SYS800 Reconnaissance de formes  
Laboratoire 2 : Algorithmes de classification

# Bayes quadratique

Pour ce premier algorithme de classification, nous allons utiliser la règle de Bayes qui calcule la probabilité à posteriori p(ωj|x) qu’un nouvel élément x appartienne à la classe ωj:

Dans notre cas, les probabilités à priori de chaque classe P(ωj) sont égales car nous savons que la base de données contient le même nombre d’élément pour chacune des dix classes. Nous avons donc pour toutes les classes.

La priorité à priori p(x| ωj) qu’un élément x appartienne à la classe ωj est définie par :

Avec d le nombre de caractéristiques, µj et Σj respectivement le vecteur moyenne et la matrice de covariance de la classe j obtenue à partir de la base d’apprentissage.

Pour classifier un élément inconnu x, nous calculerons les probabilités que cet élément appartienne à chacune des classes, et la décision sera prise selon la plus grande probabilité.

Dans les logiciels de mathématiques comme Matlab, certaines fonctions comme l’exponentielle font appel à de long développement de Taylor qui coûte cher en ressource et en temps de calcul. Il est donc courant d’utiliser des fonctions monotones croissantes afin de transformer une expression sous une forme qui ne fera pas appel à ce genre de fonction, tout en conservant l’ordre. Dans notre cas, nous pouvons utiliser la fonction monotone et croissante logarithme népérien qui retirera la fonction exponentielle de notre expression.

Nous pouvons également retirer de l’expression les éléments qui sont constant pour toutes les classes et donc qui n’influenceront pas l’ordre : puisque dans notre cas, les probabilités à priori des 10 classes sont toutes les mêmes. On obtient :

L’ordre n’ayant pas été changé, nous chercherons toujours la plus haute valeur pour classifier les nouveaux éléments x. Nous remarquons dans cette expression la présence de la distance de Mahalanobis. Pour classer un élément dans une classe, nous cherchons la distance la plus courte. Il faut donc minimiser ce terme : ce qui équivaut à maximiser ce terme : et donc notre expression.

Sur la base de test de 1000 éléments, l’algorithme nous donne un taux d’erreur de 6.5%. Afin d’analyser les performances de ce classifieur, nous dressons la matrice de confusion qui présente le nombre de fois que les 10 différentes décisions ont été prises selon les dix classes possibles. Nous pouvons alors trouver sur la diagonale le nombre de fois que la bonne décision a été prise et en analysant le reste de la matrice, nous pouvons observer les cas les plus difficiles à classifier.



Nous remarquons en analysant la matrice de confusion que les erreurs de classification se produisent le plus lorsque l’on essaye de classifier les chiffres 3 (11% d’erreur) et 5 (13% d’erreur). Les erreurs les plus courantes sont de prendre un chiffre 3 pour un 2 (7% des chiffres 3) et un chiffre 5 pour un 3 (7% des chiffres 5).

Nous remarquons également que ce sont les chiffres 2 et 8 qui sont le plus souvent sollicitées avec respectivement 11.6% et 11.2% des décisions et que les chiffres 5, 8 et 1 sont les plus difficiles à reconnaître avec respectivement 8.7%, 9.1% et 9.2% des décisions. Nous pouvons donc deviner l’allure des frontières entre les classes d’après la matrice de confusion.