SY09 - TP3 Discrimination, théorie bayésienne de la décision

Pierre Bathellier et Vançon Virgile

Université de Technologie de Compiègne

Table des matières

Introduction

Lors du TP précédent, nous avons utilisé différentes techniques de classification non supervisée. Nous allons désormais nous intéresser aux méthodes de classification ou apprentissage supervisées. Le but de ces méthodes est de construire des systèmes de prédiction permettant de classer de nouveaux individus, à partir d'une règle de décision mise en place grâce à des observations sur un ensemble d'apprentissage, dont les caractéristiques sont connues. Dans le premier exercice, nous comparerons deux classifieurs : le classifieur euclidien et le classifieur des K plus proches voisins. Dans le deuxième exercice, nous nous intéresserons à la théorie bayésienne de la décision, qui permet de trouver une règle optimale selon un critère spécifique.

1 Classifieur euclidien, K plus proches voisins

1.1 Programmation

L'objectif du rapport étant d'aller à l'essentiel, le code des fonctions de classification selon les classifieurs euclidien et KPPV est fourni en annexe à la fin de ce document.

Nos fonctions ont l'air fonctionnels dans la mesure où l'on retrouve Figure 1 et 2 les frontières de décision obtenues en utilisant toutes les données pour l'apprentissage comme présenté dans l'énoncé.

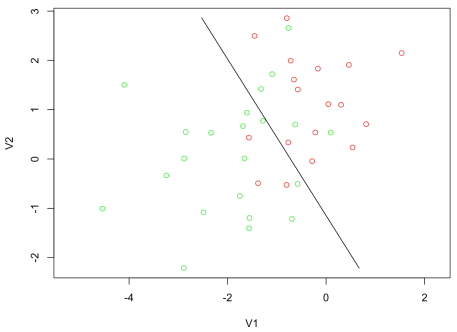

Figure 1. Classification obtenue avec le classifieur euclidien en prenant toutes les données pour l'apprentissage.

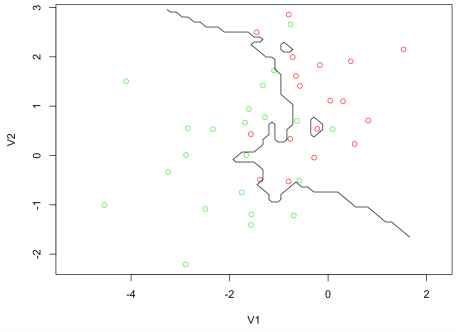

Figure 2. Classification obtenue avec le classifieur KPPV en prenant toutes les données pour l'apprentissage.

1.2 Évaluation des performances

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

Tous les jeux de données possèdent la même répartition qui ne different que par le nombre d'individus. Les classes ont été générées selon une loi normale bivariée.

1.2.1.1 Classifieur Euclidien - Estimation des paramètres μ_k, Σ_k et π_k des classes.

Si un échantillon $X_1, \ldots, X_n$ est un échantillon *iid* de vecteurs aléatoire parents $X \sim \mathcal{N}(\mu, \Sigma)$, les estimateurs $\hat{\mu}$ et $\hat{\Sigma}$ sont respectivement le vecteur moyenne empirique $\overline{X}$ et la matrice de variance empirique V (et on a dans ce cas $\hat{\mu} \sim \mathcal{N}(\mu, \frac{1}{n}\Sigma)$).

Table 1. Estimation des paramètres μ_k, Σ_k et π_k des distribution conditionnelles de chaque jeu de données.

	Synth1-40	Synth1-100	Synth1-500	Synth1-1000
μ_1	$\begin{pmatrix} -0.317 \\ 1.092 \end{pmatrix}$	$\begin{pmatrix} 0.026 \\ 0.815 \end{pmatrix}$	$\begin{pmatrix} 0.132 \\ 0.880 \end{pmatrix}$	$\begin{pmatrix} -0.013 \\ 0.917 \end{pmatrix}$
μ_2	$\begin{pmatrix} -1.883 \\ 0.105 \end{pmatrix}$	$\begin{pmatrix} -1.965 \\ -0.127 \end{pmatrix}$	$\begin{pmatrix} -1.883 \\ -0.085 \end{pmatrix}$	$\begin{pmatrix} -1.961 \\ 0.018 \end{pmatrix}$
Σ_1	$\begin{pmatrix} 0.682 & 0.119 \\ 0.119 & 1.013 \end{pmatrix}$	$\begin{pmatrix} 0.882 & -0.131 \\ -0.131 & 1.109 \end{pmatrix}$	$\begin{pmatrix} 1.050 & 0.052 \\ 0.052 & 0.983 \end{pmatrix}$	$\begin{pmatrix} 0.969 & -0.065 \\ -0.065 & 1.079 \end{pmatrix}$
Σ_2	$\begin{pmatrix} 1.375 & 0.323 \\ 0.323 & 1.439 \end{pmatrix}$	$\begin{pmatrix} 0.757 & -0.038 \\ -0.038 & 0.761 \end{pmatrix}$	$\begin{pmatrix} 0.967 & -0.111 \\ -0.111 & 0.979 \end{pmatrix}$	$\begin{pmatrix} 0.990 & 0.021 \\ 0.021 & 0.941 \end{pmatrix}$
π_1	0.450	0.540	0.528	0.496
π_2	0.55	0.46	0.472	0.504

1.2.1.2 Classifieur Euclidien - Répétition.

Nous allons répéter 20 fois la classification - via le classifieur euclidien - des différents jeux de données mis à notre disposition. A chaque séparation i, nous allons pouvoir déterminer une estimation du taux d'erreur E_i. La moyenne de ces taux d'erreur est un estimateur de ϵ.

Intervalle de confiance à 95%
Si l'on répète la classification un très grand nombre de fois (100000 fois par exemple) pour un certain ensemble de données (on prend ici Synth1-500 qui apparaît comme être l'ensemble intermédiaire) et que l'on trace l'histogramme des fréquences d'apparition des différents taux d'erreur on obtient la figure 3. On constate donc visuellement que le taux d'erreur ϵ suit la loi normale suivante :

$$\epsilon \sim \mathcal{N}(\mu, \sigma^2)$$

Ici μ, σ^2 sont inconnus mais peuvent être estimer respectivement par $\overline{X}$ qui est la moyenne empirique et par $S*^2$, qui est la variance empirique corrigé.

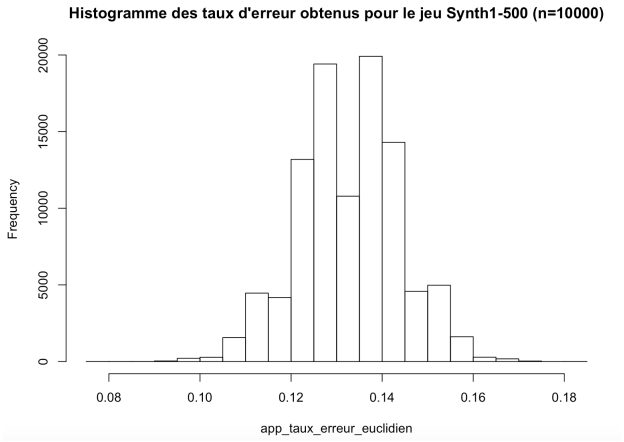

Figure 3. Répartition des taux d'erreur ϵ dans la classification du jeu de données Synth1-500 (10000 répétitions).

La formule de l'intervalle de confiance est alors :

$$IC = \left[\overline{X} - t_{n-1;1-\frac{\alpha}{2}} \frac{S*}{\sqrt{n}}; \overline{X} + t_{n-1;1-\frac{\alpha}{2}} \frac{S*}{\sqrt{n}}\right]$$

Soit $t_{n-1;1-\frac{\alpha}{2}}$ le fractile de la loi de Student à $n-1$ dégrés de liberté. On veut un intervalle ded confiance à 95% de confiance, et on prend donc $\alpha = 5$. Il est demandé de tester le classifieur 20 fois on a donc $20 - 1 = 19$ d.d.l..

D'après la table de Student $t_{19;0.975} = 2,0930$.

Le tableau récapitulatif des taux d'erreur et de leurs intervalles de confiance du classifieur euclidien sur les ensembles d'apprentissage et de test est présenté Table 2

Table 2. Resultats taux d'erreur et intervalles de confiance pour 20 séparations des données avec le classifieur euclidien

	Synth1-40	Synth1-100	Synth1-500	Synth1-1000
ϵ_{app}	0.198	0.081	0.133	0.142
IC_{app}	[0.175 ;0.220]	[0.071 ;0.092]	[0.128 ;0.138]	[0.138 ;0.146]
ϵ_{test}	0.248	0.107	0.132	0.142
IC_{test}	[0.198 ;0.298]	[0.087 ;0.127]	[0.122 ;0.142]	[0.135 ;0.150]

Observation : Les taux d'erreur de l'ensemble d'apprentissage et de test tendent à diminuer entre les jeux de données 40 et 100 puis augmentent pour les jeux de données 500 et 1000. Un résultat notable est que les intervalles de confiance se resserrent quand on augmente la taille des jeux de données testés.

Interprétation : L'évolution des taux d'erreur peut paraître assez surprenant. Ce résultat ne semble cependant pas anormal dans la mesure où les intervalles de confiance ne se chevauchent pas. Une explication pourrait être que lors de sa génération, l'ensemble Synth1-100 a généré des points dont la classification est plus facile. Si Synth1-100 est vu comme un tirage exceptionnel, la tendance du taux d'erreur serait donc de diminuer lorsqu'on augmente le nombre de données.

1.2.1.3 Nombre de voisin Optimal en prenant l'ensemble d'apprentissage en ensemble de validation.

Lorsque l'on prend comme ensemble de validation l'ensemble d'apprentissage on obtient le nombre de voisin optimal suivant :

$$Kopt = 1$$

Ce résultat n'est pas surprenant, car la fonction de calcul du nombre idéal de voisin test la fonction kppv.val pour différente valeurs de k, et retient le k pour lequel le taux d'erreur de la fonction de classification est le plus bas. Dans le cas ou k = 1 la fonction va attribuer comme classe à l'ensemble testé le voisin le plus proche.

Si l'on prend le même ensemble pour l'apprentissage et l'ensemble de validation le voisin le plus proche de chaque point sera le point lui même (ce point faisant à la fois parti de l'ensemble d'apprentissage et de test). Ainsi pour chaque point l'étiquette qui lui sera attribué sera sa propre étiquette. On obtient donc dans ce cas un taux d'erreur $\epsilon = 0$.

1.2.1.2 Classifieur KPPV - Répétition.

Encore une fois nous allons classifier 20 fois chaque jeu de données, cette fois-ci avec le classifieur KPPV. Pour chaque classfication nous avons recours à kppv.tune pour déterminer Kopt.

Le tableau récapitulatif des taux d'erreur et de leurs intervalles de confiance du classifieur KPPV sur les ensembles d'apprentissage et de test est présenté Table 3.

Table 3. Resultats taux d'erreur après 20 séparations des données pour le classifieur "K plus proches voisins"

	Synth1-40	Synth1-100	Synth1-500	Synth1-1000
ϵ_{app}	0.165	0.051	0.126	0.128
IC_{app}	[0.143 ;0.187]	[0.041 ;0.061]	[0.121 ;0.131]	[0.124 ;0.132]
ϵ_{test}	0.160	0.088	0.126	0.147
IC_{test}	[0.110 ;0.210]	[0.066 ;0.107]	[0.116 ;0.136]	[0.139 ;0.154]

Observation : De manière semblable à la classification "euclidienne", les taux d'erreur de l'ensemble d'apprentissage et de test tendent à diminuer entre les jeux de données 40 et 100 puis augmentent pour les jeux de données 500 et 1000. A nouveau, les intervalles de confiance se resserrent quand on augmente la taille des jeux de données testés.

Interprétation : Les variations du taux d'erreur lors de la classification avec le critère KPPV sont donc semblables à celles obtenues avec le classifieur euclidien. Toutefois, les taux d'erreur semblent meilleurs, à part pour Synth1-1000. On peut donc estimer que sur ces jeux de données le classifieur KPPV est plus performant. Cette performance peut néanmoins être nuancée par le fait que l'algorithme de classification selon les KPPV a une complexité plus grande, et prend davantage de temps à s'exécuter.

1.2.2 Jeu de données Synth2-1000

Contrairement aux jeux de données Synth1-50, Synth1-100, Synth1-500, Synth1-1000 la distribution du jeu de données Synth2-1000 n'est pas la même ; ces données ont été générés selon d'autres paramètres. En revanche la détermination de ces paramètres se fait de la même manière que dans la partie 1.2.2.1

Table 4. Estimation des paramètres μ_k, Σ_k et π_k des distribution conditionnelles du jeu de données Synth2-1000.

	μ_1	μ_2	Σ_1	Σ_2	π_1	π_2
Synth2-1000	$\begin{pmatrix} 3.018 \\ -0.006 \end{pmatrix}$	$\begin{pmatrix} -2.142 \\ -0.03 \end{pmatrix}$	$\begin{pmatrix} 0.990 & 0.119 \\ 0.113 & 1.093 \end{pmatrix}$	$\begin{pmatrix} 4.435 & -0.154 \\ -0.154 & 1.031 \end{pmatrix}$	0.523	0.477

On va déterminer maintenant les taux d'erreur et leurs intervalles de confiance avec le classifieur euclidien et KPPV.

Taux erreur - Classifieur euclidien

Table 5. Resultats taux d'erreur et intervalles de confiance pour 20 séparations des données avec le classifieur euclidien

	ϵ_{app}	IC_{app}	ϵ_{test}	IC_{test}
Synth2-1000	0.063	[0.061 ;0.065]	0.062	[0.057 ;0.068]

Taux erreur - Classifieur KPPV

Table 6. Resultats taux d'erreur et intervalles de confiance pour 20 séparations des données avec le classifieur KPPV

	ϵ_{app}	IC_{app}	ϵ_{test}	IC_{test}
Synth2-1000	0.043	[0.034 ;0.053]	0.056	[0.049 ;0.063]

Analyse et Interprétation

Les taux d'erreur trouvés pour les deux classifieurs sur ce jeu de données sont bien bien inférieurs à ceux obtenues pour Synth-1000 (Tous les taux d'erreur sont ici inférieurs à 7% contre entre 12 et 15% pour Synth-1000). De même les intervalles de confiance sont davantage resserrés. Notons qu'encore une fois le classifieur KPPV semble plus précis.

Si l'on compare graphiquement les répartitions des jeux Synt-1000 et Synth2-1000 (Figure 4 et 5), on constate que la distinction entre les individus du groupe 1 et 2 est plus prononcée, particulièrement à la frontière entre les deux groupes. La distance euclidienne entre les deux centres de gravité est plus élevée. Cela explique que le classifieur euclidien et KPPV aient davantage de facilités à différencier les individus des deux classes.

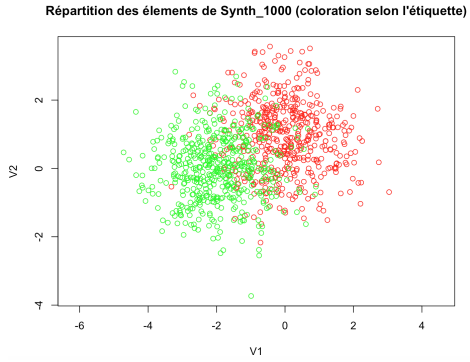

Figure 4.

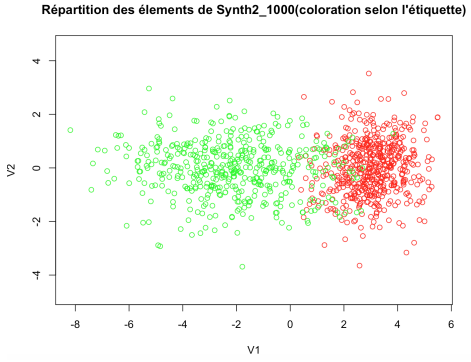

Figure 5.

1.2.3 Jeux de données réelles

On cherche maintenant à classifier des jeux de données réelles. Les deux jeux de données que l'on va classifier sont les jeux Pima et BreastCancer.

Jeu de données Pima

Table 7 et 8 les taux d'erreurs et leurs intervalles de confiance avec le classifieur euclidien et KPPV sur le jeu de données Pima.

Table 7. Resultats taux d'erreur et intervalles de confiance pour 20 séparations des données Pima avec le classifieur euclidien

	ϵ_{app}	IC_{app}	ϵ_{test}	IC_{test}
Pima	0.246	[0.237 ;0.255]	0.242	[0.225 ;0.258]

Table 8. Resultats taux d'erreur et intervalles de confiance pour 20 séparations des données Pima avec le classifieur KPPV

	ϵ_{app}	IC_{app}	ϵ_{test}	IC_{test}
Pima	0.181	[0.169 ;0.193]	0.221	[0.206 ;0.238]

Jeu de données BreastCancer

Table 9 et 10 les taux d'erreurs et leurs intervalles de confiance avec le classifieur euclidien et KPPV sur le jeu de données BreastCancer.

Table 9. Resultats taux d'erreur et intervalles de confiance pour 20 séparations des données BreastCancer avec le classifieur euclidien

	ϵ_{app}	IC_{app}	ϵ_{test}	IC_{test}
BreastCancer	0.040	[0.037 ;0.042]	0.037	[0.032 ;0.043]

Table 10. Resultats taux d'erreur et intervalles de confiance pour 20 séparations des données BreastCancer avec le classifieur KPPV

	ϵ_{app}	IC_{app}	ϵ_{test}	IC_{test}
BreastCancer	0.038	[0.035 ;0.040]	0.042	[0.036 ;0.048]

Analyse et Interprétation

Encore une fois le classifieur KPPV semble plus précis que celui euclidien et ceux dans pour les jeux pima et breastcancer. La classification du jeu BreastCancer est elle plus précise que celle du jeu pima. Si l'on représente ces jeux de données dans le premier plan factoriel de l'ACP (Figure 6 et 7) on peut se rendre compte que les individus Breastcancer sont beaucoup plus regroupés que peuvent l'être ceux du jeu pima. Il paraît donc logique que les classifieurs soient plus efficace dans les cas où chaque classe est clairement distinguable.

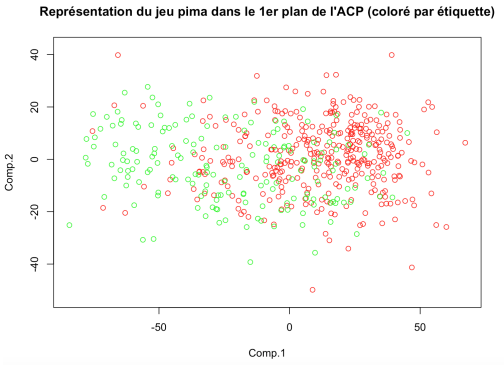

Figure 6.

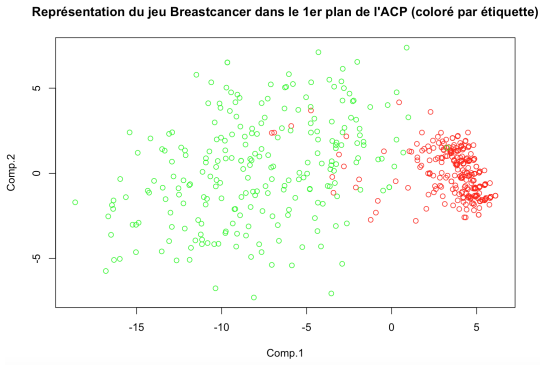

Figure 7.

2 Bayes

2.1 Distributions marginales

Une loi normale est une loi de probabilité absolument continue qui dépend de deux paramètres : son espérance, un nombre réel noté μ, et son écart type, un nombre réel positif noté σ. Une loi normale mutlivariée est une

généralisation multidimensionnelle de la loi normale. Une loi normale multivariée est paramétrée par un vecteur μ représentant son centre et une matrice semi-définie positive Σ qui est sa matrice de variance-covariance.

La classe ω_1 de notre tableau individus-variables suit une loi une loi normale bivariée de paramètres μ_1 et σ_1, et la classe ω_2 suit une loi une loi normale bivariée de paramètres μ_2 et σ_2, sachant que n_1 a été déterminé par tirage aléatoire suivant une loi binomiale de paramètres n et $\pi_1 = 0.5$ et $n_2 = n - n_1$.
Or, tout sous-vecteur du vecteur aléatoire X, c'est à dire tout sous-ensemble de l'ensemble des variables aléatoires $X_1, ..., X_p$ est lui-même un vecteur aléatoire dont la loi est la loi marginale obtenue de la manière suivante :
Imaginons une variable X suivant une loi normale multivariée telle que $X \sim \mathcal{N}(\mu, \Sigma)$, avec :

$$X = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix},\ \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}\ et\ \Sigma = \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$

Alors, la loi marginale de x_1 est :

$$x_1 \sim \mathcal{N}(\mu_1, \Sigma_{11})$$

Rappelons que dans notre cas, pour chaque jeux de données Synth1, les paramètres de la loi normale multivariée sont :

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

On en déduit que les distributions marginales des variales X^1 et X^2 sont :
Pour Synth1 :
- Classe 1 : $X_1 \sim \mathcal{N}(0,1)$ et $X_2 \sim \mathcal{N}(1,1)$
- Classe 2 : $X_1 \sim \mathcal{N}(-2,1)$ et $X_2 \sim \mathcal{N}(0,1)$

Pour Synth2 :
- Classe 1 : $X_1 \sim \mathcal{N}(3,1)$ et $X_2 \sim \mathcal{N}(0,1)$
- Classe 2 : $X_1 \sim \mathcal{N}(-2,5)$ et $X_2 \sim \mathcal{N}(0,1)$

2.2 Courbes d'isodensité

Les courbes d'isodensité d'un loi normale multivariée ont pour équation $c = (x-\mu)^T \Sigma^{-1}(x-\mu)$, où c est une constante. Ce sont des ellipsoïdes de centre μ, et lorsque la matrice Σ est diagonale (ce qui est le cas ici pour les jeux Synth1), alors les axes de ces ellipsoïdes sont parallèles aux axes de coordonnées.
En appliquant cette formule à nos donnéeés, on obtient les deux équation suivantes :

$$Classe1 : c = (x-\mu_1)^T \Sigma_1^{-1}(x-\mu_1)$$
$$Classe2 : c = (x-\mu_2)^T \Sigma_2^{-1}(x-\mu_2)$$

Après application numérique, on obtient les résultats récapitulés dans le tableau suivant :

	Classe 1	Classe 2
$Synth1$	$c = x_1^2 + (x_2-1)^2$ cercle de centre μ_1 et de rayon $\sqrt{c}$	$c = (x_1+2)^2 + x_2^2$ cercle de centre μ_2 et de rayon $\sqrt{c}$
$Synth2$	$c = (x_1-3)^2 + x_2^2$ cercle de centre μ_1 et de rayon $\sqrt{c}$	$c = 5(x_1+2)^2 + x_2^2 \iff 1 = (\sqrt{\frac{5}{c}}(x_1+2))^2 + (\frac{x_2}{\sqrt{c}})^2$ ellipsoïde de centre μ_2 grand rayon $\sqrt{c}$ et de petit rayon $\sqrt{\frac{c}{5}}$

Table 11. Expressions des courbes d'isodensité

Les courbes d'isodensité pour la classe 1 des jeux Synth1 et Synth2 sont des cercles de centres μ_1 et de rayons $\sqrt{c}$, et la courbe d'isodensité de la classe 2 pour le jeu Synth1 est un cercle de centre μ_2 et de rayons $\sqrt{c}$. En revanche la courbe de la classe 2 pour le jeu Synth2 est une ellipsoïde de centre μ_2 grand rayon $\sqrt{c}$ et de petit rayon $\sqrt{\frac{c}{5}}$, car la matrice de variance n'est pas la matrice identité I_2 (ses axes sont en revenche bien parallèles aux axes des coordonnées).

2.2 Règle de Bayes, jeux Synth1-n

En apprentissage supervisé, la règle de Bayes est une règle qui consiste à affecter chaque individu x à la classe de plus grande probabilité *a posteriori*. Contrairement à la probabilité *a priori*, qui est la proportion π_k de la classe ω_k, la probabilité a posteriori est la quantité $\mathbb{P}(Z = \omega_k | X = x)$. Cette règle a pour but de minimiser la probabilité d'erreur de Bayes $\varepsilon(\delta|x)$ pour tout x.
La décision par la règle δ peut prendre deux valeurs possibles :

$$\delta^*(x) = \begin{cases} a_1 \text{ si } \mathbb{P}(\omega_2|x) < \mathbb{P}(\omega_1|x) \\ a_2 \; sinon \end{cases}$$

Il est possible d'exprimer la règle de Bayes en fonction du rapport de vraisemblance $\frac{f_1(x)}{f_2(x)}$:

$$\delta^*(x) = a1 \iff \mathbb{P}(\omega_1|x) > \mathbb{P}(\omega_2|x)$$

$$\iff \frac{f_1(x)\pi_1}{f(x)} > \frac{f_2(x)\pi_2}{f(x)}$$

$$\iff \frac{f_1(x)}{f_2(x)} > \frac{\pi_2}{\pi_1}$$

On en déduit :

$$\delta^*(x) = \begin{cases} a_1 \text{ si } \frac{f_1(x)}{f_2(x)} > \frac{\pi_2}{\pi_1} \\ a_2 \; sinon \end{cases}$$

Les densités conditionnelles f_k ont pour expression :

$$f_k(x) = \frac{1}{(2\pi)^{\frac{p}{2}}(det\Sigma_k)^{\frac{1}{2}}} exp(-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k))$$

Pour les jeux de données Synth1-n on a : $\Sigma_1 = \Sigma_2 = \Sigma = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \pi_1 = 0.496 \; et \; \pi_2 = 0.504$. Après application numérique on obtient l'expression suivante :

$$\frac{f_1(x)}{f_2(x)} = exp(2x_1 + x_2 + \frac{3}{2}) > \frac{\pi_2}{\pi_1}$$

$$Soit : 2x_1 + x_2 > ln(\frac{\pi_2}{\pi_1}) - \frac{3}{2}$$

$$\iff 2x_1 + x_2 > k \; avec \; k = ln(\frac{\pi_2}{\pi_1}) - \frac{3}{2} = -1.483$$

Si $2x_1 + x2 > k$, alors x sera affecté à la classe ω_1, sinon il sera affecté à la classe ω_2. On obtient donc l'expression suivante de la règle de Bayes, pour $k = ln(\frac{\pi_2}{\pi_1}) - \frac{3}{2} = -1.483$:

$$\delta^*(x) = \begin{cases} a_1 \text{ si } 2x_1 + x_2 > k = -1.483 \\ a_2 \; sinon \end{cases}$$

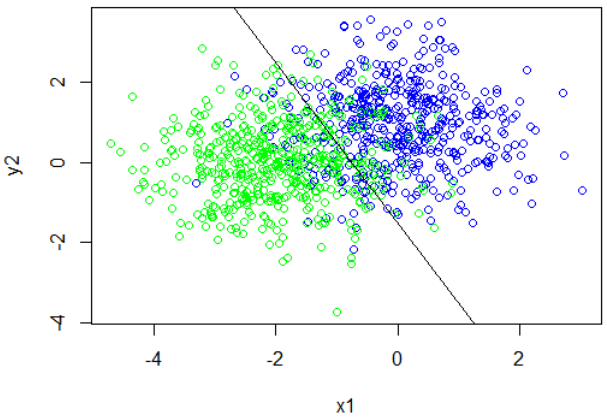

Figure 8. Données Synth1-1000 et frontière correspondante

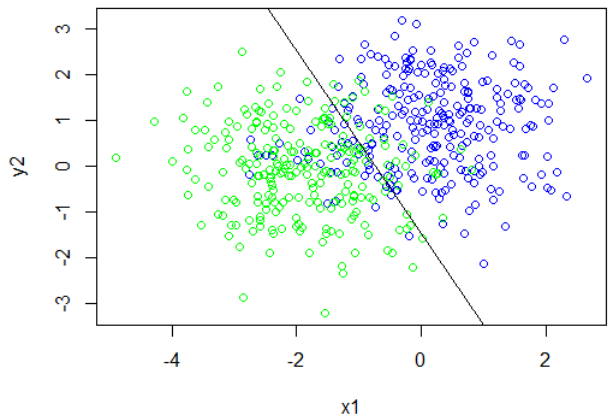

Figure 9. Données Synth1-500 et frontière correspondante

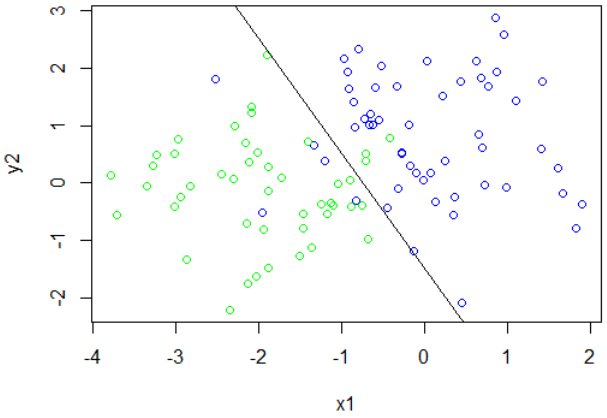

Figure 10. Données Synth1-100 et frontière correspondante

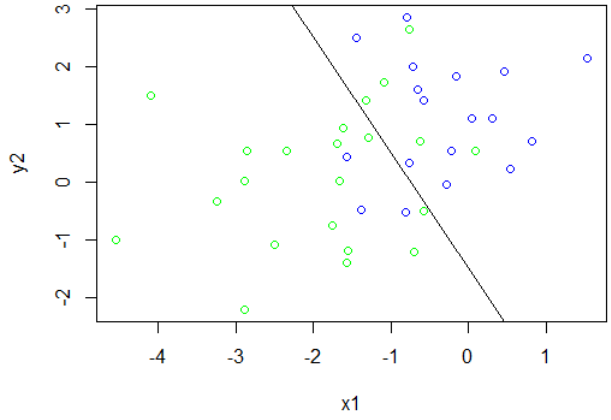

Figure 11. Données Synth1-40 et frontière correspondante

La frontière sépare clairement les deux classes sur les graphiques ci-dessus, et constitue donc une règle de décision cohérente. Tout porte à croire que les calculs effectués sont corrects.

2.2 Règle de Bayes, jeux Synth2-1000

Nous allons ici exprimer la règle de bayes en fonction de $x_1 et x_2$ pour le jeu de données Synth2-1000. Pour ce jeu de données nous avons les informations suivantes : $\Sigma_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \Sigma_2 = \begin{pmatrix} 5 & 0 \\ 0 & 1 \end{pmatrix}, \pi_1 = 0.523 \; et \; \pi_2 = 0.477.$ En reprenant l'expression des densités conditionnelles f_k présentée plus haut et après application numérique, on obtient :

$$\frac{f_1(x)}{f_2(x)} = \sqrt{5} exp(-\frac{2}{5}x_1^2 + \frac{17}{5}x_1 - \frac{41}{10}) > \frac{\pi_2}{\pi_1}$$

$$Soit : -\frac{2}{5}x_1^2 + \frac{17}{5}x_1 - \frac{41}{10} - ln(\frac{\pi_2}{\sqrt{5}\pi_1}) > 0$$

En prenant $k_1 = ln(\frac{\pi_2}{\sqrt{5}\pi_1})$ et $k_2 = \frac{41}{10}$, on obtient :

$$-\frac{41}{10}x_1^2 + \frac{17}{5}x_1 > k_1 + k_2$$

On trace ensuite la courbe d'équation $-\frac{2}{5}x_1^2 + \frac{17}{5}x_1 - \frac{41}{10}$ pour obtenir la frontière de décision :

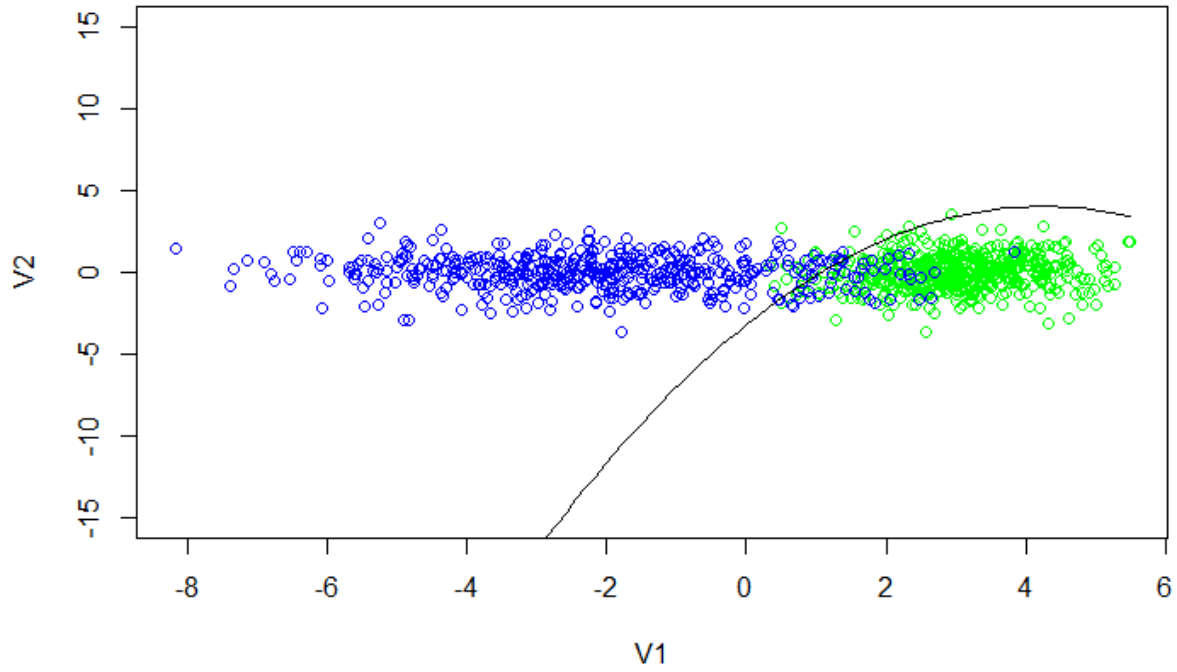

Figure 12. Données Synth2-1000 et frontière correspondante

La courbe de l'équation présentée dans la figure ci-contre représente une frontière cohérente au vue de sa superposition avec les données issues du jeux Synth2-1000.

Conclusion

En conclusion, ce TP nous a permis de découvrir et de mettre en pratique les algorithmes pour trois méthodes d'apprentissage supervisé. Nous avons notamment eu l'opportunité de coder les fonctions d'apprentissage et de test d'un classifieur euclidien et de la méthode des k plus proches voisins. Nous avons également appliqué les méthodes d'estimation de taux d'erreurs et de détermination d'intervalles de confiance. Ce TP nous aura également fait découvrir de nouveaux outils de R.

Table des figures

Annexe 1

```
ceuc.app = function(Xapp, zapp) {

  k = length(unique(as.character(zapp)));
  mu = matrix(nrow=k, ncol=dim(Xapp)[2]);
  for (i in 1:k) {
    classi = sort(unique(as.character(zapp)))[i];
    individualsi = Xapp[which(as.character(zapp) == classi),];
    mu[i,] = apply(individualsi, 2, mean);
  }
  return (mu);
}

ceuc.val <- function(mu, Xtst) {
   etiquette <- matrix(nrow=nrow(Xtst), 1)
   distTab <- distXY(mu, Xtst)
   print(distTab)
   for (i in 1:(nrow(Xtst))){
    etiquette[i,] <- which.min(distTab[,i])
   }
   etiquette
}
```

Figure 13. fonctions du classifieur euclidien

Annexe 2

```
kppv.tune = function(Xapp, zapp, Xval, zval, nppv) {
  Kopt = matrix(nrow=dim(as.matrix(nppv))[1], ncol=1);
  for (i in 1:length(nppv)) {
    ztst = kppv.val(Xapp, zapp, nppv[i], Xval);
    Kopt[i,] = (length(ztst[which(ztst != zval),]) / length(zval) * 100);
  }

  return (nppv[which.min(Kopt)]);
}

kppv.val <- function(Xapp, zapp, K, Xtst)
{
  etiquette <- matrix(nrow=nrow(Xtst), 1)
  distTab <- distXY(Xapp, Xtst)
  print(distTab)
  for (i in 1:(ncol(distTab))){
    kvoisin <- matrix(nrow=K, ncol=1)
    distRow <- distTab[,i]
    for (j in 1:K){
      kvoisin[j,] <- which.min(distRow)
      distRow[which.min(distRow)] <- 1000000
    }
    zappvoisin <- matrix(nrow=K, 1)
    for(j in 1:K){
      zappvoisin[j,1] <- zapp[kvoisin[j,1]]
    }
    scores_classes = as.data.frame(table(zappvoisin))
    max_score = max(scores_classes$Freq)
    etiquette[i,]<- as.numeric(as.character(scores_classes[which.max(scores_classes$Freq),]$z
  }
  etiquette
}
```

Figure 14. fonctions du classifieur KPPV