SY09 Printemps 2019
TP 9 — Analyse discriminante

1 Analyse discriminante de données gaussiennes

1.1 Implémentation

Introduction

On souhaite comparer les performances de trois modèles d'analyse discriminante (analyse discriminante quadratique, analyse discriminante linéaire, et classifieur bayésien naïf sous hypothèse de normalité des classes), sur les jeux de données simulées `Synth1-1000`, `Synth2-1000` et `Synth3-1000`. Pour chacun de ces jeux de données, la distribution conditionnelle à chaque classe est gaussienne, les paramètres pouvant en revanche changer d'un jeu de données à l'autre.

On implémentera l'analyse discriminante via quatre fonctions : `adq.app`, `adl.app`, `nba.app` et `ad.val`. Les trois premières font l'apprentissage de chacun des modèles considérés : elles doivent donc prendre en arguments d'entrée le tableau de données `Xapp` et les étiquettes `zapp` associées, et retourner les paramètres du modèle (proportions, moyennes et matrices de covariance). On stockera de manière générique les matrices de covariance dans un tableau à trois dimensions (`array`).

La fonction `ad.val` calcule les probabilités a posteriori des classes pour un ensemble d'individus, ainsi que le classement associé : elle prend donc en compte les paramètres du modèle et l'ensemble de données `Xtst` à classer, et retourne une structure contenant les probabilités `prob` estimées et les classes prédites `pred`. On pourra s'appuyer sur la fonction `mvdnorm`, qui permet de calculer la densité d'une loi normale multivariée pour un tableau de données.

Vérification des fonctions

La Figure 1 montre les courbes de niveau des probabilités a posteriori $\widehat{\Pr}(\omega_k|\boldsymbol{x})$ estimées lorsque la totalité des données `Synth1-40` sont utilisées pour l'apprentissage du modèle. On pourra utiliser la fonction `prob.ad`, disponible sur le site de l'UV, pour afficher les courbes de niveau des probabilités a posteriori $\widehat{\Pr}(\omega_1|\boldsymbol{x})$ estimées par un modèle. On rappelle que la frontière de décision estimée entre les deux classes ω_1 et ω_2 correspond à la courbe de niveau $\widehat{\Pr}(\omega_1|\boldsymbol{x}) = \widehat{\Pr}(\omega_2|\boldsymbol{x})$.

Traitement des données

On utilisera le même protocole expérimental que précédemment (séparation des données en ensembles d'apprentissage et de test, apprentissage, puis classement des données et évaluation des performances), et on le répétera $N = 20$ fois pour chaque jeu de données.

1. Pour chaque jeu de données, calculer le taux d'erreur (de test) moyen sur les $N = 20$ séparations effectuées. On pourra s'appuyer sur les frontières de décision obtenues pour analyser les résultats. Comment peut-on les interpréter ?

   On peut utiliser le code suivant :

   ```
   > rm(list = ls())
   > cat("\014")
   >
   ```

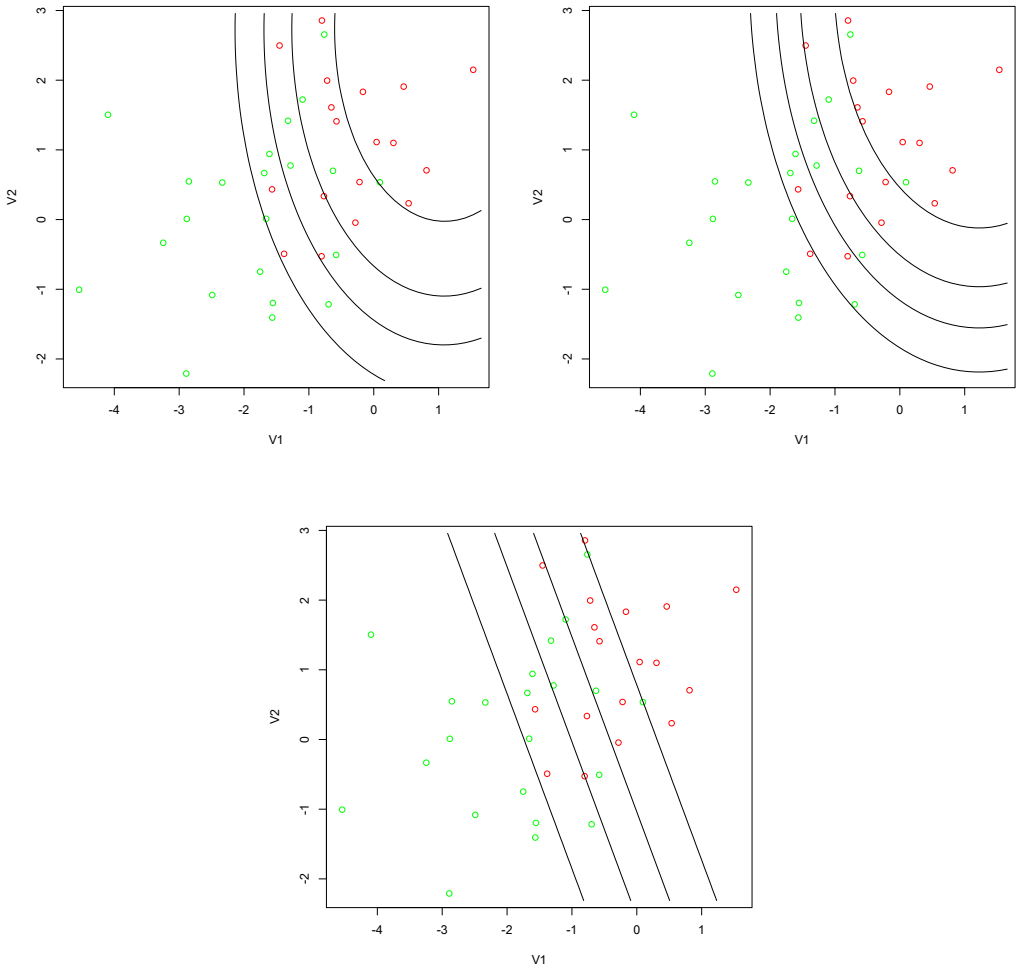

FIGURE 1 – Frontières de décision obtenues avec les données `Synth1-40` (TP6) en utilisant tous les exemples pour l'apprentissage ; haut : ADQ (gauche) et classifieur bayésien naïf (droite), bas : ADL.

```
> library(MASS)
>
> source("fonctions/separ1.R")
> source("fonctions/dmvnorm.R")
> source("fonctions/prob.ad.R")
> source("correction/anadisc.R")
>
> #donn <- read.csv("donnees/Synth1-1000.csv", header=T)
> #donn <- read.csv("donnees/Synth2-1000.csv", header=T)
> donn <- read.csv("donnees/Synth3-1000.csv", header=T)
> X <- donn[,1:2]
> z <- as.factor(donn[,3])
>
> N <- 20
>
> # erreur de test
> E.adq <- rep(0,N)
> E.adl <- rep(0,N)
> E.nba <- rep(0,N)
```

```
>
> for (t in 1:N)
> {
>     # separation de l'ensemble de donnees
>     # en ensemble d'apprentissage et de test
>     donn.sep <- separ1(X, z)
>     Xapp <- donn.sep$Xapp
>     zapp <- donn.sep$zapp
>     Xtst <- donn.sep$Xtst
>     ztst <- donn.sep$ztst
>
>     # apprentissage
>     adq <- adq.app(Xapp, zapp)
>     adl <- adl.app(Xapp, zapp)
>     nba <- nba.app(Xapp, zapp)
>
>     adq.res <- ad.val(adq, Xtst)
>     adl.res <- ad.val(adl, Xtst)
>     nba.res <- ad.val(nba, Xtst)
>
>     # calcul de l'erreur de test
>     E.adq[t] <- mean(adq.res$pred!=ztst)
>     E.adl[t] <- mean(adl.res$pred!=ztst)
>     E.nba[t] <- mean(nba.res$pred!=ztst)
>
> }
>
> cat("Erreur ADQ: ", mean(E.adq), "+/-",
  ↪  qt(0.975,N-1)*sd(E.adq)/sqrt(N), "\n")
> cat("Erreur ADL: ", mean(E.adl), "+/-",
  ↪  qt(0.975,N-1)*sd(E.adl)/sqrt(N), "\n")
> cat("Erreur NBA: ", mean(E.nba), "+/-",
  ↪  qt(0.975,N-1)*sd(E.nba)/sqrt(N), "\n")
>
> prob.ad(adq, Xapp, zapp, c(0.2,0.4,0.6,0.8))
> prob.ad(adl, Xapp, zapp, c(0.2,0.4,0.6,0.8))
> prob.ad(nba, Xapp, zapp, c(0.2,0.4,0.6,0.8))
```

On obtient les résultats consignés dans le tableau 1. Schématiquement, pour le premier jeu de données, ADQ donne de meilleurs résultats que les deux autres modèles ; pour le second jeu de données, les meilleurs résultats sont obtenus pour ADQ et NBC, et pour le troisième avec ADQ et ADL. Ces résultats s'expliquent par les caractéristiques des classes (matrices de covariance différentes dans le premier cas, différentes et diagonales dans le second, égales et non diagonales dans le troisième). Notons que les différences en termes de performances sont globalement faibles.

Données	A.D.Q.	A.D.L.	Cl. bayésien naïf
`Synth1`	0.03363363	0.04459459	0.04384384
`Synth2`	0.05915916	0.07882883	0.05945946
`Synth3`	0.08903904	0.08948949	0.09144144

TABLE 1 – Résultats obtenus pour les jeux de données synthétiques `Synth1-1000`, `Synth2-1000` et `Synth3-1000`.

2. Recommencer en ne sélectionnant que $n_{\text{app}} = 20$ exemples au total pour l'apprentissage ; comparer et interpréter.

En modifiant la fonction `separ1` pour qu'elle n'utilise plus que $1/50^{\text{e}}$ des données pour l'apprentissage, on trouve que les taux d'erreur varient beaucoup plus et que l'ADL donne globalement de meilleurs résultats que l'ADQ et le classifieur bayésien naïf. Cela est dû à une plus grande sensibilité de ces derniers modèles au manque de données. Notons qu'il est nécessaire de restreindre la taille de l'ensemble d'apprentissage de manière drastique pour observer ce phénomène.

1.2 Exercice théorique : règle de Bayes

Les jeux de données synthétiques utilisés au paragraphe 1.1 ont été obtenus par un processus génératif tel que décrit dans le TP précédent :

1. tout d'abord, l'effectif n_1 de la classe ω_1 a été déterminé par tirage aléatoire suivant une loi binomiale $\mathcal{B}(n, \pi_1)$, avec $\pi_1 = 0.5$;
2. n_1 individus ont ensuite été générés dans la classe ω_1 suivant une loi normale bivariée $\mathcal{N}(\mu_1, \Sigma_1)$, et $n_2 = n - n_1$ dans la classe ω_2 suivant une loi normale bivariée $\mathcal{N}(\mu_2, \Sigma_2)$.

Questions.

1. Quelles sont les distributions marginales des variables X^1 et X^2 dans chaque classe ?

   Le vecteur $\boldsymbol{X}$ suit une loi normale multivariée (bivariée) dans chacune des classes :

   $$\boldsymbol{X} \underset{\omega_k}{\sim} \mathcal{N}(\mu_k, \Sigma_k), \qquad \text{pour } k \in \{1, 2\}.$$

   Par conséquent, tout sous-vecteur d'un vecteur aléatoire gaussien étant lui-même gaussien (les paramètres étant obtenus en sélectionnant les lignes et colonnes du vecteur espérance et de la matrice de covariance), on en déduit que

   $$X^j \underset{\omega_k}{\sim} \mathcal{N}(\mu_{kj}, \sigma^2_{kj}),$$

   où $j \in \{1, 2\}$ est l'indice de la variable considérée.

2. Calculer l'expression des courbes d'iso-densité dans la classe ω_k, en fonction de μ_k et Σ_k. À quoi correspondent ces courbes dans le cas général (Σ_k quelconque) ? Si Σ_k est diagonale, sphérique ?

   La courbe d'iso-densité de la classe ω_k est définie par

   $$\big\{\boldsymbol{x} \text{ t.q. } f_k(\boldsymbol{x}) = K\big\},$$

   avec $0 < K \leq f_k(\mu_k)$, c'est-à-dire $0 < K \leq (2\pi)^{-\frac{p}{2}} |\Sigma_k|^{-\frac{1}{2}}$. Dans le cas général, cela donne l'équation d'une ellipse de centre μ_k :

   $$\begin{aligned} f_k(\boldsymbol{x}) = K \quad &\Leftrightarrow \quad |\Sigma_k|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \mu_k)^T \Sigma_k^{-1} (\boldsymbol{x} - \mu_k)\right) = (2\pi)^{\frac{p}{2}} K, \\ &\Leftrightarrow \quad (\boldsymbol{x} - \mu_k)^T \Sigma_k^{-1} (\boldsymbol{x} - \mu_k) = -p \log(2\pi) - 2 \log K - \log |\Sigma_k|; \end{aligned}$$

   si $K = (2\pi)^{-\frac{p}{2}} |\Sigma_k|^{-\frac{1}{2}}$, la courbe d'iso-densité se réduit à un unique point :

   $$f_k(\boldsymbol{x}) = K \Leftrightarrow \boldsymbol{x} = \mu_k;$$

   si $K \leq 0$ ou $K > (2\pi)^{-\frac{p}{2}} |\Sigma_k|^{-\frac{1}{2}}$, il n'existe pas de valeur de $\boldsymbol{x}$ vérifiant l'égalité $f_k(\boldsymbol{x}) = K$, la courbe d'iso-densité est donc un ensemble vide.

   En dimension $p = 2$, on peut aller plus loin en factorisant la matrice Σ_k :

   $$\Sigma_k = \lambda_k P_k A_k P_k^T, \quad \lambda_k > 0, \quad P_k = \begin{pmatrix} \cos\theta_k & -\sin\theta_k \\ \sin\theta_k & \cos\theta_k \end{pmatrix}, \quad A_k = \begin{pmatrix} a_k & 0 \\ 0 & 1/a_k \end{pmatrix}.$$

   L'orientation de la classe et les longueurs des axes de l'ellipse dépendent des paramètres λ_k et a_k intervenant dans cette décomposition, comme le montre la figure 2.

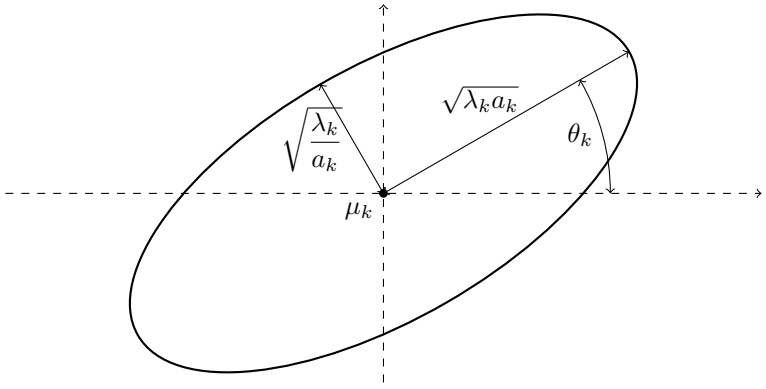

FIGURE 2 – Orientation d'une classe gaussienne de centre μ_k et de matrice de covariance Σ_k suivant la décomposition $\Sigma_k = \lambda_k P_k A_k P_k^T$.

3. Calculer l'expression de la frontière de décision de la règle de Bayes $\delta^\star$ dans le cas général.

   La frontière de décision est définie par

   $$\big\{\boldsymbol{x} \text{ t.q. } g_1(\boldsymbol{x}) = g_2(\boldsymbol{x})\big\} \Leftrightarrow \big\{\boldsymbol{x} \text{ t.q. } g_1(\boldsymbol{x}) - g_2(\boldsymbol{x}) = 0\big\},$$

   où la fonction discriminante g_k est définie par

   $$g_k(\boldsymbol{x}) = -\frac{1}{2}(\boldsymbol{x} - \mu_k)^T \Sigma_k^{-1}(\boldsymbol{x} - \mu_k) - \frac{1}{2}\log|\Sigma_k| + \log \pi_k - \frac{p}{2}\log(2\pi).$$

   Représenter cette frontière et comparer aux frontières obtenues avec les trois modèles d'analyse discriminante pour le jeu de données `Synth1-1000`, généré avec les paramètres suivants :

   $$\mu_1 = \begin{pmatrix} -1 \\ -2 \end{pmatrix}, \quad \mu_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad \Sigma_1 = \begin{pmatrix} 3 & -1.5 \\ -1.5 & 2 \end{pmatrix}, \quad \Sigma_2 = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix}.$$

   Il existe plusieurs possibilités, qui donnent rigoureusement les mêmes résultats. On peut calculer les fonctions discriminantes en utilisant les « vrais » paramètres, ou encore calculer les « vraies » probabilités a posteriori (là aussi, en utilisant les « vrais » paramètres).

   On en déduit ensuite la frontière de décision, dans le premier cas en traçant la courbe de niveau 0 de la surface $g_1(\boldsymbol{x}) - g_2(\boldsymbol{x})$, et dans le second cas en traçant la courbe de niveau 0.5 de la surface $\Pr(\omega_1|\boldsymbol{x})$.

4. Calculer l'expression de la frontière de décision de la règle de Bayes $\delta^\star$ lorsque Σ_1 et Σ_2 sont diagonales ($\Sigma_k = \text{diag}(\sigma_{k1}^2, \sigma_{k2}^2)$, pour $k = 1, 2$).

   On remarquera tout d'abord que si Σ_k est diagonale, $|\Sigma_k| = \sigma_{k1}^2\sigma_{k2}^2$. De plus, on a

   $$(\boldsymbol{x} - \mu_k)^T \Sigma_k^{-1}(\boldsymbol{x} - \mu_k) = \left(\frac{x_1 - \mu_{k1}}{\sigma_{k1}}\right)^2 + \left(\frac{x_2 - \mu_{k2}}{\sigma_{k2}}\right)^2.$$

   En utilisant le fait que $\pi_1 = \pi_2$, on a donc la frontière de décision suivante :

   $$\left(\frac{x_1 - \mu_{11}}{\sigma_{11}}\right)^2 - \left(\frac{x_1 - \mu_{21}}{\sigma_{21}}\right)^2 + \left(\frac{x_2 - \mu_{12}}{\sigma_{12}}\right)^2 - \left(\frac{x_2 - \mu_{22}}{\sigma_{22}}\right)^2 + 2\log\frac{\sigma_{11}\sigma_{12}}{\sigma_{21}\sigma_{22}} = 0.$$

   Représenter cette frontière et comparer aux frontières obtenues avec les trois modèles d'analyse discriminante pour le jeu de données `Synth2-1000`, généré avec les paramètres suivants :

   $$\mu_1 = \begin{pmatrix} -1 \\ -2 \end{pmatrix}, \quad \mu_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad \Sigma_1 = \begin{pmatrix} 3 & 0 \\ 0 & 1 \end{pmatrix}, \quad \Sigma_2 = \begin{pmatrix} 1 & 0 \\ 0 & 4.5 \end{pmatrix}.$$

5. Calculer la frontière de décision de la règle de Bayes $\delta^\star$ lorsque $\Sigma_1 = \Sigma_2 = \Sigma$.

On a dans ce cas la frontière de décision

$$\left(\Sigma^{-1}(\mu_1-\mu_2)\right)^T\left(\boldsymbol{x}-\frac{\mu_1+\mu_2}{2}+\frac{\ln(\pi_1/\pi_2)}{(\mu_1-\mu_2)^T\Sigma^{-1}(\mu_1-\mu_2)}(\mu_1-\mu_2)\right)=0.$$

Après simplification (notamment en utilisant $\pi_1=\pi_2$ et $\mu_1=-\mu_2$), on obtient une frontière définie par

$$(\mu_1-\mu_2)^T\Sigma^{-1}\boldsymbol{x}=0.$$

Représenter cette frontière et comparer aux frontières obtenues avec les trois modèles d'analyse discriminante pour le jeu de données `Synth3-1000`, généré avec les paramètres suivants :

$$\mu_1=\begin{pmatrix}-1\\-2\end{pmatrix},\quad \mu_2=\begin{pmatrix}1\\2\end{pmatrix},\quad \Sigma_1=\Sigma_2=\begin{pmatrix}3&-1.5\\-1.5&4.5\end{pmatrix}.$$

2 Détection de spams par analyse discriminante

On considère à présent les données contenues dans le fichier `spambase.csv`. On pourra les charger au moyen du code suivant :

```
> Spam <- read.csv("donnees/spambase.csv", header=T, row.names=1)
> X <- Spam[,-58]
> z <- as.factor(Spam[,58])
```

Les données sont des descripteurs calculés sur des courriers électroniques, et ont pour objectif de distinguer les spams des courriers réguliers. Les variables initiales sont :

1. 48 mesures de fréquences, dans le mail, de mots pré-définis,
2. 6 mesures de fréquences de caractères pré-définis,
3. la longueur moyenne des séquences ininterrompues de lettres majuscules dans le message,
4. la longueur de la plus longue séquence ininterrompue de lettres majuscules,
5. le nombre total de majuscules dans le message.

2.1 Utilisation des modèles précédents

Dans un premier temps, on utilisera « naïvement » les modèles d'analyse discriminante développés précédemment, que l'on sait présenter des garanties d'optimalité si les classes sont gaussiennes.

L'utilisation de ces modèles peut poser un certain nombre de problèmes d'ordre numérique. On pourra tenter de les résoudre en pré-traitant les données (centrage-réduction, sélection de variables [1]). Si s'avère que certains modèles ne sont pas utilisables malgré ces procédures, on veillera à apporter des éléments d'explication.

On obtient les résultats suivants, après centrage-réduction des données et conservation des 20 premières composantes principales (calculées sur l'ensemble d'apprentissage).

```
> Erreur ADQ:  0.2172099 +/- 0.01281418
> Erreur ADL:  0.1319752 +/- 0.004425083
> Erreur NBA:  0.2323664 +/- 0.03508289
```

2.2 Classifieur bayésien naïf pour données binaires

Simplification des données

On considère à présent une simplification des données décrites ci-dessus ; plus particulièrement, elles ont été transformées en nouvelles variables *binaires* de la manière suivante :

1. Dans un contexte d'apprentissage supervisé, *on exclura évidemment les données de test de la procédure de sélection de variables.*

1. les $48 + 6 = 54$ mesures de fréquence de mots ont été converties en indication de la présence (valeur `1`) ou absence (valeur `0`) du mot dans le message,
2. les autres mesures (longueur moyenne et maximale des séquences de lettres majuscules, nombre total de majuscules dans le message) ont été remplacées par `1` si la valeur originale était supérieure à la médiane des observations de la variable dans `spambase`, et `0` sinon.

Ces nouvelles données sont contenues dans le fichier `spambase2.csv`.

Implémentation

Les probabilités a posteriori des classes sont obtenues par la règle de Bayes :

$$\Pr(Z_{ik} = 1|\boldsymbol{X} = \boldsymbol{x}) = \frac{\Pr(\boldsymbol{X} = \boldsymbol{x}|Z_{ik} = 1)\Pr(Z_{ik} = 1)}{\Pr(\boldsymbol{X} = \boldsymbol{x})};$$

sous l'hypothèse d'indépendance des attributs conditionnellement à la classe, on a donc

$$\Pr(Z_{ik} = 1|\boldsymbol{X} = \boldsymbol{x}) = \frac{\pi_k \prod_{j=1}^{p} (p_{kj})^{x_j}(1 - p_{kj})^{1-x_j}}{\sum_{\ell=1}^{g} \pi_\ell \prod_{j=1}^{p} (p_{\ell j})^{x_j}(1 - p_{\ell j})^{1-x_j}}.$$

On admettra pour l'instant que les estimateurs du maximum de vraisemblance des paramètres du modèle sont, pour tout $k = 1, \ldots, g$ et tout $j = 1, \ldots, p$,

$$\widehat{\pi}_k = \frac{1}{n}\sum_{i=1}^{n} z_{ik}, \quad \widehat{p}_{kj} = \frac{\sum_{i=1}^{n} z_{ik}x_{ij}}{\sum_{i=1}^{n} z_{ik}}.$$

On programmera deux fonctions pour implémenter le modèle décrit ci-dessus. On fera l'apprentissage du modèle avec une fonction `bin.app`, qui prendra comme arguments d'entrée un tableau individus-variables `Xapp` et un vecteur d'indicateurs de classe `zapp`, et retournera les estimations des paramètres du modèle (probabilités a priori `pik` et probabilités conditionnelles `pkj`).

La fonction réalisant l'apprentissage du modèle est assez simple à programmer :

```
> bin.app <- function(Xapp, zapp)
> {
>     nbCl <- nlevels(zapp)
>
>     outp <- NULL
>     outp$pik <- as.numeric(table(zapp))
>     outp$pik <- outp$pik/sum(outp$pik)
>     outp$pkj <- matrix(0, nrow=nbCl, ncol=length(outp$pik))
>
>     for (k in 1:nbCl)
>     {
>         outp$pkj[k,] <- apply(Xapp[which(zapp==k),],2,mean)
>     }
>
>     outp
> }
```

On classera un ensemble de données de test au moyen d'une fonction `bin.val`, qui prendra en arguments d'entrée les estimations des paramètres du modèle (`pik` et `pkj`) de même que le tableau individus-variables des individus de test (`Xtst`), et retournera le vecteur `prob` des probabilités a posteriori calculées et les prédictions associées `pred`.

La fonction `bin.val` permet elle de réaliser le classement des individus selon ce modèle :

```
> bin.val <- function(model, Xtst)
```

```
> {
>   ntst <- dim(Xtst)[1]
>   nbCl <- length(model$pik)
>
>   outp <- NULL
>
>   outp$prob <- matrix(0, nrow=ntst, ncol=nbCl)
>   for (k in 1:nbCl)
>   {
>     Mpkj <- matrix(rep(model$pkj[k,],ntst),nrow=ntst,byrow=T)
>     outp$prob[,k] <- model$pik[k] * apply(Mpkj^Xtst*(1-Mpkj)^(1-Xtst),1,prod)
>   }
>   outp$prob <- outp$prob / apply(outp$prob,1,sum)
>   outp$pred <- max.col(outp$prob)
>
>   outp
> }
```

Test

Utiliser la procédure habituelle pour évaluer les performances du modèle sur les données `spambase2`. Comparer avec les autres modèles utilisés. Que peut-on remarquer ? Quelles performances obtient-on en comparaison de celles obtenues sur les données `spambase` ? Pourquoi ?

En gardant toutes les composantes principales :

```
> Erreur ADQ:  0.1708279 +/- 0.02455292
> Erreur ADL:  0.07105606 +/- 0.002773364
> Erreur NBA:  0.1107236 +/- 0.004545019
> Erreur BIN:  0.1154498 +/- 0.002502007
```

En gardant les 50 premières composantes principales :

```
> Erreur ADQ:  0.1041721 +/- 0.007124626
> Erreur ADL:  0.07219687 +/- 0.001679477
> Erreur NBA:  0.1072034 +/- 0.004423976
> Erreur BIN:  0.1139179 +/- 0.002448519
```

De manière intéressante, ce passage aux données binaires améliore les performances de tous les modèles, bien qu'on perde de l'information. On peut imaginer que l'information importante pour classer les données est en fait celle de présence/absence. On constate que QDA reste sensible à la dimension de l'espace.

Le classifieur bayésien naïf développé pour les données binaires fait globalement mieux que le classifieur bayésien naïf développé dans le cas gaussien (qui n'est pas ridicule). Il n'en reste pas moins qu'il fait moins bien que QDA et LDA : il est probable que l'hypothèse d'indépendance conditionnelle ne soit pas adaptée à ces données.

Formalisation (subsidiaire)

Les questions suivantes ont pour objectif de prouver l'expression des estimateurs du maximum de vraisemblance des paramètres donnés ci-dessus.

1. Quelle est la distribution conditionnelle d'un attribut X^j conditionnellement à la classe Z ? En déduire l'expression de $\Pr(X^j = x_j | Z = \omega_k)$.

On a évidemment

$$X^j \underset{\omega_k}{\sim} \mathcal{B}(p_{kj}),$$

et donc

$$\Pr(X^j = x_j | Z = \omega_k) = (p_{kj})^{x_j}(1 - p_{kj})^{1-x_j}.$$

2. En supposant l'indépendance des variables $X^1, \ldots, X^p$ conditionnellement à la classe $Z = \omega_k$, en déduire la probabilité jointe du vecteur aléatoire $\boldsymbol{X}$ conditionnellement à la classe ω_k : $\Pr(\boldsymbol{X} = \boldsymbol{x} | Z = \omega_k)$, où $\boldsymbol{x} = (x_1, \ldots, x_p)^T$ est une réalisation du vecteur aléatoire $\boldsymbol{X}$.

En supposant l'indépendance conditionnelle, on peut exprimer la probabilité d'un vecteur de variables binaires comme le produit des probabilités des valeurs le constituant :

$$\begin{aligned} \Pr(\boldsymbol{X} = \boldsymbol{x} | Z = \omega_k) &= \prod_{j=1}^{p} \Pr(X^j = x_j | Z = \omega_k), \\ &= \prod_{j=1}^{p} (p_{kj})^{x_j}(1 - p_{kj})^{1-x_j}. \end{aligned}$$

3. En considérant que le i$^{\text{e}}$ exemple d'apprentissage consiste en un couple $(\boldsymbol{x}_i, \boldsymbol{z}_i)$ où $\boldsymbol{x}_i$ est une réalisation de $\boldsymbol{X}$ et $\boldsymbol{z}_i$ une réalisation de $\boldsymbol{Z}$ du vecteur de classe (avec $\boldsymbol{z}_i = (z_{i1}, \ldots, z_{ig})^T$, et $z_{ik} = 1$ si $\boldsymbol{x}_i \in \omega_k$ et $z_{ik} = 0$ sinon), écrire la probabilité jointe $\Pr(\boldsymbol{X} = \boldsymbol{x}_i, \boldsymbol{Z} = \boldsymbol{z}_i)$.

La probabilité jointe d'un vecteur d'attributs et de la classe $\boldsymbol{X}, \boldsymbol{Z}$ est obtenue par

$$\Pr(\boldsymbol{X}, \boldsymbol{Z}) = \prod_{\boldsymbol{z}} (\Pr(\boldsymbol{Z} = \boldsymbol{z}) \Pr(\boldsymbol{X} | \boldsymbol{Z} = \boldsymbol{z}))^{\mathbb{1}_{[\boldsymbol{Z}=\boldsymbol{z}]}},$$

où $\mathbb{1}_{[\boldsymbol{Z}=\boldsymbol{z}]}$ est la fonction indicatrice de l'événement $\boldsymbol{Z} = \boldsymbol{z}$ (elle est égale à 1 si $\boldsymbol{Z} = \boldsymbol{z}$, et à 0 sinon). En utilisant l'encodage 0/1 de l'information de classe ($z_k = 1$ si et seulement $\boldsymbol{x} \in \omega_k$, et $Z_k = 0$ sinon), alors :

$$\begin{aligned} \Pr(\boldsymbol{X} = \boldsymbol{x}, \boldsymbol{Z} = \boldsymbol{z}) &= \prod_{k=1}^{g} (\Pr(Z_k = 1)\ \Pr(\boldsymbol{X} = \boldsymbol{x} | Z_k = 1))^{z_k}, \\ &= \prod_{k=1}^{g} (\pi_k\ \Pr(\boldsymbol{X} = \boldsymbol{x} | Z = \omega_k))^{z_k}. \end{aligned}$$

On a donc, pour le i$^{\text{e}}$ exemple d'apprentissage :

$$\begin{aligned} \Pr(\boldsymbol{X} = \boldsymbol{x}_i, \boldsymbol{Z} = \boldsymbol{z}_i) &= \prod_{k=1}^{g} (\pi_k\ \Pr(\boldsymbol{X} | Z = \omega_k))^{z_{ik}} \\ &= \prod_{k=1}^{g} \left(\pi_k \prod_{j=1}^{p} (p_{kj})^{x_{ij}} (1 - p_{kj})^{1-x_{ij}} \right)^{z_{ik}}. \end{aligned}$$

4. En déduire la vraisemblance jointe des paramètres du modèle p_{kj} et $\pi_k = \Pr(Z = \omega_k)$ ($k = 1, \ldots, g$, $j = 1, \ldots, p$) étant donné l'ensemble d'apprentissage $\{(\boldsymbol{x}_1, \boldsymbol{z}_1), \ldots, (\boldsymbol{x}_n, \boldsymbol{z}_n)\}$.

La vraisemblance jointe des paramètres du modèle p_{kj} (pour $k = 1, \ldots, g$ et $j = 1, \ldots, p$) et π_k (pour $k = 1, \ldots, g$) est simplement la probabilité d'observer l'échantillon d'apprentissage

étant donné ces paramètres :

$$\begin{aligned} L\left(p_{kj}, \pi_k; \boldsymbol{x}_1, \boldsymbol{z}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{z}_n\right) &= \prod_{i=1}^{n} \Pr(\boldsymbol{X} = \boldsymbol{x}_i, \boldsymbol{Z} = \boldsymbol{z}_i), \\ &= \prod_{i=1}^{n} \prod_{k=1}^{g} \left(\pi_k \prod_{j=1}^{p} (p_{kj})^{x_{ij}} (1 - p_{kj})^{1 - x_{ij}} \right)^{z_{ik}}; \end{aligned}$$

la log-vraisemblance s'écrit donc :

$$\begin{aligned} \ln L\left(p_{kj}, \pi_k; \boldsymbol{x}_i, \boldsymbol{z}_i\right) &= \ln \left(\prod_{i=1}^{n} \prod_{k=1}^{g} \left(\pi_k \prod_{j=1}^{p} (p_{kj})^{x_{ij}} (1 - p_{kj})^{1 - x_{ij}} \right)^{z_{ik}} \right), \\ &= \sum_{i=1}^{n} \sum_{k=1}^{g} \left(z_{ik} \ln \pi_k + z_{ik} \sum_{j=1}^{p} \left(x_{ij} \ln p_{kj} + (1 - x_{ij}) \ln(1 - p_{kj}) \right) \right). \end{aligned}$$

5. Montrer que l'EMV de chaque paramètre p_{kj} $(k = 1, \ldots, g,\ j = 1, \ldots, p)$ est

$$\widehat{p}_{kj} = \frac{1}{n_k} \sum_{i=1}^{n} z_{ik} x_{ij},$$

avec $n_k = \sum_{i=1}^{n} z_{ik}$ (on supposera pour cela que $0 < p_{kj} < 1$ pour tout $k = 1, \ldots, g$ et $j = 1, \ldots, p$), et que l'EMV de chaque probabilité a priori est

$$\widehat{\pi}_k = \frac{1}{n} \sum_{i=1}^{n} z_{ik} = \frac{n_k}{n}.$$

Probabilités a priori La dérivée de la log-vraisemblance par rapport à l'une des proportions π_ℓ est :

$$\frac{\partial \ln L(p_{kj}, \pi_k)}{\partial \pi_\ell} = \sum_{i=1}^{n} \frac{z_{i\ell}}{\pi_\ell}.$$

L'annuler tout en respectant la contrainte $\sum_{k=1}^{g} \pi_k = 1$ (lagrangien) donne

$$\widehat{\pi}_\ell = \frac{1}{n} \sum_{i=1}^{n} z_{i\ell} = \frac{n_\ell}{n},$$

où $n_\ell = \sum_{i=1}^{n} z_{i\ell}$ représente le nombre d'exemples d'appentissage de la classe ω_ℓ.

Probabilités conditionnelles Le calcul de la dérivée de la log-vraisemblance par rapport à l'une des probabilités $p_{\ell j}$ donne

$$\frac{\partial \ln L(p_{kj}, \pi_k)}{\partial p_{\ell j}} = \sum_{i=1}^{n} \left(z_{i\ell} \left(\frac{x_{ij}}{p_{\ell j}} - \frac{(1 - x_{ij})}{(1 - p_{\ell j})} \right) \right).$$

L'annulation de cette dérivée donne

$$\begin{aligned}
\frac{\partial \ln L(p_{kj}, \pi_k)}{\partial p_{\ell j}} = 0 \quad &\Leftrightarrow \quad \sum_{i=1}^{n} z_{i\ell} \left(\frac{x_{ij}}{p_{\ell j}} - \frac{(1 - x_{ij})}{(1 - p_{\ell j})} \right) = 0, \\
&\Leftrightarrow \quad \frac{1}{p_{\ell j}} \sum_{i=1}^{n} z_{i\ell} x_{ij} - \frac{1}{1 - p_{\ell j}} \sum_{i=1}^{n} z_{i\ell} (1 - x_{ij}) = 0, \\
&\Leftrightarrow \quad (1 - p_{\ell j}) \sum_{i=1}^{n} z_{i\ell} x_{ij} = p_{\ell j} \sum_{i=1}^{n} z_{i\ell} (1 - x_{ij}), \\
&\Leftrightarrow \quad \sum_{i=1}^{n} z_{i\ell} x_{ij} - p_{\ell j} \sum_{i=1}^{n} z_{i\ell} x_{ij} = p_{\ell j} \sum_{i=1}^{n} z_{i\ell} - p_{\ell j} \sum_{i=1}^{n} z_{i\ell} x_{ij}, \\
&\Leftrightarrow \quad p_{\ell j} = \frac{\sum_{i=1}^{n} z_{i\ell} x_{ij}}{\sum_{i=1}^{n} z_{i\ell}}.
\end{aligned}$$

En d'autres termes, l'estimateur de la probabilité que $X^j = 1$ pour les exemples de la classe $Z = \omega_k$ est la proportion (empirique) de valeurs "1" pour la variable X^j, pour les exemples d'apprentissage de la classe $Z = \omega_k$.

Dérivées secondes On peut remarquer que les dérivées croisées sont toutes nulles, puisque chaque dérivée première par rapport à un paramètre ne fait intervenir que lui (qu'il s'agisse de π_ℓ ou de $p_{\ell j}$). De plus, on peut facilement vérifier que

$$\begin{aligned}
\frac{\partial^2 \ln L(p_{kj}, \pi_k)}{\partial \pi_\ell^2} &= -\sum_{i=1}^{n} \frac{z_{i\ell}}{(\pi_\ell)^2}, \\
\frac{\partial^2 \ln L(p_{kj}, \pi_k)}{\partial p_{\ell j}^2} &= -\sum_{i=1}^{n} z_{i\ell} \left(\frac{x_{ij}}{p_{\ell j}^2} + \frac{(1 - x_{ij})}{(1 - p_{\ell j})^2} \right).
\end{aligned}$$

La matrice des dérivées secondes est donc diagonale, et tous ses éléments diagonaux sont strictement négatifs : elle est donc définie négative. En conséquence, l'annulation des dérivées premières donne bien des estimations du maximum de vraisemblance.