

TP no 5 - modèles probabilistes de classification

Master parcours SSD - UE Apprentissage Statistique I

1 Exercice 1 - classifieur de Bayes et lois normales (univariées)

L'objectif de cet exercice est d'appréhender les notions de probabilité a posteriori et de classifieur de Bayes. Pour cela, on considère un problème de classification à deux classes mettant en jeu une seule variable, distribuée normalement au sein de chacune des classes : $P(x|C_0) = \mathcal{N}(\mu_0, \sigma_0)$ et $P(x|C_1) = \mathcal{N}(\mu_1, \sigma_1)$. L'objectif sera d'expliquer et de visualiser le classifieur de Bayes, et de le comparer aux classifieurs que l'on obtient par les approches génératives (LDA et QDA) vues en cours.

1. On considère les paramètres suivants : $(\mu_0 = 3, \sigma_0 = 1)$ et $(\mu_1 = 7, \sigma_1 = 1)$. Représenter les densités théoriques¹ correspondantes pour $x \in [-10, 10]$.
2. Représenter les probabilités a posteriori $P(C_0|x)$ et $P(C_1|x)$ que l'on obtient en considérant un a priori uniforme (i.e., $P(C_0) = P(C_1) = 0.5$). En déduire le classifieur de Bayes.
3. Générer un échantillon² de taille 50 selon chacune des lois $P(x|C_0)$ et $P(x|C_1)$.
4. Représenter les probabilités a posteriori que l'on obtient à partir de ces deux échantillons par les approches LDA et QDA. En déduire les classifieurs correspondants, et comparer au classifieur de Bayes.
 - La distinction entre LDA et QDA peut sembler artificielle en présence d'une seule variable. En pratique, cela veut dire qu'avec QDA on considère une variance par classe (i.e., $\hat{\sigma}_1^2$ et $\hat{\sigma}_0^2$), alors qu'avec LDA on ne considère qu'une seule variance. On prendra la définition de la variance "poolée", ce qui revient ici à $\hat{\sigma}^2 = 0.5(\hat{\sigma}_0^2 + \hat{\sigma}_1^2)$ car les effectifs sont les mêmes dans les deux classes.
5. Reproduire cette analyse pour les paramètres suivants : $(\mu_0 = 3, \sigma_0 = 1)$ et $(\mu_1 = 7, \sigma_1 = 3)$. Commenter.

2 Exercice 2 : LDA, QDA et régression logistique

L'objectif de cet exercice est d'apprendre à manipuler les outils R permettant de mettre en oeuvre les modèles LDA, QDA et régression logistique. Pour cela nous allons comparer leurs performances de prédiction sur un jeu de données simulées en deux dimensions.

1. Charger le jeu de données `tp5-exo2.Rdata` et le visualiser.
 - il se constitue d'un jeu d'apprentissage (`X.train`, `y.train`) contenant 400 observations en 2 dimensions appartenant à deux catégories ("1" et "2"), et d'un jeu de test (`X.test`, `y.test`) contenant 200 observations.
 - on représentera sur un même graphique les données d'apprentissage et de test, avec un code couleur indiquant la catégorie ("1" ou "2"), et un symbole indiquant le jeu de données (apprentissage ou test), via l'option `pch`.
2. Mettre en oeuvre l'approche LDA :
 - (a) apprendre le modèle à partir des données d'apprentissage en utilisant la fonction `lda` du package MASS.

1. On rappelle que la fonction `dnorm` permet d'obtenir la densité de la loi normale.

2. On rappelle que la fonction `rnorm` permet d'effectuer des tirages selon une loi normale.

- (b) obtenir les prédictions sur les données de test grâce à la fonction `predict` correspondante (voir la documentation à `predict.lda`). Quelles informations fournit cette fonction ?
 - (c) représenter les probabilités a posteriori d'être dans la classe "1" en fonction des vrais classes définies par `y.test` en utilisant la fonction `boxplot`. Qu'observez-vous ?
 - (d) construire la matrice de confusion obtenue et calculer le taux de bonne classification.
3. Faire de même avec l'approche QDA. Les performances sont-elles meilleures ? Est-ce surprenant au vu du graphique construit précédemment ?
 - pour cela on s'appuie sur la fonction `qda` du package MASS qui fonctionne exactement comme la fonction `lda`.
 4. Enfin, faire de même avec la régression logistique. Comment ce modèle se positionne t'il par rapport aux approches génératives ?
 - on utilise pour cela la fonction `glm` et la fonction `predict` correspondante. Ces fonctions suivent la même syntaxe que celle de la fonction `lm` pour effectuer une régression linéaire.
 - pour construire le modèle :


```
fit = glm(y.train ~ x1+x2, data = data.frame(X.train), family = "binomial")
```
 - pour effectuer les prédictions :


```
p = predict(fit, newdata = data.frame(X.test), type = "response").
```
 - notons que le vecteur `p` contient les probabilités a posteriori d'être dans la classe "positive". La classe positive est choisie automatiquement comme le deuxième "niveau" considéré pour l'apprentissage : nous avons ici des "1" et des "2", la classe 2 est considérée comme étant la classe positive.

Pour aller plus loin :

1. Comparer les résultats obtenus par les trois approches en terme de courbe ROC.
2. Appliquer une procédure de rejet sur les prédictions fournies par la régression logistique en ne conservant que les prédictions pour lesquelles la probabilité a posteriori renvoyée par le modèle est supérieure à 0.6 ou inférieure à 0.4. Quelle est la proportion de prédictions rejetées ? Quel taux de bonne classification obtient-on sur les prédictions effectivement réalisées ?
3. Représenter la frontière définie par la régression logistique sur la figure générée à la Question 1.
 - On rappelle que la frontière est donnée par l'équation $f(x) = 0$, où $f(x)$ est la fonction linéaire définie par les coefficients du modèle linéaire, soit ici $f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$.
 - Pour tracer la frontière on peut donc utiliser la fonction `abline` qui permet de faire figurer sur un graphique existant une droite d'équation $y = a + bx$ à partir des paramètres a et b .
 - Il faut donc trouver l'expression $x_2 = a + bx_1$ à partir des coefficients $(\beta_0, \beta_1, \beta_2)$ de la régression logistique.