Université de Toulon - Var | Langage C
Licence 2 Sciences Pour l'Ingénieur | Année 2014-2015

Projet 3 : Classification supervisée : Analyse discriminante

par Faicel CHAMROUKHI

Table des matières

Ce travail devra être effectué en binôme lors des séances de TP et être présenté lors de la dernière heure de la dernière séance de TP.

1 Objectifs

L'objectif de ce projet est de développer une bibliothèque implémentant un algorithme de classification supervisée, dite aussi discrimination de données brutes.
Le package développé s'appellera `DA` (pour Discrinant Analysis (Analyse Discriminate)).

2 Contexte

2.1 Qu'est ce que la classification supervisée ?

En quelques mots, la classification supervisée, dite aussi discrimination est la tâche qui consiste à discriminer des données, de façon supervisée (ç-à-d avec l'aide préalable d'un expert), un ensemble d'objets ou plus largement de données, de telle manière que les objets d'un même groupe (appelé classes) sont plus proches (au sens d'un critère de (dis)similarité choisi) les unes au autres que celles des autres groupes. Généralement, on passe par une première étape dite d'apprentissage où il s'agit d'apprendre une règle de classification partir de données annotées (étiquetées) par l'expert et donc pour les quelles les classes sont connues, pour ıprédire les classes de nouvelles données, pour lesquelles (on suppose que) les données sons inconnues. La prédiction est une tâche principale utilisée dans de nombreux domaines, y compris l'apprentissage automatique, la reconnaissance de formes, le traitement de signal et d'images, la recherche d'information, etc.

2.2 Quelles données ?

Les données traitées en classification peuvent être des images, signaux, textes, autres types de mesures, etc. Dans le cadre de ce projet les données seront des données multidimensionnelles, par exemple une image (couleur). Chaque donnée est donc composée de plusieurs variables (descripteurs). Pour le cas de données multidimensionnelles standard, chaque donnée étant de dimension d (donc un point dans l'espace $\mathbb{R}^d$) et éventuellement étiquetée (dans le cas où l'on connaitrait sa classe d'appartenance (le "label")) et peut donc être modélisée, par exemple, par un e structure contenant au moins les coordonnées du point et le champ "label". Les n données peuvent donc être modélisées comme étant un tableau de n éléments, chaque élément du tableau étant une structure "donnée" comme décrite précédemment.
Dans le cas d'une image (couleur), l'image contenant n lignes et m colonnes et donc $n \times m$ pixels couleurs, chaque pixel est composé de trois composantes RVB, peut être modélisée par un tableaux de $n \times m$ structures. Chaque structure "pixel" est composée au moins des champs, couleur et "label".
Pour cette partie `LDA-QGA` du projet, on commencera par traiter les données simulées, des données iris et ensuite des images.

2.3 Qu'est ce qu'une classe ?

En gros, une classe (ou groupe) est un ensemble de données formée par des données homogènes (qui "se ressemblent" au sens d'un critère de similarité (distance, densité de probabilité, etc)). Par exemple, une classe peut être une région dans une image couleur, un évènement particulier dans un signal sonore, la classe spam et classes non spam dans le cas détection de spams dans un mail, etc.

2.4 Combien de classes ?

Le nombre de groupes (qu'on notera K) en prédiction est supposé fixe (donné par l'utilisateur). C'est le cas par exemple si l'on s'intéresse à classer des images de chiffres manuscrits (nombre de classes = 10 : $0, ..., 9$) ou de lettres manuscrites (nombre de classes = nombres de caractères de l'alphabet), etc.

2.5 Exemple Introductif

Considérons par exemple une image couleur, chaque image contient n pixels $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, chaque pixel $\mathbf{x}_i$ contient $d = 3$ valeurs (RGB). On peut donc représenter donc le $i$ème pixel $(i = 1, \ldots, n)$ par un vecteur $\mathbf{x}_i$ de dimension $d = 3 : \mathbf{x}_i = (x_{i1}, x_{i2}, x_{i3})^T \in \{0, 1, ..., 255\}^3$. Si l'on connaît les classes de certains pixels, on pourra prédire les classes des autres pixels on choisissant une mesure de (dis)similarité, par exemple une simple distance, ou une mesure de probabilité, etc. Chaque pixel à classer aura donc la classe de celui qui lui est le plus proche au sens de la mesure de (dis)similarité choisie. Ceci peut être utilisé par exemple en segmentation d'image. De manière générale, on peut représenter les données comme un ensemble de vecteurs $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, chaque $\mathbf{x}_i$ est composée de d composantes réelles : $\mathbf{x}_i = (x_{i1}, ..., x_{ij}, ..., x_{id})^T \in \mathbb{R}^d$.

2.6 Et les stats dans tout ça, pour quoi faire ?

Une classe peut être définie au sens probabilité. Afin de définir ce que c'est une classe au sens probabiliste, on dira qu'une classe est caractérisée par sa densité de probabilité.
Une donnée peut en effet être considérée comme étant la réalisation d'une variable aléatoire $\mathbf{x}_i = (x_{i1}, ..., x_{ij}, ..., x_{id})^T \in \mathbb{R}^d$ réelle multidimensionnelle.
On suppose qu'une donnée appartenant à la classe k est une v.a. qui suit une loi de paramètres $\boldsymbol{\theta}_k$ (celle de la classe k). Cette loi (densité dans ce cas) qui peut être notée $f_{ik}(\mathbf{x}_i; \boldsymbol{\theta}_k)$ est donc à choisir par l'analyseur pour modéliser ces données. Il suffit donc ensuite d'estimer les lois des différents classes (étape d'apprentissage) pour voir ensuite, pour chaque donnée de test de classe inconnue, quelle est la classe qui lui est la plus probable. Ceci s'effectue en maximisant la probabilité a posteriori 5.

2.7 Quelle loi de probabilité ?

Dans ce projet, on considérera la loi Gaussienne qui est bien adaptée à beaucoup de phénomènes physiques et naturels. Elle est simplement décrite par sa moyenne $\boldsymbol{\mu}_k$ et sa matrice de covariance et $\boldsymbol{\Sigma}$ est donnée par

$$f_{ik}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{d}{2}} |\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\Big(-\frac{1}{2}(\mathbf{x}_i - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_k)\Big) \tag{1}$$

3 Analyse Discriminante Linéaire

C'est une approche probabiliste qui consiste à regrouper les données où une donnée sera considérée proche de l'autre si elle provient de la même densité (loi) de probabilité plutôt qu'en effectuant un simple calcul de distance déterministe. Chaque classe de données pourra donc être résumée par les paramètres de sa loi de probabilité (c.f. cours probas discrètes :)). Chaque classe est supposée être Gaussienne.

4 Travail à réaliser

Il s'agit d'implémenter en C l'algorithme de l'analyse discriminante pour prédire les classes de nouvelles données à partir de données étiquetées (données d'apprentissage). On commencera par l'analyse discriminante linéaire (LDA).

4.1 Apprentissage du modèle LDA

L'apprentissage du modèle LDA consiste en l'estimation des paramètres $\boldsymbol{\mu}_k$ et $\boldsymbol{\Sigma}$ à partir des données d'apprentissage . Ces estimations sont données par

$$\pi_k = \frac{\sum_{i|z_i=k}}{n} = \frac{n_k}{n} \tag{2}$$

où n_k est le cardinal de la classe k, z_i est l'étiquette de classe de l'exemple $\mathbf{x}_i$ et $z_i = k$ désigne le fait que l'exemple $\mathbf{x}_i$ appartient à la classe k.
L'estimation des paramètres $\boldsymbol{\mu}_k$ (les moyennes) et $\boldsymbol{\Sigma}$ (matrice de covariance) à partir des données d'apprentissage sont données par

$$\boldsymbol{\mu}_k \quad = \quad \frac{1}{n_k} \sum_{i|z_i=k} \mathbf{x}_i \tag{3}$$

et

$$\boldsymbol{\Sigma} \quad = \quad \frac{1}{n-K} \sum_{k=1}^{K} \sum_{i|z_i=k} (\mathbf{x}_i - \boldsymbol{\mu}_k)(\mathbf{x}_i - \boldsymbol{\mu}_k)^T. \tag{4}$$

4.2 Test du modèle LDA

Une fois les paramètres sont estimés, on peut prédire les classes des données de test par la règle du MAP (maximum a posteriori). Cette règle consiste à maximiser les probabilités a posteriori, ç-à-d affecter chaque donnée de test $\mathbf{x}_i$ à la classe $\hat{z}_i$ ayant la plus grande probabilité a posteriori :

$$\tau_{ik} = \mathbb{P}(z_i = k|\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}) = \frac{\pi_k \, f_{ik}(\mathbf{x}_i; \boldsymbol{\mu}_k, \boldsymbol{\Sigma})}{\sum_{\ell=1}^{K} \pi_\ell \, f_{i\ell}(\mathbf{x}_i; \boldsymbol{\mu}_\ell, \boldsymbol{\Sigma})} \tag{5}$$

et on a donc

$$\hat{z}_i = \arg\max_{k=1}^{K} \tau_{ik} \quad (i = 1, \ldots, n) \tag{6}$$

L'algorithme d'apprentissage `train_LDA` est donné par le pseudo-code 1 ci-après.
Attention ! N'oubliez pas que les indices en C commencent à partir de zéro.

Algorithm 1: Algorithme de l'analyse Discriminante Linéaire (LDA)

Input: Données d'apprentissage ; $\mathbf{X}^{\text{train}} = (\mathbf{x}_1^{\text{train}}, \ldots, \mathbf{x}_n^{\text{train}})$; classes des données d'apprentissage $\mathbf{z}^{\text{train}} = (z_1^{\text{train}}, \ldots, z_n^{\text{train}})$

Algorithme `train_LDA` :

for $k \leftarrow 1$ ***to*** K **do**
  Calculer la proportion π_k de la classe k en utilisant l'équation (2)
  Calculer la moyenne $\boldsymbol{\mu}_k$ en utilisant l'équation (3)
end
Calculer la matrice de covariance des classes $\boldsymbol{\Sigma}$ en utilisant l'équation (4)

Result: Paramètres $(\pi_1, \ldots, \pi_K, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma})$

Ensuite, une fois les paramètres sont estimés, on peut prédire les classes des données de test. L'algorithme de test `test_LDA` est donné par le pseudo-code 2 ci-après.

Algorithm 2: Algorithme de classification de l'analyse discriminante linéaire

Input: Données de test ; $\mathbf{X}^{\text{test}} = (\mathbf{x}_1^{\text{test}}, ..., \mathbf{x}_n^{\text{test}})$; paramètres d'apprentissage $(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma})$

Algorithme `test_LDA` :

```
/* Calcul des densités pour les différentes classes */
```
for $i \leftarrow 1$ ***to*** n **do**
  for $k \leftarrow 1$ ***to*** K **do**
    Calculer la densité f_{ik} avec $\boldsymbol{\mu}_k$ et $\boldsymbol{\Sigma}$ pour chaque exemple de test $\mathbf{x}_i$ en utilisant en utilisant l'équation (1)
  end
end
```
/* Classification par la règle du MAP */
```

for $i \leftarrow 1$ ***to*** n **do**
  Calculer la classe de chaque donnée de test $\mathbf{x}_i$ en maximisant les probabilités a posteriori τ_{ik} données par l'équation (5) :
  $\hat{z}_i = \arg\max_{k=1}^{K} \tau_{ik}$
end
Result: $\hat{z}_i \quad (i = 1, \ldots, n)$: classes de test estimés

5 Données et présentation des résultats

Pour les données, vous pouvez par exemple utiliser celles-ci : Xtrain, klastrain, Xtest. Vous pouvez également tester votre programme sur les données iris téléchargeable ici (Iris, un jeu de données très utilisé en classification automatique).

Pour la présentation des résultats, pour commencer, les résultats trouvés peuvent être affichés directement à l'écran ou écrites dans un fichier.
Pour aller plus loin, la présentation des résultats pourra se faire par des graphiques en affichant par exemple les données dans l'espace, chaque ensemble de données appartenant à une même classe est colorée par une couleur différente, etc. Vous pouvez vous inspirez de l'exemple suivant de la figure 1. Les données pour cet exemple sont : Xtrain, klastrain, Xtest.

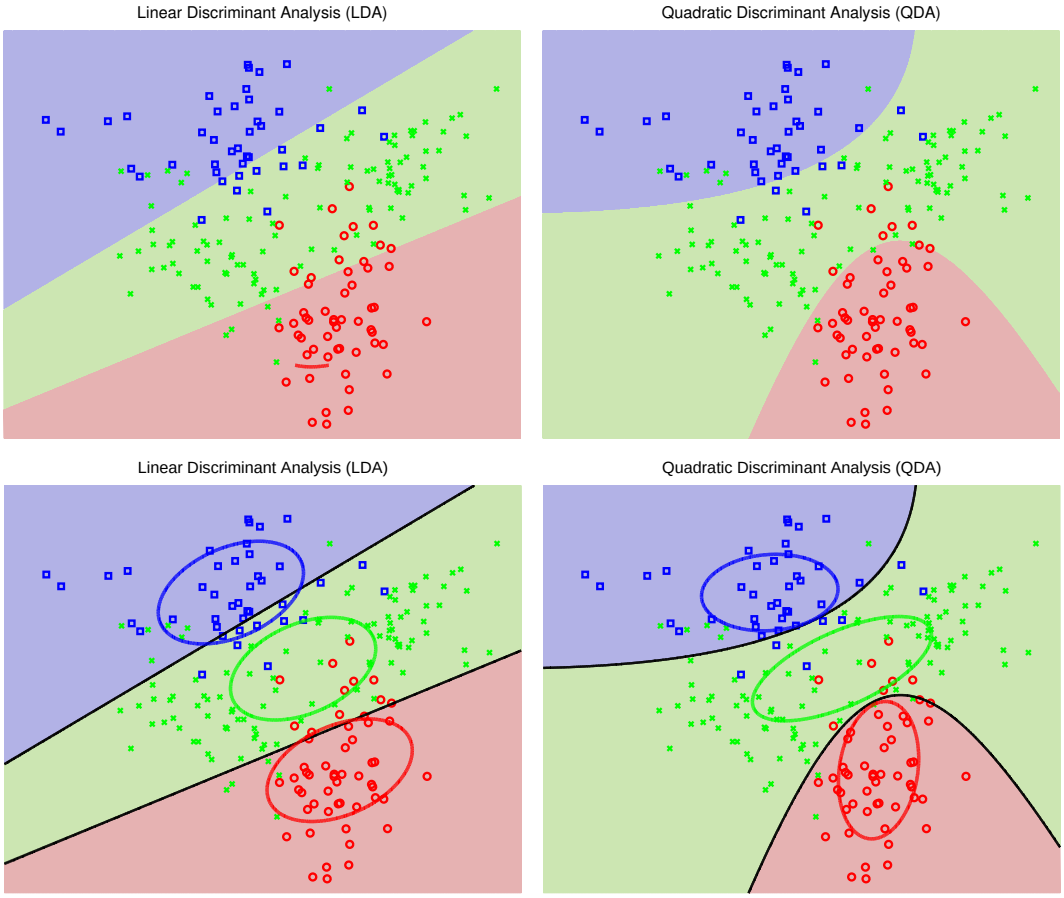

FIGURE 1 – Un exemple où l'on a trois classes. Les données d'apprentissage sont représentées par □, ×, ∘. Le reste représente les données de test et les couleurs correspondent aux classes prédites par l'algorithme d'analyse discriminante linéaire (LDA) et l'analyse discriminante quadratique (QDA).

Si vous manipulez des données pour lesquelles vos connaissez les classes des données de test (le cas des iris par exemple), vous donnerez également le taux d'erreur de classification en comparant les classes fournies par l'algorithme avec les vraies classes (par exemple en créant une fonction `classif_error`). Pour les images, essayer de regarder si les classes trouvées correspondent bien aux vrais catégories des lettres ou chiffres manuscrits . Essayer de dresser une matrice de confusion, etc

Bonne chance !