Université de Technologie de Compiègne
SY09:Analyse des données et Data-Mining , P17

Rendu TP3

Zineb Slam, Oumaima Talouka

3 juin 2017

1 Classifieur euclidien, K plus proches voisins

1.1 Programmation

1.1.1 Classifieur Euclidien

ceuc.app est une fonction qui retourne les centres d'inertie de chaque classifieur. La fonction *rowsum* sur R nous a été très utile pour sommer les lignes. La fonction *ceuc.val* prédit classe chaque individu des données tests. Nous utilisons ici la fonction *distXY* pour calculer qui sépare chaque individu a chacun des centres d'inertie.

1.1.2 K Plus proches Voisins

Dans cette partie nous avons essaye de programmer l'algorithme des k plus proches voisins en évitant les boucles.

1.1.3 Test de fonctions

1.2 Évaluation des Performances

1.2.1 Estimation des paramètres

Les résultats sont affichés dans Le tableau ci-dessous :

		Estimation des Parametres		
Jeu de données Synth1	k	μ_k	$\sum_k$	π_k
40	1	$\begin{pmatrix} -0.32 \\ 1.09 \end{pmatrix}$	$\begin{bmatrix} 0.68 & 0.12 \\ 0.12 & 1.01 \end{bmatrix}$	0.45
	2	$\begin{pmatrix} -1.883 \\ 0.105 \end{pmatrix}$	$\begin{bmatrix} 1.37 & 0.32 \\ 0.32 & 1.44 \end{bmatrix}$	0.55
100	1	$\begin{pmatrix} 0.02 \\ 0.82 \end{pmatrix}$	$\begin{bmatrix} 0.88 & -0.13 \\ -0.13 & 1.12 \end{bmatrix}$	0.54
	2	$\begin{pmatrix} -1.96 \\ -0.13 \end{pmatrix}$	$\begin{bmatrix} 0.76 & -0.04 \\ -0.04 & 0.76 \end{bmatrix}$	0.46
500	1	$\begin{pmatrix} 0.13 \\ 0.88 \end{pmatrix}$	$\begin{bmatrix} 1.05 & 0.052 \\ 0.052 & 0.98 \end{bmatrix}$	0.53
	2	$\begin{pmatrix} -1.88 \\ -0.08 \end{pmatrix}$	$\begin{bmatrix} 0.97 & -0.11 \\ -0.11 & 0.98 \end{bmatrix}$	0.47
1000	1	$\begin{pmatrix} -0.01 \\ 0.91 \end{pmatrix}$	$\begin{bmatrix} 0.97 & 0.06 \\ 1.08 & 0.06 \end{bmatrix}$	0.50
	2	$\begin{pmatrix} -1.96 \\ 0.02 \end{pmatrix}$	$\begin{bmatrix} 0.99 & 0.02 \\ 0.02 & 0.94 \end{bmatrix}$	0.50

1.2.2 Calcul du taux d'erreur du classifieur Euclidien

Dans cette partie nous allons tester les performances de chacun des algorithmes programmes precedement en utilisan le critère de l'erreur qu'on exprime comme :

$$Erreur = \frac{1}{N} \sum_{i=1}^{n} \mathbb{1}\hat{z}_i \neq z_i$$

Avec z_i la vrai valeur de z et $\hat{z}_i$ la valeur calculée a partir de l'algorithme. N est le nombre d'individus. $\mathbb{1}\hat{z}_i \neq z_i = 1$ si $\hat{z}_i = z_i$, 0 sinon. Après avoir calculé le taux d'erreur on peut utiliser la formule ci-dessous pour calculer l'intervalle de confiance :

$$Ic = [\mu_\epsilon - t\frac{\sigma_\epsilon^2}{\sqrt{N}}, \mu_\epsilon + t\frac{\sigma_\epsilon^2}{\sqrt{N}}]$$

On choisit un niveau de confiance t de 95%. Notons ici que μ et σ sont ceux calculés pour les différentes valeurs de ϵ et non ceux du jeu de données. Autre point a remarquer est le N qui est ici le nombre d'individus , or comme on a sépare notre jeu de données entre un ensemble d'application et un ensemble de test, on veillera a mettre le nombre d'individus correspondant en fonction si c'est l'intervalle de confiance d'application ou test. D'après l'énoncé en utilisant la fonction *separ1* napp $=\frac{2n}{3}$ et ntst$=\frac{n}{3}$ n étant le nombre total d'individus du jeu de donnée.

Les résultats obtenus pour les jeux de données sont affichés dans Le tableau qui suit.

	Performance du Classifieur Euclidien			
Jeu de données Synth1	ϵ_{app}	Ic_{app}	ϵ_{test}	Ic_{test}
40	0.204375	$[0.203, 0.205]$	0.2325	$[0.229, 0.236]$
100	0.0937	$[0.0936, 0.0938]$	0.099	$[0.098, 0.099]$
500	0.1335	$[0.1334, 0.1335]$	0.1347	$[0.1346, 0.1347]$
1000	0.14055	$[0.14054, 0.14055]$	0.1443	$[0.1442, 0.1443]$

Analyse de la performance du Classifieur Euclidien On remarque que plus le nombre d'individus augmente plus l'erreur diminue ce qui est logique car avec plus de donnees dans le modele d'apprentissage notre modele peut bien apprendre les data et donc mieux classer les ensembles de tests.

1.2.3 Le nombre Optimal de voisins du KPP

1.2.4 Calcul du taux d'erreur du KPP

	Performance du KPP	
Jeu de données Synth1	ϵ_{test}	Ic_{test}
40	0.25	$[0.246, 0.253]$
100	0.116	$[0.115, 0.116]$
500	0.1472	$[0.1471, 0.1472]$
1000	0.1544	$[0.1543, 0.1544]$

Analyse de la performance du KPP :

Comparaison des 2 méthodes : En comparant les 2 méthodes on remarque que l'erreur ainsi que les intervalles de confiance du Classifier Euclidien sont bien inférieur a ceux de la méthode des KPP. Il faut cependant ne pas perdre en considération que notre ensemble d'application qu'on a obtenu avec la fonction ***separ1*** est d'une taille plus importante que la taille de l'ensemble d'application utilise dans le KPP. Pour illustrer 2/3 des données ont ete utilise pour entrainer le Classifieur Euclidien alors que seulement n/4 ont été utilisées dans le KPP.

1.2.5 Jeux de données Synth2-1000

Estimation des Paramètres				
Jeu de données	k	μ_k	$\sum_k$	π_k
Synth2-1000	1	$\begin{pmatrix} 3.018 \\ -0.0063 \end{pmatrix}$	$\begin{bmatrix} 0.9904 & 0.1131 \\ 0.1131 & 1.092 \end{bmatrix}$	0.52
	2	$\begin{pmatrix} -2.142 \\ -0.0265 \end{pmatrix}$	$\begin{bmatrix} 4.435 & -0.154 \\ -0.154 & 1.030 \end{bmatrix}$	0.48

	Performance du Classifier Euclidien			
Jeu de données	ϵ_{app}	Ic_{app}	ϵ_{test}	Ic_{test}
Synth2-1000	0.061275	$[0.06127, 0.06127]$	0.0648	$[0.0647, 0.0648]$

	Performance du KPP	
Jeu de données	ϵ_{test}	Ic_{test}
Synth2-1000	0.061	$[0.0609, 0.0610]$

1.2.6 Jeux de données réelles : Pima & Breast Cancer

	Performance du Classifier Euclidien			
Jeu de données	ϵ_{app}	Ic_{app}	ϵ_{test}	Ic_{test}
Pima	0.2395	$[0.2395, 0.2395]$	0.2602	$[0.2601, 0.2602]$
Cancer	0.0403	$[0.0403, 0.0403]$	0.0397	$[0.03974, 0.03975]$

	Performance du KPP	
Jeu de données	ϵ_{test}	Ic_{test}
Pima	0.2575	$[0.2574, 0.2575]$
Cancer	0.03513	$[0.03512, 0.03515]$

2 Règle de Bayes

3 Conclusion