TP : Analyse factorielle Discriminante

Rémi Flamary

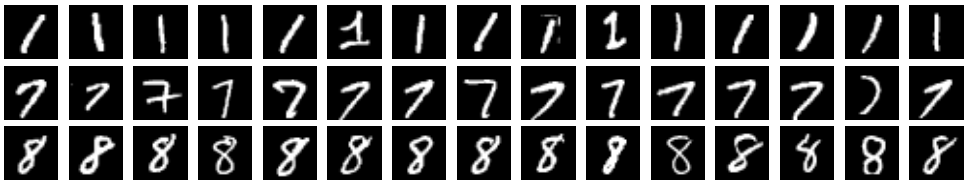
1 Chargement des données et pré-traitement

Lors du TP vous aurez besoin des bibliothèque Python `numpy` `pylab` et `scipy`. Il vous est conseillé de les importer dès le début avec le code suivant :

```
import numpy as np
import pylab as pl
import scipy as sp
```

Vous pourrez ensuite accéder aux fonctions de ces modules à l'aide du point (par exemple `np.zeros(10)` pour la fonction `zeros` de numpy). Dans la suite du TP, pour chaque question la liste des fonctions `numpy/pylab` nécessaires est donné entre parenthèses.

- Télécharger le fichier "digits.npz".
- Charger ce fichier en mémoire en utilisant la fonction `np.load`. Le fichier contient les matrices suivantes :
  - **x** et **xt** : matrices de données contenant respectivement $n = 3000$ et $nt = 1500$ exemples d'images manuscrites. Chaque ligne de ces matrices correspond à une image stockée sous la forme d'un vecteur transposé.
    Les images sont de la forme suivante :

    
  - **y** et **yt** : étiquettes des images décrites dans les matrices précédentes. Ce sont des vecteurs qui contiennent la classe $(1, 7, 8)$ de chaque image de **x** et **xt**.
- Utiliser la fonction `np.reshape` pour extraire quelques images de taille 28×28 pour chaque classe. Les visualiser avec la fonction `pl.imshow`.

2 Analyse factorielle discriminante binaire

- On veut créer un problème de classification binaire à partir des trois classes. Vous pourrez par exemple choisir de classifier la classe 8 contre 1 et 7. Stocker les étiquettes binaires $(-1, 1)$ dans les vecteurs **yb** et **ytb** (opérateur `==`) .
- Estimer les moyennes de chaque classes et les visualizer sous la forme d'images (`np.mean,np.reshape,pl.imshow`). En déduire la matrice Σ_b du problème binaire.
- Estimer les matrices de covariance Σ_k (fonction `np.cov`) pour chaque classe. En déduire la matrice Σ_w intra classe du problème binaire et la visualiser.
- Estimer la direction **u** (`np.linalg.solve` pour inverser la matrice). Quel est le problème numérique et comment le résoudre ?
- Projeter les exemples d'apprentissage **x** et de test **xt** sur la direction **u** (`np.dot,pl.hist`). Superposer deux histogrammes montrant la séparation des classes pour les données d'apprentissage et de test. Conclusions ?

— Trouver un seuil permettant de prédire la classe à partir de la projection. Calculer le taux de bonne reconnaissance sur les données d'apprentissage et de test (`np.sign,np.mean,==`).
— Comparer les différentes projections $\mathbf{u}_{\text{rsb}}$, $\mathbf{u}_{\text{fisher}}$ et $\mathbf{u}_{\text{deflex}}$. Quelles sont leurs TBR (taux de bonne reconnaissance) ?

— Refaire les étapes précédentes pour la détection de 1 contre 7 et 8 et 7 contre 1 et 8. Quelles sont les différences de performance ? À quoi sont-elles dues ?

3 Analyse multiclasse

— On travaille maintenant sur le cas multiclasse avec 3 classes.
— Estimer les moyennes et matrices de covariance de chaque classes.
— Calculer la projection $\mathbf{U}$ à partir de la matrice $\Sigma_w^{-1}\Sigma_b$ (fonction `np.linalg.eigh`. Une fois encore il existe des problème numériques, comment les gérer ?
— Projeter les données d'apprentissage et de test sur le sous espace de dimension 2 et tracer les exemples (`pl.scatter`). les classes sont elles bien séparées ?
— Comparer cette projection à une projection de type ACP (vue au TP précédent). Laquelle est la plus adapté pour une tâche de classification ?
— Proposer une méthode de décision de la classe à partir de la projection 2D. Calculer les TBR pour les données d'apprentissage et de test.
— Quel est l'effet de la régularisation sur les performances de prédiction ?