pour visualiser les frontières de décision obtenues à l'aide des fonctions ceuc.val et kppv.val que vous avez développées, vous pouvez télécharger les fonctions front.ceuc et front.kppv en ligne sur le site de l'UV [1].

1.2 Évaluation des performances

On dispose de cinq jeux de données (téléchargeables sur le site de l'UV) : Synth1-40, Synth1-100, Synth1-500, Synth1-1000, et Synth2-1000. Pour chacun de ces jeux de données, chaque classe a été générée suivant une loi normale bivariée. Les distributions sont les mêmes pour les jeux de données Synth1-40, Synth1-100, Synth1-500 et Synth1-1000, qui ne diffèrent que par le nombre d'exemples disponibles. En revanche, la distribution du jeu de données Synth2 est différente.

Jeux de données Synth1-40, Synth1-100, Synth1-500 et Synth1-1000

On considère dans un premier temps les jeux de données Synth1-40, Synth1-100, Synth1-500 et Synth1-1000.

1. Pour chacun des jeux de données, estimer les paramètres μ_k et Σ_k des distributions conditionnelles, ainsi que les proportions π_k des classes.
2. Pour chaque jeu de données, on souhaite estimer le taux d'erreur ε du classifieur enclidien. Pour ne pas baser l'analyse sur une unique observation, on propose de répéter N fois l'expérience suivante :
   - on sépare l'ensemble de données disponible aléatoirement de manière à former un ensemble d'apprentissage et un ensemble de test ;
   - on apprend les paramètres du modèle sur l'ensemble d'apprentissage formé, et on calcule le taux d'erreur obtenu sur l'ensemble de test.

   En déterminant de la sorte N séparations aléatoires de l'ensemble de données en un ensemble d'apprentissage et un ensemble de test, on peut ainsi recueillir un échantillon de N estimations $E_1, \ldots, E_N$ du taux d'erreur (généralement effectuées sur l'ensemble de test).

1. Ces fonctions échantillonnent l'espace des caractéristiques en déterminant une grille de points. On peut déterminer les décisions prises pour les points de cette grille, puis tracer les frontières de décision par la fonction contour.

On pourra utiliser la fonction separ1, disponible en ligne sur le site de l'UV, pour séparer aléatoirement l'ensemble de données en un ensemble d'apprentissage et un ensemble de test (de tailles respectives napp $= 2n/3$ et ntst $= n/3$, avec n le nombre total d'exemples) :

```
donn.sep <- separ1(X, z)
Xapp <- donn.sep$Xapp
zapp <- donn.sep$zapp
Xtst <- donn.sep$Xtst
ztst <- donn.sep$ztst
```

(a) On remarquera que le taux d'erreur obtenu sur un ensemble de m points étiquetés $(\boldsymbol{x}_i, z_i), i = 1, \ldots, m$ s'écrit

$$E = \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}_{z_i \neq \widehat{z}_i},$$

où $\widehat{z}_i$ est l'étiquette prédite pour le i$^{\text{e}}$ individu de l'ensemble, et où $\mathbf{1}_{z_i \neq \widehat{z}_i}$ vaut 1 si $z_i \neq \widehat{z}_i$ et 0 sinon. En supposant que m est grand, et que l'ensemble de données est représentatif de la population totale, quelle est la loi approchée de E_j, pour $j = 1, \ldots, N$? Proposer un intervalle de confiance de niveau $1 - \alpha$ sur le taux d'erreur ε, basé sur le taux d'erreur moyen $\overline{E} = \sum_{j=1}^{N} E_j / N$ calculé sur les N expériences (on fera l'hypothèse d'indépendance des taux d'erreur E_j).

(b) Écrire un script qui effectue $N = 20$ séparations aléatoires de chaque jeu de données en ensembles d'apprentissage et de test, et qui calcule (et stocke) pour chacune le taux d'erreur d'apprentissage et le taux d'erreur de test. En considérant ensuite l'ensemble des résultats obtenus lors des $N = 20$ expériences, donner l'estimation ponctuelle du taux d'erreur ε obtenues sur l'ensemble d'apprentissage, l'estimation ponctuelle obtenue sur l'ensemble de test, ainsi que les réalisations de l'intervalle de confiance. Qu'observez-vous ?

3. Effectuer une séparation aléatoire de l'ensemble de données en un ensemble d'apprentissage et un ensemble de test. Déterminer le nombre optimal de voisins à l'aide de la fonction kppv.app, en utilisant l'ensemble d'apprentissage comme ensemble de validation. Quel est le nombre optimal de voisins déterminé ? Pourquoi ?
4. Comme pour le classifieur euclidien, écrire un script qui effectue $N = 20$ séparations aléatoires de chaque jeu de données en ensembles d'apprentissage et de test, et qui pour chacune détermine le nombre optimal de voisins à partir d'un ensemble de validation spécifique, et calcule (et stocke) le taux d'erreur sur l'ensemble d'apprentissage et sur l'ensemble de test.

   Comme précédemment, déterminer les estimations ponctuelles du taux d'erreur et les réalisations des intervalles de confiance obtenues à partir des erreurs d'apprentissage, puis des erreurs de test. Comparer avec les résultats obtenus pour le classifieur euclidien, et commenter.

   Remarque : on pourra utiliser la fonction separ2 pour déterminer aléatoirement les ensembles d'apprentissage, de validation et de test (de tailles respectives napp $= n/2$, nval $= n/4$ et ntst $= n/4$) :

```
donn.sep <- separ2(X, z)
Xapp <- donn.sep$Xapp
zapp <- donn.sep$zapp
Xval <- donn.sep$Xval
zval <- donn.sep$zval
Xtst <- donn.sep$Xtst
ztst <- donn.sep$ztst
```

Jeu de données Synth2-1000

On considère maintenant le jeu de données Synth2-1000.

1. Estimer les paramètres μ_k et Σ_k des distributions conditionnelles, ainsi que les proportions π_k des classes.
2. En utilisant les mêmes procédures que précédemment, calculer les estimations ponctuelles et par intervalles de confiance du taux d'erreur sur l'ensemble d'apprentissage et sur l'ensemble de test. Commenter et interpréter les résultats obtenus.

2 Règle de Bayes

En réalité, les jeux de données étudiés précédemment ont été obtenus de la manière suivante :

1. tout d'abord, l'effectif n_1 de la classe ω_1 a été déterminé par tirage aléatoire suivant une loi binomiale de paramètres n et $\pi_1 = 0.5$;
2. n_1 individus ont ensuite été générés dans la classe ω_1 suivant une loi normale bivariée de paramètres μ_1 et Σ_1, et $n_2 = n - n_1$ individus ont été générés dans la classe ω_2 suivant une loi normale bivariée de paramètres μ_2 et Σ_2.

   Pour les jeux de données Synth1-40, Synth1-100, Synth1-500 et Synth1-1000, on a utilisé comme paramètres $n = 40$, $n = 100$, $n = 500$, et $n = 1000$, respectivement ; et

$$\mu_1 = \begin{pmatrix} -2 \\ 1 \end{pmatrix}, \quad \mu_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \Sigma_1 = \Sigma_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix};$$

   Pour le jeu de données Synth2-1000, on a utilisé $n = 1000$, et

$$\mu_1 = \begin{pmatrix} -4 \\ 1 \end{pmatrix}, \quad \mu_2 = \begin{pmatrix} 4 \\ 1 \end{pmatrix}, \quad \Sigma_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \Sigma_2 = \begin{pmatrix} 5 & 0 \\ 0 & 5 \end{pmatrix}.$$

Pour chacun des jeux de données :

1. quelles sont les distributions marginales des variables X^1 et X^2 dans chaque classe ?
2. Montrer que dans chaque classe, les courbes d'iso-densité sont des cercles dont on précisera les centres et les rayons.
3. Déterminer l'expression de la règle de Bayes pour le problème de discrimination des classes ω_1 et ω_2.
4. Pour les quatre premiers jeux de données, tracer avec R les frontières de décision dans le plan formé par les variables X_1 et X_2.
5. Calculer l'erreur de Bayes pour les quatre premiers jeux de données, et comparer aux résultats obtenus dans l'exercice précédent.