TP : Introduction à l'apprentissage statistique

1 Introduction

L'apprentissage statistique est un domaine de l'*intelligence artificielle* qui exploite les outils fournis par les statistiques (descriptives et/ou inférentielles) afin de concevoir des algorithmes capables d'*apprendre* un modèle mathématique à partir de données disponibles, pour ensuite effectuer certaines tâches comme la *classification*, le *clustering*, la *régression*, etc.

Dans ce TP, nous nous intéressons au problème de la *reconnaissance de visages*, dont l'objectif est de concevoir des algorithmes qui, à partir d'une base d'images de visages d'individus identifiés, est capable d'associer à une nouvelle image l'individu qui lui correspond.

Les méthodes les plus performantes à l'heure actuelle sont celles utilisant le *deep learning* (e.g. le système DeepFace de Facebook [3]), qui exploitent les *réseaux de neurones profonds*. Néanmoins, ces méthodes ont d'une part un coût calculatoire très élevé, et nécessitent d'autre part une base d'images identifiées très importante pour effectuer un apprentissage correct, ce qui n'est pas toujours possible en pratique. Nous utiliserons ici une approche alternative, appelée *eigenface* [4], et reposant un outil classique d'analyse de données, appelé *analyse en composantes principales* (ACP). L'ACP, inventée par Karl Pearson en 1901, a pour objectif d'effectuer une *réduction de dimension* des données de départ, en transformant un grand nombre de variables corrélées entre elles en un nombre réduit de variables décorrélées, appelées composantes principales, et décrivant de manière suffisante les données d'origine. La classification d'images se fera alors sur les données réduites.

Après une étude du fonctionnement de l'ACP en section 2, nous l'appliquons au problème de réduction de dimension dans une base d'images de visages [1] en section 3. Finalement, nous abordons en section 4 la problématique de la reconnaissance automatique de visages à travers une classification sur les images réduites par ACP.

2 Principes de l'ACP (à préparer chez soi)

Considérons un ensemble de données $\mathbf{x}_1, \ldots, \mathbf{x}_n$ de $\mathbb{R}^p$, dite *données d'apprentissage*, dont on note

$$\overline{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_i$$

la moyenne empirique et

$$\hat{\mathbf{R}} = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{x}_i - \overline{\mathbf{x}}) (\mathbf{x}_i - \overline{\mathbf{x}})^T$$

la matrice de covariance empirique. Rappelons qu'une telle matrice étant semi-définie positive, elle diagonalise en particulier dans une base orthonormée avec des valeurs propres positives. On notera ainsi $\lambda_1 \geq \ldots \lambda_p \geq 0$ ses valeurs propres par ordre décroissant et $\mathbf{u}_1, \ldots, \mathbf{u}_p$ les vecteurs propres orthonormés associés.

Composantes principales. L'ACP consiste à projeter les données $\mathbf{x}_1, \dots, \mathbf{x}_n$ dans les ℓ directions orthogonales qui contiennent le plus d'« énergie » au sens statistique du terme, c'est-à-dire en terme de variance empirique. Plus formellement, on cherche une base de ℓ vecteurs orthonormés $\mathbf{v}_1, \dots, \mathbf{v}_\ell \in \mathbb{R}^p$, tels que les variances empiriques

$$E_j = \frac{1}{n} \sum_{i=1}^{n} \langle \mathbf{v}_j, \mathbf{x}_i - \overline{\mathbf{x}} \rangle^2 , \quad j = 1, \dots, \ell,$$

soient maximales. Pour une nouvelle donnée $\mathbf{x} \in \mathbb{R}^p$, on ne travaillera ensuite que sur les ℓ coefficients des projections $\langle \mathbf{v}_j, \mathbf{x} - \overline{\mathbf{x}} \rangle$, $j = 1, \dots, \ell$, qui constituent un vecteur de dimension réduite par rapport à $\mathbf{x}$ si tant est que $\ell \ll p$.

1. Montrer que le vecteur propre $\mathbf{u}_1$ de $\hat{\mathbf{R}}$ maximise l'énergie de projection parmi tous les vecteurs unitaires, c'est-à-dire

$$\mathbf{u}_1 \in \operatorname*{argmax}_{\mathbf{v} \in \mathbb{R}^p : \|\mathbf{v}\|_2 = 1} \frac{1}{n} \sum_{i=1}^{n} \langle \mathbf{v}, \mathbf{x}_i - \overline{\mathbf{x}} \rangle^2 ,$$

   Que vaut alors l'énergie de la projection $E_1 = \frac{1}{n} \sum_{i=1}^{n} \langle \mathbf{u}_1, \mathbf{x}_i - \overline{\mathbf{x}} \rangle^2$?

2. Etendre le résultat précédent en montrant que pour tout $i \geq 2$,

$$\mathbf{u}_j \in \operatorname*{argmax}_{\substack{\mathbf{v} \in \mathbb{R}^d : \|\mathbf{v}\|_2 = 1, \\ \mathbf{v} \perp \mathbf{u}_1, \dots, \mathbf{u}_{j-1}}} \frac{1}{n} \sum_{i=1}^{n} \langle \mathbf{v}, \mathbf{x}_i - \overline{\mathbf{x}} \rangle^2 .$$

   et en donnant la valeur de $E_j = \frac{1}{n} \sum_{i=1}^{n} \langle \mathbf{u}_j, \mathbf{x}_i - \overline{\mathbf{x}} \rangle^2$.

Réduction de dimension. La dimension ℓ du sous-espace de projection $\mathcal{S} = \text{vect}\,(\mathbf{u}_1, \dots, \mathbf{u}_\ell)$ est généralement fixée de manière à garantir la conservation d'une certaine fraction de l'énergie totale. Notons $\pi_{\mathcal{S}}$ la projection orthogonale de $\mathbb{R}^p$ sur $\mathcal{S}$, i.e.

$$\pi_{\mathcal{S}}(\mathbf{x}) = \sum_{j=1}^{\ell} \langle \mathbf{u}_j, \mathbf{x} \rangle \, \mathbf{u}_j .$$

Les *composantes principales* d'un vecteur $\mathbf{x} \in \mathbb{R}^p$ désignent les coordonnées de $\pi_{\mathcal{S}}(\mathbf{x} - \overline{\mathbf{x}})$ dans la base $\{\mathbf{u}_1, \dots, \mathbf{u}_\ell\}$, i.e. les quantités $\langle \mathbf{u}_1, \mathbf{x} - \overline{\mathbf{x}} \rangle, \dots, \langle \mathbf{u}_\ell, \mathbf{x} - \overline{\mathbf{x}} \rangle$. Les *directions principales* désignent quant à elles les vecteurs propres $\mathbf{u}_1, \dots, \mathbf{u}_\ell$. Définissons

$$\kappa(\ell) = \frac{\frac{1}{n} \sum_{i=1}^{n} \|\pi_{\mathcal{S}}(\mathbf{x}_i - \overline{\mathbf{x}})\|_2^2}{\frac{1}{n} \sum_{i=1}^{n} \|\mathbf{x}_i - \overline{\mathbf{x}}\|_2^2} \tag{1}$$

comme le ratio de l'énergie de projection (i.e. l'énergie associée aux composantes principales) sur l'énergie totale associées aux données.

3. Donner une expression de $\kappa(\ell)$ en fonction de $\lambda_1, \dots, \lambda_p$.

La dimension ℓ_* optimale est alors fixée de telle manière que

$$\ell_* = \min \{\ell : \kappa(\ell) \geq \alpha\} ,$$

où α est un taux de reconstruction fixé par l'utilisateur (par exemple, $\alpha = 0.9$ ou $\alpha = 0.99$).

Implémentation. En pratique, lorsque l'on travaille sur des données de grande dimension p, le coût calculatoire de la décomposition spectrale de $\hat{\mathbf{R}}$ pour obtenir les vecteurs propres $\mathbf{u}_1, \ldots, \mathbf{u}_\ell$ est prohibitif ; par exemple, pour une image de 256×256 en niveaux de gris et vectorisée, on a $p = 65536$. Dans le cas où le nombre de données n est tel que $n \ll p$, il est possible d'obtenir les vecteurs propres de $\hat{\mathbf{R}}$ de manière moins coûteuse. On supposera par la suite que $\mathbf{x}_1, \ldots, \mathbf{x}_n$ sont linéairement indépendants pour simplifier les développements. Remarquons que

$$\hat{\mathbf{R}} = \mathbf{X}\mathbf{X}^T,$$

où $\mathbf{X}$ est la matrice $p \times n$ définie par ses colonnes comme

$$\mathbf{X} = \frac{1}{\sqrt{n}} \left[\mathbf{x}_1 - \overline{\mathbf{x}}, \ldots, \mathbf{x}_n - \overline{\mathbf{x}}\right]. \tag{2}$$

4. Montrer que si $\mathbf{v}$ est un vecteur propre de la matrice de Gram $\mathbf{X}^T\mathbf{X}$, alors $\mathbf{u} = \mathbf{X}\mathbf{v}$ est un vecteur de propre de $\mathbf{X}\mathbf{X}^T$ associé à la même valeur propre. Montrer également que $\mathbf{X}^T\mathbf{X}$ est définie positive.
5. On considère $\mathbf{v}_1, \ldots, \mathbf{v}_n$ un ensemble de vecteurs propres orthonormés de $\mathbf{X}^T\mathbf{X}$, ainsi que les matrices $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$ et

$$\mathbf{U} = \mathbf{X}\mathbf{V}\left(\mathbf{V}^T\mathbf{X}^T\mathbf{X}\mathbf{V}\right)^{-1/2}. \tag{3}$$

   Montrer que $\mathbf{U}$ est une matrice à colonnes orthonormées de vecteurs propres de $\hat{\mathbf{R}}$ associés aux valeurs propres non nulles.

3 Réduction de dimension et eigenfaces

Dans cette partie, on s'appuiera sur le fichier eigenfaces.m *à compléter, ainsi que sur les différentes images fournies, disponibles à l'adresse*

`http://sites.google.com/site/valletp/stats/`.

On considère ici des images de dimensions 192×168 pixels en niveaux de gris (i.e. à valeurs dans $[\![0, 255]\!]$, issues de la base *the Extended Yale Face Database B* [1, 2], qui contient les visages de 28 sujets photographiés sous différentes conditions de luminosité. Une des spécificités de cette base tient au fait que les images ont été préalablement centrées et découpées afin que seuls les visages apparaissent. [1]

Dans l'archive database.zip, une partie des images de la base correspondant à 6 sujets sont fournies (mais le reste des images peut être téléchargé à l'adresse [1]). Elle contient des dossiers du type training (données d'apprentissage) ou test (données de test), où :

— /training1/ : 10 images par individus, éclairage de face ;
— /test1/ : 7 images par individus (différentes de /training1/), éclairage de face ;
— /training2/ : 5 images par individus, éclairage de face ;
— /test2/ : 12 images par individus (différentes de /training2/), éclairage de face ;
— /test3/ à /test6/ : même nombre d'images par individus (variable suivant le dossier), angle d'éclairage de plus en plus fort.

FIGURE 1: Images de la base training1

Les images du dossier /training1/ sont donnés à titre d'exemple en FIGURE 1. Chaque image, de format pgm, possède un nom de la forme yaleBn_xxx.pgm où n représente le numéro du sujet photographié, et xxx contient des informations sur la luminosité.

Chargement/affichage des images. Pour se ramener au formalisme de l'ACP, chaque image sera chargée en mémoire sous la forme d'une matrice, puis *vectorisée* pour former un vecteur de données de dimension $p = 32256$. Ceci s'effectue grâce aux deux commandes :

```
1- % img contient la matrice 192 × 168 de pixels de l'image à l'adresse adr
   img = double(imread(adr));

2- % imgv contient un vecteur de taille p = 32256
   imgv = img( :);
```

Pour afficher une image, stockée sous forme de matrice, on utilise les commandes

```
3- % colormap(gray) indique que l'affichage de l'image doit se faire en niveaux de gris
   imagesc(img);
   colormap(gray);
```

Par ailleurs, le fichier eigenfaces.m fournit les commandes pour parcourir toutes les images des dossiers type training et test.

Eigenfaces. Pour effectuer l'ACP, nous allons calculer la matrice de covariance empirique $\hat{\mathbf{R}}$ sur la base d'images dédiées à l'apprentissage (i.e contenues dans les dossiers type training).

1. Intégrer au fichier eigenfaces.m le calcul des valeurs propres non nulles et vecteurs propres orthonormés associés de $\hat{\mathbf{R}}$. *Pour des raisons de temps de calcul, on utilisera l'astuce proposée en fin de section 2.*

1. Les images faisant également apparaître des éléments du décor en plus du visage peuvent accroître artificiellement les performances des algorithmes de reconnaissance de visages.

Dans le contexte de la reconnaissance de visages, les directions principales sont appelées *eigenfaces* ; ce sont les images correspondant aux vecteurs propres $\mathbf{u}_1, \dots, \mathbf{u}_n$, définis comme les colonnes de la matrice $\mathbf{U}$ en (3). Les eigenfaces constituent une base orthonormée décrivant les visages de la base d'apprentissage. Le sous-espace vectoriel $\mathcal{S}$ engendré par les ℓ eigenfaces $\mathbf{u}_1, \dots, \mathbf{u}_\ell$ les plus énergétiques sera appelé *facespace*.

2. Pour la base training1, à l'aide de la commande subplot, représenter sur une même figure les n eigenfaces. Commenter.
3. Représenter sur une même figure, pour une image de chacun des 6 sujets de la base, l'évolution de l'image reconstruite après ACP en faisant varier la dimension ℓ du facespace (penser à ajouter la moyenne $\overline{\mathbf{x}}$ pour un meilleur rendu). Commenter les résultats obtenus.
4. Tracer finalement le ratio de reconstruction $\ell \mapsto \kappa(\ell)$ défini en (1), et déterminer de manière automatique la dimension ℓ_* du sous-espace de reconstruction de telle manière à garantir un ratio de 0.9. Commenter au regard des items précédents.
5. Si l'on utilise la base training2, à quelles limitations peut-on s'attendre ?

4 Classification

Pour réaliser la reconnaissance de visages, nous allons mettre en œuvre un *classifieur*, c'est-à-dire une fonction

$$\varphi : \mathbb{R}^p \to [\![1, m]\!],$$

prenant en entrée un vecteur de données de dimension p, et renvoyant un entier représentant l'indice de la classe (ici le nombre de classes m représente le nombre de sujets dans la base, i.e $m = 6$). Notons pour la suite que l'ensemble $[\![1, n]\!]$ se partitionne en k sous-ensembles $\mathcal{C}_1, \dots, \mathcal{C}_m$, où $\mathcal{C}_j$ contient les indices des données de la classe j (i.e du j-ème sujet). Pour un image $\mathbf{x} \in \mathbb{R}^p$, on notera

$$\omega(\mathbf{x}) = \begin{pmatrix} \langle \mathbf{x} - \overline{\mathbf{x}}, \mathbf{u}_1 \rangle \\ \vdots \\ \langle \mathbf{x} - \overline{\mathbf{x}}, \mathbf{u}_\ell \rangle \end{pmatrix}$$

le vecteur de ses ℓ composantes principales (*noter le recentrage des données par la moyenne empirique $\overline{\mathbf{x}}$, qui est essentiel*).

Classifieur k-NN. La classification par *plus proches voisins* (ou k-NN pour « k-Nearest Neighbors ») consiste à déterminer les k plus proches voisins de $\omega(\mathbf{x})$ parmi l'ensemble des composantes principales des données de la base d'apprentissage. On estime ensuite la classe de $\mathbf{x}$ comme la classe la plus représentée parmi ces k voisins. Plus formellement, notons

$$\mathcal{V}(\mathbf{x}) = \underset{\{i_1,\dots,i_k\} \subset [\![1,n]\!]}{\operatorname{argmin}} \sum_{j=1}^{k} \left\| \omega(\mathbf{x}) - \omega(\mathbf{x}_{i_j}) \right\|_2 .$$

l'ensemble des k plus proches voisins de $\mathbf{x}$ dans la base d'apprentissage $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$. Alors le classifieur k-NN est donné par [2]

$$\varphi(\mathbf{x}) = \underset{j \in [\![1,m]\!]}{\operatorname{argmax}} \left| \mathcal{V}(\mathbf{x}) \cap \mathcal{C}_j \right| .$$

2. Dans le cas où les argmax / argmin ne sont pas uniques, on choisit un élément arbitrairement.

1. Implémenter le classifieur k-NN, et commenter, *sans faire d'expérimentation*, les compromis autour du choix de l'hyperparamètre k.

Le classifieur construit précédemment sera appliqué sur les images de la base de test contenues dans les dossiers type `test`. Pour évaluer les performances du classifieur, on calculera la *matrice de confusion*, qui est une matrice $m \times m$ affichant les taux de bonne et mauvaise classifications. Un exemple est donné en FIGURE 2 ; en particulier, plus un classifieur est performant, plus sa matrice de confusion est proche de la matrice identité.

Classes	**1**	**2**	**3**
1	0.7	0.2	0.1
2	0.2	0.8	0
3	0.1	0	0.9

FIGURE 2: Exemple de matrice de confusion ($m = 3$ classes)

La fonction `confmat`, qui est fournie, permet de calculer la matrice de confusion ainsi que le taux d'erreur global de la classification (incluant tous les individus).

2. Calculer les matrices de confusion sur les bases de test, ainsi que le taux d'erreur global. Au vu des spécificités de chaque base d'images décrites précédemment, commenter les résultats obtenus.

Classifieur gaussien. Le classifieur k-NN n'exploite aucun modèle sur les données, et dépend par ailleurs du choix délicat du nombre de plus proches voisins à sélectionner. Une approche alternative peut être envisagée en proposant un modèle statistique sur les vecteurs de composantes principales des images, et en exploitant les *statistiques inférentielles* afin de concevoir un classifieur adapté. Pour la suite, on supposera alors que pour une image $\mathbf{x}$ de la i-ème classe, le vecteur des composantes principales est tel que

$$\omega(\mathbf{x}) \sim \mathcal{N}_{\mathbb{R}^\ell}\left(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}\right),$$

c'est-à-dire suit une loi gaussienne multivariée de moyenne $\boldsymbol{\mu}_i \in \mathbb{R}^\ell$ et de matrice de covariance $\boldsymbol{\Sigma}$ (de taille $\ell \times \ell$) Rappelons que la loi $\mathcal{N}_{\mathbb{R}^\ell}\left(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}\right)$ admet pour densité

$$f_i(\mathbf{y}) = \sqrt{\frac{1}{2\pi \det(\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2}(\mathbf{y} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}^{-1}(\mathbf{y} - \boldsymbol{\mu}_i)\right).$$

On propose alors d'estimer la classe de la donnée $\mathbf{x}$ par *maximum de vraisemblance*, en considérant le classifieur [3]

$$\begin{aligned}\varphi(\mathbf{x}) &= \underset{j \in [\![1,m]\!]}{\operatorname{argmax}} f_j\left(\omega(\mathbf{x})\right) \\ &= \underset{j \in [\![1,m]\!]}{\operatorname{argmin}} \left\| \boldsymbol{\Sigma}^{-1/2}\left(\omega(\mathbf{x}) - \boldsymbol{\mu}_j\right)\right\|_2^2.\end{aligned}$$

En pratique, les moyennes intra-classe $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m$ et la covariance $\boldsymbol{\Sigma}$ sont inconnues, mais peuvent être estimées empiriquement à partir des données d'apprentissage par

$$\hat{\boldsymbol{\mu}}_j = \frac{1}{|\mathcal{C}_j|} \sum_{i \in \mathcal{C}_j} \omega(\mathbf{x}_i)$$

3. Si l'argmin n'est pas unique, on en sélectionne un arbitrairement.

et

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n} \sum_{j=1}^{m} \sum_{i \in \mathcal{C}_j} (\omega(\mathbf{x}_i) - \hat{\boldsymbol{\mu}}_j) \left(\omega(\mathbf{x}_i) - \hat{\boldsymbol{\mu}}_j\right)^T .$$

3. Représenter sur une même figure et 4 graphiques distincts, les nuages de points des couples de composantes principales $(1,2), (2,3), (3,4), (4,5)$ pour toutes les images de 3 individus sélectionnés au choix dans la base d'apprentissage. Ajouter également les moyennes intra-classes $\hat{\boldsymbol{\mu}}_1, \ldots, \hat{\boldsymbol{\mu}}_m$ correspondantes. Commenter les résultats obtenus au regard du modèle statistique adopté.
4. Implémenter le classifieur gaussien. Sachant que $\operatorname{rg}\left(\hat{\boldsymbol{\Sigma}}\right) \leq \min(\ell - m, n)$, mettre en lumière une limitation potentielle.
5. Reprendre les expérimentations menées pour le classifieur k-NN et commenter les différences de performances entre les deux méthodes.
6. Proposer une modification du classifieur gaussien pour intégrer la situation plus générale où les matrices de covariance de chaque classe sont potentiellement différentes. Quelle sont les avantages et les inconvénients ?

Extension (Bonus). L'algorithme précédent permet de rapprocher un visage inconnu d'un visage de la base d'apprentissage. Néanmoins, il ne permet pas de *détecter* un visage n'appartenant pas à la base.

7. Proposer et mettre en œuvre une modification du classifieur gaussien, permettant de détecter un visage n'appartenant pas à la base.
8. Tester les performances de la méthode proposée en retirant un ou plusieurs individus des bases d'apprentissage fournies. Commenter les résultats.

Références

[1] `http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html`.

[2] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many : Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6) :643–660, 2001.

[3] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface : Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[4] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1) :71–86, 1991.