Classification d'images de chats et de chiens

Projet de Statistique 1, 2ème année

Ensae ParisTech

Hugo Thimonier & Dimitri Meunier

23 janvier 2019

Table des matières

I Découverte de la base de données et réduction de la dimension

Remarque. *Dans toute la suite du sujet nous noterons p et non π la paramètre de la variable aléatoire de Bernouilli Y afin d'éviter les confusions.*

I.1 Question (1)

```
## [1] "Number of observations: " "363"
## [1] "Number of observations in train set: "
## [2] "315"
## [1] "Number of observations in test set: "
## [2] "48"
## [1] "Dimension of each observation: " "40000"
```

Image of a cat, labelled 1

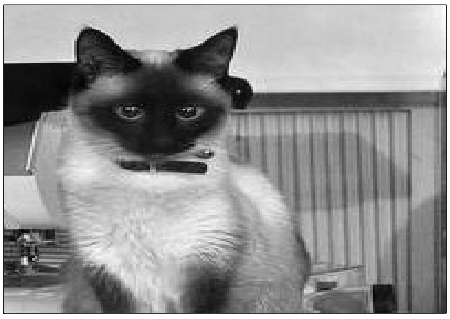

Image of a cat, labelled 1

Image of a dog, labelled 0

Image of a dog, labelled 0

```
## [1] "Percentage of positive targets (cats) in train set"
## [2] "0.498412698412698"
## [1] "Percentage of positive targets (cats) in test set"
## [2] "0.479166666666667"
```

Le label 1 correspond aux images de chats (et donc le label 0 correspond aux chiens), il y a 363 observations dont 315 dans le train set et 48 dans le test set. Il y a 49.8% de chats dans le train set et 47,9% dans le test set. Chaque image est représentée comme un vecteur de $\mathbb{R}^{40000}$

I.2 Question (2)

La fonction SVD nous retourne directement la matrice V de la décomposition en valeurs singulières, les 15 premières composantes principales sont ensuite selectionées en conservant les 15 premières colonnes de la matrice de PCA CV. La fonction *compute_PCA* s'occupe du calcul des composantes principales, de découper à nouveaux les données et de sauvegarder les données réduites dans $C_{train}.Rdata$ et $C_{test}.Rdata$ et retourne la variance expliquée (voir ci-dessous).

```
compute_PCA <- function(Xtrain,Xtest) {
  X = rbind(Xtrain,Xtest) #concatenation
  X = scale(X, center = TRUE, scale = FALSE)  #centrage
  PCA = svd(X,nu=0,nv=15) #nv=15 on ne conserve que 15 composantes
  C = X %*% PCA$v
  Ctrain = C[1:315,]
  Ctest = C[316:dim(C)[1],]
  save(Ctrain, file = "Ctrain.RData") #sauvegarde des donnees reduites
  save(Ctest, file = "Ctest.RData")

  return(sum(PCA$d[1:15]**2)/sum(PCA$d**2)) #variance expliquee
}
```

Soit $\lambda_1 > \cdots > \lambda_{373}$ les valeurs singulières ordonées par ordre décroissant. Le pourcentage de variance expliqué par les 15 premières composantes principales est donné par :

$$\frac{\sum_{i=1}^{15} \lambda_i^2}{\sum_{i=1}^{373} \lambda_i^2}$$

```
var_expl = compute_PCA(Xtrain,Xtest)
var_expl

## [1] 0.7134303
```

La fonction *compute_PCA* retourne un pourcentage de variance expliqué de 71,34%.

I.3 Question (3)

Les notations choisies pour formaliser le modèle statistique sont : $\mathcal{B}(\mathbb{R}^{15})$ pour la tribu borélienne engendrée par $\mathbb{R}^{15}$, $\mathcal{P}(\{0,1\})$ pour l'ensemble des parties de $\{0,1\}$, $\otimes$ represente l'opération produit de tribus [1], $\mathbb{M}^{a\times b}$ est l'espace des matrices réelles à a lignes et b colonnes. Nous définissons de plus

$$\forall B \in \mathcal{B}(\mathbb{R}^{15}), \forall j \in \{0,1\}, m_j(B) = \int_B \frac{1}{(2\pi)^{15/2}} (\det \Sigma_j)^{-1/2} exp\{-\frac{1}{2}(c-\mu_j)^T \Sigma_j^{-1}(c-\mu_j)\} dc$$

et

$$\theta = (p, \mu_0, \mu_1, \Sigma_0, \Sigma_1)$$

Le modèle statistique associé aux observations $((c_1, y_1), \ldots, (c_n, y_n))$ est l'espace de probabilité $(\Omega, \mathcal{F}, (\mathbb{P}_\theta, \theta \in \Theta))$ donné par

1. Soient $\mathcal{A}$ et $\mathcal{B}$ deux tribus, $\mathcal{A} \otimes \mathcal{B}$ est la tribu engendrée par les pavés $A \times B$ où $A \in \mathcal{A}, B \in \mathcal{B}$

$$\Omega = (\mathbb{R}^{15} \times \{0,1\})^n$$

$$\mathcal{F} = (\mathcal{B}(\mathbb{R}^{15}) \otimes \mathcal{P}(\{0,1\}))^{\otimes^n}$$

$$\Theta = [0,1] \times \mathbb{R}^{15} \times \mathbb{R}^{15} \times \mathbb{M}^{15\times 15} \times \mathbb{M}^{15\times 15}$$

$$\forall \theta \in \Theta, \forall B \in \mathcal{B}(\mathbb{R}^{15}), \forall A \in \mathcal{P}(\{0,1\}),$$

$$\mathbb{P}_\theta(A \times B) = p\delta_1(A)m_1(B) + (1-p)\delta_0(A)m_0(B)$$

Remarque. *On vérifie bien que* $\mathbb{P}_\theta(\{0,1\} \times \mathbb{R}^{15}) = p\delta_1(\{0,1\}) \times 1 + (1-p)\delta_0(\{0,1\}) \times 1 = p+1-p = 1$. *De plus, en prenant* $B = \mathbb{R}^{15}$ *on retombe bien sur une loi de Bernouilli de paramètre* p *pour la loi marginale de* Y

$$\mathbb{P}_\theta(A \times \mathbb{R}^{15}) = p\delta_1(A) + (1-p)\delta_0(A)$$

II Analyse Discriminante Quadratique

II.1 Question (4)

Par définition de la vraisemblance on a que

$$L((c_1,y_1),\dots,(c_n,y_n)|\theta) = \prod_{i=1}^n f_{c,Y}(c_i,y_i) = \prod_{i=1}^n f_Y(y_i) f_{c|Y_i=y_i}(c)$$

Dès lors on a que

$$\begin{aligned}
L((c_1,y_1),\dots,(c_n,y_n)|\theta) &= \prod_{i=1}^n \Bigg\{ p^{y_i}(1-p)^{1-y_i} \\
&\quad \left[\tfrac{1}{(2\pi)^{15/2}}(\det\Sigma_1)^{-1/2} exp\{-\tfrac{1}{2}(c_i-\mu_1)^T\Sigma_1^{-1}(c_i-\mu_1)\}\right]^{\mathbb{1}_{y_i=1}} \\
&\quad \left[\tfrac{1}{(2\pi)^{15/2}}(\det\Sigma_0)^{-1/2} exp\{-\tfrac{1}{2}(c_i-\mu_0)^T\Sigma_0^{-1}(c_i-\mu_0)\}\right]^{\mathbb{1}_{y_i=0}} \Bigg\} \\
&= p^{\sum_i y_i}(1-p)^{n-\sum_i y_i} \\
&\quad \left[\tfrac{1}{(2\pi)^{15/2}}(\det\Sigma_1)^{-1/2}\right]^{\sum_i y_i} \left[\tfrac{1}{(2\pi)^{15/2}}(\det\Sigma_0)^{-1/2}\right]^{n-\sum_i y_i} \\
&\quad exp\Big\{ -\tfrac{1}{2}\big[\textstyle\sum_{i,y_i=1}(c_i-\mu_1)^T\Sigma_1^{-1}(c_i-\mu_1) + \sum_{i,y_i=0}(c_i-\mu_0)^T\Sigma_0^{-1}(c_i-\mu_0)\big]\Big\} \\
&= p^{N_1}(1-p)^{N_2} \\
&\quad \left[\tfrac{1}{(2\pi)^{15/2}}(\det\Sigma_1)^{-1/2}\right]^{N_1} \left[\tfrac{1}{(2\pi)^{15/2}}(\det\Sigma_0)^{-1/2}\right]^{N_2} \\
&\quad exp\Big\{ -\tfrac{1}{2}\big[\textstyle\sum_{i,y_i=1}(c_i-\mu_1)^T\Sigma_1^{-1}(c_i-\mu_1) + \sum_{i,y_i=0}(c_i-\mu_0)^T\Sigma_0^{-1}(c_i-\mu_0)\big]\Big\}
\end{aligned}$$

Ainsi, la log-vraisemblance est de la forme,

$$\begin{aligned} l((c_1,y_1),\ldots,(c_n,y_n)|\theta) &= N_1\log(p)+N_2\log(1-p)-\frac{N_1}{2}\log(\det(\Sigma_1)) \\ &\quad -\frac{1}{2}\sum_{i,y_i=1}(c_i-\mu_1)^T\Sigma_1^{-1}(c_i-\mu_1)-\frac{N_2}{2}\log(\det(\Sigma_0)) \\ &\quad -\frac{1}{2}\sum_{i,y_i=0}(c_i-\mu_0)^T\Sigma_0^{-1}(c_i-\mu_0)+\text{constante} \end{aligned} \tag{1}$$

avec constante $= -\frac{15n}{2}log(2\pi)$

II.2 Question (5)

L'estimateur du maximum de vraisemblance pour p est obtenu par la CPO, on a :

$$\begin{aligned} \frac{\partial l((c_1,y_1),\ldots,(c_n,y_n)|\theta)}{\partial p} &= 0 \\ \Leftrightarrow \frac{N_1}{p}-\frac{N_2}{1-p} &= 0 \\ \Leftrightarrow (1-p)N_1 &= N_2p \\ \Leftrightarrow N_1 &= (N_1+N_2)p \\ \Leftrightarrow \hat{p} &= \frac{N_1}{n} \end{aligned}$$

Pour μ_1 et μ_0 le calcul est rigoureusement le même, voici celui pour μ_1 à nouveau grâce à la CPO :

$$\begin{aligned} \frac{\partial l((c_1,y_1),\ldots,(c_n,y_n)|\theta)}{\partial \mu_1} &= 0 \\ \Leftrightarrow \frac{\partial}{\partial\mu_1}\sum_{i,y_i=1}(c_i-\mu_1)^T\Sigma_1^{-1}(c_i-\mu_1) &= 0 \\ \Leftrightarrow \frac{\partial}{\partial\mu_1}\left\{\sum_{i,y_i=1}c_i^T\Sigma_1^{-1}c_i-c_i^T\Sigma_1^{-1}\mu_1-\mu_1^T\Sigma_1^{-1}c_i+\mu_1^T\Sigma_1^{-1}\mu_1\right\} &= 0 \\ \Leftrightarrow \sum_{i,y_i=1}(-(c_i^T\Sigma_1^{-1})^T-\Sigma_1^{-1}c_i+\left(\Sigma_1^{-1}+(\Sigma_1^{-1})^T\right)\mu_1 &= 0 \end{aligned}$$

En tant que matrice de variance-covariance Σ_1 et Σ_0 sont symétriques, il en va de même pour leur inverse, ainsi $\Sigma_1^{-1} = (\Sigma_1^{-1})^T$ et $\Sigma_0^{-1} = (\Sigma_0^{-1})^T$. Dès lors,

$$\begin{aligned}
\frac{\partial l((c_1, y_1), \ldots, (c_n, y_n)|\theta)}{\partial \mu_1} &= 0 \\
\Leftrightarrow 2 \sum_{i, y_i = 1} \Sigma_1^{-1} c_i &= 2 N_1 \Sigma_1^{-1} \mu_1 \\
\Leftrightarrow \sum_{i, y_i = 1} \Sigma_1 \Sigma_1^{-1} c_i &= N_1 \mu_1 \\
\Leftrightarrow \hat{\mu}_1 &= \frac{1}{N_1} \sum_{i, y_i = 1} c_i
\end{aligned}$$

De même,

$$\hat{\mu}_0 = \frac{1}{N_2} \sum_{i, y_i = 0} c_i$$

De même pour Σ_1 et Σ_0 les résultats sont rigoureusement identiques, en s'aidant des formules de dérivées matricielles données on a,

$$\begin{aligned}
\frac{\partial l((c_1, y_1), \ldots, (c_n, y_n)|\theta)}{\partial \Sigma_1} &= 0 \\
\Leftrightarrow \frac{N_1}{2} \Sigma_1^{-1} - \frac{1}{2} \sum_{i, y_i = 1} \Sigma_1^{-1} (c_i - \mu_1)(c_i - \mu_1)^T \Sigma_1^{-1} &= 0 \\
\Leftrightarrow \frac{1}{N_1} \sum_{i, y_i = 1} \Sigma_1^{-1} (c_i - \mu_1)(c_i - \mu_1)^T \Sigma_1^{-1} &= \Sigma_1^{-1} \\
\Leftrightarrow \Sigma_1^{-1} \frac{1}{N_1} \sum_{i, y_i = 1} (c_i - \mu_1)(c_i - \mu_1)^T &= I_{15} \\
\Leftrightarrow \hat{\Sigma_1} = \frac{1}{N_1} \sum_{i, y_i = 1} (c_i - \mu_1)(c_i - \mu_1)^T
\end{aligned}$$

Ainsi en subtituant les estimateurs de μ_1 et μ_0 on obtient que,

$$\begin{aligned}
\hat{\Sigma}_1 &= \frac{1}{N_1} \sum_{i, Y_i = 1} (c_i - \hat{\mu}_1)(c_i - \hat{\mu}_1)^T \\
\hat{\Sigma}_0 &= \frac{1}{N_2} \sum_{i, Y_i = 0} (c_i - \hat{\mu}_0)(c_i - \hat{\mu}_0)^T
\end{aligned}$$

Les estimateurs du maximum de vraisemblance sont donc :

$$\begin{cases} \hat{p} &= \dfrac{N_1}{n} \\ \hat{\mu}_1 &= \dfrac{1}{N_1}\displaystyle\sum_{i,y_i=1} c_i \\ \hat{\mu}_0 &= \dfrac{1}{N_2}\displaystyle\sum_{i,y_i=0} c_i \\ \hat{\Sigma}_1 &= \dfrac{1}{N_1}\displaystyle\sum_{i,y_i=1} (c_i-\hat{\mu}_1)(c_i-\hat{\mu}_1)^T \\ \hat{\Sigma}_0 &= \dfrac{1}{N_2}\displaystyle\sum_{i,y_i=0} (c_i-\hat{\mu}_0)(c_i-\hat{\mu}_0)^T \end{cases}$$

II.3 Question (6)

Le sous-gradient $\nabla_{p,\mu_1,\mu_0} l(\theta)$ est de la forme

$$\nabla_{p,\mu_0,\mu_1} l(\theta) = \begin{pmatrix} \frac{N_1}{p} - \frac{N_2}{1-p} \\ -2\sum_{i,y_i=1} \Sigma_1^{-1} c_i + \Sigma_1^{-1}\mu_1 \\ -2\sum_{i,y_i=0} \Sigma_1^{-1} c_i + \Sigma_1^{-1}\mu_0 \end{pmatrix}$$

Remarque. $\nabla_{p,\mu_1,\mu_0} l(\theta) \in \mathbb{R}^{15}$

Par conséquent on obtient la sous-hessienne suivante

$$\nabla^2_{p,\mu_0,\mu_1} l(\theta) = \begin{pmatrix} -\frac{N_1}{p^2} - \frac{N_2}{(1-p)^2} & 0_{\mathbb{M}^{1\times 15}} & 0_{\mathbb{M}^{1\times 15}} \\ 0_{\mathbb{R}^{15}} & -2N_1\Sigma_1^{-1} & 0_{\mathbb{M}^{15\times 15}} \\ 0_{\mathbb{R}^{15}} & 0_{\mathbb{M}^{15\times 15}} & -2N_1\Sigma_0^{-1} \end{pmatrix}$$

La matrice est diagonale par blocs et chacun de ses blocs est défini négatif. En effet, nous savons que les matrices de covariances Σ_0 et Σ_1 sont définies positives donc leur inverse aussi et donc $-2N_1\Sigma_0^{-1}$ et $-2N_1\Sigma_1^{-1}$ sont définies négatives. C'est un exercice facile mais fastidieux à rédiger de voir que si la matrice est diagonale par blocs avec chacun de ses blocs défini négatif alors la matrice entière est définie négative.

Remarque. $\nabla^2_{p,\mu_0,\mu_1} l(\theta) \in \mathbb{M}^{31\times 31}$

II.4 Question (7)

$$\mathbb{E}[\hat{p}] = \mathbb{E}\left[\frac{\sum_{i=1}^n y_i}{n}\right] = \frac{1}{n}\sum_i \mathbb{E}[y_i] = \mathbb{E}[y_1] = p$$

$$\begin{aligned} \mathbb{E}[\hat{\mu}_1] &= \mathbb{E}\Big[\mathbb{E}[\hat{\mu}_1|y_1,\dots,y_n]\Big] = \mathbb{E}\left[\tfrac{1}{N_1}\mathbb{E}\left[\sum_i c_i \mathbb{1}_{y_i=1}\middle| y_1,\dots,y_n\right]\right] \\ &= \mathbb{E}\left[\tfrac{1}{N_1}\sum_i \mathbb{1}_{y_i=1}\mathbb{E}\left[c_i\middle|y_1,\dots,y_n\right]\right] = \mathbb{E}\left[\tfrac{1}{N_1}\sum_i \mathbb{1}_{y_i=1}\mathbb{E}[c_i|y_i]\right] \end{aligned}$$

Ici nous avons utilisé l'indépendance de (c_i, y_i) avec $(y_j)_{j\neq i}$ ce qui permet d'obtenir $\mathbb{E}[c_i|y_1,\dots,y_n] = \mathbb{E}[c_i|y_i]$ [2], or $\mathbb{E}[c_i|y_i]\mathbb{1}_{y_i=1} = \mu_1\mathbb{1}_{y_i=1}$, on obtient donc

$$\mathbb{E}[\hat{\mu}_1] = \mathbb{E}\left[\frac{1}{N_1}\sum_i \mathbb{1}_{y_i=1}\mu_1\right] = \mu_1\mathbb{E}[\frac{N_1}{N_1}] = \mu_1$$

De même, en effectuant exactement le même calcul pour μ_0 :

$$\mathbb{E}[\hat{\mu}_0] = \mu_0$$

II.5 Question (8)

On a que $\mathbb{E}[Y] = p$, dès lors l'estimateur de la méthode des moments qui est l'équivalent empirique de $\mathbb{E}[Y]$ est

$$p^{MM} = \bar{Y} = \frac{1}{n}\sum_i y_i = \frac{N_1}{n} = \hat{p}$$

De même on a que $\mathbb{E}[C|Y=1] = \mu_1$, avec,

$$\mathbb{E}[C|Y=1] = \frac{\mathbb{E}[C\mathbb{1}\{Y=1\}]}{\mathbb{E}[\mathbb{1}\{Y=1\}]}$$

ainsi l'estimateur de la méthode des moments est l'équivalent empirique de cette quantité :

$$\mu_1^{MM} = \frac{\frac{1}{n}\sum_{i=1}^n c_i\mathbb{1}_{y_i=1}}{\frac{1}{n}\sum_{i=1}^n \mathbb{1}_{y_i=1}} = \frac{\sum_{i,y_i=1} c_i}{\sum_i y_i} = \frac{\sum_{i,y_i=1} c_i}{N_1} = \hat{\mu}_1$$

De même on a que $\mathbb{E}[C|Y=0] = \mu_0$, ainsi l'estimateur de la méthode des moments est le moment empirique de

$$\mathbb{E}[C|Y=0] = \frac{\mathbb{E}[C\mathbb{1}\{Y=0\}]}{\mathbb{E}[\mathbb{1}\{Y=0\}]}$$

à savoir

$$\mu_0^{MM} = \frac{\frac{1}{n}\sum_{i,y_i=0} c_i}{\frac{1}{n}\sum_{i,y_i=0} y_i} = \frac{\sum_{i,y_i=1} c_i}{\sum_i(1-y_i)} = \frac{\sum_{i,y_i=1} c_i}{N_2} = \hat{\mu}_0$$

Enfin, on a que $\mathbb{V}(C|Y=1) = \Sigma_1$, l'estimateur de la méthode des moments est donc le moment empirique de

$$\mathbb{V}(C|Y=1) = \mathbb{E}[(c-\mu_1)(c-\mu_1)^T|Y=1]$$

à savoir, en se servant des estimateurs de μ_0 et μ_1 :

$$\Sigma_1^{MM} = \frac{1}{N_1}\sum_{i,y_i=1}(c_i-\hat{\mu}_1)(c_i-\hat{\mu}_1)^T = \hat{\Sigma}_1$$

De même pour Σ_0^{MM},

$$\Sigma_0^{MM} = \frac{1}{N_2}\sum_{i,y_i=0}(c_i-\hat{\mu}_0)(c_i-\hat{\mu}_0)^T = \hat{\Sigma}_0$$

II.6 Question (9)

La fonction *computeML* se charge de calculer les estimateurs déterminés à la question 5. Nous avons ensuite comparé les résultats obtenus avec ceux de la fonction *qda* du package *MASS*. On oberve uniquement une légère différence sur les valeurs des log de déterminant. Ces erreurs viennent probablement d'approximations numériques et de la façon de calculer les déterminants.

2. Pour une preuve rigoureuse de ce résultat voir *Probability with Martingales* de David Williams, page 88 : proposition (k) "Role of Independence"

```r
load("Ctrain.RData")
load("Ctest.RData")

computeML <- function(C, Y){
  n = length(Y)
  N1 = sum(Y==1)
  C0 = C[Y==0,]
  C1 = C[Y==1,]

  p_hat = N1/n
  mu_hat0 = colMeans(C0)
  mu_hat1 = colMeans(C1)
  C0_centered = sweep(C0,2,mu_hat0) #susbtract mu_hat0 to C0
  C1_centered = sweep(C1,2,mu_hat1)
  sigma_hat0 = t(C0_centered)%*%C0_centered/(n-N1)
  sigma_hat1 = t(C1_centered)%*%C1_centered/N1

  out = list(p_hat,mu_hat0,mu_hat1,sigma_hat0,sigma_hat1)

  return(out)
}

ML = computeML(Ctrain,Ytrain)

#QDA from MASS package
qda.model = qda(Ctrain,Ytrain)
```

```
## Estimation of p :  0.4984127
## QDA estimation of p :  0.4984127
## Estimation of log(det(sigma0)) :  102.7835
## QDA estimation of log(det(sigma0)) :  102.8787
## Estimation of log(det(sigma1)) :  105.5119
## QDA estimation of log(det(sigma1)) :  105.6077
## Estimation of mu0[1:4] :  51.136 4.886618 0.9851159 10.69928
## QDA estimation of mu0[1:4] :  51.136 4.886618 0.9851159 10.69928
## Estimation of mu1[1:4] :  -52.21073 -2.780416 0.3652852 -11.93721
## QDA estimation of mu1[1:4] :  -52.21073 -2.780416 0.3652852 -11.93721
```

II.7 Question (10)

Soit $j \in \{0,1\}$, d'après la formule de Bayes (appliquée deux fois),

$$\mathbb{P}(Y=j|c) = \frac{f_{C,Y}(c,j)}{f_C(c)} = \frac{\mathbb{P}(Y=j)f_{C|Y=j}(c)}{f_C(c)}$$

Comme $C|Y=j \sim N(\mu_j, \Sigma_j)$ et $\mathbb{P}(Y=j) = p^j(1-p)^{1-j}$ et

$$f_C(c) = \int f_{C,Y}(c,j)d_\mu(j)) = f_{C|Y=0}(c)\mathbb{P}(Y=0)+f_{C|Y=1}(c)\mathbb{P}(Y=1) = (1-p)\varphi(c;\mu_0,\Sigma_0)+p\varphi(c;\mu_1,\Sigma_1)$$

avec μ la mesure de comptage, on obtient :

$$\mathbb{P}(Y=1|c) = \frac{p\varphi(c;\mu_1,\Sigma_1)}{(1-p)\varphi(c;\mu_0,\Sigma_0)+p\varphi(c;\mu_1,\Sigma_1)}$$

$$\mathbb{P}(Y=0|c) = \frac{(1-p)\varphi(c;\mu_0,\Sigma_0)}{(1-p)\varphi(c;\mu_0,\Sigma_0)+p\varphi(c;\mu_1,\Sigma_1)}$$

II.8 Question (11)

On observe que le terme de normalisation $m = (1-p)\varphi(c;\mu_0,\Sigma_0)+p\varphi(c;\mu_1,\Sigma_1)$ va se simplifier lorqu'on fait le ratio des deux quantités :

$$\frac{\mathbb{P}(Y=1|c)}{\mathbb{P}(Y=0|c)} = \frac{p\varphi(c;\mu_1,\Sigma_1)}{(1-p)\varphi(c;\mu_0,\Sigma_0)}$$

En passant en log on obtient :

$$\begin{aligned}
\log\left\{\tfrac{\mathbb{P}(Y=1|c)}{\mathbb{P}(Y=0|c)}\right\} &= \log(p) - \log(1-p) + \log(\varphi(c;\mu_1,\Sigma_1)) - \log(\varphi(c;\mu_0,\Sigma_0)) \\
&= \log(p) - \log(1-p) - \tfrac{1}{2}\log(\det(\Sigma_1)) + \tfrac{1}{2}\log(\det(\Sigma_0)) \\
&\quad -\tfrac{1}{2}(c-\mu_1)^T\Sigma_1^{-1}(c-\mu_1) + \tfrac{1}{2}(c-\mu_0)^T\Sigma_0^{-1}(c-\mu_0) + \tfrac{15}{2}log(2\pi) - \tfrac{15}{2}log(2\pi) \\
&= -\tfrac{1}{2}\log(\det(\Sigma_1)) - \tfrac{1}{2}(c-\mu_1)^T\Sigma_1^{-1}(c-\mu_1) + \log(p) \\
&\quad +\tfrac{1}{2}\log(\det(\Sigma_0)) + \tfrac{1}{2}(c-\mu_0)^T\Sigma_0^{-1}(c-\mu_0) - \log(1-p)
\end{aligned}$$

On obtient bien le résultat souhaité.

II.9 Question (12)

Soit A l'évènement tel que :

$$A = \left\{\log\left(\frac{\mathbb{P}(Y=1|c)}{\mathbb{P}(Y=0|c)}\right) > 0\right\} = \left\{\frac{\mathbb{P}(Y=1|c)}{\mathbb{P}(Y=0|c)} > 1\right\} = \left\{\mathbb{P}(Y=1|c) > \mathbb{P}(Y=0|c)\right\}$$

Donc si $\mathbb{1}_A = 1$, A est réalisé et $\arg\max_{y\in\{0,1\}} \mathbb{P}(Y=y|c) = 1$ et si $\mathbb{1}_A = 0$ alors A^c est réalisé et $\arg\max_{y\in\{0,1\}} \mathbb{P}(Y=y|c) = 0$. Dans tous les cas nous avons bien :

$$\mathbb{1}_{\{\log(\frac{\mathbb{P}(Y=1|c)}{\mathbb{P}(Y=0|c)})>0\}} = \arg\max_{y\in\{0,1\}} \mathbb{P}(Y=y|c)$$

Remarque. *La règle de décision qui en dècoule est bien une fonction* ***quadratique*** *de l'observation c ce qui a donné son nom à la méthode.*

II.10 Question (13)

Soit $x, y \in \{0,1\}$, on a alors quatre cas possibles :
— $x = 0, y = 0$ donne $(x-y)^2 = |x-y| = \mathbb{1}_{x\neq y} = 0$
— $x = 1, y = 1$ donne $(x-y)^2 = |x-y| = \mathbb{1}_{x\neq y} = 0$
— $x = 0, y = 1$ donne $(x-y)^2 = |x-y| = \mathbb{1}_{x\neq y} = 1$
— $x = 1, y = 0$ donne $(x-y)^2 = |x-y| = \mathbb{1}_{x\neq y} = 1$

Cela implique que si X et Y sont deux variables aleatoires à valeur dans $\{0,1\}$ alors

$$(X-Y)^2 = |X-Y| = \mathbb{1}_{X=Y} p.s$$

Comme $\mathbb{E}[\mathbb{1}_{y=\hat{y}}] = \mathbb{P}(y=\hat{y})$ et $(\hat{y}, y) \in \{0,1\}^2$, en passant à l'espérance on obtient bien

$$\mathbb{E}[(y-\hat{y})^2] = \mathbb{E}[|y-\hat{y}|] = \mathbb{P}(y=\hat{y})$$

II.11 Question (14)

La fonction *computeLogRatio* prend en entrée une observation (un vecteur de $\mathbb{R}^{15}$) et les statistiques de la question 5 afin de calculer le log ratio associé à l'observation.

La fonction *computePred* prend en entrée une matrice d'observations et effectue pour chaque observation une prédiction selon la règle de décision de la question 12 : le label prédit est 1 si le log ratio est positif et 0 sinon.

```
computeLogRatio <- function(cvect,p,mu0,mu1,Sigma0,Sigma1) {

  logratio = (0.5*(-log(det(Sigma1)) + log(det(Sigma0))
                   - t(cvect-mu1)%*%ginv(Sigma1)%*%(cvect-mu1)
                   + t(cvect-mu0)%*%ginv(Sigma0)%*%(cvect-mu0))
              + log(p) - log(1-p))

  return(logratio)
}

computePred <- function(C,p,mu0,mu1,Sigma0,Sigma1) {

  toapply <- function(cvect,p,mu0,mu1,Sigma0,Sigma1) {
    return(as.integer((computeLogRatio(cvect,p,mu0,mu1,Sigma0,Sigma1)>0)))
  }

  pred = apply(C, MARGIN = 1, FUN = toapply, p = p, mu0 = mu0,
               mu1 = mu1, Sigma0 = Sigma0, Sigma1 = Sigma1)

  return(pred)
}
```

II.12 Question (15)

À l'aide des fonctions des questions précédentes, les prédictions se font aisément comme le montre les deux lignes de codes ci-dessous. L'erreur sur le test set est de 12,5%.

```
stats = computeML(Ctrain,Ytrain)
prediction = computePred(Ctest,stats[[1]],stats[[2]],stats[[3]],stats[[4]],stats[[5]])

prediction

##  [1] 1 1 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 0 0
## [36] 0 0 0 0 0 0 0 0 0 0 0 1 0

#Erreur de prediction :
sum(prediction==Ytest)/length(Ytest)

## [1] 0.875
```

Nous comparons ces résultats avec ceux fournies par la fonction *qda* :

```
computeQDA <- function(Ctrain, Ctest, Ytrain, Ytest){
  qda.model = qda(Ctrain,Ytrain)
  pred = predict(qda.model, Ctest)
```

```
  print(pred$class)
  return(sum(pred$class==Ytest)/length(Ytest))
}

computeQDA(Ctrain, Ctest, Ytrain, Ytest)

##  [1] 1 1 1 1 1 1 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 0 0 1 0 0 0 0 0 0 0 0
## [36] 0 0 0 0 0 0 0 0 0 0 0 1 0
## Levels: 0 1
## [1] 0.875
```

Les prédictions sont exactement les mêmes et donc le pourcentage d'erreur aussi. Comme vu question 9 ce résultat était attendu car seuls d'infimes écarts surgissaient entre les estimations des log de déterminant.

Il y a environ 50% de chats et 50% de chiens dans le test set donc si on considère le prédicteur aléatoire qui consisterait à tirer à pile ou face chaque prédiction, il ne ferait pas mieux que 50%. Nous atteignons ici un précision de 87.5% ce qui est beaucoup plus raisonnable.

II.13 Bonus, Analyse Discriminante Linéaire

Dans *The Elements of Statistical Learning* est abordée l'Analyse Discriminante Linéaire, la seule différence avec l'Analyse Discriminante Quadratique est que l'on fait une hypothèse d'**homoscédacticité** i.e. les deux groupes ont la même variance Σ (mais toujours des moyennes μ_0 et μ_1 différentes). Les estimateurs de $\hat{p}$, $\hat{\mu}_0$ et $\hat{\mu}_1$ restent les mêmes et celui de Σ est donné par :

$$\hat{\Sigma} = \frac{\sum_{i,y_i=0}(c_i - \hat{\mu}_0)(c_i - \hat{\mu}_0)^T + \sum_{i,y_i=1}(c_i - \hat{\mu}_1)(c_i - \hat{\mu}_1)^T}{n-2}$$

Le log ratio prend alors une forme légèrement différente :

$$\log\left(\frac{\mathbb{P}(Y=1|c)}{\mathbb{P}(Y=0|c)}\right) = \log(\frac{\hat{p}}{1-\hat{p}}) - \frac{1}{2}(\hat{\mu}_1 + \hat{\mu}_0)^T\hat{\Sigma}^{-1}(\hat{\mu}_1 - \hat{\mu}_0) + c^T\hat{\Sigma}^{-1}(\hat{\mu}_1 - \hat{\mu}_0)$$

La règle de décision est ensuite la même, le label prédit est 1 si cette quantité est positive et 0 sinon. On observe que la règle de décision est maintenant **linéaire** en l'observation c et non **quadratique** comme dans les questions précédentes (d'où la différence de nom entre les deux méthodes). À l'aide des fonctions ci-dessous nous conduisons une prédiction sur la base de test avec deux algorithmes d'analyse discriminante linéaire : notre algorithme "maison" et l'algorithme *lda* du package *MASS*.

```
computeMLlin <- function(C, Y){
  #Calcul les statistiques associees a la LDA
  n = length(Y)
  N1 = sum(Y==1)
  C0 = C[Y==0,]
  C1 = C[Y==1,]
  p_hat = N1/n
  mu_hat0 = colMeans(C0)
  mu_hat1 = colMeans(C1)
  C0_centered = sweep(C0,2,mu_hat0)
  C1_centered = sweep(C1,2,mu_hat1)
  Sigma = (t(C0_centered)%*%C0_centered + t(C1_centered)%*%C1_centered)/(n-2)

  out = list(p_hat,mu_hat0,mu_hat1,Sigma)
  return(out)
```

```
}

  computeLogRatiolin <- function(c,p,mu0,mu1, Sigma) {
    #Calcul les log ratio associees a la LDA

    logratio = (log(p) - log(1-p) - 0.5*t(mu1 + mu0)%*%ginv(Sigma)%*%(mu1 - mu0)
                + t(c)%*%ginv(Sigma)%*% (mu1-mu0))

    return(logratio)
  }

computePredlin <- function(C,p,mu0,mu1,Sigma) {

  nbobs = dim(C)[1]
  pred = rep(0,nbobs)
  for (i in 1:nbobs) {

    logratio = computeLogRatiolin(C[i,],p,mu0,mu1,Sigma)
    predi = logratio>0
    pred[i] = predi
  }

  return(pred)
}

computeLDA <- function(Ctrain, Ctest, Ytrain, Ytest){
  lda.model = lda(Ctrain,Ytrain)
  pred = predict(lda.model, Ctest)
  print(pred$class)
  return(sum(pred$class==Ytest)/length(Ytest))
}


statsbis = computeMLlin(Ctrain,Ytrain)
prediction = computePredlin(Ctest,statsbis[[1]],statsbis[[2]],statsbis[[3]],statsbis[[4]])
prediction

##  [1] 1 0 1 1 1 0 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [36] 0 0 0 1 0 0 1 0 0 0 0 0 0

sum(prediction==Ytest)/length(Ytest)

## [1] 0.8125

computeLDA(Ctrain, Ctest, Ytrain, Ytest)

##  [1] 1 0 1 1 1 0 1 1 1 1 0 1 0 0 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [36] 0 0 0 1 0 0 1 0 0 0 0 0 0
## Levels: 0 1
## [1] 0.8125
```

Comme pour l'Analyse Discriminante Quadratique ici nos résultats sont exactement les mêmes que ceux de la fonction *lda*. Les hypothèses de l'Analyse Discriminante Linéaire sont plus fortes et on observe une augmentation du taux d'erreur, on passe de 12,5% d'erreur à 18.75% d'erreur.