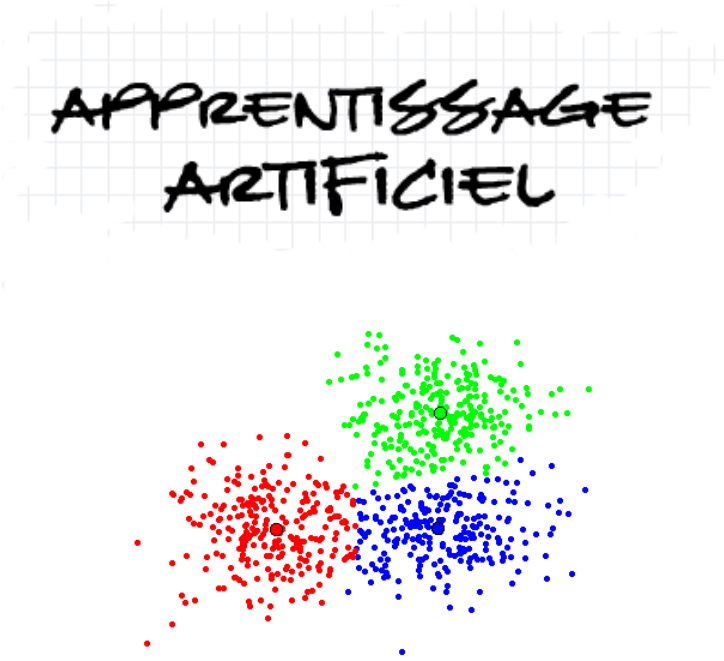
Cours, Exercices et Travaux Pratiques

Vincent Charvillat



INP ENSEEIHT

Département Informatique et Mathématiques Appliquées

Tables des matières

Avant-propos

Ce document regroupe des notes de cours, des exercices et des sujets de travaux pratiques utiles à l'unité d'enseignement intitulée «Apprentissage et Applications». Ce cours est dispensé au sein de la majeure mathématique en seconde année du département informatique et mathématiques appliquées de l'ENSEEIHT. Ce document repose naturellement sur un socle bibliographique important. Certains passages du cours sont ainsi directement inspirés des ouvrages suivants :

- The Elements of Statistical Learning, Trevor Hastie, Robert Tibshirani, Jerome Friedman, second edition, Springer, 2009
- Pattern Recognition and Machine Learning, Christopher M. Bishop, Springer, 2006
- Pattern Classification, R. Duda, P. Hart, G. Stork, second edition, Wiley, 2001
- Reinforcement Learning: An Introduction, R. Sutton and A. Barto, MIT Press, 1998
- Apprentissage Artificiel, A. Cornuéjols L. Miclet, Eyrolles, 2003

Le premier ouvrage, bien qu'assez technique, est vraisemblablement *la* référence principale. Certaines figures (mises à la disposition des enseignants) sont issues de ces ouvrages. Les cours de J-Y. Tourneret , P. Besse, M. Marchand, T. Jaakkola, F. Garcia ainsi que plusieurs thèses de doctorat (C. Goutte, H. La Rochelle), m'ont également permis de reprendre et d'adapter quelques démarches que j'ai trouvées particulièrement didactiques. Enfin, les sujets de travaux pratiques doivent beaucoup aux intervenants qui m'ont récemment accompagné dans cet enseignement : P. Parisot, J. Guenard et J.D. Durou.

Thème 1

Introduction à l'apprentissage

Vincent Charvillat
Septembre 2010

Ce premier thème introduit le cours d'apprentissage et les premiers éléments de terminologie au travers d'applications de reconnaissance de formes. Quelques classifieurs élémentaires sont également présentés.

1.1 Terminologie illustrée

En apprentissage articificiel, on s'intéresse à des *mécanismes de prédiction* dans des contextes où les données manipulées sont *aléatoires*. Ces dernières étant vues comme des réalisations de variables aléatoires, on parlera donc aussi d'*apprentissage statistique*.

L'exemple introductif classique est celui de la reconnaissance automatique de l'écriture manuscrite. La figure 1.1 montre des réalisations manuscrites aléatoires des chiffres. Un algorithme d'apprentissage artificiel permet de mettre au point un mécanisme de prédiction du nombre écrit à partir d'une donnée d'entrée (par exemple une imagette numérique de 64×64 pixels). En d'autres mots, on veut prédire la classe d'appartenance d'une imagette quelconque ($\in \mathbb{R}^{64\times 64}$) parmi les dix classes possibles associées aux chiffres $\{0, 1, \ldots, 9\}$ possibles. Un tel problème de prédiction d'une classe d'appartenance est aussi appelé un problème de *classification*.

Plus formellement, on considèrera pour la classification, un espace d'entrée $\mathcal{X}$ ($\mathbb{R}^{64\times 64}$ dans l'exemple ci-dessus) et un espace discret de sortie à k classes $\mathcal{Y} = \{0, \ldots k-1\}$ ($k = 10$ dans l'exemple). Un *prédicteur* (ou *classifieur*) est une fonction $h : \mathcal{X} \to \mathcal{Y}$ qui prédit pour une donnée d'entrée $x_i \in \mathcal{X}$ (une imagette dans l'exemple), sa classe : $\widehat{y_i} = h(x_i) \in \mathcal{Y}$.

Pour savoir si cette prédiction est correcte, on peut utiliser un ensemble de données supervisées. Une paire $z_i = (x_i, y_i)$ forme une donnée supervisée, si un expert a étiqueté la donnée d'entrée x_i avec sa classe correcte y_i. Plus formellement,

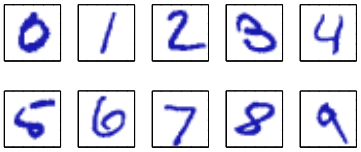


Figure 1.1: Réalisations manuscrites de chiffres.

un ensemble de n données supervisées est, par exemple, noté $D = \{z_i\}_{i=1...n}$ avec $z_i = (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$. Pour savoir si une prédiction $\widehat{y_i} = h(x_i) \in \mathcal{Y}$ d'un classifieur h est bonne, on la compare alors à l'étiquette y_i correcte (donnée par l'expert). Une fonction non-négative $e : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_+$ dite de *perte* exprime l'erreur de prédiction (ou, entre d'autres mots, l'ampleur de l'écart entre y_i et la prédiction $\widehat{y_i}$). Naturellement, on prend souvent une perte nulle (resp. strictement positive) si la prédiction est correcte (resp. fausse) : $e(\widehat{y_i}, y_i) = 0$ si $\widehat{y_i} = y_i$ (resp. $e(\widehat{y_i}, y_i) > 0$ si $\widehat{y_i} \neq y_i$).

Un bon classifieur devra conduire à des pertes minimales sur l'ensemble des données (ou exemples) qu'il aura à classer (c'est-à-dire potentiellement tout $\mathcal{X} \times \mathcal{Y}$). Mais, toutes les paires $z_i = (x_i, y_i)$ ne sont pas équiprobables. Chaque paire est la réalisation d'une v.a. $Z = (X, Y)$. Les distributions de probabilités associées aux variables aléatoires X, Y, Z et leurs dépendances [1] sont des ingrédients clés en apprentissage statistique où l'on cherchera à minimiser l'espérance des pertes.

1.2 Une multitude de prédicteurs possibles

Ce principe général de réduction de l'espérance des pertes induites par de mauvaises prédictions sera formalisé dans la suite. Il faut toutefois noter qu'il permet de comparer une multitude de prédicteurs de natures parfois très différentes.

Les prédicteurs peuvent être définis mathématiquement (un prédicteur peut être une fonction dont on connaît la forme analytique) ou de manière procédurale (un prédicteur peut être associé au parcours d'un arbre de décision ou de régression). Les prédicteurs peuvent être déterministes (une entrée x conduit invariablement à la même prédiction) ou probabiliste (sachant l'entrée, une prédiction est émise avec une certaine probabilité). Pour un même type de prédicteur (qu'il s'agisse, par exemple, d'une fonction polynomiale ou d'un arbre), *les prédicteurs peuvent être de complexité variable* : le degré du polynôme peut croître, la profondeur de l'arbre peut varier. Si les prédicteurs sont paramétriques, la complexité croît avec le nombre de paramètres (ou degré de liberté) considérés.

Dans la suite, nous illustrons cette multiplicité de prédicteurs possibles en introduisant plusieurs classifieurs différents : un classifieur binaire linéaire, un classifieur quadratique, un arbre de décision...

[1] Y doit dépendre de X si on espère faire une prédiction à partir d'une réalisation de X

1.2.1 Classifieurs binaires et quadratiques

Un classifieur binaire linéaire $h^{\theta}_{\mathbf{w}}$ est défini, de manière paramétrique, par :

- un vecteur $\mathbf{w} = (w_1, .., w_p)^T$ de p poids,
- un seuil de décision θ.

Les espaces d'entrée et de sortie sont $\mathcal{X} \subset \mathbb{R}^p$, $\mathcal{Y} = \{+1, -1\}$
Une prédiction s'exprime alors selon :

$$\widehat{y} = h^{\theta}_{\mathbf{w}}(x) = \text{sgn}(\mathbf{w}^T x - \theta) \tag{1.1}$$

où $\text{sgn}(s) = +1$ lorsque $s > 0$ et -1 lorsque $s \leq 0$

On dit que ce classifieur binaire (cas à deux classes) est linéaire car il combine linéairement les composantes du vecteur d'entrée x avec les poids rangés dans le vecteur de paramètres $\mathbf{w}$. Un algorithme d'apprentissage consiste à ajuster les paramètres $\mathbf{w}$ et θ pour qu'ils soient cohérents avec un ensemble de données supervisées ou *échantillon d'apprentissage* $D = \{(x_i, y_i)\}_{i=1...n} \in \mathcal{X} \times \mathcal{Y}^n$.

EXERCICE On considère un classifieur binaire linéaire comme défini ci-dessus avec $p = 2$, $\mathcal{X} = [0.0, 1.0]^2$ et $\theta = 0$.

1. Dessiner arbitrairement 10 points $x_i = (x_i^1, x_i^2)^T, i = 1, \ldots, 10$ dans le pavé $[0.0, 1.0]^2$. Soit $\mathbf{w_0} = (-1/\sqrt{2}, 1/\sqrt{2})^T$, donner une interprétation géométrique dans le plan 2D ($\mathcal{X} \subset \mathbb{R}^2$), des prédictions $h^{\theta=0}_{\mathbf{w_0}}(x_i)$: on montrera que le classifieur sépare le plan en deux régions (hyperplans) de décision. On dit qu'il s'agit d'un *séparateur linéaire*.

2. Donner l'équation de la *frontière de décision* séparant les deux *régions de décision*.

3. Etant donné un échantillon d'apprentissage $D = \{(x_i, y_i)\}_{i=1...n} \in \mathcal{X} \times \mathcal{Y}^n$. $\mathcal{Y} = \{+1, -1\}$. Donner un exemple d'échantillon D (avec $n \geq 4$) pour lequel il n'existe pas de vecteur $\mathbf{w}$ tel que les prédictions $\widehat{y_i} = h^0_{\mathbf{w}}(x_i) = \text{sgn}(\mathbf{w}^T x_i)$ soient toutes correctes (c'est-à-dire $\forall i, \widehat{y_i} = y_i$). On dira que D n'est pas linéairement séparable.

4. Quand une prédiction $\widehat{y_t} = \text{sgn}(\mathbf{w}^T x_t)$ diffère de (resp. coincide avec) l'étiquette correcte y_t, quel est le signe de $y_t \mathbf{w}^T x_t$?

5. La règle de mise à jour dite du *perceptron* consiste, étant donné un vecteur de paramètres $\mathbf{w}_k$ et une prédiction infructueuse pour une donnée de D, à modifier le vecteur de paramètre selon :

$$\mathbf{w_{k+1}} \leftarrow \mathbf{w_k} + y_t * x_t \quad \text{si} \quad \widehat{y_t} = \text{sgn}(\mathbf{w_k}^T x_t) \neq y_t$$

On vous demande de montrer (grâce aux observations faites à la question précédente) que cette règle de mise à jour conduit, si on l'utilise itérativement, à une prédiction correcte pour l'entrée x_t.

6. Proposer un algorithme d'apprentissage dont vous pensez qu'il pourrait converger vers un vecteur $\mathbf{w}_D$ satisfaisant au mieux un échantillon d'apprentissage $D = \{(x_i, y_i)\}_{i=1\ldots n}$.

7. Un tel classifieur linéaire binaire prend-il en compte explicitement les relations entre les composantes (par exemple $x_i^{j_1}$ et $x_i^{j_2}$) d'un vecteur d'entrée x_i ? Citer une application de reconnaissance de formes où ces relations pourraient avoir un intérêt.

Par extension avec le classifieur linéaire précédent, on peut introduire un classifieur quadratique dont la prédiction est donnée par :

$$\widehat{y} = h_\omega(x) = \text{sgn}\left(\omega_0 + \sum_i \omega_i x_i + \sum_{i,j} \omega_{i,j} x_i x_j\right) \tag{1.2}$$

Il est clair que les classifieurs quadratiques sont plus généraux que les classifieurs linéaires. Ils conduisent à des régions et frontières de décisions plus complexes que celles issues d'une séparation linéaire. Si on nomme $\mathcal{H}_1$ (resp. $\mathcal{H}_2$, resp. $\mathcal{H}_k$), la classe des prédicteurs linéaires (resp. quadratiques, resp. d'ordre k), on a une hiérarchie de classes telle que :

$$\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_k \tag{1.3}$$

La croissance de la *complexité des classifieurs* va avec celle du nombre de paramètres à estimer. On peut donc d'emblée distinguer deux problèmes complémentaires :

- Etant donnée une classe de fonction $\mathcal{H}$, quel est le meilleur prédicteur $h \in \mathcal{H}$ possible pour un problème d'apprentissage donné ? Il s'agit, par exemple, d'estimer les meilleurs paramètres ($\mathbf{w}$ et θ) d'un classifieur linéaire connaissant un échantillon d'apprentissage D. C'est un problème d'estimation (au sens des statistiques) ou d'optimisation (au sens de l'analyse).

- Etant donnée une hiérarchie de modèles de prédiction possibles ($\mathcal{H}_1 \subset \mathcal{H}_2 \subset \cdots \subset \mathcal{H}_k$), quelle est la meilleure classe possible ? Il s'agit par exemple de choisir entre un classifieur linéaire ou quadratique. En apprentissage, effectuer ce choix revient à résoudre un problème de *sélection de modèle* sur lequel nous reviendrons plus en détail.

1.2.2 Arbre de décision

Les séparateurs précédents sont définis analytiquement. On peut aussi utiliser des approches algorithmiques ou procédurales pour définir des prédicteurs dans le domaine de la classification ou de la régression. L'exemple des arbres de décision est révélateur de cette possibilité. La figure 1.2 présente la structure arborescente d'un classifieur binaire qui opère sur un espace d'entrée $\mathcal{X} = [0.0, 1.0]^2 \subset \mathbb{R}^2$ et peut prédire les deux valeurs qualitatives de $\mathcal{Y} = \{+1, -1\}$. Cet arbre de décision est un arbre de discrimination binaire qui consiste à classer une donnée x de $\mathcal{X}$ à l'issue du parcours d'une séquence de *noeuds*.

Un noeud est défini par le choix conjoint d'une variable (parmi les composantes de la variable d'entrée) et d'un prédicat qui induit une *division* des données en deux classes. A titre d'exemple, la racine de l'arbre de la figure 1.2 est un noeud qui sépare les données d'entrée $x \subset \mathbb{R}^2$ selon que la variable associée à la première composante (notée $X1$) de x est supérieure ou non à un seuil (0.5). Si les variables ne sont pas quantitatives, l'utilisation d'un seuil doit être remplacée par une autre fonction logique. Il est aisé à partir de cet arbre de décision de déterminer les régions et frontières de décisions du classifieur binaire ainsi obtenu.

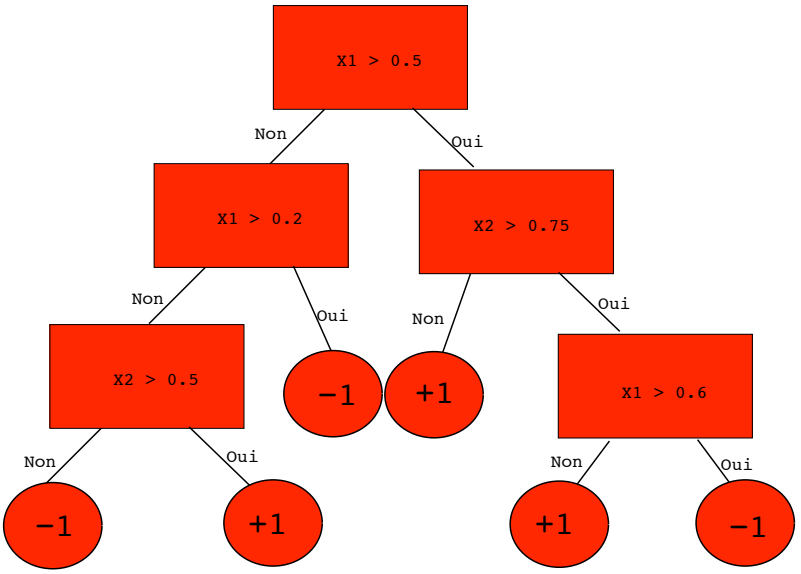


Figure 1.2: Un arbre de décision

Il reste trois points délicats à aborder pour construire (apprendre) automatiquement un tel classifieur :

i) Parmi toutes les *divisions* admissibles à un instant donné lors de la création d'un tel arbre, il faut pouvoir identifier la meilleure division possible. Ce critère de sélection est à bien choisir.

ii) Il faut également choisir une règle permettant de décider si un noeud est terminal ou pas.

iii) Enfin, les noeuds terminaux (c'est-à-dire les feuilles de l'arbre), doivent être étiquetés par les valeurs de l'espace de sortie $\mathcal{Y}$ selon un principe à mettre au point.

Le second point ii), qui vise à interrompre le développement en profondeur de l'arbre, revient à gérer la *complexité* du classifieur obtenu au sens exactement équivalent à ce qui a été vu plus haut. Un arbre trop développé, c'est-à-dire trop compliqué, peut être obtenu si on cherche à expliquer «coûte que coûte» un ensemble de données supervisées. L'arbre maximal, le plus développé, conduit à des régions (et frontières) de décision très complexes. Ce classifieur peut se révéler défaillant pour la

prédiction de nouvelles données comme nous l'expliquerons formellement plus loin. L'élagage d'un arbre maximal revient alors à simplifier le modèle pour atteindre le meilleur compromis entre fidélité aux données d'apprentissage et bonne capacité de prédiction sur de nouvelles données.

1.3 Les trois modes d'apprentissage

On distingue généralement trois modes principaux d'apprentissage :

- l'apprentissage supervisé (par «instruction»),
- l'apprentissage par renforcement (par «évaluation»),
- l'apprentissage non supervisé.

1.3.1 Apprentissage supervisé

L'**apprentissage supervisé** est celui que nous avons tacitement utilisé dans les illustrations précédentes. Il s'agit du scénario d'apprentissage pour lequel un expert fournit d'emblée des exemples de ce qu'il faut apprendre. Exemples à partir desquels un prédicteur est bâti. On s'intéresse typiquement à un prédicteur h qui prédit une sortie $\widehat{y}$ à partir d'un stimulus $x \in \mathcal{X}$ selon le schéma suivant :

$$x \in \mathcal{X} \rightarrow \boxed{\text{h ?}} \rightarrow \widehat{y} = h(x) \in \mathcal{Y}$$

Comme nous l'avons déjà dit, un ensemble de n données supervisées, appelé également «échantillon d'apprentissage», est donné par l'expert et usuellement noté $D = \{(x_i, y_i)\}_{i=1..n}$. Il regroupe les dires de l'expert qui a indiqué qu'à la donnée d'entrée x_i devait correspondre la valeur de sortie y_i. Le prédicteur h peut alors être appris automatiquement en minimisant, pour toutes les données disponibles dans D, les écarts entre ses prédictions $\widehat{y_i} = h(x_i)$ et les sorties correctes y_i. Une fonction de *perte* à bien choisir mesure numériquement ces écarts (les erreurs de prédiction). Retenons que la solution au problème de prédiction posé est connue, grâce aux instructions fournies par l'expert, pour un ensemble de données d'entrées $x_i, i = 1, .., n$.

1.3.2 Apprentissage par renforcement

Par opposition, en **apprentissage par renforcement** la solution au problème n'est jamais accessible (même pour un ensemble limité de données d'entrée). Il n'y a pas d'«oracle» comme dans le cas précédent. On considère le plus souvent un agent qui prédit des actions $\widehat{a}$ (appartenant à un ensemble discret et fini d'actions noté $\mathcal{A}$) à partir des états de son environnement. Au temps t, l'agent perçoit cet état comme une entrée $s_t \in \mathcal{S}$ dans le schéma suivant :

$$\begin{array}{rl} s_t \in \mathcal{S} \rightarrow & \boxed{\pi\ ?} \rightarrow \widehat{a} = \pi(s_t) \in \mathcal{A} \\ r + s_{t+1} \nwarrow & \boxed{\text{feedback}} \swarrow \end{array}$$

Comme précédemment, l'agent doit faire une prédiction mais celle-ci n'est pas à considérer seule : elle s'insère dans une séquence d'actions dont on souhaite qu'elle soit optimale. Plus précisément, l'agent cherche une politique (notée π) qui prédit en s_t la meilleure action selon $\widehat{a} = \pi(s_t)$. Cette action a un effet séquentiel sur l'environnement dynamique dans lequel opère l'agent. Ce dernier perçoit en guise de «feedback», un nouvel état s_{t+1} et un «signal de renforcement» noté r ci-dessus dans la figure. Ce signal numérique est à interpréter comme une récompense qui donne à l'agent une information sur la pertinence des actions réalisées précédemment. L'objectif de l'agent est alors, en chaque état, de maximiser l'espérance des récompenses futures. On parle d'un apprentissage par évaluation au sens où on laisse l'agent opérer par essai-erreur. Les choix d'actions conduisant à de bons cumuls de récompenses seront ainsi renforcés. Cette stratégie d'apprentissage par évaluation est plus délicate que celle qui opère selon les instructions de l'expert. La quantité d'information utilisable pour apprendre par renforcement est moindre que celle disponible dans le cas supervisé.

1.3.3 Apprentissage non-supervisé

La quantité d'information externe disponible pour apprendre dans le cas de l'**apprentissage non supervisé** est encore inférieure puisque réduite à rien. Dans le cas non-supervisé, on ne dispose ni d'«oracle» ni de signaux de renforcement pour guider l'apprentissage. Seule une collection de données brutes ou de mesures $D = \{x_i\}_{i=1...n}$ est disponible. Le schéma fonctionnel se réduit au suivant :

$$x \in \mathcal{X} \rightarrow \boxed{\text{h ?}}$$

La prédiction de similarités présentes au sein des données ou la découverte d'autres structures descriptives sont les objectifs principaux des méthodes d'apprentissage non-supervisé. Les techniques de «clustering» visant le regroupement d'individus similaires et donc, potentiellement, de même classe (notée $h(x)$ dans la figure) sont nombreuses. Se posent aussi, de manière non-supervisée des problèmes de représentation compacte ou de visualisation de données multidimensionnelles.

1.4 Bilan

Dans la suite du cours, les trois modes d'apprentissage seront abordés. Dans ces trois cas, il s'agit d'apprendre, par «induction», des lois les plus générales possibles à partir de données observées, d'exemples et, éventuellement, de contre-exemples.

Thème 2

Généralisation

Vincent Charvillat
Septembre 2010

Dans ce second thème, on formalise la notion de généralisation qui est centrale en apprentissage statistique. On souligne aussi la similitude et les différences entre régression et classification. Le cadre est celui de l'apprentissage supervisé.

2.1 Complexité optimale d'un classifieur

A titre d'exemple introductif, on considère un classifieur dont l'espace d'entrée est $\mathcal{X}$ et l'espace discret de sortie à k classes est $\mathcal{Y} = \{0, \dots k-1\}$. On considère un ensemble de données supervisées $D = \{z_i\}_{i=1\dots n}$. On décompose classiquement les données selon $z_i = (x_i, y_i)$ avec $x_i \in \mathcal{X}$ et $y_i \in \mathcal{Y}$. Chaque donnée z_i est la réalisation d'une v.a. $Z = (X, Y)$. On notera $P_Z(z) = P_{X,Y}(x, y)$ (resp. $P_X(x)$, $P_{Y|X}(y)$), les distributions jointes, (resp. d'entrée, conditionnelle) associées.

Nous allons décrire trois choix importants à effectuer en vue de l'apprentissage d'un tel classifieur :

- choix de la fonction perte,
- choix des classes de prédicteurs considérées (un ou plusieurs modèles de complexités distinctes)
- apprentissage, au sein de chaque classe, des meilleurs classifieurs par minimisation du risque empirique.

2.1.1 Fonctions de perte pour la classification

Comme nous l'avons vu précédemment, les instructions données par l'expert ($\forall i$, «classer la donnée d'entrée x_i avec l'étiquette y_i») permettent d'apprendre un classifieur à partir de l'ensemble d'apprentissage D en minimisant les écarts entre ses

prédictions et les vraies valeurs. En classification, la fonction de perte «non informative» (notée e_z) est la plus simple à utiliser pour mesurer cet écart. Elle est définie par :

$$e_z(\widehat{y}, y) = \begin{cases} 0 \ si \ y = \widehat{y} \\ 1 \ si \ y \neq \widehat{y} \end{cases} \tag{2.1}$$

On dit que la perte e_z est non-informative car elle se contente de dire, de manière binaire, si la prédiction $\widehat{y}$ est correcte (resp. fausse) et a donc un coût nul (resp. unitaire). On parle aussi de perte 0-1 pour e_z. D'autres pertes peuvent être imaginées pour quantifier de manière plus fine l'écart entre une prédiction imparfaite et la classe correcte. Dans certaines applications, il est clair que toutes les erreurs de classification ne sont pas de gravité équivalente et doivent donc avoir des coûts différenciés. Une fonction perte générale e doit donc simplement vérifier :

$$e(\widehat{y}, y) = \begin{cases} 0 \ si \ y = \widehat{y} \\ > 0 \ si \ y \neq \widehat{y} \end{cases} \tag{2.2}$$

2.1.2 Hiérarchie de modèles de complexité croissante

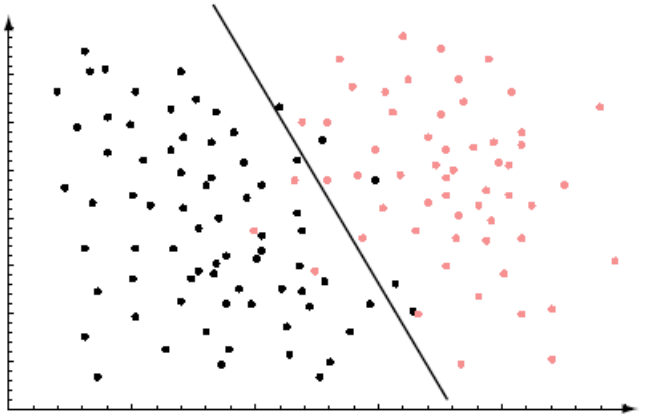


Figure 2.1: Un classifieur «trop simple»

Dans notre exemple, on choisit trois modèles de complexité croissante :

- les classifieurs linéaires (classe $\mathcal{H}_1$),
- les classifieurs quadratiques (classe $\mathcal{H}_2$ avec $\mathcal{H}_1 \subset \mathcal{H}_2$),
- des classifieurs beaucoup plus compliqués (classe $\mathcal{H}_c$).

Les formes possibles des frontières de décision associées à cette hiérarchie de modèles sont illustrées par la figure 2.1 pour un classifieur linéaire, la figure 2.2 pour un classifieur quadratique et la figure 2.3 pour un classifieur de complexité la plus forte. Ces figures présentent un exemple de classification binaire pour lequel les données des deux classes présentes dans D sont colorées en noir et rouge.

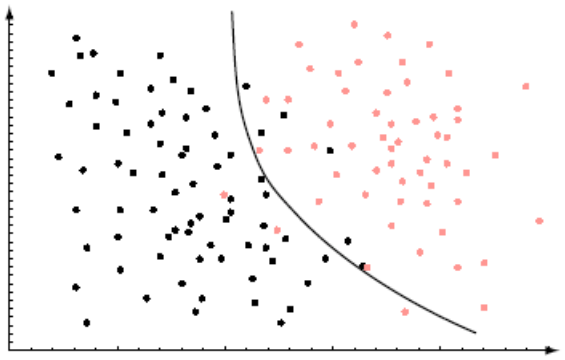


Figure 2.2: Un classifieur de complexité «moyenne»

2.1.3 Minimisation du risque empirique

Les classifieurs précédents sont obtenus à l'issue de la minimisation du risque empirique. On appelle, pour un classifieur candidat h (pris dans une classe de fonctions $\mathcal{H}$), un ensemble de données supervisées $D = \{(x_i, y_i)\}_{i=1\ldots n}$ et une perte e, «risque empirique» la quantité :

$$R_D(h) = \frac{1}{n} \sum_{i=1}^{n} e(h(x_i), y_i) \tag{2.3}$$

Le classifieur dont la frontière de décision est montrée sur la figure 2.1 est donc le classifieur h_1^* solution de :

$$h_1^* = \underset{h \in \mathcal{H}_1}{\operatorname{argmin}}\ R_D(h) \tag{2.4}$$

Apprendre le meilleur classifieur d'une classe donnée à partir de D revient donc à résoudre un problème d'optimisation. On appelle aussi «erreur d'apprentissage» le risque empirique minimal obtenu pour la solution h_1^* (après apprentissage) :

$$\texttt{err}(h_1^*) = R_D(h_1^*) \tag{2.5}$$

Cette erreur évalue le cumul des pertes engendrées par les mauvaises prédictions du classifieur sur l'échantillon d'apprentissage D. De même on notera h_2^* (resp. h_c^*), les meilleurs classifieurs obtenus par minimisation du risque empirique sur les classes $\mathcal{H}_2$ (resp. $\mathcal{H}_c$). On remarquera, qu'en supposant que la perte e_z est celle utilisée pour apprendre le classifieur linéaire de la figure 2.1, l'erreur d'apprentissage est alors facile à déterminer grâce à la figure.

Lorsque les modèles choisis pour les classifieurs deviennent de plus en plus complexes, c'est-à-dire qu'ils disposent de plus en plus de degrés de liberté, les régions de décisions peuvent plus facilement s'ajuster aux données d'apprentissage présentes dans D. On observe ainsi sur la figure 2.3 que l'erreur d'apprentissage $\texttt{err}(h_c^*)$ est nulle : $\texttt{err}(h_c^*) = R_D(h_c^*) = 0$. Grâce à la flexibilité du classifieur, toutes les prédictions sont correctes et la frontière de décision sépare les données sans commettre d'erreurs parmi les données d'apprentissage de D.

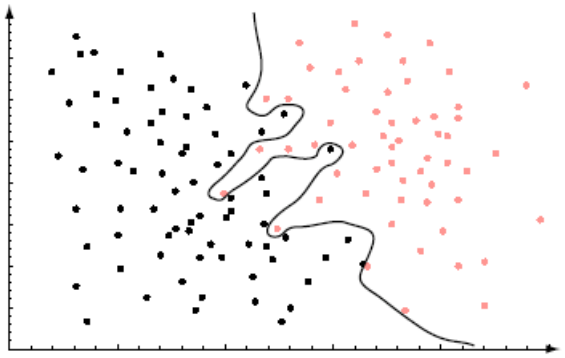


Figure 2.3: Un classifieur «trop complexe»

Un classifieur qui réussit à ramener une erreur d'apprentissage à zéro est-il pour autant le meilleur ? La réponse est négative. La seule «minimisation du risque empirique» ne suffit pas pour obtenir un «apprentissage optimal».

2.2 Risque espéré et généralisation

Le meilleur classifieur est celui dont la performance ne se limite pas à correctement classifier les données appartenant à l'ensemble d'apprentissage : on souhaite aussi qu'il prédise correctement la classe de *nouvelles données* pour lesquelles l'expert ne s'est pas prononcé. On dira que le meilleur classifieur est celui qui possède la plus forte capacité de *généralisation*. En guise d'illustration, le classifieur de la figure 2.3 possède des frontières qui satisfont parfaitement l'ensemble d'apprentissage ($\texttt{err}(h_c^*) = R_D(h_c^*) = 0$) mais dont on pressent qu'elles sont «trop complexes» pour correctement prédire de nouveaux exemples situés à la frontière des deux classes.

Formellement, on appelle "Expected Prediction Error (EPE)" ou risque espéré (R) d'un classifieur $h : \mathcal{X} \to \mathcal{Y}$ pour une perte non négative e, la quantité :

$$R(h) = EPE(h) = E_{(X,Y)}[e(h(X), Y)] \tag{2.6}$$

Cette quantité mesure l'espérance des pertes dues aux prédictions de h sur l'ensemble des données susceptibles d'être générées par l'application (et non pas uniquement sur les données supervisées). Chaque nouvelle donnée (ou exemple) se réalise selon une mesure d'entrée x (réalisation de la variable aléatoire X) et correspond à une classe y inconnue (réalisation de la variable aléatoire Y). Avec l'EPE ou le risque espéré, on mesure donc une espérance des écarts de prédiction lorsque le couple de variables aléatoires X, Y varie selon une distribution $P_{X,Y}(x, y)$. Le meilleur classifieur est celui qui rend l'EPE minimale. En apprentissage statistique, la difficulté principale vient du fait que la loi des données n'est généralement pas connue. Seul l'échantillon d'apprentissage nous informe «empiriquement» sur la distribution des données. La taille de l'échantillon est donc un élément fondamental. Plus on dispose de données supervisées, plus notre échantillon d'apprentissage est significatif et révélateur de la loi générale et inconnue des données. Dans le cas d'un

échantillon limité, on va donc chercher à contourner cette difficulté en introduisant des connaissances ou hypothèses supplémentaires. On tentera ainsi d'approcher (mathématiquement ou empiriquement) l'EPE en vue d'identifier les meilleurs prédicteurs possibles.

Dans le cas particulier de la classification, la loi conditionnelle de Y sachant X est discrète. On peut donc développer l'EPE sous la forme suivante :

$$R(h) = EPE(h) = E_{(X,Y)}[e(h(X), Y)] = \int_{\mathcal{X}} P_X(x) \left[\sum_{y \in \mathcal{Y}} e(h(x), y) P_{Y|X=x}(y) \right] dx \tag{2.7}$$

On utilise dans cette dernière expression le fait que les valeurs à prédire pour la classification sont prises dans un ensemble discret et fini. L'ensemble des idées précédentes peut être repris lorsque l'on veut prédire des valeurs continues, réelles. Dans ce dernier cas, le problème abordé au sens de l'apprentissage supervisé est celui, bien connu, de la régression.

2.3 Prédiction et régression

Un problème de régression fait appel aux mêmes ingrédients qu'un problème de classification supervisée. On s'intéresse aussi à un prédicteur h qui prédit une sortie $\widehat{t}$ à partir d'un stimulus $x \in \mathcal{X}$ selon le schéma suivant :

$$x \in \mathcal{X} = \mathbb{R}^q \rightarrow \boxed{\text{h ?}} \rightarrow \widehat{t} = h(x) \in \mathcal{T} = \mathbb{R}^m$$

On dispose également d'un ensemble de données supervisées qui nous sert d'échantillon d'apprentissage : $D = \{(x_i, t_i)\}_{i=1\ldots n}$. La différence fondamentale est que la variable de sortie (celle qu'il faut prédire) est une variable continue de $\mathcal{T} = \mathbb{R}^m$. On parle aussi de variable de sortie quantitative pour la régression par opposition avec les variables qualitatives utilisées en classification. L'espace d'entrée est le plus souvent de même nature que celui de sortie ($\mathcal{X} = \mathbb{R}^q$) mais de dimension éventuellement distincte ($q \neq m$).

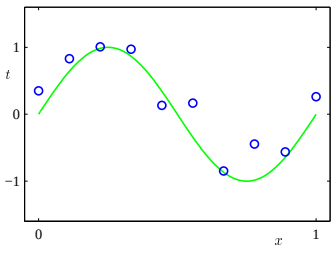


Figure 2.4: Régression et apprentissage.

La figure 2.4 illustre cette nouvelle situation et présente, grâce aux cercles bleus, un ensemble de données d'apprentissage $D = \{(x_i, t_i)\}_{i=1\ldots n}$ avec $x_i \in \mathbb{R}^1$ (en abscisse) et $t_i \in \mathbb{R}^1$ (en ordonnée). Ces points du plan se répartissent autour du

graphe d'une fonction (en vert) inconnue. Très souvent on considèrera que les données d'apprentissage sont des observations bruitées de points situés sur le graphe de cette fonction inconnue f que l'on va chercher à modéliser grâce à une fonction de prédiction (un prédicteur). On considèrera par exemple que les sorties t_i mises en jeu dans les données d'apprentissage de D s'expliquent selon $t_i = f(x_i) + \epsilon_i$ avec ϵ_i une réalisation d'un bruit additif aléatoire. On parlera de données fonctionnelles bruitées additivement.

La figure 2.5 suivante illustre le principe d'une fonction de prédiction h dont le graphe apparaît en rouge. Le prédicteur h qui joue un rôle similaire aux classifieurs vus précédemment prédit des sorties $\widehat{t_i} = h(x_i)$ pour chaque donnée d'entrée x_i. Les prédictions dans ce cas simple sont tout simplement les ordonnées sur le graphe du prédicteur h prises aux abscisses d'entrées. Intuitivement encore apprendre le meilleur prédicteur revient à trouver celui qui minimise l'écart entre prédictions $\widehat{t_i}$ et données disponibles t_i. Dans le cas de données fonctionnelles avec bruit blanc additif (et quelques autres détails), on retrouvera formellement plus loin le résultat intuitif suivant : le meilleur prédicteur h est la fonction inconnue f elle-même.

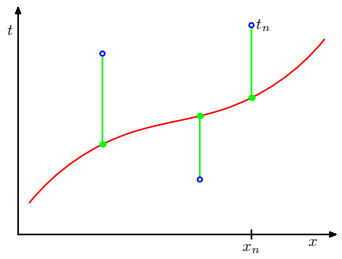


Figure 2.5: Prédiction en régression.

On peut alors reprendre la même démarche que celle décrite précédemment pour l'apprentissage de classifieurs. Les trois choix importants à effectuer sont strictement similaires :

- choix de la fonction perte,
- choix des classes de prédicteurs considérées (un ou plusieurs modèles de complexités distinctes)
- apprentissage, au sein de chaque classe, des meilleurs classifieurs par minimisation du risque empirique.

2.3.1 Fonctions de perte pour la régression

La différence fondamentale entre classification et régression se situe dans la dimension de l'espace de sortie : ici les sorties sont continues et appartiennent à $\mathcal{T} = \mathbb{R}^m$. Les normes L_p sont des choix naturels pour mesurer l'écart entre une sortie prédite $\widehat{t}$ et le vecteur correct t.

$$e(\widehat{t}, t) = ||\widehat{t} - t||_p^p \tag{2.8}$$

En pratique, on utilise très souvent la norme 2 ($p = 2$). Elle est liée aux méthodes des moindres carrés et à leurs interprétations statistiques au sens du maximum de vraisemblance dans le cas de bruits gaussiens [1].

La norme 1 a des propriétés de robustesse (aux données aberrantes) qui la rendent parfois intéressante. D'autres fonctions de pertes robustes sont aussi utilisables. Elles sont de forme générale $c(||\widehat{t} - t||)$ avec une fonction de pondération c choisie de sorte que les très grands écarts entre prédictions et données correctes soient pénalisés de manière (asymptotiquement) constante.

2.3.2 Hiérarchie de modèles de complexité croissante

Comme pour les classifieurs, une hiérarchie de prédicteurs de complexité croissante peut être considérée dans le cas de modèles de prédiction polynomiaux. Il s'agit de considérer un prédicteur paramétrique de type :

$$h_\omega(x) = \omega_0 + \omega_1 x + \omega_2 x^2 + \cdots + \omega_M x^M = \sum_{j=0}^{M} \omega_j x^j \tag{2.9}$$

Le degré M du prédicteur h_ω défini par le vecteur de paramètres $\omega = (\omega_0, \ldots \omega_M)^T$ donne une idée immédiate de sa complexité. De manière cohérente avec les notations précédentes, on dispose d'une hiérarchie de prédicteurs de complexité croissante :

- les prédicteurs constants de degré $M = 0$ (classe $\mathcal{H}_0$),
- les prédicteurs affines $M = 1$ (classe $\mathcal{H}_1$ avec $\mathcal{H}_0 \subset \mathcal{H}_1$),
- les prédicteurs quadratiques (classe $\mathcal{H}_2$ avec $\mathcal{H}_0 \subset \mathcal{H}_1 \subset \mathcal{H}_2$),
- des prédicteurs arbitrairement complexes avec le degré M (classe $\mathcal{H}_M$ avec $\mathcal{H}_0 \subset \mathcal{H}_1 \cdots \subset \mathcal{H}_{M-1} \subset \mathcal{H}_M$).

Dans les illustrations suivantes (figure 2.6) nous retrouvons le besoin de correctement *sélectionner le modèle* le plus approprié. Le prédicteur de degré nul (cas $M = 0$ et graphe rouge de la figure 2.6 (a)) prédit la même sortie pour toutes les entrées x. Il est trop «simple» vis-à-vis de la fonction autour de laquelle les données se structurent (cf. la courbe en vert reprenant la fonction de la figure 2.4). Les erreurs de prédictions sont importantes si les sorties attendues sont autour de 1 ou -1. Les commentaires sont les mêmes pour $M = 1$, (figure 2.6 (b)). Ces prédicteurs n'ont pas assez de degrés de liberté pour s'ajuster correctement aux données d'apprentissage et approcher correctement la fonction inconnue qui les explique (qui est visiblement «plus qu'affine»).

La figure 2.6 (d) présente une situation inverse pour laquelle $M = 9$. Ici le graphe du prédicteur est suffisamment compliqué pour que chaque donnée de l'ensemble d'apprentissage soit expliquée sans erreur. On a pour un vecteur $\omega^* \in \mathbb{R}^{10}$ bien choisi : $\forall i, \widehat{t_i} = h_{\omega^*}(x_i) = t_i$. Cette situation fait écho à celle rencontrée pour la classification et illustrée par la figure 2.3. Dans les deux cas, on n'observe aucune

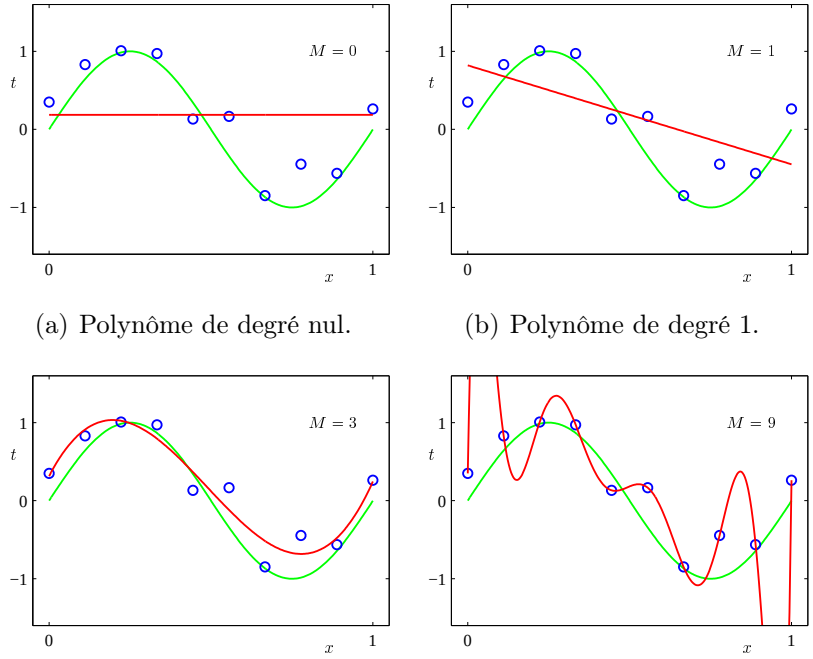
[1] Des rappels sur ce sujet sont proposés plus loin.

erreur/écart de prédiction sur l'ensemble d'apprentissage D. Le prédicteur est suffisamment complexe pour expliquer toutes les données d'apprentissage. Par contre, il est vraisemblablement «trop complexe» pour expliquer de nouvelles données prises hors de D. En particulier, dans la figure 2.6 (d), il apparaît clairement que les prédictions pour de nouvelles données proches du graphe vert, pour les abscisses immédiatement supérieures (resp. inférieures) à 0 (resp. 1), seront totalement erronées. On parle de «sur-apprentissage» (ou «overfitting» en anglais) pour expliquer que de tels prédicteurs, trop complexes, tentent d'expliquer (ici d'interpoler) toutes les données bruitées au lieu d'approcher la fonction f.

Ni trop simple, ni trop compliqué, le prédicteur présenté en rouge dans la figure 2.6 (c) semble être un prédicteur plus approprié pour expliquer à la fois :

- les données d'apprentissage (en bleu) et
- de nouvelles données susceptibles d'être générées comme celles de D (par exemple selon $t'_i = f(x'_i)+\epsilon'_i$ avec la même f mais de nouvelles réalisations pour le bruit ϵ'_i et/ou de nouvelles abscisses distinctes de celles des données présentes dans D).

On retrouve le point central en apprentissage statistique : la capacité de *généralisation* dans des termes exactement similaires à ceux de la section 2.2 précédente.



(a) Polynôme de degré nul.

(b) Polynôme de degré 1.

(c) Polynôme de degré 3.

(d) Polynôme de degré 9.

Figure 2.6: Hiérarchie de modèles pour la régression.

2.3.3 Minimisation du risque empirique pour la régression

L'apprentissage de chaque prédicteur de la figure 2.6 précédente revient à minimiser un risque empirique défini par l'équation (2.3) et déjà introduit pour la classification. La transposition entre classification et régression est immédiate. Seule la fonction de perte change.

Pour apprendre un prédicteur de degré M, à partir d'un ensemble de données supervisées $D = \{(x_i, y_i)\}_{i=1\dots n}$[2] on minimise pour $h \in \mathcal{H}_M$ le risque empirique $R_D(h)$ avec une perte e adaptée à la régression :

$$h^*_M = \underset{h\in\mathcal{H}_M}{\operatorname{argmin}} \left\{ R_D(h) = \frac{1}{n}\sum_{i=1}^{n} e(h(x_i), y_i) \right\} \tag{2.10}$$

Pour une perte quadratique (en norme L_2), apprendre le meilleur prédicteur au sein de $\mathcal{H}_M$ revient à résoudre un problème aux moindres carrés :

$$h^*_M = \underset{h\in\mathcal{H}_M}{\operatorname{argmin}} \left\{ \frac{1}{n}\sum_{i=1}^{n} ||y_i - h(x_i)||_2^2 \right\} \tag{2.11}$$

Dans le cas particulier précédent, on regroupe les coefficients polynomiaux dans le vecteur de paramètre $\omega = (\omega_0, \dots \omega_M)^T$. Les prédictions $h_\omega(x_i) = \sum_{j=0}^{M} \omega_j x_i^j$ sont linéaires en ω. Ce qui conduit à un problème aux moindres carrés linéaires :

$$\omega^*_M = \underset{\omega\in\mathbb{R}^M}{\operatorname{argmin}} \left\{ \frac{1}{n}\sum_{i=1}^{n} ||y_i - (1, x_i, x_i^2, \dots x_i^j, \dots x_i^M)\omega||_2^2 = \frac{1}{n}||\mathsf{Y} - \mathsf{A}\omega||_2^2 \right\} \tag{2.12}$$

où le vecteur Y s'écrit $\mathsf{Y} = (y_1 \dots y_n)^T$ et la matrice A est de taille $n \times M + 1$:

$$\mathsf{A} = \begin{pmatrix} 1 & \cdots & x_1^j & \cdots & x_1^M \\ \vdots & & \vdots & & \vdots \\ 1 & \cdots & x_i^j & \cdots & x_i^M \\ \vdots & & \vdots & & \vdots \\ 1 & \cdots & x_n^j & \cdots & x_n^M \end{pmatrix}$$

Des solutions analytiques pour le problème (2.12) sont connues lorsque le problème est bien posé. Pour un degré M suffisamment élevé, on peut ainsi trouver des polynômes qui interpolent les données d'apprentissage. Dans la figure 2.6 (d), on a $\texttt{err}(h^*_9) = R_D(h^*_9) = 0$. C'est, du point de vue de l'apprentissage et pour la régression, la situation similaire à celle présentée dans la figure 2.3 dans le cas de la classification. L'erreur d'apprentissage (risque empirique obtenu pour la solution h^*_9 ou ω^*_9) est nulle. Pour autant, le meilleur prédicteur n'est pas dans notre exemple celui de degré le plus élévé. Augmenter la complexité (ici le degré) des prédicteurs permet de minimiser l'erreur d'apprentissage mais ne permet pas de minimiser le risque espéré, c'est-à-dire l'EPE. Dans le cas de la régression, l'EPE a une forme similaire à celle rencontrée pour la classification (équation 2.7) mais les données de sortie et la loi conditionnelle associée sont continues :

[2]Dans la suite, les ordonnées à prédire seront généralement notées y_i.

$$EPE(h) = E_{(X,Y)}[e(h(X), Y)] = \int_{\mathcal{X}} P_X(x) \left[\int_{\mathcal{Y}} e(h(x), y) P_{Y|X=x}(y) dy \right] dx \quad (2.13)$$

2.4 Bilan autour du vrai risque : l'EPE

Le cadre de l'apprentissage statistique permet d'aborder de manière unifiée les problèmes de régression et de classification. Dans les deux cas, il s'agit de trouver des prédicteurs dont la capacité de généralisation est la meilleure possible. Il ne faut pas confondre le risque empirique (ou erreur d'apprentissage) utilisé pour ajuster un prédicteur à un échantillon d'apprentissage et le «vrai risque espéré» (ou erreur de généralisation).

Pour un échantillon d'apprentissage donné, le risque empirique décroît progressivement en fonction de la complexité de la classe au sein de laquelle on apprend un prédicteur. Dans la figure 2.7, l'erreur d'apprentissage devient même nulle pour une classe $\mathcal{H}_k$ suffisamment complexe.

Le vrai risque $R(h)$ est quant à lui défini par l'EPE ou Expected Prediction Error pour un prédicteur h et une perte e. Il dépend de la loi inconnue des données :

$$R(h) = EPE(h) = E_{(X,Y)}[e(h(X), Y)] \quad (2.14)$$

Cette quantité mesure l'espérance des pertes dues aux prédictions de h sur l'intégralité des données susceptibles d'être générées (et non pas uniquement sur un ensemble d'apprentissage). Ce «vrai risque» présente un minimum pour la classe de prédicteurs qui a la meilleure capacité de généralisation. Parmi la hiérarchie de classes de prédicteurs de complexités croissantes illustrée par la figure 2.7, l'EPE est minimale pour une classe ni trop simple ni trop complexe.

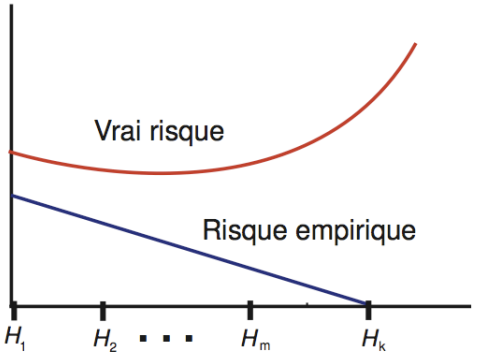


Figure 2.7: Risque empirique et vrai risque espéré.

Dans la suite du cours, après une brève introduction sur les prétraitements utiles en apprentissage et quelques rappels sur la régression, nous présenterons les solutions théoriques de la minimisation de l'EPE et leurs traductions pratiques si la loi des données est inconnue.

Thème 3

Prétraitements

.

Vincent Charvillat et Pierre Gurdjos
Février 2011

On s'intéresse ici aux vecteurs $\mathbf{x} \in \mathcal{X}$ donnés en entrée à un prédicteur h dans le schéma de boîte noire suivant :

$$\mathbf{x} \in \mathcal{X} \rightarrow \boxed{\text{h ?}} \rightarrow h(\mathbf{x}) \in \mathcal{Y}$$

Dans l'exemple introductif, consacré à la reconnaissance automatique de l'écriture manuscrite (Figure 1.1), nous avons considéré des imagettes de 64×64 pixels. Dans ce cas, l'espace d'entrée $\mathcal{X}$ peut donc être un sous-ensemble de $\mathbb{R}^{64\times64}$. Les $64 \times 64 = 4096$ composantes (valeurs des pixels) de $\mathbf{x}$ sont-elles nécessaires pour construire un classifieur efficace ? Il est connu que les pixels voisins d'une image naturelle sont fortement corrélés. Cette dépendance statistique implique une forte redondance de l'information apportée par des pixels voisins. *Réduire la dimension* de l'espace d'entrée est souvent possible et souhaitable. *Sélectionner les caractéristiques* (ou composantes) les plus «informatives» ou les plus «discriminantes» est une étape préliminaire fondamentale en apprentissage artificiel. Des connaissances a priori ou des méthodes de prétraitement peuvent être utilisées pour cela. Ces idées importantes seront rediscutées dans le thème 8.

Nous détaillons ci-après deux méthodes de prétraitement célèbres. La première (ACP ou Analyse en Composantes Principales) est générale et adopte une approche non supervisée. La seconde est adaptée à la classification supervisée (AFD ou Analyse Factorielle Discriminante).

3.1 Analyse en Composantes Principales (ACP)

3.1.1 Principe

L'ACP est une méthode d'analyse de données. On commence par placer les données d'entrée disponibles dans une matrice $\mathbf{X}$. On suppose que l'on dispose de n vecteurs d'entrées $\mathbf{x}_i \in \mathcal{X} = \mathbb{R}^p$, $i \in 1 \dots n$. Pour suivre la terminologie usuelle en analyse de données, on dira que $\mathbf{x}_i$ est vu comme un individu d'indice i c'est-à-dire une collection de p variables (ou attributs numériques). L'objectif est de représenter chaque individu avec un nombre réduit q d'attributs ou de variables. Si $q \ll p$, on cherche une représentation «compacte» des vecteurs donnés en entrée d'un prédicteur.

Grâce à cette réduction du nombre de variables d'entrée, on pourra simplifier le prédicteur mais aussi revenir à une dimension permettant la *visualisation* des données si $q = 2$ ou 3. Cela permet d'analyser graphiquement la structure des données, d'observer d'éventuels regroupements.

On souhaite donc approcher des individus $\mathbf{x}_i \in \mathcal{X}$ de grande taille dans un espace de dimension réduite en les déformant le moins possible. Si chaque individu est rangé en ligne dans $\mathbf{X}$, on a au départ $\mathbf{X} \in \mathcal{M}_{\mathbb{R}}(n,p)$ (noté abusivement $\mathbb{R}^{n\times p}$ plus loin)

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_i^T \\ \vdots \\ \mathbf{x}_n^T \end{pmatrix} = \begin{pmatrix} & & \vdots & & \\ & & \vdots & & \\ \cdots & \cdots & x_{ij} & \cdots & \cdots \\ & & \vdots & & \\ & & \vdots & & \end{pmatrix}$$

On a :

- ligne $\mathbf{x}_i^T \in \mathbb{R}^p \rightarrow$ valeurs des p variables de l'individu i
- colonne $\mathbf{v}_j \in \mathbb{R}^n \rightarrow$ valeurs de la variable j prises pour les n individus

On dira qu'un individu est un vecteur $\mathbf{x}_i$ de l'e.v. $\mathbb{R}^p$ appelé *espace des individus*. Dans $\mathbb{R}^p$, on étudie les distances entre individus. De manière complémentaire, on dira qu'une variable est un vecteur $\mathbf{v}_j$ de l'e.v. $\mathbb{R}^n$ appelé *espace des variables*. Dans $\mathbb{R}^n$, on étudie les angles entre variables.

3.1.2 Caractéristique d'ordre 1 : tendance centrale des variables

On note $\bar{\mathbf{x}} \in \mathbb{R}^p$, le vecteur des **moyennes arithmétiques des variables**. Ce vecteur $\bar{\mathbf{x}}$ est aussi appelé **individu moyen** (centre de gravité du nuage de points).

Il s'écrit matriciellement

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i = \frac{1}{n}\underbrace{\begin{bmatrix} \mathbf{x}_1 & | \cdots | & \mathbf{x}_n \end{bmatrix}}_{p\times n}\underbrace{\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}}_{n\times 1} = \frac{1}{n}\mathbf{X}^\top \mathbf{1}_n$$

Centrage des données : application d'une translation dans $\mathbb{R}^p$ telle que l'individu moyen $\bar{\mathbf{x}}$ soit à l'origine.

- Individu centré : $\mathbf{x}_i^c = \mathbf{x}_i - \bar{\mathbf{x}}$
- Tableau de données centré : $\mathbf{X}^c = \mathbf{X} - \mathbf{1}_n \bar{\mathbf{x}}^\top$

3.1.3 Caractéristiques d'ordre 2 : dispersion et dépendance des variables

Matrice de covariance « empirique » $\mathbf{\Sigma} \in \mathbb{R}^{p \times p}$, définie matriciellement[1] par

$$\mathbf{\Sigma} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^\top = \frac{1}{n}(\mathbf{X}^c)^\top \mathbf{X}^c = \frac{1}{n}\mathbf{X}^\top\mathbf{X} - \bar{\mathbf{x}}\bar{\mathbf{x}}^\top$$

La matrice de covariance mesure :

- la dispersion des p variables autour de leurs moyennes arithmétiques :
  $\rightarrow$ la diagonale

$$\begin{bmatrix} \ddots & & \\ & \sigma_{jj} = \frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2 & \\ & & \ddots \end{bmatrix}$$

  contient les p « **variances empiriques** » des variables notées σ_j^2 ;
- les dépendances linéaires entre deux variables $\mathbf{v}_i$ et $\mathbf{v}_k$, $i \neq k$
  $\rightarrow$ les éléments hors-diagonale, notés σ_{ik}, contiennent les « **covariances empiriques** » entre les variables $\mathbf{v}_i$ et $\mathbf{v}_k$.

L'**inertie du nuage de points**, c.-à-d. la dispersion de ceux-ci autour du centre de gravité $\bar{\mathbf{x}}$, mesurée par la moyenne des carrés des distances des individus à $\bar{\mathbf{x}}$, est donnée par

$$\mathcal{I}(\mathbf{X}) = trace(\mathbf{\Sigma}) = \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i - \bar{\mathbf{x}}\|^2$$

Elle est aussi appelée **variance totale**, sachant que

$$trace(\mathbf{\Sigma}) = \sum_{j=1}^{p}\sigma_j^2$$

[1] On montrera, à titre d'exercice que $(\mathbf{X}^c)^\top\mathbf{X}^c = \mathbf{X}^\top\mathbf{X} - n\bar{\mathbf{x}}\bar{\mathbf{x}}^\top$.

$$\begin{aligned}(\mathbf{X}^c)^\top\mathbf{X}^c &= (\mathbf{X}^\top - \bar{\mathbf{x}}\mathbf{1}_n^\top)(\mathbf{X} - \mathbf{1}_n\bar{\mathbf{x}}^\top) \\ &= \mathbf{X}^\top\mathbf{X} - \underbrace{(\mathbf{X}^\top\mathbf{1}_n)}_{=n\bar{\mathbf{x}}}\bar{\mathbf{x}}^\top - \bar{\mathbf{x}}\underbrace{(\mathbf{1}_n^\top\mathbf{X})}_{=n\bar{\mathbf{x}}^\top} + \bar{\mathbf{x}}\underbrace{(\mathbf{1}_n^\top\mathbf{1}_n)}_{=n}\bar{\mathbf{x}}^\top \\ &= \mathbf{X}^\top\mathbf{X} - n\bar{\mathbf{x}}\bar{\mathbf{x}}^\top - n\bar{\mathbf{x}}\bar{\mathbf{x}}^\top + n\bar{\mathbf{x}}\bar{\mathbf{x}}^\top = \mathbf{X}^\top\mathbf{X} - n\bar{\mathbf{x}}\bar{\mathbf{x}}^\top\end{aligned}$$

3.1.4 Corrélation entre variables

Il faut réduire (normaliser) les données pour analyser les éventuelles corrélations entre variables. On appelle matrice diagonale (p, p) de réduction, la matrice :

$$\mathbf{D}_{1/\sigma} = (\operatorname{diag}\sigma)^{-1} = \begin{pmatrix} \frac{1}{\sigma_1} & & & & \\ & \ddots & & & \\ & & \frac{1}{\sigma_i} & & \\ & & & \ddots & \\ & & & & \frac{1}{\sigma_p} \end{pmatrix}$$

On peut alors réduire le tableau de données centrées :

$$\begin{aligned} \mathbf{X}_{\mathbf{0,1}} &= \begin{pmatrix} & & \frac{x_{1j}-\bar{x}_j}{\sigma_j} & & \\ & & \vdots & & \\ \cdots & \cdots & \frac{x_{ij}-\bar{x}_j}{\sigma_j} & \cdots & \cdots \\ & & \vdots & & \\ & & \frac{x_{nj}-\bar{x}_j}{\sigma_j} & & \end{pmatrix} \\ &= \mathbf{X}^{\mathrm{c}}\mathbf{D}_{1/\sigma} \end{aligned}$$

On appelle finalement matrice de corrélation, la matrice suivante :

$$\begin{aligned} \mathbf{R} &= \begin{pmatrix} & \vdots & \\ \cdots & \frac{1}{n}\sum_{k=1}^{n} \frac{(x_{ki}-\bar{x}_i)}{\sigma_i} \frac{\left(x_{kj}-\bar{x}_j\right)}{\sigma_j} & \cdots \\ & \vdots & \end{pmatrix} \\ &= \frac{1}{n}\mathsf{X}_{\mathbf{0,1}}^{\top}\mathsf{X}_{\mathbf{0,1}} \end{aligned}$$

On peut, grâce à cette matrice, identifier empiriquement une dépendance affine entre les variables $\mathbf{v}_i$ et $\mathbf{v}_j$: si $R_{ij} = 1$ alors $\mathbf{v}_j = a\mathbf{v}_i + b$ (avec $a > 0$, corrélation positive).

Une corrélation positive $R_{ij} \approx 1$ (sachant $|R_{ij}| \leq 1$) signifie donc une forte dépendance entre les deux variables de sorte que si $\mathbf{v}_i$ augmente, il en va statistiquement de même pour $\mathbf{v}_j$.

3.1.5 Analyse en $q = 1$ Composante Principale

L'ACP est une méthode *factorielle linéair*e pour analyser la structure des données. Elle permet d'obtenir une représentation approchée du nuage de points dans un sous-espace de dimension faible $q \ll p$ de $\mathbb{R}^p$, qui conserve le « maximum » d'information et qui soit la plus « proche » possible du nuage initial.

Le problème $\mathscr{P}_{q=1}$

Formulation 1 *Il s'agit de chercher une droite D de $\mathbb{R}^p$, passant par le centre de gravité $\bar{\mathbf{x}}$ et de vecteur directeur $\mathbf{u} \in \mathbb{R}^p$, qui maximise l'inertie du nuage des points projetés sur D, c.-à-d. solution de*

$$\max_{\mathbf{u}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{y}_i - \bar{\mathbf{y}}\|^2 \tag{3.1}$$

où $\mathbf{y}_i$ dénote la projection de $\mathbf{x}_i$ sur D et $\bar{\mathbf{y}}$ dénote le centre de gravité des points projetés.

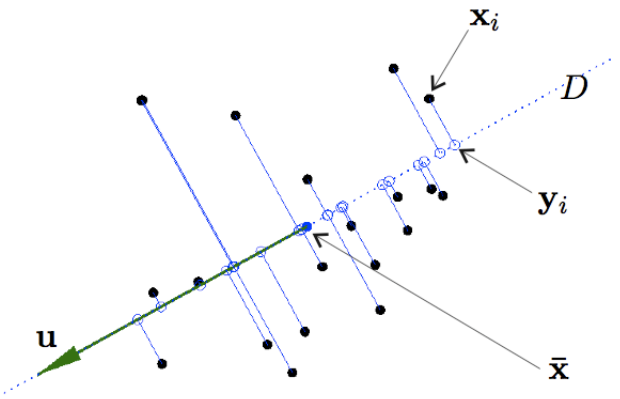


Figure 3.1: La droite recherchée est telle que la moyenne des carrés des distances entre projections sur D, c.-à-d. telle que l'inertie est la plus grande possible : la projection du nuage doit être la plus étalée possible.

> Dit différemment : « l'ACP cherche à remplacer les p variables d'un individu par une nouvelle —appelée (première) composante principale— qui soit de variance maximale et obtenue à partir d'une combinaison linéaires des des variables initiales ».

Un **résultat important** est que maximiser l'inertie revient à minimiser l'erreur d'approximation, c.-à-d. la somme des carrés des distances entre points originaux et points projetés.

$$\arg\max_{\mathbf{u}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{y}_i - \bar{\mathbf{y}}\|^2 = \arg\min_{\mathbf{u}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{y}_i - \mathbf{x}_i\|^2$$

Du fait de cette équivalente, l'approximation du nuage de points est la « meilleure » possible.

Projection orthogonale sur une droite

On rappelle que si $\mathbf{u} \in \mathbb{R}^p$ représente un vecteur directeur unitaire ($\|\mathbf{u}\| = 1$) de la droite D, alors la projection orthogonale de $\mathbf{x}_i$ sur D s'écrit

$$\mathbf{y}_i = c_i\mathbf{u} + \bar{\mathbf{x}} \quad \text{où} \quad c_i = \mathbf{u}^\top(\mathbf{x}_i - \bar{\mathbf{x}})$$

La matrice $\mathbf{\Pi} = \mathbf{u}\mathbf{u}^\top$ est un projecteur sur la droite vectorielle parallèle à D ; il vérifie $\mathbf{\Pi} = \mathbf{\Pi}^2$

Dans $\mathbb{R}^p$, l'**approximation du tableau de données** projetées sur D est donc donnée par la matrice $n \times q$ de rang 1

$$\mathbf{Y} = (\mathbf{X} - \mathbf{1_n}\bar{\mathbf{x}}^\top)\mathbf{u}\mathbf{u}^\top + \mathbf{1_n}\bar{\mathbf{x}}^\top$$

dont la ligne i correspond à la projection sur D de l'individu i

$$\mathbf{Y} \overset{\text{déf}}{=} \begin{bmatrix} \vdots \\ \mathbf{y}_i^\top \\ \vdots \end{bmatrix}$$

Définition 1 *On appelle* ***Composante Principale****, la coordonnée c_i du point projeté $\mathbf{y}_i$, relativement au repère $\{\bar{\mathbf{x}}; \mathbf{u}\}$. La droite D est appelée* ***Axe Principal*** *et le vecteur directeur $\mathbf{u}$ est appelé* ***Vecteur Principal****.*

Résolution de $\mathscr{P}_{q=1}$

On montre que le problème de l'analyse en $q = 1$ Composante Principale est strictement équivalent au problème suivant :

Formulation 2 *Chercher $\mathbf{u} \in \mathbb{R}^p$ solution de*

$$\max_{\mathbf{u}\in\mathbb{R}^p} \mathbf{u}^\top\mathbf{\Sigma}\mathbf{u} \quad \textit{sous la contrainte} \quad \|\mathbf{u}\|^2 = 1 \tag{3.2}$$

Pour montrer l'équivalence des deux formulations 1 et 2 précédentes,

- on montre facilement[2] que $\bar{\mathbf{y}} = \bar{\mathbf{x}}$.
- Ainsi le carré de la distance entre $\mathbf{y}_i$ et $\bar{\mathbf{y}}$ s'écrit :

$$\|\mathbf{y}_i - \bar{\mathbf{y}}\|^2 = \left\|\mathbf{u}\mathbf{u}^\top(\mathbf{x}_i - \bar{\mathbf{x}})\right\|^2 = (\mathbf{x}_i - \bar{\mathbf{x}})^\top\mathbf{u}\mathbf{u}^\top\mathbf{u}\mathbf{u}^\top(\mathbf{x}_i - \bar{\mathbf{x}}) = (\mathbf{x}_i - \bar{\mathbf{x}})^\top\mathbf{u}\mathbf{u}^\top(\mathbf{x}_i - \bar{\mathbf{x}})$$

$$= (\mathbf{u}^\top(\mathbf{x}_i - \bar{\mathbf{x}}))^2$$

[2] $\bar{\mathbf{y}} = \frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{u}\mathbf{u}^\top(\mathbf{x}_i - \bar{\mathbf{x}}) + \bar{\mathbf{x}}\right) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{u}\mathbf{u}^\top\mathbf{x}_i - \frac{1}{n}\sum_{i=1}^{n}\mathbf{u}\mathbf{u}^\top\bar{\mathbf{x}} + \frac{1}{n}\sum_{i=1}^{n}\bar{\mathbf{x}} = \mathbf{u}\mathbf{u}^\top\bar{\mathbf{x}} - \mathbf{u}\mathbf{u}^\top\bar{\mathbf{x}} + \bar{\mathbf{x}} = \bar{\mathbf{x}}$

- Il nous suffit de montrer que $\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{y}_i-\bar{\mathbf{y}}\|^2=\mathbf{u}^\top\mathbf{\Sigma}\mathbf{u}$ pour faire le lien entre (3.1) et (3.2):

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{y}_i-\bar{\mathbf{y}}\|^2 &= \frac{1}{n}\sum_{i=1}^{n}(\mathbf{u}^\top(\mathbf{x}_i-\bar{\mathbf{x}}))^2\\ &= \frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{u}^\top(\mathbf{x}_i-\bar{\mathbf{x}})\right)\left((\mathbf{x}_i-\bar{\mathbf{x}})^\top\mathbf{u}\right)\\ &= \frac{1}{n}\mathbf{u}^\top\underbrace{\left(\sum_{i=1}^{n}(\mathbf{x}_i-\bar{\mathbf{x}})(\mathbf{x}_i-\bar{\mathbf{x}})^\top\right)}_{n\mathbf{\Sigma}}\mathbf{u}=\mathbf{u}^\top\mathbf{\Sigma}\mathbf{u}\end{aligned}$$

■

Le problème d'optimisation sous contrainte $\mathscr{P}_{q=1}$ formulé en équation (3.2) peut être reformulé comme le problème non-contraint

$$\max_{\mathbf{u}\in\mathbb{R}^p}\mathscr{L}(\mathbf{u})$$

en introduisant le Lagrangien

$$\mathscr{L}(\mathbf{u})=\mathbf{u}^\top\mathbf{\Sigma}\mathbf{u}+\lambda(1-\|\mathbf{u}\|^2)$$

Puisque

$$\frac{\partial\mathscr{L}}{\partial\mathbf{u}}=2\mathbf{\Sigma}\mathbf{u}-2\lambda\mathbf{u}.$$

La condition nécessaire d'optimalité $\frac{\partial\mathscr{L}}{\partial\mathbf{u}}=\mathbf{0}$ est exprimée par l'équation aux valeurs propres suivante :

$$\mathbf{\Sigma}\mathbf{u}=\lambda\mathbf{u}. \tag{3.3}$$

La solution exacte de (3.3) est un couple $(\lambda,\mathbf{u})$, correspondant à une valeur propre et un vecteur propre de $\mathbf{\Sigma}$.

Comment choisir ce couple ?

- Comme $\mathbf{\Sigma}$ est symétrique et semi-définie positive, ses valeurs propres sont réelles et positives (ou nulles).
- Puisque $\mathbf{u}^\top\mathbf{\Sigma}\mathbf{u}=\lambda\mathbf{u}^\top\mathbf{u}$, cf. (3.3), maximiser $\mathbf{u}^\top\mathbf{\Sigma}\mathbf{u}$ sous la contrainte $\|\mathbf{u}\|^2=1$, revient à maximiser λ !

Théorème 1 (solution de $\mathscr{P}_{q=1}$) *La solution du problème $\mathscr{P}_{q=1}$ pour $\mathbf{u}_1$ est donnée par le vecteur propre associé à la plus grande valeur propre de $\mathbf{\Sigma}$.*

3.1.6 Cas général : analyse en $q > 1$ composantes principales

Le problème $\mathscr{P}_q$

Formulation 3 *Chercher un sous-espace affine S_q de dimension $q < p$ de $\mathbb{R}^p$, passant par le centre de gravité $\bar{\mathbf{x}}$ et de direction le sous-espace vectoriel $F_q \subset \mathbb{R}^p$, qui maximise l'inertie du nuage des points projetés sur S_q, c.-à-d. solution de*

$$\max_{\substack{F \subset \mathbb{R}^p \\ \dim F = q}} \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{y}_i - \bar{\mathbf{y}}\|^2$$

où $\mathbf{y}_i$ dénote la projection de $\mathbf{x}_i$ sur S_q et $\bar{\mathbf{y}}$ dénote le centre de gravité des points projetés.

On rappelle tout d'abord qu'un sous-espace vectoriel de dimension q peut être engendré par une base formée de q vecteurs indépendants, notés $\{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_q\}$, qui forment la matrice

$$\mathbf{U}_q = \left[\begin{array}{c|c|c} \mathbf{u}_1 & \cdots & \mathbf{u}_q \end{array} \right]. \tag{3.4}$$

Si la base $\{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_q\}$ est orthonormée, alors $\mathbf{U}_q$ est orthogonale c.-à-d.

$$\mathbf{U}_q^\top \mathbf{U}_q = \mathbf{I}.$$

Donc, rechercher F_q revient à rechercher une base $\{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_q\}$ orthonormée qui l'engendre.

Projection orthogonale sur un sous-espace affine

Si $\{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_q\} \subseteq \mathbb{R}^p$ est une base orthonormée engendrant F_q, alors la projection orthogonale de $\mathbf{x}_i$ sur S_q de direction F_q s'écrit

$$\mathbf{y}_i = \mathbf{U}_q \mathbf{c}_i + \bar{\mathbf{x}} \quad \text{où} \quad \mathbf{c}_i = \mathbf{U}_q^\top (\mathbf{x}_i - \bar{\mathbf{x}}) \tag{3.5}$$

où $\mathbf{U}_q$ est défini en (3.4). La matrice $\mathbf{\Pi} = \mathbf{U}_q \mathbf{U}_q^\top$ est un projecteur sur F_q (direction de S_q) ; elle vérifie $\mathbf{\Pi} = \mathbf{\Pi}^2$.

Définition 2 *On appelle **Composantes Principales** les coordonnées $(c_1, ..., c_q)$ du point projeté $\mathbf{y}_i$, relativement au repère orthornormé $\{\bar{\mathbf{x}}; \mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_q\}$.*

On note $\mathbf{c}_i$ le vecteur des Composantes Principales de l'individu i. L'ensemble des q Composantes Principales sera donné par la matrice notée

$$\mathbf{C}_q = (\mathbf{X} - \mathbf{1}_n \bar{\mathbf{x}}^\top) \mathbf{U}_q = \mathbf{X}^c \mathbf{U}_q$$

sachant que la ligne i correspondra aux Composantes Principales de l'individu i

$$\mathbf{C}_q = \underset{n \times q}{\begin{bmatrix} & \vdots & \\ \cdots & c_{ij} & \cdots \\ & \vdots & \end{bmatrix}} = \begin{bmatrix} \vdots \\ \mathbf{c}_i^\top \\ \vdots \end{bmatrix}$$

Dans $\mathbb{R}^p$, l'**approximation du tableau de données** projetées sur S_q, appelée **reconstruction**, est donnée par la matrice $n \times q$ de rang q

$$\mathbf{Y} = (\mathbf{X} - \mathbf{1_n}\bar{\mathbf{x}}^\top)\mathbf{U}_q\mathbf{U}_q^\top + \mathbf{1}_n\bar{\mathbf{x}}^\top = \mathbf{X^c}\mathbf{U}_q\mathbf{U}_q^\top + \mathbf{1}_n\bar{\mathbf{x}}^\top \tag{3.6}$$

dont la ligne i correspond à la projection sur D de l'individu i

$$\mathbf{Y} \overset{\text{déf}}{=} \begin{bmatrix} \vdots \\ \mathbf{y}_i^\top \\ \vdots \end{bmatrix}$$

Un **résultat important** est que l'approximation du nuage de points est la « meilleure » possible. En effet, maximiser l'inertie revient à minimiser l'erreur d'approximation, c.-à-d. à minimiser la somme des carrés des distances entre points et points projetés (voir le théorème d'Eckart–Young en §3.1.7)

$$\arg \max_{\{\mathbf{u}_1,\mathbf{u}_2,\ldots,\mathbf{u}_q\}} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{y}_i - \bar{\mathbf{y}}\|^2 = \arg \min_{\{\mathbf{u}_1,\mathbf{u}_2,\ldots,\mathbf{u}_q\}} \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{y}_i - \mathbf{x}_i\|^2$$

Résolution du problème $\mathscr{P}_q$

On supposera ici, sans perte de généralité, que le centre de gravité du nuage de points coïncide avec $\mathbf{0} \in \mathbb{R}^p$. Pour se ramener à ce cas, on effectuera au préalable un centrage des données de telle façon que :

$$\mathbf{x}_i^c = \mathbf{x}_i \quad \text{et} \quad \mathbf{X}^c = \mathbf{X}$$

Théorème 2 *Soit $F_{q-1} \subset \mathbb{R}^p$ un sous-espace vectoriel de dimension $q-1$, solution de $\mathscr{P}_{q-1}$. Alors le sous-espace vectoriel F_q de dimension q, solution de $\mathscr{P}_q$, est la somme directe[a] de F_{q-1} et de la droite vectorielle F_1 de $\mathbb{R}^p$, orthogonale à F_{q-1}, solution de $\mathscr{P}_1$: les solutions sont « emboîtées ».*

[a] La somme directe de deux sev F et G est définie par $F \oplus G = \{f + g \mid f \in F, \quad g \in G\}$

Ce que cela veut dire : pour obtenir F_q, on procède de proche en proche, on cherche d'abord le sous-espace vectoriel de dimension 1 maximisant l'inertie du nuage projeté, puis le sous-espace vectoriel de dimension 1 orthogonal au précédent et maximisant l'inertie, etc.

Théorème 3 *Le sous-espace vectoriel F_q de dimension q, solution de $\mathscr{P}_q$, est engendré par les q vecteurs propres de Σ associés aux q plus grandes valeurs propres.*

On rappelle que les vecteurs propres de $\mathbf{\Sigma}$ sont orthogonaux, sans perte de généralité, on les supposera unitaires. On peut démontrer ces théorèmes du fait que le problème de l'analyse en q Composantes Principales est strictement équivalent au problème $\mathscr{P}_q$ suivant :

résoudre

$$\max_{\mathbf{U}_q \in R^{p\times q}} \mathit{trace}(\mathbf{U}_q^\top \mathbf{\Sigma} \mathbf{U}_q) \quad \text{sous la contrainte} \quad \mathit{rang}\mathbf{U}_q = q$$

Les vecteurs de base recherchés sont alors les colonnes de la solution $\mathbf{U}_q$, cf. (3.4).

3.1.7 Reconstruction du tableau des données au moyen des composantes principales et des facteurs principaux

Bien que l'objectif soit en général de n'utiliser qu'un petit nombre de Composantes Principales, l'ACP en construit initialement p, autant que de variables originales.

Reconstruction de l'individu i

Voir l'équation (3.5).

Reconstruction du tableau des données

Voir l'équation (3.6).

- Si $q = p$: reconstitution/reconstruction exacte ;
- si $q < p$: meilleure approximation de $\mathbf{X}$ par une matrice de rang q au sens des moindres carrés (théorème d'Eckart–Young) :

  $$\mathbf{Y} = \arg\min_{\mathbf{M}} \|\mathbf{M} - \mathbf{X}\|_F \quad \text{sous la contrainte} \quad rang(\mathbf{M}) = q$$

  Aucune autre matrice de rang q ne peut rendre l'erreur d'approximation plus faible.

3.1.8 Conclusion et propriétés de l'ACP

Statistiquement parlant, l'ACP a pour objectif de décrire le nuage de points par q nouvelles variables (les Composantes Principales) qui soient une combinaison linéaire des variables initiales et dont la variance totale est maximale.

- **Nombre de CP :**

  « Retenir q Composantes Principales »
  veut dire
  « Remplacer les observations originales par leur projections orthogonales dans le sous-espace à q dimensions défini par les q premières Composantes Principales. »

- **Orthogonalité :**

  Les Composantes Principales sont associées à des directions de l'espace des observations qui sont deux à deux orthogonales. Autrement dit, l'ACP procède à un changement de repère orthogonal, les axes originaux étant remplacées par les Axes Principaux.

- **Décorrélation :**

  D'une part, on a le schéma suivant, permettant de calculer la matrice de covariance $\mathbf{\Sigma}'$ associées aux Composantes Principales

$$\begin{array}{ccc} \{\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_p\} & \overset{\text{changement de base vectorielle}}{\longrightarrow} & \{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_p\} \\ \mathbf{X}^c & \longrightarrow & \mathbf{C}^c = \mathbf{X}^c\mathbf{U} \\ \mathbf{\Sigma} = (\mathbf{X}^c)^\top \mathbf{X}^c & \longrightarrow & \mathbf{\Sigma}' = (\mathbf{C}^c)^\top \mathbf{C}^c = \mathbf{U}^\top \mathbf{\Sigma} \mathbf{U} \end{array}$$

D'autre part,

$$\begin{aligned} \mathbf{\Sigma}\mathbf{u}_j &= \lambda_j \mathbf{u}_j \\ &\Leftrightarrow \mathbf{\Sigma}\mathbf{U} = \mathbf{U} \begin{bmatrix} \ddots & & \\ & \lambda_j & \\ & & \ddots \end{bmatrix} \Leftrightarrow \mathbf{U}^\top \mathbf{\Sigma} \mathbf{U} = \begin{bmatrix} \ddots & & \\ & \lambda_j & \\ & & \ddots \end{bmatrix} \end{aligned}$$

Les Composantes Principales sont des variables qui s'avèrent être deux à deux décorrélées.

- **Ordre d'importance - contraste :**

  La propriété fondamentale des Composantes Principales est de pouvoir être classées par ordre décroissant d'importance : le meilleur sous-espace à q dimensions dans lequel projeter les données est celui engendré par les q premières Composantes Principales.

  La variance totale des Composantes Principales (c.-à-d. des nouvelles variables) correspond à la somme des valeurs propres

$$\sum_{j=1}^{q} \sigma_j^2 = trace(\mathbf{\Sigma}) = \sum_{j=1}^{q} \lambda_j$$

  Chaque valeur propre mesure la part de variance expliquée par l'Axe Principal correspondant. Le contraste conservé par q Composantes Principales est

$$\frac{\sum_{j=1}^{q} \lambda_j}{\sum_{j=1}^{p} \lambda_j}$$

3.2 Application directe de l'ACP : les «eigenfaces»

Le travail de M. Turk et A. Pentland intitulé «Eigenfaces for Recognition» et publié en 1991 dans la revue «Journal of Cognitive Neuroscience» (vol. 3, pp 71-86) constitue une des applications les plus intéressantes et populaires de l'ACP au domaine de la reconnaissance de forme. Il s'agit tout simplement d'appliquer l'ACP à partir de données de très grandes tailles : des images de visages.

La figure 3.2 suivante montre une base de $n = 9$ visages présentant 3 personnes dans 3 positions différentes relativement à la caméra. Dans cet exemple, chaque image est de taille : 480 lignes $\times$ 640 colonnes. Il est immédiat, par empilement des colonnes, de se ramener à la situation où chaque image est un vecteur $\mathbf{x}_i \in \mathbb{R}^p$ avec $p = 480 \times 640 = 307200$ composantes (niveaux de gris / pixels).

Nous pouvons donc adopter à nouveau les notations usuelles pour l'ACP en disant que la base d'images forme un tableau de données $\mathbf{X} \in \mathcal{M}_{\mathbb{R}}(n = 9, p = 307200)$ avec, en ligne i, le vecteur $\mathbf{x}_i^T \in \mathbb{R}^p$ qui représente la i^{ieme} image de la base.

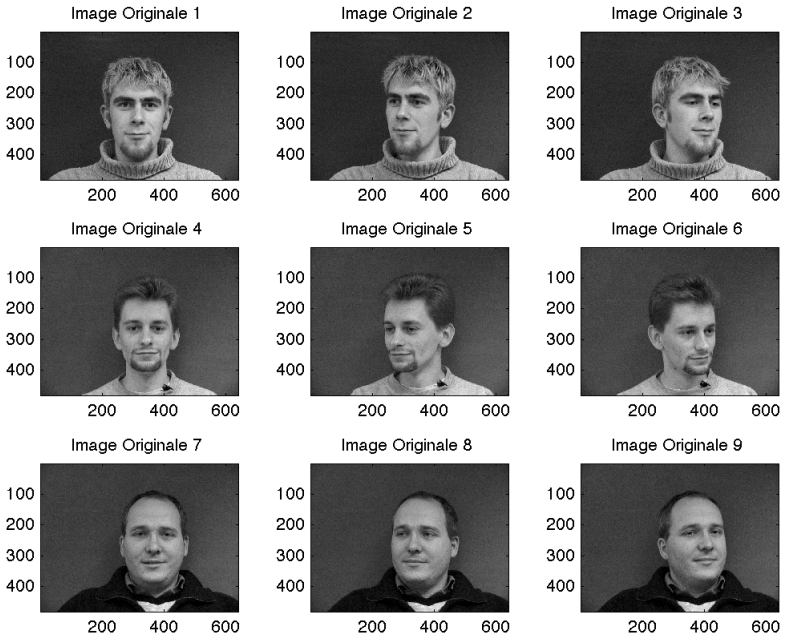


Figure 3.2: Une base de visages

L'individu moyen $\bar{\mathbf{x}} \in \mathbb{R}^p$ est le vecteur des **moyennes arithmétiques des variables** : il représente alors, tout simplement, la version «vectorisée» de l'image moyenne $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$ des images de la base. Cette image moyenne est montrée en haut et à gauche de la figure 3.4. Il est alors possible de centrer les données (c'est-à-dire les images de la base) pour former le tableau centré $\mathbf{X}^c = \mathbf{X} - \mathbf{1}_n \bar{\mathbf{x}}^\top$.

On s'intéresse alors à un mécanisme de prédiction visant la reconnaissance de visages. L'idée est de donner une image requête $\mathbf{x} \in \mathbb{R}^{307200}$ en entrée d'un prédicteur dont l'objectif est de prédire l'image (ou les images) de la base la (ou les) plus similaire(s) à cette requête selon le schéma de boîte noire suivant :

$$\mathbf{x} \in \mathbb{R}^{307200} \rightarrow \boxed{\text{h}} \rightarrow h(\mathbf{x})$$

Une requête simple est illustrée par la figure 3.3 suivante. L'image requête est celle d'un des trois individus présents dans la base précédente mais dans une nouvelle position (bouche ouverte). En d'autres termes, l'image requête n'appartient pas à la base de visage mais est similaire à trois des images de cette base. L'objectif est donc d'identifier ces similarités pour, éventuellement, reconnaître l'individu. Dans la figure 3.3, les trois images du même individu présentes dans la base sont renvoyées comme résultat (correct) de la requête. Ce résultat est ordonné : la première image trouvée (choix 1) est mathématiquement (mais pas visuellement) la plus similaire à la requête. Précisons maintenant comment ce résultat a été obtenu.

Comme nous l'avons dit en introduction de ce thème consacré aux prétraitements, il est inutile et inopportun d'utiliser les $p = 307200$ niveaux de gris pour comparer l'image requête avec chacune des images de la base. L'ACP est donc un

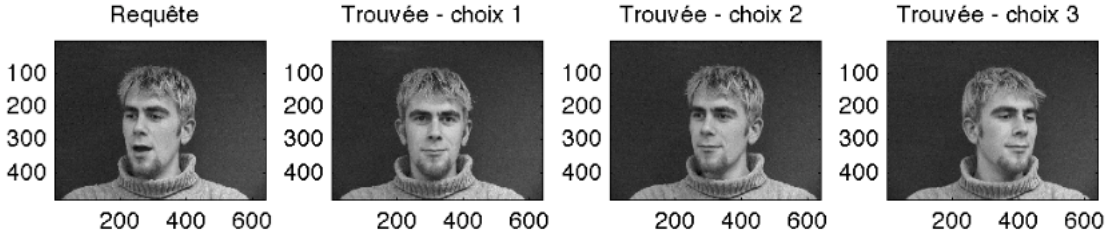


Figure 3.3: Résultat d'une requête sur une base de visages.

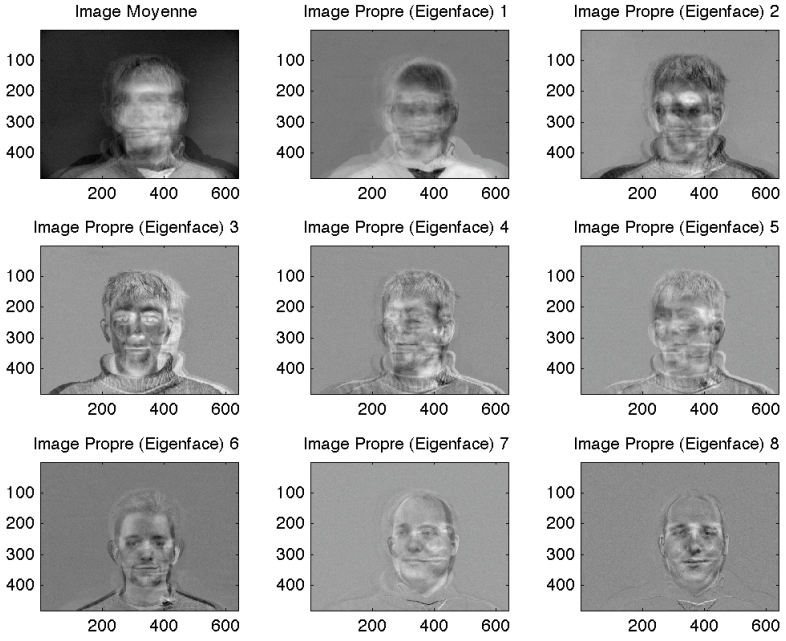


Figure 3.4: Les «eigenfaces»

prétraitement qui consiste à réduire la dimension des données selon le schéma suivant où le prédicteur prend en entrée la sortie de l'ACP:

$$\mathbf{x} \in \mathbb{R}^{307200} \rightarrow \boxed{\text{ACP}} \rightarrow \phi(\mathbf{x}) \in \mathbb{R}^{q \ll 307200} \rightarrow \boxed{\text{h}} \rightarrow h(\mathbf{x})$$

Le vecteur caractéristique $\phi(\mathbf{x})$ est tout simplement formé des q premières composantes principales résultant de la projection de l'image requête $\mathbf{x}$ sur les q premiers vecteurs propres unitaires $\{\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_q\} \subseteq \mathbb{R}^p$ de $\mathbf{\Sigma} = \frac{1}{n}(\mathbf{X}^c)^\top \mathbf{X}^c$. Ces vecteurs propres sont des vecteurs de $\mathbb{R}^{307200}$ est donc également des images que l'on appelle «eigenfaces». Ces images sont présentées dans la figure 3.4. On a ainsi :

$$\phi(\mathbf{x}) \stackrel{\text{déf}}{=} \begin{bmatrix} \mathbf{u}_1^\top \mathbf{x} \\ \mathbf{u}_2^\top \mathbf{x} \\ \vdots \\ \mathbf{u}_q^\top \mathbf{x} \end{bmatrix}$$

On dit que $\phi(\mathbf{x})$ est une représentation compacte préservant au mieux l'information contenue dans $\mathbf{x}$ avec seulement q composantes principales. La prédiction h peut alors être définie de diverses manières. Parmi les possibilités les plus simples, on peut chercher les images de la base dont les représentations compactes sont les plus proches (au sens d'une distance d bien choisie) de celle associée à l'image requête :

$$h(\mathbf{x}) = \arg\min_{i \in 1 \ldots n} d(\phi(\mathbf{x_i}), \phi(\mathbf{x}))$$

C'est en choisissant $q = 3$ et $d(\phi(\mathbf{x_i}), \phi(\mathbf{x})) = ||\phi(\mathbf{x_i}) - \phi(\mathbf{x})||_2$ que les résultats des figures précédentes ont été produits.

Cette présentation occulte toutefois un problème délicat : la grande taille de $\mathbf{\Sigma} = \frac{1}{n}(\mathbf{X}^c)^\top \mathbf{X}^c$ ($p \times p = 307200 \times 307200$) ne permet pas de calculer ses vecteurs propres de manière aisée. L'astuce est la suivante. Soient $\mathbf{A} \in \mathcal{M}_{\mathbb{R}}(n, p)$ et $\lambda \in \mathbb{R}^*$ on a :

$$\lambda \text{ est valeur propre de } \mathbf{A}^\top \mathbf{A} \iff \lambda \text{ est valeur propre de } \mathbf{A}\mathbf{A}^\top$$

Plus précisèment, si on appelle $\mathbf{v}$ un vecteur propre associé à la valeur propre λ de $\mathbf{A}^\top \mathbf{A}$, on peut montrer que $\mathbf{Av}$ est un vecteur propre associé à la valeur propre λ de $\mathbf{A}\mathbf{A}^\top$. Et de manière symétrique, si on appelle $\mathbf{w}$ un vecteur propre associé à la valeur propre λ de $\mathbf{A}\mathbf{A}^\top$ alors $\mathbf{A}^\top \mathbf{w}$ est un vecteur propre associé à la valeur propre λ de $\mathbf{A}^\top \mathbf{A}$.

Ce résultat permet de calculer les «eigenfaces» à partir d'une analyse spectrale de la matrice $\mathbf{X}^c(\mathbf{X}^c)^\top$ dont la taille est réduite à $n \times n$ dans notre application.

3.3 Analyse Factorielle Discriminante (AFD)

3.3.1 Principe

Alors que l'ACP est une méthode de prétraitement générale, l'Analyse Factorielle Discriminante traite les données fournies en entrée d'un classifieur. Le cadre est restreint à la classfication supervisée. On souhaite encore réduire la dimension des données d'entrée mais avec un nouvel objectif. Il s'agit de déterminer des axes (dits «factoriels») sur lequels les projections des variables caractéristiques des individus vont permettre de bien séparer les classes en présence. L'exercice suivant permet une introduction illustrée à l'AFD.

3.3.2 Exercice

Dans cet exercice, les vecteurs sont notés en gras. On considère un problème de classification supervisée d'individus appartenant à deux classes notées C_1 et C_2. On

connait chaque individu grâce à un vecteur de caractéristiques noté $\mathbf{x}$ de dimension d. Pour déterminer des caractéristiques discriminantes en vue d'une classification, on projette un vecteur $\mathbf{x}$ sur un axe de vecteur directeur $\mathbf{w}$ selon $y = \mathbf{w}^T\mathbf{x}$. On choisit un seuil $-\omega_0$ tel que les décisions de classifications sont : on décide C_1 si $y \geq -\omega_0$, on décide C_2 sinon.

De manière supervisée, on se donne N_1 (resp. N_2) points dans C_1 (resp. C_2). Et on cherche à séparer au mieux les classes en choisissant correctement le vecteur/la direction de projection $\mathbf{w}$. Les individus de C_1 (resp. C_2) se structurent autour des centres de classe $\mathbf{m}_1$ (resp. $\mathbf{m}_2$) avec $\mathbf{m}_i = \frac{1}{N_i}\sum_{n\in C_i}\mathbf{x}_n$. La mesure de séparation entre classes la plus simple (après la projection selon $\mathbf{w}$) est la séparation des projections $m_i = \mathbf{w}^T\mathbf{m}_i$ des centres de classes $\mathbf{m}_i$. Si $\mathbf{w}$ est de norme 1, on peut tenter de bien séparer les projections des centres des classes en maximisant :

$$m_2 - m_1 = \mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1) \tag{3.7}$$

Fisher propose une approche plus riche qui consiste à maximiser cet écart entre projections des centres tout en minimisant la dispersion des classes après projection. Cette dispersion est définie par $s_i^2 = \sum_{n\in C_i}(y_n - m_i)^2$ pour la classe C_i (avec $y_n = \mathbf{w}^T\mathbf{x}_n$). Le critère de Fisher $J(\mathbf{w})$ suivant peut alors être maximisé pour trouver le meilleur axe $\mathbf{w}$:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \tag{3.8}$$

1. On vous demande d'illustrer graphiquement (pour $d = 2$) les idées précédentes : les individus de C_1 (resp. C_2) seront représentés dans le plan par des petits ronds "o" (resp. des petites croix "x"). On vous demande de dessiner deux figures (deux populations d'individus et/ou deux axes) pour lesquelles les critères de l'équation (3.7) sont les mêmes mais qui diffèrent du point de vue du critère de Fisher de l'équation (3.8). Sur chaque figure on fera apparaître avec attention *l'allure des frontières de décision* et les ingrédients mathématiques $\mathbf{w}$, $\mathbf{m}_i$, m_i, s_i etc. On commentera ces figures permettant une séparation linéaire de deux classes.

2. Soit $\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T$ une matrice de covariance dite interclasse (Between-class). On vous demande de reformuler le numérateur du critère de Fisher (c'est-à-dire $(m_2 - m_1)^2$) en fonction de $\mathbf{w}$ et $\mathbf{S}_B$.

3. Soit $\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2$ matrice de covariance (Within-class) avec $\mathbf{S}_i = \sum_{n\in C_i}(\mathbf{x}_n - \mathbf{m}_i)(\mathbf{x}_n - \mathbf{m}_i)^T$. Exprimer le critère de Fisher $J(\mathbf{w})$ en fonction de $\mathbf{w}$, $\mathbf{S}_B$ et $\mathbf{S}_W$.

4. On peut remarquer que $J(a\mathbf{w}) = J(\mathbf{w})$ pour tout scalaire a. On peut donc maximiser le numérateur du critère de Fisher sous la contrainte d'un dénominateur unitaire (égal à 1). Exprimer alors en fonction de de $\mathbf{w}$, $\mathbf{S}_B$ et $\mathbf{S}_W$, le problème d'optimisation sous contrainte que l'on veut résoudre pour déterminer un axe factoriel discriminant (c'est-à-dire le meilleur $\mathbf{w}$) d'après Fisher.

5. Donner le Lagrangien associé au problème précédent et sa dérivée selon $\mathbf{w}$.

6. En remarquant que $\mathbf{S}_B\mathbf{w}$ est proportionnel à $(\mathbf{m}_2-\mathbf{m}_1)$ (ce qui est noté $\mathbf{S}_B\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1)$), montrer finalement que la solution/direction recherchée $\mathbf{w}_{fisher}$ est telle que $\mathbf{w}_{fisher} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$ en supposant $\mathbf{S}_W$ inversible.

3.4 Exercice reliant classification et régression

Cet exercice prolonge le précédent On reprend exactement les mêmes notations que dans l'exercice précédent. Les décisions de classifications étaient : on décide C_1 si $y \geq -\omega_0$ on décide C_2 sinon. Pour un individu $\mathbf{x}_n$ parmi les $N = N_1 + N_2$ dont on dispose, on décide donc qu'il appartient à la classe C_1 si $\mathbf{w}^T\mathbf{x}_n + \omega_0 \geq 0$ et qu'il appartient à C_2 sinon. On a donc une fonction discriminante $\delta(\mathbf{x}_n) = \mathbf{w}^T\mathbf{x}_n + \omega_0$ qui est positive pour une classe et négative pour l'autre.

On peut utiliser cette propriété pour apprendre un classifieur linéaire au sens de la régression. L'idée est d'associer des valeurs cibles t_n positives (et bien choisies) pour les individus $\mathbf{x}_n$ issus de C_1 et des valeurs cibles négatives pour les individus de C_2. On peut alors apprendre un classifieur linéaire en minimisant l'écart E aux cibles :

$$E = \frac{1}{2}\sum_{n=1}^{N}(\delta(\mathbf{x}_n) - t_n)^2 = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n + \omega_0 - t_n)^2 \tag{3.9}$$

Parmi les choix possibles de «cibles», on propose ici :

- $t_n = \frac{N}{N_1}$ pour les N_1 individus $\mathbf{x}_n$ de C_1
- $t_n = -\frac{N}{N_2}$ pour les N_2 individus $\mathbf{x}_n$ de C_2

1. Calculer $\sum_{n=1}^{N} t_n$
2. Donner une expression de $\mathbf{m} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$ en fonction de N, N_1, N_2, et des centres de classes introduits dans l'exercice précédent $\mathbf{m}_1$, $\mathbf{m}_2$.
3. Pour minimiser E, calculer sa dérivée partielle par rapport à ω_0 et donner une condition nécessaire associée.
4. Montrer qu'il est aussi nécessaire que :

$$\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n + \omega_0 - t_n)\mathbf{x}_n = 0 \tag{3.10}$$

5. Montrer que ω_0 vérifie $\omega_0 = -\mathbf{w}^T\mathbf{m}$
6. Déduire de l'équation (3.10), que le $\mathbf{w}$ recherché vérifie, avec les notations de l'exercice précédent :

$$\Big(\mathbf{S}_W + \frac{N_1N_2}{N}\mathbf{S}_B\Big)\mathbf{w} = N(\mathbf{m}_1 - \mathbf{m}_2) \tag{3.11}$$

7. Montrer alors que nous retrouvons comme précédemment $\mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$.

8. Nous avons cependant dans cet exercice un résultat un peu plus riche que dans l'exercice précédent sur l'Analyse Factorielle Discriminante. Pourquoi ? Redonner la forme définitive de la décision.

Thème 4

Rappels sur la régression

Vincent Charvillat
Décembre 2010

4.1 Cadre

On se place dans le cadre de la régression (au sens de l'apprentissage) selon un «modèle de données fonctionnelles» que nous allons expliquer. Nous rappelons que résoudre un problème de régression revient à trouver le prédicteur h qui prédit au mieux des sorties aléatoires appartenant à $\mathcal{Y}$ à partir d'entrées appartenant à $\mathcal{X}$ selon le schéma de boîte noire suivant :

$$x \in \mathcal{X} \rightarrow \boxed{\text{h ?}} \rightarrow \widehat{y} = h(x) \in \mathcal{Y}$$

Ayant choisi une classe de prédicteur ($h \in \mathcal{H}$) et une perte e, on a vu que le prédicteur le plus intéressant est celui qui minimise l'erreur de généralisation (ou risque espéré ou EPE) :

$$R(h) = EPE(h) = E_{(X,Y)}[e(h(X), Y)] = \int_{\mathcal{X}} \int_{\mathcal{Y}} e(h(x), y) P_{X,Y}(x, y) dx dy \qquad (4.1)$$

En pratique la loi jointe des données $P_{X,Y}(x, y) = P_{Y|X=x}(y) P_X(x)$ est inconnue. On cherche donc à faire des hypothèses simplificatrices qui permettent de modéliser les dépendances entre entrées et sorties.

En détaillant les solutions théoriques de la minimisation de l'EPE, nous verrons plus loin que la connaissance de la loi conditionnelle $P_{Y|X=x} = P_{X,Y}(x, y)/P_X(x)$ banalise le problème.

On peut donc simplifier le problème en supposant l'existence d'une dépendance statistique de Y sur X selon :

$$Y = f(X) + \epsilon \tag{4.2}$$

où :

- f est une fonction réelle inconnue $f : \mathbb{R} \to \mathbb{R}$
- ϵ est un bruit (variable aléatoire réelle) d'espérance nulle : $E(\epsilon) = 0$ et indépendant de X.
- Y est une variable aléatoire réelle (v.a comme fonction de v.a.)

Ce cadre s'avère simplificateur puisque la distribution conditionnelle $P_{Y|X=x}$ ne dépend alors de X que via la moyenne $f(x)$ c'est-à-dire l'espérance conditionnelle $f(x) = E(Y|X = x)$. On parle donc de modèle de données fonctionnelles puisqu'on bruite additivement les valeurs d'une fonction.

4.2 Des échantillons simples à manipuler

Il est fréquent de supposer que X et donc $f(X)$ sont déterministes dans l'équation (4.2). Dans ce cas, une mesure de sortie Y est vue comme la réponse fonctionnelle, bruitée aléatoirement, à un signal d'entrée X lequel est supposé/connu sans erreur.

Pour bâtir un premier échantillon d'apprentissage $D_1 = \{(x_i, y_i)\}_{i=1\dots n}$, on considère alors n valeurs d'entrée connues $\{x_i\}_{i=1\dots n}$ et n réalisations indépendantes et identiquement distribuées du bruit selon la loi d'ϵ:

$$y_i = f(x_i) + \epsilon_i \tag{4.3}$$

Si l'on conserve les mêmes entrées $\{x_i\}_{i=1\dots n}$, un second ensemble d'apprentissage D_2 ne diffèrera de D_1 qu'en raison de nouvelles réalisations du bruit lesquelles conduiront à de nouvelles observations $\{y'_i \neq y_i\}_{i=1\dots n}$ en sortie : $D_2 = \{(x_i, y'_i)\}_{i=1\dots n}$.

La figure 4.1 illustre cette situation : en bleu, le graphe de $f : x \to f(x)$, en rouge deux échantillons D_1 et D_2 représentés avec des cercles 'o' et des croix '+'. Ces échantillons partagent les mêmes entrées scalaires $\{x_i\}_{i=1\dots n}$ en abscisses. Ce modèle de données fonctionnelles est intéressant car, grâce à lui, on pourra facilement analyser le comportement (aléatoire) d'algorithmes d'apprentissage lorsque les échantillons d'apprentissage varient (aléatoirement).

4.3 Modèle gaussien et maximum de vraisemblance

4.3.1 Modèle gaussien

On peut affiner ce qui précéde en adoptant un modèle paramètrique pour la fonction f au voisinage de laquelle sont générées les données. La fonction dépend alors d'un ensemble de paramètres regroupés dans un vecteur noté β. Un modèle de régression gaussien s'écrit alors :

$$Y = f(X, \beta) + \epsilon \tag{4.4}$$

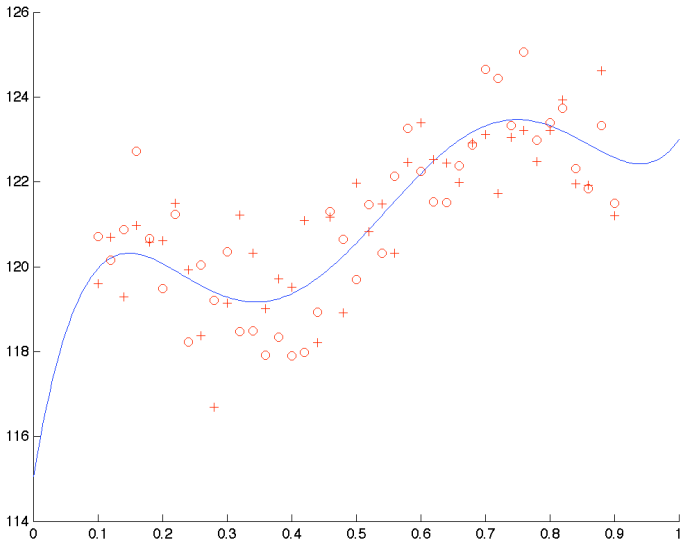


Figure 4.1: Deux échantillons selon le modèle de données fonctionnelles

avec $\epsilon \sim \mathcal{N}(0, \sigma^2)$ et β et σ^2 inconnus. La figure 4.2 illustre cette nouvelle situation où un bruit gaussien, additif et vertical apparaît.

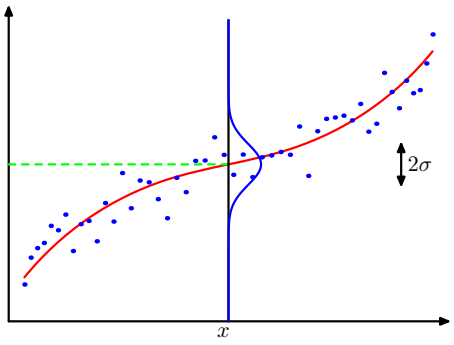


Figure 4.2: Bruit vertical additif gaussien pour le modèle de données fonctionnelles

La loi du bruit additif ϵ est telle que la densité conditionnelle de y s'écrit :

$$p(y \mid x, \beta, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \left(\frac{\overbrace{(y - f(x, \beta))^2}^{\epsilon^2}}{2\sigma^2} \right) = p(\epsilon) \tag{4.5}$$

Pour n abscisses x_i, on considère alors n réalisations indépendantes des sorties y_i (ou de manière équivalente des bruits ϵ_i) donnant un ensemble d'apprentissage $D = \{x_i, y_i\}_{i=1\ldots n}$. Les bruits ϵ_i sont iid.

4.3.2 Maximum de vraisemblance

En postulant un modèle tel que le précédent, on se donne les moyens d'apprendre les paramètres inconnus β et σ^2 liés à l'équation (4.4) à partir d'un ensemble d'apprentissage. Dans l'approche d'estimation de β et σ^2 au «Maximum de Vraisemblance» (MV), on veut choisir les valeurs de β et σ^2 qui rendent *maximale* la vraisemblance :

$$L(D, \beta, \sigma^2) = \prod_{i=1}^{n} p(y_i \mid x_i, \beta, \sigma^2) = \prod_{i=1}^{n} p(\epsilon_i) \tag{4.6}$$

Cela revient intuitivement à supposer que, observant D, nous observons l'échantillon le plus probable (qui peut donc nous permettre de retrouver β et σ^2) en résolvant :

$$[\widehat{\beta}_{MV}, \widehat{\sigma^2}_{MV}] = \underbrace{\text{argmax}}_{\beta,\sigma^2} L(D, \beta, \sigma^2) = \underbrace{\text{argmax}}_{\beta,\sigma^2} lnL(D, \beta, \sigma^2) \tag{4.7}$$

Après réécriture, de la «log-vraisemblance» on a :

$$l(D, \beta, \sigma^2) = lnL(D, \beta, \sigma^2) = -\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - f(x_i, \beta))^2 - nln(\sqrt{2\pi\sigma^2}) \tag{4.8}$$

Si on introduit $R_0(\beta)^2 = \sum_{i=1}^{n} (y_i - f(x_i, \beta))^2$ et $\theta = \sigma^2 > 0$ pour simplifier, on peut alors écrire les CN1. Pour l'inconnue β, on obtient :

$$\frac{\partial l(D, \beta, \theta)}{\partial \beta}) = 0 \Leftrightarrow \frac{\partial R_0(\beta)^2}{\partial \beta}) = 0 \rightarrow \widehat{\beta}_{MV} \text{ Ordinary Least Squares solution} \tag{4.9}$$

On retrouve, dans ce cas de bruit additif gaussien, une équivalence entre «Maximum de Vraisemblance» et «Moindres Carrés Ordinaires». En particulier, dans le cas de régression linéaire simple que nous traitons dans le paragraphe suivant ($f(x, \beta) = x^T\beta$), on retrouve la solution usuelle des moindres carrés linéaires faisant appel à la pseudo-inverse. En outre, la CN1 pour notre seconde inconnue c'est-à-dire l'échelle (ou la variance) du bruit donne :

$$\frac{\partial l(D, \widehat{\beta}_{MV}, \theta)}{\partial \theta}) = 0 \Leftrightarrow \frac{\partial}{\partial \theta} \left(-\frac{1}{2\theta} R_0(\widehat{\beta}_{MV})^2 - \frac{n}{2} ln(2\pi\theta) \right) = 0 \tag{4.10}$$

qui conduit à

$$\theta = \frac{R_0(\widehat{\beta}_{MV})^2}{n} \tag{4.11}$$

soit

$$\widehat{\sigma^2}_{MV} = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i, \widehat{\beta_{MV}}))^2 \tag{4.12}$$

4.4 Modèle de régression linéaire simple

4.4.1 Modèle linéaire avec bruits gaussiens

Dans ce paragraphe, on part du cas gaussien précédent. Mais en plus de ce qui précède, on suppose que f est linéaire (ou affine) en β dans les n équations obtenues à partir d'un ensemble d'apprentissage D et des bruits iid gaussiens $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$:

$$y_i = f(\mathbf{x_i}, \beta) + \epsilon_i \qquad \forall i \in 1 \dots n \tag{4.13}$$

Les deux situations classiques sont :

a) $f(x, \beta) = \mathbf{x}^T\beta$ (cas linéaire simple) avec β et $\mathbf{x} \in \mathbb{R}^p$

b) $f(x, \beta) = \mathbf{C}(\mathbf{x})^T\beta + d(x)$ (cas affine) avec $\beta, \mathbf{C}(\mathbf{x}) \in \mathbb{R}^p$ et $d(x) \in \mathbb{R}$

Pour le cas a), on définit alors un modèle de régression linéaire simple en adoptant les notations vectorielles suivantes :

$$Y = X\beta + E \tag{4.14}$$

Avec les sorties y_i (resp. bruits ϵ_i et entrées vectorielles $\mathbf{x_i}$) rangé(e)s dans un vecteur Y (resp. vecteur E et une matrice $X \in \mathcal{M}_{\mathbb{R}}(n, p)$) selon :

$$Y = \begin{pmatrix} y_1 \\ \vdots \\ y_i \\ \vdots \\ y_n \end{pmatrix} \quad X = \begin{pmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_i^T \\ \vdots \\ \mathbf{x}_n^T \end{pmatrix} \quad E = \begin{pmatrix} \epsilon_1 \\ \vdots \\ \epsilon_i \\ \vdots \\ \epsilon_n \end{pmatrix}$$

Ainsi Y est un *vecteur aléatoire* de dimension n, X est une matrice $n \times p$ connue (déterministe) dite parfois «du plan d'expérience»[1], β est le vecteur de dimension p des paramètres inconnus du modèle, E est le vecteur gaussien centré, de dimension n des erreurs. Le passage de l'équation (4.4) à l'équation (4.14) ainsi que les notations adoptées méritent quelques mots. Ces équations sont de même nature à ceci prés que f est supposée linéaire et que les variables aléatoires dans l'équation (4.4) sont multidimensionelles dans l'équation (4.14). En particulier la variable aléatoire associé au bruit ϵ qui contamine chaque sortie devient un vecteur aléatoire E pour lequel on a :

$$E \sim \mathcal{N}(\mathbf{0}_n, \sigma^2\mathbf{I}_n) \tag{4.15}$$

Avec ces notations, la valeur observée du vecteur aléatoire Y est la somme d'une composante déterministe $X\beta$ et d'une composante aléatoire gaussienne E qui modélise le bruit. L'hypothèse $E[E] = \mathbf{0}_n$ signifie que la composante déterministe du vecteur Y est sa moyenne. L'hypothèse $\mathrm{Var}[E] = \sigma^2\mathbf{I}_n$ signifie que les composantes ϵ_i sont des variables aléatoires non corrélées. Une hypothèse légèrement plus générale sur la matrice de variance-covariance[2] permettrait d'introduire une corrélation entre les observations ou une précision différente sur les composantes Y_i de Y.

[1] Elle contient les n expériences associées aux entrées $\mathbf{x_i}$

[2] $\mathrm{Var}[E] = \sigma^2 G$ avec G matrice symétrique définie positive.

4.4.2 Moindres carrés linéaires

Dans la suite, on on considère en outre que la matrice X est de rang plein (c'est-à-dire que la matrice carrée $X^T X$ d'ordre p est inversible.

On montre facilement que le terme $R_0(\beta)^2 = \sum_{i=1}^{n}(y_i - f(\mathbf{x_i}, \beta))^2$ introduit précédemment pour l'estimation de β revient ici à :

$$R_0(\beta)^2 = ||Y - X\beta||^2 \tag{4.16}$$

La norme euclidienne choisie correspond aux approches classiques de l'estimation aux moindres carrés (Ordinary Least Squares, OLS) et au maximum de vraisemblance (MV). Dans le paragraphe de bilan qui suit, nous montrerons clairement que minimiser ce critère aux moindres carrés linéaires (pour apprendre les paramètres qui expliquent au mieux un ensemble d'apprentissage) est tout simplement équivalent à minimiser un risque empirique avec une perte quadratique.

L'estimateur $\widehat{\beta} = \widehat{\beta_{MV}} = \widehat{\beta_{OLS}}$ minimisant $R_0(\beta)^2$ est solution du système des équations dites normales :

$$X^T X\widehat{\beta} - X^T Y = 0 \tag{4.17}$$

Deux caractérisations usuelles conduisent à ce résultat. La caractérisation différentielle consite à dériver le critère $R_0(\beta)^2$. La caractérisation algébrique (ou géométrique) consiste à projeter Y orthogonalement sur $\mathrm{Im}(X)$. On retrouve alors les équations de la pseudo-inverse de X si elle est de rang plein :

$$\widehat{\beta} = (X^T X)^{-1} X^T Y \tag{4.18}$$

Il faut d'emblée comprendre que l'estimation dépend de X et de Y. Elle dépend donc de l'ensemble d'apprentissage $D = \{\mathbf{x_i}, y_i\}_{i=1\ldots n}$ considéré : lorsque D varie $\widehat{\beta}$ varie. On adoptera la notation $\widehat{\beta}_D$ pour souligner cette dépendance à l'ensemble d'apprentissage.

Il est fréquent à ce niveau de supposer que le modèle (eq. 4.14) est exact. Ce qui signifie qu'il existe un vecteur β^* de p paramètres qui explique les données d'apprentissage $D = \{\mathbf{x_i}, y_i\}_{i=1\ldots n}$ disponibles et rangées dans Y et X.

$$Y = X\beta^* + E \tag{4.19}$$

On peut reformuler l'estimateur $\widehat{\beta}$ en fonction des paramètres exacts en combinant les équations (4.18) et (4.19) selon :

$$\widehat{\beta} = \beta^* + (X^T X)^{-1} X^T E$$

On peut finalement montrer :

- que $\widehat{\beta}$ est sans biais,
- que sa matrice de variance-covariance est :

$$\Sigma_{\widehat{\beta}} = \sigma^2 (X^T X)^{-1}$$

- que $\widehat{\beta}$ est gaussien :

$$\widehat{\beta} \sim \mathcal{N}\left(\beta^*, \sigma^2(X^TX)^{-1}\right)$$

Ces résultats sont fondamentaux. Ils permettent en particulier d'étudier la variance des estimateurs selon les données d'entrées $\mathbf{x_i}, i = 1 \dots n$ c'est-à-dire selon X. On peut, en choisissant/sélectionnant bien ces données d'entrées, réduire la variabilité des prédictions.

Ayant estimé (appris) les paramètres $\widehat{\beta}$, une prédiction pour une donnée d'entrée $\mathbf{x}$ s'écrit $\widehat{y} = \mathbf{x}^T\widehat{\beta}$ dans le modèle de régression linéaire simple.

Vectoriellement, les prédictions associées aux données d'apprentissage s'expriment selon :

$$\widehat{Y} = X\widehat{\beta} \tag{4.20}$$

On appelle matrice de prédiction (ou matrice chapeau) la matrice

$$H = X(X^TX)^{-1}X^T$$

telle que, dans le cas du modèle de régression linéaire simple :

$$\widehat{Y} = HY \tag{4.21}$$

On montre que la matrice chapeau H (comme «Hat») associée à une matrice X (d'un plan d'expérience) est la matrice de la projection orthogonale sur le sous-espace $\mathrm{Im}(X)$

4.5 Bilan vis-à-vis de l'apprentissage

En guise de résumé, on peut reformuler les étapes précédentes permettant une interprétation de la régression au sens de l'apprentissage :

1. On dispose d'un ensemble d'apprentissage $D = \{\mathbf{x_i}, y_i\}_{i=1\dots n}$ dont les données de sortie et d'entrée peuvent être rangées respectivement dans un vecteur Y et une matrice X .

2. On postule un modèle de régression (paramétrique) qui modélise simplement la dépendance statistique des entrées et des sorties dans l'équation fondamentale de l'EPE (équation 5.1). Par exemple $Y = f(X, \beta) + E$ ou plus simplement encore $Y = X\beta + E$ dans le cas de la régression linéaire simple. Le modèle de données fonctionnelles sous-jacent simplifie beaucoup les choses en limitant la source d'aléas au vecteur aléatoire E et en supposant $f(X, \beta)$ (ou $X\beta$) déterministes. Lorsque E varie (sachant X, conditionnellement à X), D varie selon le principe simple illustré par la figure 4.1.

3. A partir d'un échantillon d'aprentissage D, on apprend les paramètres par minimisation d'un risque empirique avec une perte adaptée à la régression. En formulation paramétrique, cela revient à résoudre :

$$\widehat{\beta} = \widehat{\beta}_D = \underset{\beta}{\operatorname{argmin}}\ R_D(\beta) = \frac{1}{n}\sum_{i=1}^{n} e(f(\mathbf{x_i}, \beta), y_i) \tag{4.22}$$

Le critère à optimiser revient souvent à l'un de ceux utiliser en estimation de paramètre. On a en particulier rappelé le lien entre ce risque empirique avec perte quadratique[3], l'estimation aux moindres carrés et le maximum de vraisemblance.

4. Ayant appris les paramètres (qui dépendent de l'ensemble d'apprentisage D, $\widehat{\beta} = \widehat{\beta}_D$), on dispose désormais d'un prédicteur dont la forme est la suivante :

$$\mathbf{x} \rightarrow \boxed{\widehat{\beta} = \widehat{\beta}_D} \rightarrow \widehat{y} = f(\mathbf{x}, \widehat{\beta}) \quad (= \mathbf{x}^T\widehat{\beta} \text{ dans le cas linéaire}) \tag{4.23}$$

5. On peut alors se poser les questions clés en apprentissage : la complexité du modèle est-elle bien choisie ? quelle est la capacité de généralisation du prédicteur obtenu ? etc.

Dans le thème consacré à la généralisation, nous avons introduit la régression au sens de l'apprentissage avec les modèles polynomiaux (paragraphe 2.3). Notre introduction, que le lecteur est invité à parcourir à nouveau, correspond bien aux étapes précédentes. Le modèle polynomial choisi (équation 2.9) est linéaire en ω. Il est compatible avec un modèle de régression linéaire simple pour lequel le risque empirique (équations 2.10 et 2.11) revient à un critère aux moindres carrés (équation 2.12). Le lecteur se convaincra facilement que le modèle de régression simple, de forme générale donnée par l'équation (4.14), revient ici à : $\mathsf{Y} = \mathsf{A}\omega + E$. Résoudre en ω le problème associé à l'équation (2.12) est équivalent à minimiser en β le critère de l'équation (4.16) ce qui conduit, si X (c'est-à-dire A) est de rang plein, à une solution via la pseudo-inverse donnée par l'équation (4.18). Enfin, les prédictions associées à un modèle de régression polynomial (figure 2.5) :

- répondent au principe donné par l'équation (4.23) pour une entrée unique,
- sont données mathématiquement par l'équivalent des équations (4.20) ou (4.21) pour les prédictions vectorielles associées à plusieurs entrées.

4.6 Exercice

Dans cet exercice on se place dans le cadre précédent de la régression linéaire simple avec un modèle exact et des bruits gaussiens qui conduisent à :

$$\widehat{\beta} = (X^TX)^{-1}X^TY = \beta^* + (X^TX)^{-1}X^TE$$

1. Que vaut H^2 ? avec H la matrice chapeau associée à X. Commenter le résultat.
2. Exprimer l'erreur d'apprentissage $\texttt{err}(\widehat{\beta})$ en fonction de Y, $\widehat{Y}$ et n

[3] $e(\widehat{y}, y) = ||\widehat{y} - y||_2^2$ pour une variable de sortie y vectorielle ou $e(\widehat{y}, y) = (\widehat{y} - y)^2$ pour y scalaire

3. Etablir que :
$$\texttt{err}(\widehat{\beta}) = \frac{1}{n}(E^T E - E^T H E)$$

4. On veut maintenant obtenir une espérance de cette erreur d'apprentissage lorsque l'ensemble d'apprentissage D varie. Etablir que :
$$\mathrm{E}_D(\texttt{err}(\widehat{\beta})) = \sigma^2(1 - \frac{p}{n})$$
indication :
$\mathrm{E}_{E|X}(E^T M E) = \sigma^2 \texttt{trace}(M)$ pour toute matrice M carrée et un vecteur gaussien $E \sim \mathcal{N}(0, \sigma^2 \texttt{Id}_N)$

Thème 5

Approximations de l'EPE

Vincent Charvillat
Décembre 2010

Dans ce thème, on s'intéresse à l'évaluation de l'EPE dont la forme générale, pour un prédicteur ($h \in \mathcal{H}$) et une perte e, est la suivante :

$$R(h) = EPE(h) = E_{(X,Y)}[e(h(X), Y)] = \int_{\mathcal{X}} \int_{\mathcal{Y}} e(h(x), y) P_{X,Y}(x, y) dxdy \qquad (5.1)$$

Pour évaluer l'EPE, la difficulté vient du fait que la loi des données est inconnue. On présente donc d'abord des estimateurs basés sur le risque empirique. Ces estimateurs permettent d'approcher l'EPE, de comparer différents prédicteurs et de sélectionner celui dont la capacité de généralisation est la meilleure. La validation croisée est à utiliser en pratique lorsqu'on dispose d'une quantité limitée de données supervisées.

5.1 Estimateur empirique de l'EPE

5.1.1 Cadre

Soit h^* un prédicteur minimisant le risque empirique associé à un échantillon d'apprentissage $D = \{x_i, y_i\}_{i=1...n}$:

$$h^* = \underset{h \in \mathcal{H}}{\operatorname{argmin}}\ R_D(h) = \frac{1}{n} \sum_{i=1}^{n} e(h(x_i), y_i) \qquad (5.2)$$

Nous avons déjà vu que, pour des classes de prédicteurs suffisamment complexes, l'erreur d'apprentissage $\texttt{err}(h^*) = R_D(h^*)$, c'est-à-dire le risque empirique à la solution h^*, est une estimation excessivement optimiste du vrai risque $R(h)$. La figure

2.7 résume ce phénomène dit de «sur-apprentissage» pour des classes de prédicteurs $\mathcal{H}_j$ de complexité croissante avec j. Evaluer la capacité de généralisation d'un prédicteur à partir de l'échantillon D qui a permis de le construire n'est pas satisfaisant.

Pour estimer correctement la capacité de généralisation de h^*, c'est-à-dire $R(h^*) = EPE(h^*)$, nous pouvons utiliser un autre ensemble de données supervisées , **distinct de** D. On appelle cet ensemble de $m \neq n$ nouvelles données, un échantillon de test et on le note généralement T. Si on note à nouveau[1] $z_i = (x_i, y_i)$ les données supervisées contenues dans $T \neq D$, on a simplement m nouvelles réalisations z_i du vecteur aléatoire $Z = (X, Y)$ de loi $P_Z(z) = P_{X,Y}(x, y)$ qui a déjà permis de réaliser (indépendamment) les données de D.

Un estimateur empirique de l'EPE pour h^* est déduit du risque empirique calculé sur T selon :

$$R_T(h^*) = \frac{1}{m} \sum_{i=1}^{m} e(h^*(x_i), y_i) \tag{5.3}$$

Ce risque empirique mesure effectivement une capacité de généralisation via la moyenne empirique des écarts entre les prédictions $h^*(x_i)$ et les sorties attendues y_i sur les nouvelles données de $T = \{x_i, y_i\}_{i=1\ldots m} \neq D$.

5.1.2 Propriétés de l'estimateur du risque empirique

Plus formellement, si on note Z^m la variable aléatoire associée à T : $Z^m = (Z_1, Z_2, \ldots, Z_m)$. Les composantes $Z_i = (X_i, Y_i)$ sont iid ($\forall i, Z_i$ est distribuée selon P_Z). L'estimateur de $R(h) = EPE(h)$ considéré est le suivant :

$$R_{Z^m}(h) = \frac{1}{m} \sum_{i=1}^{m} e(h(X_i), Y_i) \tag{5.4}$$

Cet estimateur a de bonnes propriétés :

- Il est non biaisé : $E_{Z^m}\left[R_{Z^m}(h)\right] = R(h)$
- Sa variance diminue avec le nombre m d'exemples dans l'échantillon de test (propriété de convergence) :

$$\mathrm{Var}_{Z^m}\left[R_{Z^m}(h)\right] = \frac{\mathrm{Var}_{Z=(X,Y)}\left[e(h(X), Y)\right]}{m} \tag{5.5}$$

L'absence de biais est immédiate à démontrer et la réduction de variance découle du théorème suivant : Soit $\{V_i\}$ n v.a. iid tq $\forall i, E[V_i] = \mu$ et $Var[V_i] = \sigma^2$ alors $Var\left[\frac{1}{n}\sum_{i=1}^{n} V_i\right] = \frac{\sigma^2}{n}$. De sorte que le risque empirique $R_{Z^m}(h)$ est un bon estimateur de l'EPE lorsque m est elevé et pour autant que $\mathrm{Var}_{Z=(X,Y)}\left[e(h(X), Y)\right]$ soit bornée.

Ces bonnes propriétés encouragent l'utilisation du risque empirique comme estimateur du vrai risque. En pratique, si l'on dispose d'énormément de données supervisées, il est possible de scinder les données en, d'une part un échantillon D

[1] On pourrait distinguer, sans intérêt réel, $D = \{x_i, y_i\}_{i=1\ldots n}$ et $T = \{x'_i, y'_i\}_{i=1\ldots m}$.

d'apprentissage et, d'autre part, un échantillons de test T. Pour respectivement apprendre des prédicteurs pris dans différentes classes et tester celui qui, parmi eux, généralise le mieux.

5.1.3 Du bon usage du risque empirique

Les bonnes propriétés asymptotiques du risque empirique justifient son utilisation lorsque la taille des échantillons est importante. A titre d'exemple, le sur-apprentissage d'un polynôme de degré 9 illustré par la figure 2.6(d) intervient en raison d'un ensemble d'apprentissage de taille limitée. Dans la figure 5.1, la taille de l'échantillon d'apprentissage augmente considérablement. Cette figure montre qu'apprendre un polynôme assez complexe (degré 9) en minimisant le risque empirique sur un échantillon important retrouve du sens.

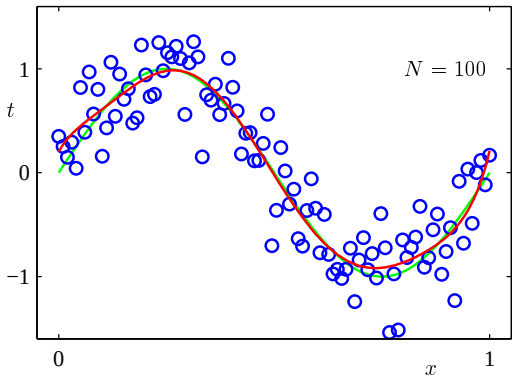


Figure 5.1: Effet positif associé à un grand ensemble d'apprentissage.

En fait, l'approximation du vrai risque par un risque empirique calculé sur un ensemble $S = \{x_i, y_i\}_{i=1...m}$ consiste à estimer empiriquement la loi jointe $P_{X,Y}(x, y)$ par :

$$\widehat{P}_{X,Y}(x, y) = \frac{1}{m} \sum_{i=1}^{m} \delta(x - x_i)\delta(y - y_i) \tag{5.6}$$

Ce qui conduit effectivement à :

$$R(h) = EPE(h) \approx \int_{\mathcal{X}} \int_{\mathcal{Y}} e(h(x), y)\widehat{P}_{X,Y}(x, y)dxdy = R_S(h) \tag{5.7}$$

Pour m suffisamment grand, $R_S(h)$ nous donne une bonne estimation de $R(h)$ ce qui motive l'utilisation de la minimisation du risque empirique pour l'apprentissage et/ou le test d'un prédicteur. Bien entendu, en pratique, le nombre de données supervisées est toujours le facteur limitant.

5.2 Validation croisée

Les techniques précédentes ne s'appliquent donc pas lorsque l'on dispose d'un (unique) ensemble d'apprentissage de taille $n = |D|$ limitée. Séparer $D = \{x_i, y_i\}_{i=1...n}$ pour en extraire un échantillon de test de taille suffisante n'est pas possible.

5.2.1 Validation croisée, "K-fold"

Pour approcher l'EPE lorsqu'on manque de données, la méthodologie de la validation croisée permet d'utiliser :

- toutes les données pour tester un prédicteur h,
- presque toutes les données pour construire h.

L'idée est d'itérer l'estimation de l'erreur de généralisation sur plusieurs échantillons de validation extraits de D puis d'en calculer la moyenne. Moyenner permet de réduire la variance et ainsi d'améliorer la précision de l'approximation associée de l'EPE si la taille de D est limitée.

L'algorithme de la validation croisée (**K-fold Cross-Validation**) est décrit ci-après :

o Découper aléatoirement l'échantillon D en K ensembles disjoints $\{T_1, T_2 \dots T_K\}$ (approximativement) de même taille $|T_i|$ et réalisant une partition de D.

o **Pour** $i = 1, 2, \dots K$ **Faire**

1. Apprendre $h^*_{D-T_i}$ par minimisation de $R_{D-T_i}(h)$
2. Evaluer ce prédicteur : $R_i = R_{T_i}(h^*_{D-T_i})$

Fin Pour

o Calculer $CV_{Kfold} = R^K_{CV} = \frac{1}{K} \sum_{i=1}^{K} R_i$

5.2.2 Validation croisée, "Leave-One-Out"

Lorsque $K = |D|$, la validation croisée consiste à exclure successivement chaque donnée de D pour tester le modèle appris en privant D d'une seule donnée. On parle alors de validation croisée **Leave-One-Out (LOO)**.

Dans la séance de travaux pratiques (TP) consacrée à la validation croisée, on montre que le calcul de l'approximation $CV_{LOO} = R^{K=1}_{CV}$ peut parfois s'effectuer rapidement. Si l'on sait facilement passer du prédicteur appris à partir de tout D à celui appris sur D privé d'une seule donnée, on peut éviter les n apprentissages (étape 1.) de la boucle **Pour** ci-dessus.

La notion de **Validation Croisée Généralisée** (GCV en anglais) peut alors être introduite et reliée aux critères de sélection de modèles comme cela est expliqué dans le TP. Le lecteur est encouragé à compléter ces éléments de cours sur la validation croisée avec les explications, compléments et interprétations vues en TP.

5.2.3 Variance du score de validation croisée

Très souvent, les résultats de validation croisée sont présentés sous la forme d'un intervalle :

$$R_{CV}^K \pm \sigma_{CV}^K$$

avec

$$\sigma_{CV}^K \approx \sqrt{\frac{1}{K(K-1)} \sum_{i=1}^{K} (R_i - R_{CV}^K)^2}$$

Cette approximation est partiellement justifiée par les deux points suivants :

(i) la variance σ^2 des R_i vues comme des v.a. peut être estimée empiriquement par la moyenne des écarts quadratiques à la moyenne R_{CV}^K :

$$\sigma^2 \approx \frac{1}{(K-1)} \sum_{i=1}^{K} (R_i - R_{CV}^K)^2$$

(ii) l'estimateur R_{CV}^K est une moyenne de K v.a. de variance σ^2 d'où $\sigma_{CV}^K \approx \sigma/\sqrt{K}$.

La justification n'est que partielle car on a (ii) si les R_i sont indépendantes. Ce n'est pas tout à fait le cas puisque les échantillons d'apprentissage se recouvrent pour $K > 2$. Un tel intervalle visant à estimer l'incertitude liée au score de la validation croisée est, sous cette forme, assez grossier.

5.3 Exercice

Dans cet exercice, on reprend les arguments développés dans le paragraphe 5.1.2. On s'intéresse plus précisément aux classifieurs avec la perte non-informative 0-1 (e_z) donnée par l'équation (2.1), on peut :

1. Déterminer la loi de la v.a. $e_z(h(X), Y)$ pour un h donné et donner ses deux premiers moments en fonction de $R(h)$.

2. En déduire :
$$Var\left[R_{Z^m}(h)\right] = \frac{R(h)(1-R(h))}{m} \tag{5.8}$$

3. Déterminer la loi de $m * R_{Z^m}(h)$, donner ses deux premiers moments et montrer la cohérence de la réponse avec les résultats précédemment établis.

4. Pour m grand, si on approche, via le théorème central limite, la distribution de $m * R_{Z^m}(h)$ par une distribution normale, on vous demande de donner les équations à résoudre pour caractériser l'intervalle de confiance $t_{1-\delta}$ tel que, lorsque T varie selon les tirages de Z^m :

$$Pr_{Z^m}\{|R_{Z^m}(h) - R(h)| \leq t_{1-\delta}\} \geq 1 - \delta \quad (\approx 0.9) \tag{5.9}$$

Thème 6

Solutions de l'EPE et méthodes dérivées

Vincent Charvillat
Décembre 2010

Dans ce thème, on s'intéresse à la minimisation de l'EPE dont la forme générale, pour un prédicteur ($h \in \mathcal{H}$) et une perte e, est la suivante :

$$R(h) = EPE(h) = E_{(X,Y)}[e(h(X), Y)] = \int_{\mathcal{X}} \int_{\mathcal{Y}} e(h(x), y) P_{X,Y}(x, y) dx dy \qquad (6.1)$$

On peut résoudre formellement le problème de la minimisation de l'EPE. Nous donnons les solutions théoriques pour la régression et la classification. Ces solutions ne sont que «théoriques» puisqu'elles dépendent de la loi inconnue des données $P_{X,Y}(x, y) = P_{Y|X=x}(y) P_X(x)$. Toutefois, ces solutions justifient le bien-fondé (et les limites) de techniques très utilisées en pratique : prédicteurs non-paramétriques aux plus proches voisins, classifieur Bayésien, etc.

6.1 Solution de l'EPE pour la régression

6.1.1 Espérance conditionnelle

On souhaite minimiser l'EPE pour la régression mise sous la forme suivante :

$$R(h) = EPE(h) = \int_{\mathcal{X}} \left[\int_{\mathcal{Y}} (y - h(x))^2 p(y|x) dy \right] p(x) dx \qquad (6.2)$$

Les sorties sont scalaires ($\mathcal{Y} \subset \mathbb{R}$), la perte choisie est $e(y, \widehat{y}) = (y - \widehat{y})^2$ et les lois (continues) sont à densité.

On définit la régression (ou espérance conditionnelle) comme la fonction :

$$r(x) = E[Y|X = x] = \int_{\mathcal{Y}} yp(y|x)dy \tag{6.3}$$

Théorème La régression $r(x)$ est la fonction qui minimise l'EPE (eq. 6.2)

La démonstration est un exercice simple si l'on observe que la minimisation de l'EPE peut se faire indépendamment pour chaque x fixé ou bien si on vérifie que le théorème est vrai en développant :

$$R(h) = \int_{\mathcal{X}} \int_{\mathcal{Y}} (y - r(x) + r(x) - h(x))^2 p(x, y)dxdy \tag{6.4}$$

Dans le paragraphe 4.1 introduisant les modèles usuels de régression, nous avions indiqué que la connaissance de la loi des sorties conditionnelle à l'entrée $p(y|x) = p(x, y)/p(x)$ banaliserait le problème de la minimisation de l'EPE. Nous retrouvons donc, avec le théorème précédent, qu'en supposant l'existence d'une dépendance statistique de Y sur X selon l'approche du pararaphe 4.1 :

$$Y = f(X) + \epsilon \tag{6.5}$$

la solution de la minimisation de l'EPE est évidemment la fonction $x \to f(x)$ qui est la régression $f(x) = E[Y|X = x]$ puique ϵ est un bruit d'espérance nulle, $E[\epsilon] = 0$. Ainsi en régression, la meilleure prédiction de Y pour une entrée $X = x$ donnée est intuitivement l'espérance (c'est-à-dire une moyenne) des sorties pouvant être générées. Bien entendu, si l'on change la perte, la solution change. Un exercice donné en fin de chapitre traite en particulier des pertes L_p pour $p \neq 2$.

6.1.2 Méthode des k plus proches voisins (kppv)

Les méthodes des k plus proches voisins (kppv) sont justifiées par l'intuition précédente (prédire en moyennant les sorties possibles pour une entrée donnée). Si on considère un ensemble d'apprentissage $D = \{x_i, y_i\}_{i=1..n}$, il est improbable d'avoir au sein de D plusieurs entrées égales à un x_0 fixé et donc plusieurs sorties à «moyenner». Le prédicteur aux kppv calcule donc une moyenne empirique des sorties en utilisant les plus proches voisins de x_0 :

$$\widehat{f_{kppv}}(x_0) = \text{moyenne}\left(y_{\sigma(i)}|x_{\sigma(i)} \in V_k(x_0)\right) = \frac{1}{k}\sum_{i=1}^{k} y_{\sigma(i)} \tag{6.6}$$

Dans cette formule, les k plus proches voisins de x_0 appartiennent au voisinage $V_k(x_0)$. Le choix d'une distance permettant une définition formelle de $V_k(x_0)$ est à discuter. Deux approximations sont utilisées vis-à-vis de l'équation (6.3) :

a) l'espérance est approchée par une moyenne empirique,

b) le conditionnement à $X = x_0$ est relâché en prenant des entrées voisines de x_0.

Pour de grands échantillons d'apprentissage (n grand), les points de $V_k(x_0)$ ont des chances d'être proches de x_0 et si k devient grand, la moyenne deviendra suffisamment stable pour obtenir un prédicteur régulier. Il faut en effet bien comprendre que le prédicteur de l'équation (6.6) est par construction constant par morceaux : pour un point très proche de x_0, disons $x_0 + \epsilon$, les k voisins ne changent pas...

En fait, sous des hypothèses de régularité suffisantes pour la loi jointe $p(x, y)$, on peut montrer que :

- Si $n \to \infty$, $k \to \infty$ tq. $k/n \to 0$
- Alors $\widehat{f_{kppv}}(x) \to E[Y|X = x]$

Ces propriétés sont très prometteuses. Est-il alors raisonnable, pour construire un prédicteur, de postuler des dépendances élémentaires entres entrées et sorties, de postuler un modèle hypothétiquement linéaire (eq 4.14) alors que le prédicteur empirique de l'équation (6.6) tend vers la solution ? En d'autres termes, tenons-nous le prédicteur idéal ? Malheureusement la réponse est négative. Le fléau de la dimension (de l'espace d'entrée $\mathcal{X}$) va limiter l'intérêt de cette approche.

6.1.3 Fléau de Bellman ("Curse of dimensionality")

Le prédicteur aux kppv de l'équation (6.6) est une méthode de prédiction «locale» opérant au voisinage $V_k(x_0)$ d'un x_0 donné. Insistons sur le fait que nous *souhaitons* qu'elle soit locale pour que l'approximation b) du paragraphe précédent soit la plus raisonnable possible. Dans de nombreux problèmes pratiques où on veut mettre au point un mécanisme de prédiction à partir de mesures (par exemple physiques, biologiques etc.), les (mesures) d'entrée sont nombreuses. Ce qui revient souvent à $x_0 \in \mathcal{X} \subset \mathbb{R}^p$ avec p grand. Le problème du fléau de la dimension est qu'une méthode par voisinage dans $\mathbb{R}^p$ avec p grand n'est plus du tout «locale».

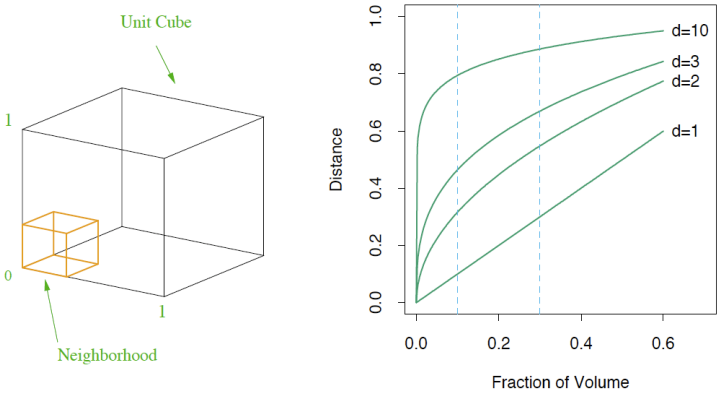


Figure 6.1: Illustration du "Curse of dimensionality", d'après Hastie et al.

Pour comprendre le problème, considérons une procédure aux plus proches voisins pour des entrées x_i uniformément réparties dans un (hyper)cube unité de $\mathbb{R}^p$. La

figure 6.1 donne, à gauche, une représentation de ce cube unité en dimension p. Considérons un voisinage V autour d'un point d'intérêt x_0 qui représente une fraction r des obsservations disponibles : on veut par exemple que le voisinage V intègre 5 % ou 10 % des données. Le voisinage représente une fraction r du volume du cube unité. En dimension p, un tel voisinage a un volume équivalent à un cube dont les côtés ont pour longueur $l_p(r) = r^{1/p}$: un tel cube de voisinage apparaît en rouge dans le cube unité et vérifie :

$$\text{Volume}(V) = \underbrace{r^{1/p} \times \cdots \times r^{1/p}}_{p \text{ fois}} = r$$

La partie droite de la figure 6.1 donne les graphes de $r \to l_d(r) = r^{1/d}$ avec la dimension d variant de 1 à 10 et r, en abscisse, variant de 0.0 à 0.6. On peut voir sur ce type de figure que pour capturer 1% du volume en dimension 10, le côté de V mesure 0.63 : $l_{10}(0.01) = 0.63$. Dès lors V n'est plus un voisinage local ce qui rend la prédiction aux kppv inappropriée. La réduction de r n'est pas une bonne solution puisqu'elle conduira, comme nous le verrons dans le thème suivant à une augmentation de variance dans la prédiction.

De surcroît, il est évident que pour échantillonner l'espace $\mathcal{X}$ avec une densité constante, le fléau de la dimension intervient : si nous échantillonnons $[0, 1]$ avec $N_1 = 100$ exemples, il faut échantillonner $[0, 1]^{10}$ avec $N_{10} = 100^{10}$ exemples pour obtenir la même densité. En d'autres termes, la densité des échantillons d'entrée (les x_i) en haute dimension est toujours faible. Cela limite également l'intérêt des méthodes de type kppv en grande dimension.

Disons pour conclure cette discussion, que l'utilisation des prétraitements visant une réduction de dimension (voir le Thème 2) prend donc tout son sens en amont d'une prédiction aux plus proches voisins.

6.2 Solution de l'EPE pour la classification

On considère un classifieur $h : \mathcal{X} \to \mathcal{Y}$ dont :

- les données d'entrée sont dans $\mathcal{X}$ et sont distribuées selon la loi de densité p,
- l'espace discret de sortie à K classes est $\mathcal{Y} = \{\omega_1, \ldots, \omega_K\}$ avec une loi conditionnelle discrète sachant l'entrée $X = x$ qui est définie par les K probabilités $P_{Y|X=x}(\omega_i) = Pr(Y = \omega_i|X = x) = Pr(\omega_i|X = x)$

L'EPE que l'on veut minimiser prend la forme suivante :

$$EPE(h) = \int_{\mathcal{X}} p(x) \left[\sum_{y \in \{\omega_1, \ldots, \omega_K\}} e(h(x), y) Pr(Y = y|X = x) \right] dx \tag{6.7}$$

Ou de manière plus compacte encore :

$$EPE(h) = E_X \left[\sum_{i=1}^{K} e(h(X), \omega_i) Pr(\omega_i|X) \right] \tag{6.8}$$

6.2.1 Solution pour une perte générale

Comme vu précédemment pour la régression, il faut observer que la minimisation de l'EPE (pour des pertes positives) peut s'effectuer ponctuellement pour chaque x. Ainsi le classifieur optimal en x_0 est :

$$\widehat{h}(x_0) = \omega_{i_0} = \underset{g \in \{\omega_1, \ldots, \omega_K\}}{\operatorname{argmin}} \sum_{i=1}^{K} e(g, \omega_i) Pr(\omega_i | X = x_0)$$

Les interprétations suivantes sont possibles :

- le terme $e(g, \omega_i)$ représente le coût de décider la classe g alors que la classe correcte est ω_i,
- le terme $\sum_{i=1}^{K} e(g, \omega_i) Pr(\omega_i | X = x_0)$ représente le risque (coût) de prédire g observant x_0 en entrée

Lorsque l'on modélise un problème de classification, le choix des $K \times K$ valeurs $e(\omega_j, \omega_i)$ avec $i, j \in \{1, \ldots K\}^2$ est donc une étape fondamentale. On remarquera en particulier qu'il est parfois pertinent de prendre : $e(\omega_j, \omega_i) \neq e(\omega_i, \omega_j)$ avec $i \neq j$. On pourra aussi observer que le choix de chaque perte permet naturellement de considérer une $K + 1^{\text{ième}}$ classe, dite *classe de rejet*, pour laquelle aucune décision de classification n'est prise. Classe additionnelle pour laquelle des pertes adaptées au problème et à l'application doivent être adroitement choisies.

6.2.2 Solution pour une perte non-informative

On dit que la perte 0-1 notée e_z est non-informative par opposition au cas précédent où chaque erreur de classification a un coût particulier. Avec la perte 0-1, c'est du tout ou rien : ou bien on a raison (on classe/prédit bien) ou bien on a tort :

$$e_z(\widehat{y}, y) = \begin{cases} 0 \ si \ y = \widehat{y} \\ 1 \ si \ y \neq \widehat{y} \end{cases} \tag{6.9}$$

En choisissant la perte e_z, il est immédiat de montrer que le classifieur optimal devient :

$$Opt(x_0) = \omega_{i_0} = \underset{g \in \{\omega_1, \ldots, \omega_K\}}{\operatorname{argmax}} Pr(g | X = x_0)$$

Certains appellent ce classifieur, le classifieur Bayésien, car il maximise la probabilité a posteriori de la classe (c'est-à-dire sachant la mesure d'entrée).

Grâce au résultat précédent, $Opt(x) = \text{argmax}_g Pr(g|X = x)$, on peut aussi bâtir un classifieur aux k plus proches voisins en approchant empiriquement la probabilité $Pr(g|X = x)$. Il s'agit de prédire pour une entrée x la classe la plus probable ω_0 compte-tenu d'un ensemble d'apprentissage $D = \{x_i, y_i\}$ au sein duquel certains x_i sont voisins de x et majoritairement associés à la classe ω_0. Ce classifieur fait l'objet d'un exercice ci-après.

6.3 Exercices

6.3.1 Solution de l'EPE pour la régression avec une perte L_p

Dans le cours, nous avons utilisé une perte quadratique pour la formulation de l'EPE dans le cas de la régression. Nous changeons ici la perte en utilisant une perte L_p selon :

$$EPE(h, x) = \int |y - h(x)|^p p(y|x) dy \tag{6.10}$$

1. Montrer que la condition nécessaire pour minimiser l'EPE en x conduit à :

$$\int_{-\infty}^{h(x)} |y - h(x)|^{p-1} p(y|x) dy = \int_{h(x)}^{\infty} |y - h(x)|^{p-1} p(y|x) dy \tag{6.11}$$

2. Que devient cette condition si $p = 1$? Donner votre interprétation. On dit que le meilleur prédicteur est la médiane de y conditionnellement à x.

3. Dans le cours, nous avons déduit de la solution de l'EPE avec une perte quadratique une classe de prédicteurs non paramétriques aux k plus proches voisins. Rappeler la forme des prédicteurs «kppv» utilisant la «moyenne empirique» et leur relation avec la solution de l'EPE.

4. Déduire des questions 2 et 3, une classe de prédicteurs non paramétriques aux kppv utilisant la «médiane» et non plus la «moyenne». Justifier votre choix. Indication d'après le dictionnaire des mathématiques (éd. PUF) : médiane statistique d'un caractère quantitatif X prenant les valeurs $a_1, a_2, \ldots a_n$ sur un échantillon de n individus : réel m tel que le nombre des valeurs de X inférieures à m soit égal au nombre des valeurs supérieures à m. Si n est impair, la médiane est l'une des valeurs $a_1, a_2, \ldots a_n$. Par exemple, si $a < b < c$ alors $m =$médiane$\{a, b, c\} = b$. Si n est pair, m est un nombre choisi entre 2 valeurs de l'échantillon : par exemple, si $a < b < c < d$, alors on peut prendre $m =$médiane$\{a, b, c, d\} = (b + c)/2$.

5. Quelle différence y a-t-il entre un prédicteur à base de «moyenne» et un prédicteur à base de «médiane» si des données aberrantes contaminent des données d'apprentissage ? Pour répondre on peut se donner un exemple d'ensemble d'apprentissage $D = \{x_i, y_i\}_i$ construit classiquement en bruitant des sorties

fonctionnelles selon $y_i = f(x_i) + \epsilon_i$ mais tel qu'au voisinage d'un x', une donnée de D d'abscisse x_{i_0} a une ordonnée y_{i_0} arbitrairement grande (donc aberrante). Comparer alors deux prédicteurs en moyenne et en médiane reposant sur les 3 plus proches voisins de x'.

6.3.2 Classifieur binaire optimal

On considère un classifieur dont l'espace d'entrée est $\mathcal{X} = [0.0, 1.0] = [0, 1]$ (l'intervalle réel) et l'espace discret de sortie est $\mathcal{Y} = \{0, 1\}$ (cas binaire à deux classes). On suppose de plus que la densité pour l'espace d'entrée est $p(x) = 1$, $\forall x \in \mathcal{X}$ et $Pr(Y = 1|X = x) = x^2$, $\forall x \in \mathcal{X}$.

1. Grâce à un dessin dans le plan ($\mathcal{X}$, $\mathcal{Y}$) et sans aucun calcul, on vous demande d'abord de donner deux allures vraisemblables pour *deux* ensembles d'apprentissage $D_1 = \{x_i, y_i\}_{i=1...10}$ et $D_2 = \{x'_i, y'_i\}_{i=1...10}$. Chaque ensemble comportera dix exemples, il suggèrera bien la nature stochastique de la réalisation des échantillons *tout en respectant globalement les hypothèses sur les lois de probabilités* en entrée et en sortie. On vous demande de commenter vos propositions. Indication : les entrées réelles x_i (tout comme les x'_i) seront uniformément réparties sur $\mathcal{X}$. Les sorties binaires y_i (tout comme les y'_i) seront d'autant plus nombreuses à valoir 1 (resp. 0) que les entrées correspondantes seront proches de 1.0 (resp. 0.0).

2. On vous demande de rappeler ce qu'est un prédicteur au plus proche voisin dans ce cas de classification binaire. Grâce à un dessin, on vous demande de donner l'allure de ce prédicteur $1ppv_{D_1}(x)$ bâti à partir de votre ensemble D_1.

3. Discuter (sans calcul) la capacité de généralisation de votre prédicteur $1ppv_{D_1}(x)$ relativement à l'ensemble D_2.

4. On vous demande de rappeler ce qu'est la perte 0-1 (coût non informatif selon le cours) pour la classification.

5. On vous demande de rappeler *dans ce cas à deux classes et pour cette perte 0-1* qui nous intéressent, le passage de l'EPE à la règle de décision bayésienne via un risque conditionnel.

6. Expliquer ainsi que le classifieur optimal est donné par :

$$Opt(x) = \begin{cases} 1 & si \quad Pr(Y = 1|X = x) > \frac{1}{2} \\ 0 & sinon \end{cases} \tag{6.12}$$

7. Expliquer que cela revient à :

$$Opt(x) = \begin{cases} 1 & si \quad x > \frac{1}{\sqrt{2}} \\ 0 & si \quad x \leq \frac{1}{\sqrt{2}} \end{cases} \tag{6.13}$$

8. Montrer que l'EPE vaut $\frac{2-\sqrt{2}}{3}$

9. Dans le cadre général de l'apprentissage artificiel, on ne peut généralement pas calculer une valeur numérique pour l'EPE. Pourquoi pouvons-nous le faire ici dans notre cas particulier ? Que connaissons-nous ici qui est d'habitude inconnu ?

10. Pour un espace de sortie dénombrable $\mathcal{Y}$, la régression $r(x)$ est définie par

$$r(x) = E_{Y|X=x}[Y] = \sum_{y \in \mathcal{Y}} y Pr(Y = y|X = x) \tag{6.14}$$

Que vaut $r(x)$ ici ? Donner une interprétation.

11. Calculer le risque quadratique de la régression que nous venons de calculer. Le risque est défini par :

$$R_r = \int_{\mathcal{X}} \sum_{y \in \mathcal{Y}} (r(x) - y)^2 Pr(Y = y|X = x) p(x) dx \tag{6.15}$$

6.3.3 Classification aux kppv

Nous avons défini et utilisé dans le cours sur l'apprentissage supervisé, des prédicteurs aux k plus proches voisins (kppv) pour la régression. Nous n'avons pas défini formellement les prédicteurs équivalents pour le problème de classification. C'est l'objet de cet exercice.

1. Soit $D = \{x_i, y_i\}, i = 1..n$ un ensemble d'apprentissage avec $y_i \in \{\omega_1, \omega_2\}$ (classification à 2 classes).
   Soit $x_0 \in \mathbb{R}^p$ un nouvel élément de classe inconnue (x_0 est à classer).
   Soit d une distance permettant de comparer x_0 aux $x_i \in \mathbb{R}^p$.
   Définir formellement la règle de classification du plus proche voisin (1ppv ou kppv avec k=1).

2. Illustrer le fonctionnement de ce classifieur (1ppv) :

   - en dessinant un exemple bien choisi d'ensemble D tel que $x_i \in [0, 1]^2, \forall i, (p = 2)$ et tel que les y_i sont représentés par des $\times$ et des $\circ$ selon qu'ils valent ω_1 ou ω_2

   - en dessinant *l'allure* des régions de décision $\mathcal{R}_1$ et $\mathcal{R}_2$ pour chacune des deux classes

3. Montrer [1] avec votre exemple précédent, ou grâce à un nouvel exemple, qu'un élement $(x_{i_0}, y_{i_0} = \omega_2)$ de D *isolé* parmi de nombreux éléments appartenant à l'autre classe ω_1 conduit à une région de décision $\mathcal{R}_2$ non connexe (fragmentée).

4. Généraliser votre définition donnée en 1) pour définir formellement un prédicteur aux 3ppv (k=3) *en adoptant une règle de décision favorisant la classe "majoritaire" dans le voisinage*

[1] toujours avec des illustrations et aucun calcul

5. Illustrer le fonctionnement de ce nouveau prédicteur (en particulier sur l'exemple donné en 3.). Il s'agit de donner *l'allure* des régions et des frontières de décision de ce classifieur aux 3ppv.

6. Commenter la différence entre vos illustrations données en 3. et 5. en termes de complexités des modèles de décision sous-jacents.

7. Est-il facile selon votre réponse donnée en 4. de définir un prédicteur (classifieur) aux 4ppv (kppv avec k=4) ? Si oui, expliquez pourquoi, Si non, tentez de redéfinir un prédicteur 4ppv.

8. On va maintenant relier le classifieur 1ppv (k=1) au classifieur Bayésien.
   Soit un voisinage $\mathcal{V} \subset \mathbb{R}^p$ centré autour de $x_0 \in \mathbb{R}^p$
   Soit $p(x|\omega_j)$ la densité du vecteur x conditionnelle à son appartenance à la classe ω_j
   Expliquer ce qu'est la quantité :

$$P_{\mathcal{V}}^j = \int_{\mathcal{V}} p(x|\omega_j)dx$$

   Dire sous quelle condition on peut l'approcher selon :

$$P_{\mathcal{V}}^j \approx p(x_0|\omega_j) * Vol(\mathcal{V})$$

   où $Vol(\mathcal{V})$ est la longueur si $p = 1$, la surface si $p = 2$ ou, plus généralement, le "volume" du voisinage $\mathcal{V}$

9. Sachant qu'il y a n_j éléments parmi les n de D qui appartiennent à la classe ω_j
   Sachant que parmi ces n_j éléments seuls N_j appartiennent à $\mathcal{V}$
   Donner une estimation de la densité conditionnelle en fonction de $Vol(\mathcal{V}), N_j, n_j$:

$$\widehat{p(x_0|\omega_j)} = \frac{?}{?}$$

10. On revient à votre classifieur du plus proche voisin (1ppv) pour deux classes $j = 1, 2$
    On veut toujours classer x_0
    Définir le plus petit voisinage $\mathcal{V}_1$ (resp. $\mathcal{V}_2$) centré autour de x_0 qui contient le plus proche voisin de x_0 pris dans D et appartenant à la classe ω_1 (resp. ω_2).
    Reformuler la règle de décision du classifieur 1ppv en comparant les volumes de $\mathcal{V}_1$ et $\mathcal{V}_2$.

11. Rappeler la règle de décision (classification) Bayesienne simple (celle obtenue avec le coût non informatif $c_{ji} = 1 - \delta_{ij}$ et utilisée en TP).

12. Montrer finalement que cette dernière règle revient à celle du classifieur 1ppv (établie en 10.) grâce a l'estimation de la densité conditionnelle établie en 9. et à une estimation empirique simple des probabilités à priori $P(\omega_1)$ et $P(\omega_2)$ par des proportions dépendantes des n_j et de n.

6.3.4 Classification Bayésienne

On considère un ensemble d'apprentissage $D = \{(\mathbf{x_i}, y_i)\}, i = 1 \dots n$ où les y_i décrivent l'étiquetage supervisé par un expert des données caractéristiques $\mathbf{x_i} \in \mathbf{R}^2$ en deux classes $y_i = \omega_1$ ou $y_i = \omega_2$.

1. Dans cette question, on représentera graphiquement les données de D dans le plan 2D. Une donnée de D de la classe ω_1 sera représentée par un petit rond 'o' et une donnée de la classe ω_2 par une petite croix 'x'.

   Les données de ω_2 échantillonnent de manière dense tout le tube $\{\mathbf{x} \in \mathbf{R}^2 | 3.9 \leq \mathbf{x}^T\mathbf{x} \leq 4.1\}$. Les données de ω_1 échantillonnent de même le tube $\{\mathbf{x} \in \mathbf{R}^2 | 0.9 \leq \mathbf{x}^T\mathbf{x} \leq 1.1\}$. Grâce à ces indications, on vous demande de dessiner l'allure des données de D.

2. Les deux classes ainsi obtenues sont-elles linéairement séparables ? Donner l'équation d'une courbe permettant la séparation des deux classes.

3. On désire montrer qu'un classifieur Bayésien simple peut permettre de séparer correctement ces deux classes. Comme en Cours-TD-TP, on introduit les densités conditionnelles suivantes :

$$p(\mathbf{x}|\omega_i) = \frac{1}{(2\pi)^{d/2} \left|\Sigma_\mathbf{i}\right|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu_i})^T {\Sigma_\mathbf{i}}^{-1}(\mathbf{x} - \boldsymbol{\mu_i})\right] \tag{6.16}$$

   - que vaut d ici ?
   - Choisir des valeurs numériques pour $\boldsymbol{\mu_1}$ et $\boldsymbol{\mu_2}$. Justifier votre choix.
   - Choisir la forme de la matrice de covariance Σ_1 (resp. Σ_2) dépendant uniquement d'un scalaire σ_1^2 (resp. σ_2^2)). Justifier vos choix vis-à-vis (de l'allure) des données.

4. Soient les fonctions discriminantes suivantes :

$$\delta_i(\mathbf{x}) = -\frac{1}{2} log \left|\Sigma_\mathbf{i}\right| - \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu_i})^T {\Sigma_\mathbf{i}}^{-1}(\mathbf{x} - \boldsymbol{\mu_i}) + logP(\omega_i) \tag{6.17}$$

   - Rappeler le principe de la décision Bayésienne simple exprimée à partir de ces fonctions discriminantes.
   - Donner l'équation de la frontière de décision en fonction des $\delta_i(\mathbf{x})$.
   - On fait désormais l'hypothèse que les deux classes sont *a priori* équiprobables. Montrer qu'une équation de la frontière de décision pour notre problème est :

$$\mathbf{x}^T\mathbf{x} = \frac{d\sigma_1^2\sigma_2^2}{\sigma_2^2 - \sigma_1^2} log\left(\frac{\sigma_2^2}{\sigma_1^2}\right) \tag{6.18}$$

   - Choisir deux valeurs numériques pour σ_1 et σ_2 qui conduisent à une séparation correcte des données.

Thème 7

Compromis Biais-Variance

Vincent Charvillat
Janvier 2010

Dans ce thème, on aborde les conditions défavorables rencontrées en pratique lorsqu'on met en œuvre un mécanisme de prédiction. Par conditions défavorables, on entend une quantité limitée de données supervisées et la difficulté à sélectionner des prédicteurs de complexité satisfaisante. En décomposant l'EPE, c'est-à-dire le risque espéré, on montre clairement que des prédicteurs trop complexes fournissent des prédictions trop dépendantes des ensembles d'apprentissage considérés et, qu'inversement, des prédicteurs trop simples peuvent conduire à des erreurs systématiques de prédiction. Le compromis sous-jacent est appelé «compromis biais-variance». On peut établir un compromis acceptable en sélectionnant un modèle avec les méthodes empiriques vues dans le thème 5 (par validation croisée par exemple) ou en utilisant les mécanismes de contrôle de la complexité présentés dans ce thème : la sélection de modèle ou la régularisation.

7.1 Décomposition de l'EPE

7.1.1 Cadre

Comme au thème 4, on se place dans le cadre de la régression, vue comme un problème d'apprentissage supervisé. On cherche des prédicteurs dans une classe $\mathcal{H}$ fixée. Pour un ensemble d'apprentissage D et grâce à la minimisation d'un risque empirique, on peut donc apprendre un prédicteur $h_D \in \mathcal{H}$ que l'on évalue par :

$$EPE(h_D) = \int_{\mathcal{X}} \int_{\mathcal{Y}} (y - h_D(x))^2 P_{X,Y}(x,y) dxdy \tag{7.1}$$

La dépendance du prédicteur appris vis-à-vis de D (notée h_D) est centrale dans ce qui suit. En particulier, il est clair que le prédicteur n'est pas forcément le

meilleur atteignable dans $\mathcal{H}$. Comme D est de taille limitée, on a vraisemblablement $h_D \neq h^*_{\mathcal{H}}$ avec $h^*_{\mathcal{H}} = \underset{h \in \mathcal{H}}{\operatorname{argmin}}\, EPE(h)$. Nous allons étudier la sensibilité de h_D au choix de l'échantillon D ou, en d'autres termes, comment se comportent les prédictions lorsque D varie.

Pour avancer, on suppose l'existence d'une dépendance fonctionnelle de Y sur X selon :

$$Y = f(X) + \epsilon \tag{7.2}$$

où :

- f est une fonction réelle inconnue $f : \mathbb{R} \to \mathbb{R}$,
- ϵ est un bruit (variable aléatoire réelle) d'espérance nulle : $E(\epsilon) = 0$, indépendant de X,
- on suppose en outre $\mathrm{Var}(\epsilon) = \sigma^2$,
- Y est une variable aléatoire réelle (v.a comme fonction de v.a.).

7.1.2 Décompositions bruit-biais-variance

Nous allons décomposer l'EPE. Dans un premier temps, grâce à nos hypothèses, il est assez aisé de décomposer l'EPE sous la forme suivante :

$$EPE(h_D) = \sigma^2 + \int_{\mathcal{X}} (f(x) - h_D(x))^2 P_X(x) dx \tag{7.3}$$

On retrouve bien alors que $f(x) = E(Y|X = x)$ est la solution optimale de la minimisation de l'EPE (qui n'est pas forcément dans $\mathcal{H}$). La même décomposition opérée ponctuellement en x_0 conduit à :

$$EPE(x_0, h_D) = E_{Y|X}\left[(y - h_D(x))^2 | x = x_0\right] = \sigma^2 + (f(x_0) - h_D(x_0))^2 \tag{7.4}$$

Lorsque D varie, h_D varie. De même, ponctuellement, lorsque D varie, $h_D(x_0)$ varie autour de $E_D\left[h_D(x_0)\right]$. L'introduction de l'espérance des prédictions permet d'obtenir la décomposition bruit-biais-variance suivante :

$$EPE(x_0) = E_D\left[EPE(x_0, h_D)\right] = \sigma^2 + (f(x_0) - E_D\{h_D(x_0)\})^2 + Var_D\{h_D(x_0)\} \tag{7.5}$$

On écrit usuellement :

$$\text{risque espéré} = \text{bruit} + (\text{biais})^2 + \text{variance} \tag{7.6}$$

L'erreur de généralisation est décomposée en les trois termes suivants (voir aussi la figure 7.1) :

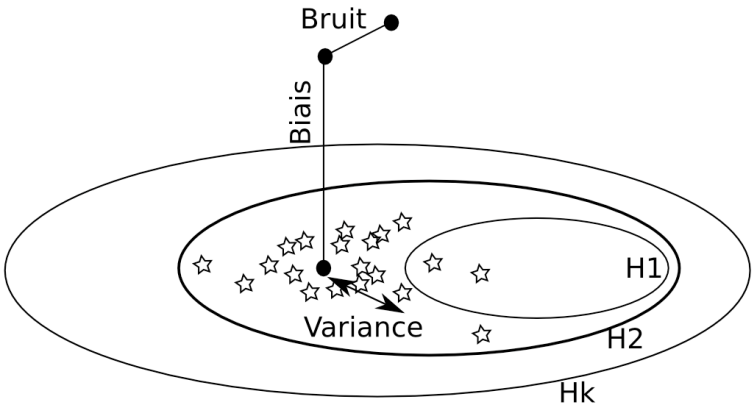


Figure 7.1: Illustration de la décomposition bruit-biais-variance.

- l'erreur due au fait que la classe de prédicteurs choisie ne contient pas nécessairement la solution optimale au problème de minimisation de l'EPE. On appelle cette partie de l'erreur le biais car l'espérance des prédictions, lorsque les ensembles d'apprentissage varient, peut ne pas être égale à la valeur optimale.

- le terme de variance qui mesure l'écart entre les prédictions issues de chaque ensemble d'apprentissage et l'espérance des prédictions. Il s'agit en d'autres termes de mesurer la sensibilité de la prédiction au choix d'un ensemble d'apprentissage particulier.

- le terme de bruit est quant à lui irréductible. Il s'agit de la variance du bruit additif qui contamine intrinsèquement les données de sortie. Ce bruit ne peut pas être prédit.

Le terme de bruit minore donc l'EPE : si la classe des prédicteurs choisie contenait la solution, si l'ensemble d'apprentissage était de taille illimitée (et si nos capacités de calcul l'étaient aussi) on pourrait espérer réduire l'EPE à ce seul terme et retrouver la solution optimale $f(x) = E[Y|x]$.

7.1.3 Illustration pratique

La décomposition précédente permet de bien comprendre les faibles capacités de généralisation de prédicteurs trop simples ou trop compliqués. Un prédicteur trop simple présente une faible variance mais la paye au prix d'un biais important. Inversement, un prédicteur trop compliqué conduira à un faible biais mais de fortes variances de prédiction. Les figures 7.2 et 7.3 illustrent ce phénomène. La solution $f(x)$ polynomiale de degré 6 au problème de régression est donnée par la courbe de Bézier en bleu. Deux ensembles d'apprentissage D_1 et D_2 sont utilisés dans chacune des figures avec, chaque fois, deux modèles de prédiction : l'un trop simple de degré 2 (< 6) et l'autre trop complexe de degré 11 (> 6). Pour des abscisses bien choisies, il est très facile de visualiser les termes en $(f(x_0) - E_D\{h_D(x_0)\})^2$ et $Var_D\{h_D(x_0)\}$.

La figure 7.1 suggère la même chose : elle montre des classes imbriquées de prédicteurs $\mathcal{H}_1 \subset \mathcal{H}_2 \cdots \subset \mathcal{H}_k$. La classe $\mathcal{H}_1$ des prédicteurs les plus simples implique un biais important. La taille plus importante de $\mathcal{H}_k$, regroupant des prédicteurs complexes, implique une variance significative lorsque l'ensemble d'apprentissage varie.

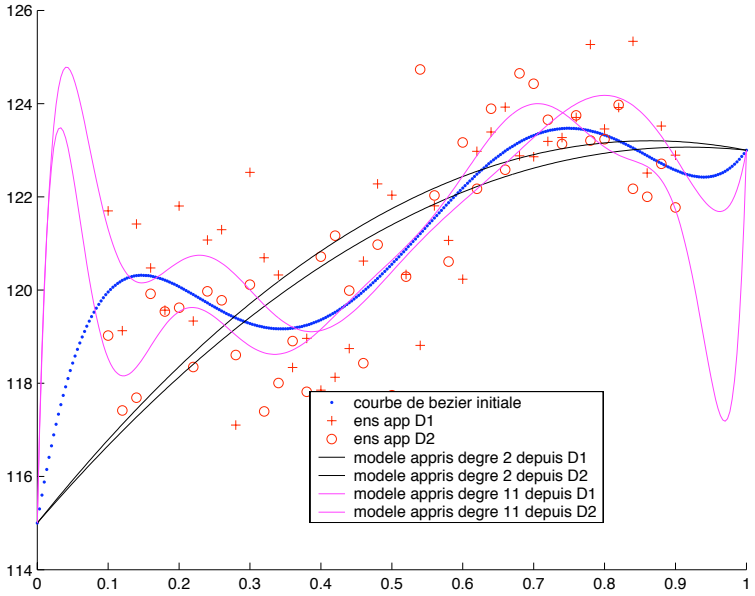


Figure 7.2: Illustration de la décomposition bruit-biais-variance.

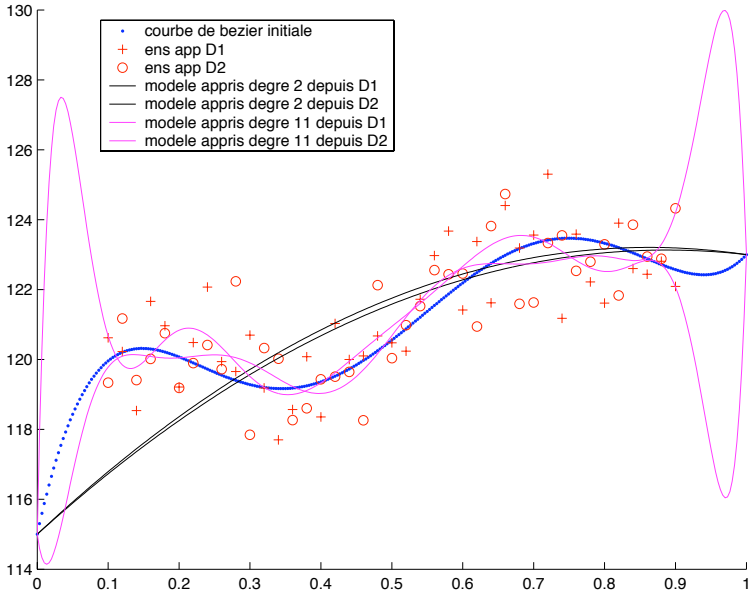


Figure 7.3: Illustration de la décomposition bruit-biais-variance.

7.1.4 Cas du prédicteur aux kppv

On veut décomposer l'EPE pour les prédicteurs aux kppv. On reprend les hypothèses (cf. section 7.1.1) qui nous ont conduit à la décomposition suivante :

$$EPE(x_0) = \sigma^2 + \left(f(x_0) - E_D\{h_D^{\text{kppv}}(x_0)\}\right)^2 + Var_D\left\{h_D^{\text{kppv}}(x_0)\right\} \tag{7.7}$$

Dans l'équation précédente h_D^{kppv} est le prédicteur aux k plus proches voisins déduit d'un ensemble d'apprentissage $D = \{(x_i, y_i)\}_{i=1\ldots n}$:

$$h_D^{\text{kppv}}(x_0) = \text{moyenne}\left(y_{\sigma(i)} | x_{\sigma(i)} \in V_k(x_0), (x_{\sigma(i)}, y_{\sigma(i)}) \in D\right) = \frac{1}{k}\sum_{i=1}^{k} y_{\sigma(i)} \tag{7.8}$$

On considère des ensembles d'apprentissage comme ceux montrés à la figure 4.1. Pour deux ensembles d'apprentissage $D_1 = \{(x_i, y_i)\}_{i=1\ldots n}$, $D_2 = \{(x_i, y_i')\}_{i=1\ldots n}$, les abscisses $\{x_i\}_{i=1\ldots n}$ sont constantes et seules les n réalisations indépendantes et identiquement distribuées des bruits additifs diffèrent : $y_i = f(x_i)+\epsilon_i$, $y_i' = f(x_i)+\epsilon_i'$. Lorsque D varie, seul le vecteur des bruits iid $E = (\epsilon_1, \ldots, \epsilon_i, \ldots, \epsilon_n)^T$ varie. On aura donc :

$$E_D\{.\} = E_X E_{E|X}\{.\} \to E_E\{.\} \tag{7.9}$$

On peut alors montrer pour $x_{\sigma(i)} \in V_k(x_0)$:

$$EPE(x_0) = \sigma^2 + \left(f(x_0) - \frac{1}{k}\sum_{i=1}^{k} f(x_\sigma(i))\right)^2 + \frac{\sigma^2}{k} \tag{7.10}$$

On en déduit que le paramètre k des «k plus proches voisins» permet de réaliser un compromis biais-variance en contrôlant la régularité du prédicteur.

7.2 Régularisation

Dans l'exemple précédent, le contrôle de complexité peut donc s'opérer via un (hyper)paramètre qui permet de régulariser le prédicteur en limitant sa variance au prix d'un biais potentiellement plus élevé.

Le mécanisme général sous-jacent est appelé *régularisation*. L'idée principale consiste à chercher un prédicteur dans une classe la plus générale possible mais de contraindre ces prédicteurs (potentiellement trop complexes) à être suffisamment réguliers pour éviter l'écueil du sur-apprentissage et l'excès de variance associé. En revenant au cas illustré par la figure 7.3, il s'agirait d'utiliser les prédicteurs polynomiaux les plus complexes (par exemple de degré 11) pour modéliser les données disponibles tout en contraignant ces modèles à être suffisamment réguliers pour éviter les phénomènes d'«oscillations» excessives du prédicteur lorsqu'il cherche à interpoler les données d'apprentissage.

Nous présentons trois approches classiques de la régularisation dans ce paragraphe :

- les splines de lissage,
- l'introduction de connaissances a priori via l'estimation MAP,
- la *ridge regression*, que nous pourrions traduire par «régression écrêtée».

7.2.1 Splines de lissage

Nous nous intéressons ici à la régression à partir d'un ensemble d'apprentissage $D = \{(x_i, y_i)\}, i = 1 \dots n$ tel que $\forall i \ (x_i, y_i) \in \mathbb{R} \times \mathbb{R}$. Nous considérons des prédicteurs généraux qui sont les fonctions de classe C_2 (deux fois continûment dérivables). On cherche alors à apprendre le meilleur prédicteur h^* en minimisant pour $h \in C_2$ une somme pénalisée de résidus aux moindres carrés [1]:

$$PRSS_\lambda(h) = \sum_{i=1}^{n} (y_i - h(x_i))^2 + \lambda \int \{h''(t)\}^2 dt \tag{7.11}$$

On dit que l'(hyper)paramètre λ est un paramètre de lissage à bien choisir. Le premier terme du critère $PRSS$ est un terme d'attache aux données d'apprentissage. Le second terme pénalise des courbures excessives pour le prédicteur qui est ainsi régularisé[2]. L'hyperparamètre λ établit un compromis entre les deux termes. On a en particulier :

- $\lambda = 0$ revient à accepter que le prédicteur interpole les données de D,
- $\lambda = \infty$ revient à chercher une droite de régression (aucune courbure n'est tolérée).

Entre ces deux situations extrêmes allant du modèle affine le plus simple au modèle d'interpolation le plus compliqué, on espère trouver le paramètre λ qui revient à travailler dans la classe de fonctions de complexité optimale. Cette optimalité est à comprendre au sens de l'apprentissage, elle consiste à trouver le meilleur λ^* associé à la capacité de généralisation la plus grande. En pratique, il est possible de trouver ce λ^* en utilisant des techniques vues précédemment comme la validation croisée. On parle aussi de «degré de liberté effectif» pour λ car il joue le rôle d'un nombre «continu» de degrés de liberté.

La solution h^* au problème de minimisation du critère précédent est connue sous le nom de «spline cubique naturelle». Le développement de la preuve est hors sujet ici. Disons simplement qu'une telle «spline naturelle» se met finalement sous la forme simple suivante :

$$h^*(x) = \sum_{j=1}^{n} \beta_j N_j(x) \tag{7.12}$$

Sous cette forme, les $N_j(x)$ forment pour $j \in 1 \dots n$ une base de fonctions pour représenter cette famille de splines. Le critère (équation 7.11) se ramène alors au cas (linéaire) de la *ridge regression* discuté ci-après.

[1] Le critère $PRSS$ est mis pour *Penalized Residual Sum of Squares*.

[2] En toute rigueur un espace de Sobolev doit être considéré avec ce terme.

7.2.2 Régularisation via l'estimateur MAP

Dans le thème 4 consacré aux rappels sur la régression, nous avons redit les relations existant entre la minimisation du risque empirique via les moindres carrés ordinaires et l'estimation au sens du Maximum de Vraisemblance (MV). Dans ce contexte, l'apprentissage d'un prédicteur paramétrique $x \rightarrow h(\beta_D, x) = h_{\beta_D}(x)$ pris dans une classe $\mathcal{H}$ revient à trouver les paramètres β_D^{MV} qui expliquent au mieux un ensemble d'apprentissage $D = \{(x_i, y_i)\}, i = 1 \dots n$ selon :

$$\beta_D^{MV} = \underset{\beta}{\text{argmax}}\ L(D|\beta) = p(D|\beta) \Leftrightarrow \beta_D^{MV} = \underset{\beta}{\text{argmin}}\ R_D(h_\beta) \tag{7.13}$$

Equations pour lesquelles $L(D|\beta)$ ou $p(D|\beta)$ représentent la vraisemblance des données d'apprentissage D sachant β. On a montré qu'une maximisation de L (ou de la «log-vraisemblance» lnL) revient à minimiser le risque empirique R_D. On peut reconnaître aisément R_D dans l'équation 4.8 (critère aux moindres carrés noté $R_0(\beta)^2$). Il faut noter aussi que l'échelle σ^2 du bruit est ici supposée connue.

L'estimateur du Maximum A Posteriori (MAP) de β consiste à adopter une posture Bayésienne en introduisant une connaissance a priori sur β ou plus précisèment sur la *loi de* β. C'est cette connaissance qui va être l'élément régularisant recherché. Le paramètre β est donc considéré comme une variable aléatoire dont la loi a priori est donnée et notée $L(\beta)$. La démarche Bayésienne consiste finalement, pour l'estimateur MAP, à maximiser la loi a posteriori de β sachant D selon

$$\beta_D^{MAP} = \underset{\beta}{\text{argmax}}\ L(\beta|D) \propto L(D|\beta)L(\beta) \tag{7.14}$$

Un exemple simple, convaincra aisément le lecteur que le critère $L(D|\beta)L(\beta)$ conduit comme précédemment à pénaliser le terme d'attache aux données (la vraisemblance $L(D|\beta)$) avec un terme $L(\beta)$ qui pénalise un écart potentiel entre les paramètres appris et ceux attendus. L'exemple le plus simple est celui de la droite de régression pour laquelle on considère un ensemble d'apprentissage $D = \{(x_i, y_i)\}, i = 1 \dots n$ bâti selon :

$$y_i = a^* x_i + b^* + \epsilon_i \qquad \epsilon_i \sim \mathcal{N}(0, \sigma^2) \quad \text{iid} \tag{7.15}$$

Les paramètres du modèle exact $\beta^* = (a^*, b^*)^T$ étant inconnus, on veut les estimer en introduisant des connaissances a priori sur la loi de $\beta = (a, b)^T$ vu comme un vecteur aléatoire. On peut par exemple supposer que :

- la pente a suit une loi $\mathcal{N}(0, \sigma_a^2)$. Ce qui indique que la droite de régression recherchée est plutôt horizontale ou, plus précisément, de pente statistiquement d'autant plus faible que la variance σ_a^2 est petite,
- l'ordonnée à l'origine b est uniformément répartie dans l'intervalle $[0, 1]$: $b \sim \mathcal{U}[0, 1]$.

Sous ces hypothèses, on peut immédiatement vérifier que :

$$L(\beta|D) \propto \exp -\frac{1}{2}\left(\frac{1}{\sigma^2}\sum_{i=1}^{n}(y_i - ax_i - b)^2 + \frac{1}{\sigma_a^2}a^2\right) \tag{7.16}$$

On retrouve exactement la même idée que précédemment selon laquelle l'hyper paramètre $\frac{1}{\sigma_a^2}$ pondère un terme aux moindres carrés pour l'attache aux données et un terme de régularisation (en a^2) qui limite ici l'amplitude de la pente. Plus la variance σ_a^2 est faible, plus le poids du terme régularisant est fort dans le compromis recherché. Cette modélisation conduit à l'estimation MAP suivante pour la pente :

$$\widehat{a}_{MAP} = \frac{\frac{1}{n}\sum_i x_i y_i - \bar{x}\bar{y}}{\overline{x^2} - \bar{x}^2 + \frac{\sigma^2}{n\sigma_a^2}} \tag{7.17}$$

Ce qui nous permet de vérifier que l'effet de régularisation attendu est bien présent : $\sigma_a^2 \to 0 \Rightarrow \widehat{a}_{MAP} \to 0$. Cet exemple souligne la généralité d'une méthode Bayésienne de régularisation via l'introduction de connaissances/lois a priori. En pratique, on s'éfforce simplement d'introduire des «lois a priori conjuguées» pour que les modèles régularisés conduisent, autant que possible, à des solutions analytiques.

7.2.3 Ridge regression

L'approche de la régularisation par *ridge regression* est une généralisation de l'exemple précédent pour lequel on limitait l'amplitude d'un paramètre à apprendre (la pente a de la droite de régression). En *ridge regression*, on souhaite écrêter ou «rétrécir» un ou plusieurs paramètres[3] pour régulariser le prédicteur h associé. On pénalise donc de trop fortes amplitudes pour le ou les paramètres concernés selon :

$$\widehat{\beta}_{\text{ridge}} = \underset{\beta=(\beta_1 \ldots \beta_p)^T \in \mathbb{R}^p}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - h(\beta, x_i))^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \tag{7.18}$$

Les notations sont les mêmes que précédemment pour les données (les x_i, y_i de D). L'hyperparamètre ($\lambda \geq 0$) contrôle ici aussi la complexité du prédicteur (pour lequel le choix d'une paramétrisation compatible avec cette approche est un problème clé). Plus λ est grand plus le retrécissement des composantes β_j s'imposera vis-à-vis du terme d'attache aux données d'apprentissage. De manière équivalente, on peut aussi présenter la *ridge regression* comme la solution du problème d'optimisation avec contrainte suivant :

$$\begin{aligned} \widehat{\beta}_{\text{ridge}} = \underset{\beta=(\beta_1 \ldots \beta_p)^T \in \mathbb{R}^p}{\operatorname{argmin}} & \textstyle\sum_{i=1}^{n} (y_i - h(\beta, x_i))^2 \\ \text{s.c.} & \textstyle\sum_{j=1}^{p} \beta_j^2 \leq s \end{aligned} \tag{7.19}$$

On considère de manière privilégiée le cas d'un prédicteur linéaire (ou affine) en β pour lequel on peut reformuler vectoriellement le critère régularisé. Pour $h(\beta, x) = C(x)^T\beta$ on a trivialement un vecteur Y et une matrice X similaires à ceux utilisés dans le modèle de régression linéaire simple qui ramènent le critère de la *ridge regression* à l'expression suivante :

$$PRSS_\lambda(\beta) = (Y - X\beta)^T(Y - X\beta) + \lambda\beta^T\beta \tag{7.20}$$

Il est alors aisé de montrer que :

[3] On parle donc aussi en anglais de *shrinkage method*.

$$\widehat{\beta}_{\text{ridge}} = \underset{\beta=(\beta_1 \ldots \beta_p)^T \in \mathbb{R}^p}{\operatorname{argmin}} PRSS_\lambda(\beta) \Rightarrow \widehat{\beta}_{\text{ridge}} = (X^T X + \lambda \text{Id}_{p\times p})^{-1} X^T Y \qquad (7.21)$$

On peut naturellement donner de nombreuses interprétations de cette dernière équation par comparaison avec celle de la pseudo-inverse (équation 4.18). Une des plus simples est d'observer que la matrice de lissage S_λ remplace la matrice chapeau $H = X(X^T X)^{-1} X^T$ selon :

$$\widehat{Y} = X(X^T X + \lambda \text{Id}_{p\times p})^{-1} X^T Y = S_\lambda Y \qquad (7.22)$$

Là où la trace de H était égale au nombre p de paramètres, les degrés effectifs de liberté (valeurs continues) sont désormais définis par :

$$df(\lambda) = Trace(S_\lambda) \qquad (7.23)$$

Le nombre effectif de degrés de liberté est donc d'autant plus faible que λ est grand. Il tend vers 0 si $\lambda \to \infty$ et, inversement, $df(\lambda) \to Trace(H) = p$ si $\lambda \to 0$ (absence de régularisation).

Nous dirons pour conclure ce paragraphe sur la régularisation par *ridge regression*, que des pénalisations de l'amplitude des paramètres β_j peuvent aussi s'entendre au sens de la norme L_1 (et non au sens de L_2 comme présenté ici). La pénalité dite de «Lasso» s'exprime simplement par $\sum_{j=1}^{p} |\beta_j| \leq t$. C'est une bonne manière, si t diminue, de forcer certains des β_j à s'annuler est de sélectionner les composantes grâce auxquelles on veut prédire !

7.3 Sélection de modèle

Une approche duale de la régularisation pour réaliser un compromis biais-variance consiste à pénaliser les modèles de prédiction trop complexes. L'idée force est d'observer que l'erreur d'apprentissage sous-estime le risque espéré (l'EPE) pour un prédicteur trop complexe. On dit que l'écart (le biais) entre l'EPE et l'erreur d'apprentissage est un terme d'optimisme[4]. Pour un prédicteur h, ce terme d'opti misme pourrait être défini comme :

$$\texttt{op} = EPE(h) - \texttt{err}(h) \qquad (7.24)$$

Ce terme d'optimisme est d'autant plus grand que le modèle est complexe. Le principe de la sélection de modèle consiste à utiliser une estimation $\widehat{\texttt{op}_p}$ de ce terme d'optimisme pour un prédicteur de complexité p (p est typiquement le nombre de paramètres ou ddl considéré). Cette estimation permet alors de pénaliser un terme d'attache aux données d'apprentissage $D = \{(x_i, y_i)\}, i = 1 \ldots n$. On apprend alors en minimisant un critère de sélection de modèle C_{sm} pour une classe de prédicteurs h_p de complexité p :

[4] On parle d'optimisme puisque la capacité de généralisation est surévaluée en utilisant les données ayant servi à l'apprentissage comme données de test.

$$C_{sm}(h_p) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \underbrace{h_p(x_i)}_{\widehat{y_i}})^2 + \widehat{\mathtt{op}_p} \tag{7.25}$$

7.3.1 Critère AIC d'Akaïke

Dans le cadre de la régression avec modèle gaussien et utilisation du risque quadratique, le critère d'information d'Akaïke (critère AIC mis pour *Akaïke Information Criterion*) coïncide avec le premier critère de pénalisation apparu dans la litérature (critère C_p de Mallows). Le critère AIC prend la forme suivante :

$$AIC_p = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2 + 2\frac{p}{n}\widehat{\sigma^2} \tag{7.26}$$

Dans ce critère, $\widehat{\sigma^2}$ est une estimation de la variance de l'erreur par un modèle de faible biais. La qualité de cette estimation est un préalable fondamental pour que la sélection du bon modèle s'opère. Mais il s'agit d'un problème de poules et d'œufs...Si le modèle le plus général n'est pas le vrai modèle (ou du moins si son biais n'est pas limité), l'estimation correcte de la variance nécessite de connaître la complexité optimale qui, à son tour, ne peut être identifiée par AIC sans connaître la variance...

Plusieurs justifications théoriques d'AIC existent. Nous présentons ci-après celle de Hastie qui définit le terme d'optimisme de l'équation 7.25 comme l'écart entre une approximation de l'EPE et une espérance (comparable) de l'erreur d'apprentissage. En moyenne, lorsque les sorties $Y = (y_1, \dots y_i \dots y_n)^T$ varient, l'erreur d'apprentissage moyenne est :

$$E_Y\left[\mathtt{err}(\widehat{h})\right] = E_Y\left[\frac{1}{n}\sum_{i=1}^{n} e(y_i, \widehat{h}(x_i))\right] \tag{7.27}$$

Pour approcher raisonnablement l'EPE, on peut évaluer la capacité de généralisation d'un prédicteur en remplaçant les sorties utilisées pour l'apprentissage, par de nouvelles sorties notées Y^{new}. L'écart moyen de prédiction pour la i^{ieme} donnée s'exprime alors en $E_{Y^{new}}\left[e(y_i^{new}, \widehat{h}(x_i))\right]$. Ce qui nous conduit à comparer l'espérance donnée par l'équation 7.27 avec :

$$\mathrm{Err}_{in} = E_Y\left[\frac{1}{n}\sum_{i=1}^{n} E_{Y^{new}}\left[e(y_i^{new}, \widehat{h}(x_i))\right]\right] \tag{7.28}$$

Le terme d'optimisme est alors défini par :

$$\widehat{\mathtt{op}_p} = \mathrm{Err}_{in} - E_Y\left[\mathtt{err}(\widehat{h})\right] \tag{7.29}$$

On montre pour la perte quadratique que le terme d'optimisme ainsi défini revient à :

$$\widehat{\mathtt{op}_p} = \frac{2}{n}\sum_{i=1}^{n} Cov(\widehat{y_i}, y_i) \tag{7.30}$$

Ainsi, l'optimisme de l'erreur d'apprentissage dépend de combien une prédiction $\widehat{y_i}$ est dépendante de la sortie y_i (ou, autrement dit, de l'influence de la sortie y_i sur sa propre prédiction $\widehat{y_i}$). Il devient clair alors que l'optimisme est d'autant plus fort que le modèle est complexe puisqu'il possède alors suffisamment de degrés de libertés pour s'ajuster à chaque sortie y_i. Dans le cas inverse, un modèle plus simple est trop rigide pour subir l'influence de chaque sortie y_i.

Il reste à montrer qu'avec un modèle de régression linéaire simple et des bruits iid additifs gaussiens centrés de variance σ^2 sur les sorties, on a :

$$\sum_{i=1}^{n} Cov(\widehat{y_i}, y_i) = Trace(H)\sigma^2 = p\sigma^2 \tag{7.31}$$

Ce qui justifie entièrement le critère de l'équation (7.26).

7.3.2 Autres Critères

Il existe de nombreux autres critères de sélection de modèles issus de différentes développements théoriques (ou approximations). Une version affinée de AIC qui est dédiée au cas gaussien et qui est adaptée aux petits échantillons est connue sous le nom d'AIC corrigée ou AIC_c :

$$AIC_{c_p} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2 + \frac{n+p}{n-p-2}\widehat{\sigma^2} \tag{7.32}$$

Une démarche Bayésienne conduit quant à elle au critère BIC (*Bayesian Information Criterion*) qui cherche asymptotiquement le modèle associé à la plus grande probabilité a posteriori. Il revient dans notre cadre à :

$$BIC_p = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2 + log(n)\frac{p}{n}\widehat{\sigma^2} \tag{7.33}$$

Dès que n est suffisamment grand, BIC tend à pénaliser les modèles complexes plus fortement que AIC. On peut montrer théoriquement que la probabilité de choisir, via BIC, le bon modèle tend vers 1 avec n tendant vers l'infini. Ce n'est pas vrai pour AIC qui tend plutôt à choisir des modèles trop complexes. Néanmoins, pour de petites tailles d'échantillon, BIC risque de se limiter à des modèles trop simples. En pratique, une comparaison empirique entre les différentes pénalisations disponibles est à opérer avant le choix du meilleur mécanisme de contrôle de la complexité pour un problème donné.

7.4 Exercices

7.4.1 Régression aux moindres carrés et Apprentissage supervisé

Dans cet exercice, on se place dans le cas de l'apprentissage supervisé appliqué à la régression. On se donne un ensemble d'apprentissage $D = \{(x_i, y_i)\}, i = 1 \dots m$. Les x_i sont des vecteurs de $\mathbf{R}^n$. On considère le sous-espace V engendré par ces vecteurs : $V = \{\sum_{i=1}^{m} \alpha_i x_i | \alpha_i \in \mathbf{R}\}$

On apprend un modèle prédictif linéaire en minimisant le risque suivant (où $C > 0$) :

$$R(\omega) = \frac{1}{2}\sum_{i=1}^{m}(x_i^T\omega - y_i)^2 + \frac{C}{2}\|\omega\|_2^2 \tag{7.34}$$

1. Comment appelle-t-on ce type de régression ? Comment l'a-t-on utilisé en cours/TP pour superviser la complexité de modèles appris ?

2. Montrer que la solution $\omega^* = argmin_\omega R(\omega)$ est dans V. indication :
   On ne résoudra pas explicitement le problème d'opti mais on se contentera de prendre un ω arbitraire dans V de lui rajouter une composante orthogonale à tous les x_i et d'observer le comportement de $R(\omega)$.

3. Donner les matrices ou vecteurs X, F, y, g permettant de mettre le critère $R(\omega)$ précédent sous la forme suivante :

$$RSS(\omega) = \frac{1}{2}\|y - X\omega\|_2^2 + \frac{C}{2}\|g - F\omega\|_2^2 \tag{7.35}$$

4. Calculer pour des X, F, y, g quelconques

$$\frac{\partial RSS}{\partial \omega}(\omega_o) = ? \tag{7.36}$$

5. Caractériser la solution $\hat{\omega} = argmin_\omega RSS(\omega)$ à partir de la condition nécessaire d'ordre 1.

6. Retrouver ainsi le résultat du cours donné pour un critère similaire à $R(\omega)$.

7.4.2 Décomposition Biais-Variance pour un modèle linéaire

Dans le cours consacré à la décomposition de l'EPE pour la régression, nous avons montré que la décomposition biais-variance en x_0 lorsque l'ensemble d'apprentissage D varie (et qu'en conséquence la prédiction $h_D(x_0)$ varie également) peut s'écrire :

$$EPE(x_0) = \sigma^2 + (f(x_0) - E_D\{h_D(x_0)\})^2 + Var_D\{h_D(x_0)\} \tag{7.37}$$

1. On vous demande de rappeler les hypothèses effectuées pour arriver à ce résultat.

2. On suppose que les données disponibles (par exemple $D = \{x_i, y_i\}_{i=1...n}$) s'expliquent par un modèle fonctionnel linéaire. En d'autres mots, il existe un vecteur $\beta^* \in \mathbb{R}^p$ qui permet d'expliquer les relations entre les entrées ($x_i \in \mathbb{R}^p$) et les sorties ($y_i \in \mathbb{R}$) selon : $y_i = x_i^T \beta^* + \epsilon_i$. Les hypothèses sur les **bruits iid additifs** ϵ_i sont les mêmes que celles rappelées à la question précédente. Si on place les y_i (resp. x_i, ϵ_i) dans un vecteur Y (resp. une matrice X, un vecteur aléatoire E) on obtient une relation vectorielle entre les données de D : $Y = X\beta^* + E$. On vous demande de rappeler (sans démonstration) la loi de E et celle de $\widehat{\beta}_D = (X^T X)^{-1} X^T Y = \beta^* + (X^T X)^{-1} X^T E$ en précisant bien les dimensions de X, Y, E.

3. On veut expliciter pour le modèle linéaire de la question 2, la décomposition donnée par l'équation 7.37. Pour établir le lien avec les notations de l'équation 7.37, on exprimera la prédiction $h_D(x_0)$ en fonction de x_0 et $\widehat{\beta}_D$. On vous demande alors de montrer que :

$$E_D(h_D(x_0)) = E_X E_{E|X}(h_D(x_0)) = x_0^T \beta^* \tag{7.38}$$

et

$$Var_D(h_D(x_0)) = \sigma^2 E_X \left(x_0^T (X^T X)^{-1} x_0 \right) \tag{7.39}$$

et d'en déduire une décomposition de $EPE(x_0)$.

En moyennant ce résultat ponctuel sur la loi des vecteurs aléatoires d'entrée («les x») on obtient donc :

$$EPE = E_x EPE(x) = \sigma^2 + \sigma^2 \int_x E_X \left(x^T (X^T X)^{-1} x \right) p(x) dx \tag{7.40}$$

4. Si n est grand (les données contenues dans X permettent d'approcher empiriquement les moments de la loi des x) et si $E(x) = 0$, montrer qu'une estimation raisonnable de $Cov(x)$ en fonction de X et n conduit à :

$$EPE \approx \sigma^2 + \frac{\sigma^2}{n} \int_x \left(x^T Cov(x)^{-1} x \right) p(x) dx \tag{7.41}$$

5. Le terme sous l'intégrale précédente se simplifie si on utilise la propriété algébrique suivante : pour une matrice carrée A et des vecteurs u, v de bonne taille on a : $u^T A v = Trace(A.v.u^T)$. On vous demande donc de montrer que :

$$EPE \approx \sigma^2 (1 + \frac{p}{n}) \tag{7.42}$$

6. Dans un exercice précédent (section 4.6) dédié aussi au modèle de régression simple linéaire et gaussien, une décomposition de l'erreur d'apprentissage (risque empirique) notée `err` a conduit quant à elle à

$$\mathrm{E}_D(\texttt{err}) = \sigma^2 (1 - \frac{p}{n})$$

On vous demande d'interpréter le plus attentivement possible ces deux derniers résultats. On pourra aussi justifier l'allure des courbes de généralisation obtenues en TP (où nous avons utilisé aussi un modèle linéaire avec p lié au degré des courbes). On pourra interpréter ce qui se passe lorsque $n \to \infty$ etc.

Thème 8

Apprentissage non-supervisé

Vincent Charvillat
Février 2010

Dans le cas non-supervisé, on ne dispose plus d'«oracle» ou d'expert pour guider l'apprentissage selon ses «instructions». Seule une collection de données brutes ou de mesures $D = \{x_i\}_{i=1...n}$ est disponible. Le schéma fonctionnel se réduit à :

$$x \in \mathcal{X} \rightarrow \boxed{\quad h\ ? \quad}$$

La prédiction de similarités présentes au sein des données ou la découverte d'autres structures descriptives sont les objectifs principaux des méthodes d'apprentissage non-supervisé. Il s'agit, pour résumer, d'identifier tout ou partie des éléments expliquant la véritable distribution P_X ayant généré D. Ces éléments sont notés h de manière abstraite dans la figure ci-dessus. On liste ci-après les types de problèmes concernés et on détaille deux techniques classiques, l'une de coalescence (*clustering* en anglais) et l'autre d'estimation de densité.

8.1 Problèmes usuels

On introduit brièvement dans ce paragraphe quelques problèmes usuels rencontrés en apprentissage non-supervisé.

8.1.1 Estimation de densité

Dans ce cas, l'élement h recherché est tout simplement la fonction densité de probabilité associée à la distribution ayant conduit à l'ensemble d'apprentissage D. Afin de résoudre ce problème, une approche possible pour un algorithme d'apprentissage consiste à spécifier une forme paramétrique précise de densité puis à ajuster les paramètres β pour maximiser la vraisemblance $L(D|\beta)$. Grâce à un tel apprentissage, une donnée manquante au sein d'un échantillon peut être prédite par une

valeur vraisemblable. Un modèle de densité peut aussi être utilisé afin d'identifier (voire d'éliminer) une entrée corrompue par un bruit aberrant. Il faut toutefois s'assurer que la densité apprise n'a pas elle-même était contaminée par des données aberrantes. Finalement, à partir d'une estimation de densité, il est normalement possible de générer de nouvelles données selon cette densité. Cela peut être utile dans certaines applications, par exemple en simulation, ou simplement pour mieux comprendre ce qui a été appris par le modèle. Le paragraphe 8.3 suivant présente une approche possible pour l'estimation d'un mélange de gaussiennes.

8.1.2 Réduction de dimension

Des techniques de prétraitement visant la réduction de la taille de x ont été présentée dans le thème 3. L'Analyse en Composantes Principales est, par essence, une technique de réduction de dimension non-supervisée. Dans ce cadre, l'élément recherché $h(x)$ est une représentation vectorielle de x plus compacte mais conservant l'essentiel de l'information initialement contenue dans x. Idéalement, une telle représentation devrait en outre permettre de mesurer efficacement des similarités éventuelles entre les données d'apprentissage (ou d'identifier des regroupements).
Via la décomposition spectrale de la matrice de covariance des données de D, l'ACP permet de bien comprendre l'importance des corrélations lorsque l'on souhaite définir des distances (ou mesures de dissemblance, dissimilarité) entre données de D. La distance de Mahalanobis (cf. figure 8.1) est l'outil privilégié pour la prise en compte des corrélations et des différences d'échelle entre les composantes des données d'apprentissage.

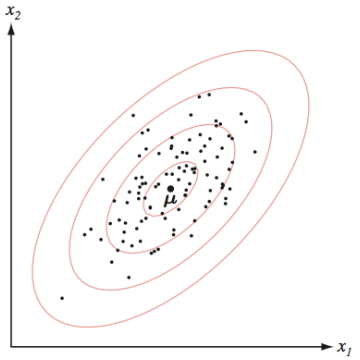


Figure 8.1: Isovaleurs (elliptiques) de la distance de Mahalanobis.

Au-delà de l'ACP, il existe de nombreuses autres techniques non-supervisées pour réduire la dimension des données d'apprentissage. Parmi elles, nous pouvons citer, les techniques de *Multidimensional Scaling* (MDS), *Locally Linear Embedding* (LLE) ou encore les *Isomap*. Ces algorithmes supposent normalement que les données de D se concentre au voisinage d'un sous-espace, de plus petite dimension que l'espace original $\mathcal{X}$. La dimension correcte du sous-espace en question peut être fixée par l'utilisateur ou recherchée par une technique non-supervisée.

8.1.3 Extraction de caractéristiques

Un mécanisme visant à extraire les caractéristiques $h(x)$ de x (les *features* en anglais) à partir desquelles on peut aisément décrire la structure des données de D, identifier des similarités, ou améliorer les performances d'un mécanisme décisionnel ultérieur peut être considéré comme une technique d'apprentissage non-supervisée.

Dans ce contexte, $h(x)$ est, comme pour l'ACP, une représentation alternative de x qui possède de bonnes propriétés. Souvent, la qualité de l'extraction de caractéristiques est mesurée vis-à-vis de l'amélioration des performances (...) d'un mécanisme de prédiction qui opère avec $h(x)$ (et non plus x) en entrée. Il existe plusieurs approches pour l'extraction de caractéristiques. Une technique de prétraitement simple consiste à simplement sélectionner les composantes de x les plus utiles [1]. Cette approche dite de sélection de caractéristiques peut être étendue. Un large ensemble de caractéristiques peut en effet être généré en considérant comme caractéristique candidate le résultat de la multiplication d'un sous-ensemble des composantes de x. On peut alors appliquer un algorithme de sélection de caractéristiques sur toutes les caractéristiques candidates. On retrouve ici des idées déjà abordées dans le thème introductif où des classifieurs quadratiques opèrent conjointement sur les composantes de x et leurs produits (équation 1.2).

Il est enfin possible de formuler le problème d'extraction de caractéristiques à partir d'un problème d'estimation de densité. En choisissant une fonction de densité utilisant un vecteur de variables latentes $h_1, h_2, \ldots h_H$ il est possible d'utiliser (les modes) des distributions conditionnelles $p(h_i|x)$ des variables latentes comme caractéristiques. C'est en particulier le cas pour les représentations creuses de x (*sparse coding*). Dans ce cas, c'est la modélisation de la densité qui conditionne l'extraction de caractéristiques.

8.1.4 Classification non-supervisée

La classification non-supervisée ou *clustering* est l'application typique et majeure de l'apprentissage non-supervisé. On cherche une partition de l'ensemble d'apprentissage D (voire de l'espace des données $\mathcal{X}$) en K classes regroupant des données similaires. A chaque classe peut éventuellement être associée une donnée «moyenne» (vue comme la représentante de la classe) qui résume bien les données de la classe. A chaque donnée x est donc associée une classe prédite $h(x)$ prise parmi les K possibles.

La figure 8.2 suivante illustre un résultat de *clustering* pour des données bidimensionnelles, avec $K = 3$ et des représentants centrés sur les trois *clusters* détectés par l'algorithme nommé *Mean Shift*. Ceci est possible en optimisant un **critère** visant à regrouper des données dans des classes, chacune la plus homogène possible et, entre elles, les plus distinctes possible. Par opposition à la classification supervisée aucun étiquetage de données préalable ne nous guide ici. Le coût généralement élevé de la supervision[2] est réduit à rien. C'est un premier avantage. L'approche

[1] Nous avons abordé ce point en évoquant la régularisation «Lasso» dans le thème précédent.

[2] Le coût de l'expertise en mode supervisé est d'autant plus grand que la base d'apprentissage est grande.

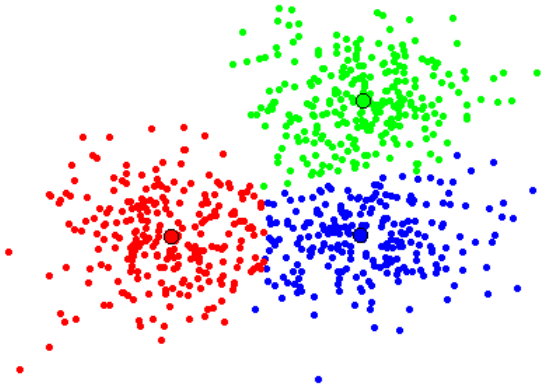


Figure 8.2: Classification non-supervisée.

non-supervisée peut aussi être utile si les classes évoluent avec le temps (par exemple si une nouvelle classe apparaît !). C'est un second avantage.

La combinatoire indique que le nombre de partitions de D (contenant n éléments) en K classes est :

$$P(n,K) = \frac{1}{K!}\sum_{i=0}^{K} C_K^i i^n (-1)^{K-i} \quad K < n \tag{8.1}$$

avec $P(n,1) = 1$, $P(n,n) = 1$ et $P(n,K) = 0$ si $K > n$. Une recherche exhaustive est donc très vite impossible. Dès $n = 20$ et pour quelques classes, P dépasse 10^{10}. Les méthodes classiques de classification non-supervisée sont donc de nature itérative et elles convergent vers un optimum local du **critère** cité plus haut. C'est sans doute l'un des inconvénients majeurs de ces méthodes.

D'autres difficultés intrinsèques ou points délicats sont à mentionner. Le nombre K de classe (ou un paramètre équivalent) est à fixer. Le choix de la mesure de distance (ou de dissemblance) entre données est laissé à l'utilisateur. Il faut donc faire des choix pertinents dans ce registre (cf remarques précédentes sur la distance de Mahalanobis). Le choix du **critère** d'homogénéité des classes est aussi laissé à l'appréciation de l'utilisateur.

Les deux méthodes de classification non-supervisée les plus connues sont vraisemblablement la classification ascendante hiérarchique (CAH) et les méthodes dites de réallocation dynamique (comme les k-moyennes, *ISODATA*, les nuées dynamiques, les techniques PAM -*Partitionning Around Medoïds*). Elles sont brièvement introduites dans la section 8.2 suivante.

8.1.5 Inférence semi-supervisée

On parle d'apprentissage semi-supervisé, lorsqu'en plus d'un ensemble de données supervisées $D = \{(x_i, y_i)\}, i = 1 \ldots n$, on dispose d'un ensemble de données non-supervisées $U = \{x'_i\}, i = 1 \ldots m$ ne contenant que des entrées sans sorties associées. De façon générale, l'apprentissage semi-supervisé consiste alors à utiliser U en plus

de D afin d'influencer la procédure d'un algorithme d'apprentissage et d'améliorer sa performance. Nous ne donnerons pas d'exemple particulier de ces méthodes très utilisées lorsque les bases d'apprentissage sont extrêmement coûteuses à obtenir (catégorisation visuelle d'objets par exemple).

8.2 Méthodes de *clustering*

8.2.1 Méthode des k-moyennes.

Nous présentons ici la méthode de *clustering* des k-moyennes (*k-means*). On se donne un ensemble d'apprentissage $D = \{\mathbf{x_i}\}_{i=1...n}$ de n observations d'un vecteur aléatoire $\mathbf{x}$ de dimension d. Nous souhaitons partitionner D en K clusters (K est fixé). Afin d'exprimer la notion d'homogénéité des données de chaque classe, on introduit le représentant $\mu_\mathbf{k}$ de la classe numéro k autour duquel les données de la classe vont se structurer/regrouper. Les K représentants $\mu_\mathbf{k}$ sont à l'image de ceux montrés dans la figure 8.2, ils seront le plus souvent de simples centres de gravités des clusters. Nous cherchons donc conjointement une allocation des n points au sein des K clusters et les K représentants $\mu_\mathbf{k}$.

Il est pratique d'introduire à ce niveau une fonction indicatrice d'allocation $\delta(k|i)$ qui alloue chaque donnée i (c'est-à-dire $\mathbf{x_i}$) au cluster k si $\delta(k|i) = 1$ ou qui indique, au contraire, que la donnée i n'est pas allouée au cluster k si $\delta(k|i) = 0$. On n'alloue ici qu'un cluster par donnée.

La mesure de dispersion associée au cluster k est alors définie, pour une certaine distance d^2 entre la donnée et le représentant du cluster, par :

$$J_k = \sum_{i=1}^{n} \delta(k|i) d^2(\mathbf{x_i}, \mu_\mathbf{k}) \tag{8.2}$$

La mesure de dispersion (ou erreur intraclasse) associée à la partition est alors définie par :

$$J = \sum_{k=1}^{K} J_k = \sum_{i=1}^{n} \sum_{k=1}^{K} \delta(k|i) d^2(\mathbf{x_i}, \mu_\mathbf{k}) \tag{8.3}$$

On souhaite naturellement[3] minimiser J. L'algorithme des k-moyennes opère itérativement en alternant l'optimisation des allocations des n points au sein des K clusters et celle relative au positionnement optimal des K représentants $\mu_\mathbf{k}$. L'algorithme est le suivant :

1. Pour K fixé, tirer au hasard, ou sélectionner pour des raisons extérieures à la méthode[4], K positions $\mu_\mathbf{k}$ dans l'espace $\mathcal{X}$ des données.

[3] C'est le **critère** de *clustering* évoqué plus haut.

[4] Toute connaissance a priori sur les données à classer est utilisable à ce niveau.

2. Allouer chaque individu au représentant $\mu_{\mathbf{k}}$ le plus proche au sens de la distance choisie :

$$\delta(k|i) = \begin{cases} 1 & \text{si} \quad k = \underset{\mathbf{l}\in 1\ldots K}{\operatorname{argmin}}\, d^2(\mathbf{x_i}, \mu_{\mathbf{l}}) \\ 0 & \text{sinon} \end{cases} \tag{8.4}$$

Il s'agit bien de minimiser trivialement J selon les termes $\delta(k|i)$, les $\mu_{\mathbf{k}}$ étant fixés (J est linéaire en les $\delta(k|i)$).

3. Optimiser la position des $\mu_{\mathbf{k}}$ de façon à minimiser J à allocation fixée. Pour $d^2(\mathbf{x_i}, \mu_{\mathbf{k}}) = ||\mathbf{x_i} - \mu_{\mathbf{k}}||^2$, la CN1 pour $\mu_{\mathbf{k}}$ conduit à :

$$2\sum_{i=1}^{N} \delta(k|i)(\mathbf{x_i} - \mu_{\mathbf{k}}) = 0 \tag{8.5}$$

Ce qui donne :

$$\mu_{\mathbf{k}} = \frac{\sum_i \delta(k|i)\mathbf{x_i}}{\sum_i \delta(k|i)} \tag{8.6}$$

On détermine ainsi le nouveau représentant (ici le centre de gravité) du cluster après la réallocation des données.

4. Répéter les étapes 2. et 3. au fil d'itérations d'indices a. Si on note J_a le critère obtenu à l'issue de l'itération a, on peut s'arrêter si $|(J_a - J_{a-1})/J_a| < \xi$ pour un seuil ξ fixé.

Le représentant $\mu_{\mathbf{k}}$ est bien ainsi une moyenne (équation 8.6) des $\sum_i \delta(k|i)$ données allouées au cluster k. Ce qui justifie le nom de l'algorithme des «k-moyennes». La convergence (locale) de l'algorithme est assurée puisque chaque étape réduit J. Cet algorithme rentre dans la classe générale des algorithmes de type *Expectation-Maximization*. La phase E d' *Expectation* est celle où on affine l'allocation (étape 2.). La phase M est celle dite de *Maximization* (étape 3.)pour laquelle on optimise les moyennes. La figure 8.3 suivante illustre le fonctionnement des k-moyennes sur un jeu de données relativement simple.

En pratique, plusieurs démarrages aléatoires de l'algorithme des k-moyennes sont intéressants. On peut, à l'issue de plusieurs exécutions, conserver la partition de critère optimal. On peut aussi ne regrouper/classer que les données systématiquement associées à l'issue de plusieurs démarrages des k-moyennes. Cet algorithme présente quelques défauts de fonctionnement si le nombre de données varie fortement d'un cluster «attendu» à l'autre. Il est également assez évident que des données très structurées et peu compactes posent des problèmes. On peut penser par exemple dans $\mathbb{R}^2$ à un nuage de points circulaires ou à des points alignés le long d'une droite. Les résultats les plus fiables sont obtenus si chaque cluster «attendu» présente un nuage de données plutôt compact.

8.2.2 Classification Ascendante Hiérarchique

Une méthode de Classification Ascendante Hiérarchique (CAH) consiste à regrouper itérativement des données d'apprentissage en construisant un arbre ou dendrogramme

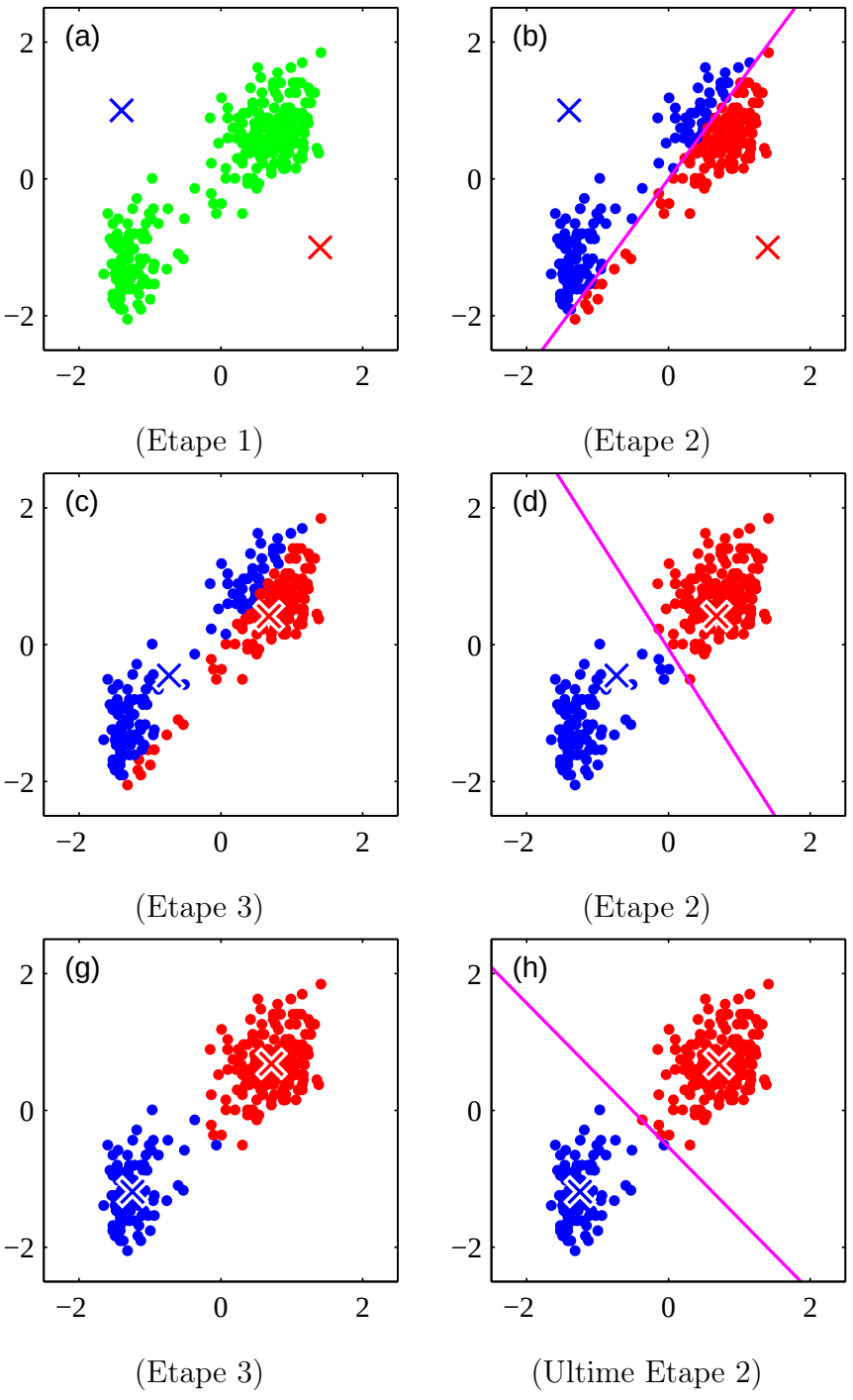


Figure 8.3: Illustration du fonctionnement des k-moyennes.

(cf figure 8.4) de bas en haut. Les feuilles de l'arbres sont associées à chaque singleton de donnée. La racine du dendrogramme regroupe toutes les données en une seule classe. Plus précisément, la CAH est initialement basée sur la connaissance d'un tableau $n \times n$ de distances (ou dissemblances) entre données et nécessite donc son stockage en mémoire. C'est une limite. L'algorithme démarre alors de la partition triviale des n singletons et cherche à regrouper les deux données les plus proches. Les regroupements d'une donnée et d'un groupe de données, puis de groupes de données deviennent alors possibles lorsqu'on remonte dans l'arbre. Cela suppose de savoir calculer, à chaque étape de remontée, la distance entre une donnée et un groupe de données ou la distance entre deux groupes de données. Par rapport à ce qui précède, l'utilisateur doit définir plus qu'une simple distance entre deux données d'apprentissage. Différents choix, appelés sauts en français ou *linkage* en anglais sont possibles.

Dans le cas élémentaire d'une distance euclidienne d, on appelle par exemple «saut de Ward», la distance définie entre deux regroupements A et B de barycentres respectifs g_A et g_B et de poids respectifs ω_A et ω_B par :

$$d(A,B) = \frac{\omega_A \omega_B}{\omega_A + \omega_B} d(g_A, g_B) \tag{8.7}$$

On montre que le saut de Ward pour la distance euclidienne induit une minimisation de la variance interclasse. Un autre saut possible est naturellement la distance simple entre barycentres $d(g_A, g_B)$. Les sauts choisis influencent notablement la forme du dendrogramme obtenu par la CAH (cf. figure 8.4).

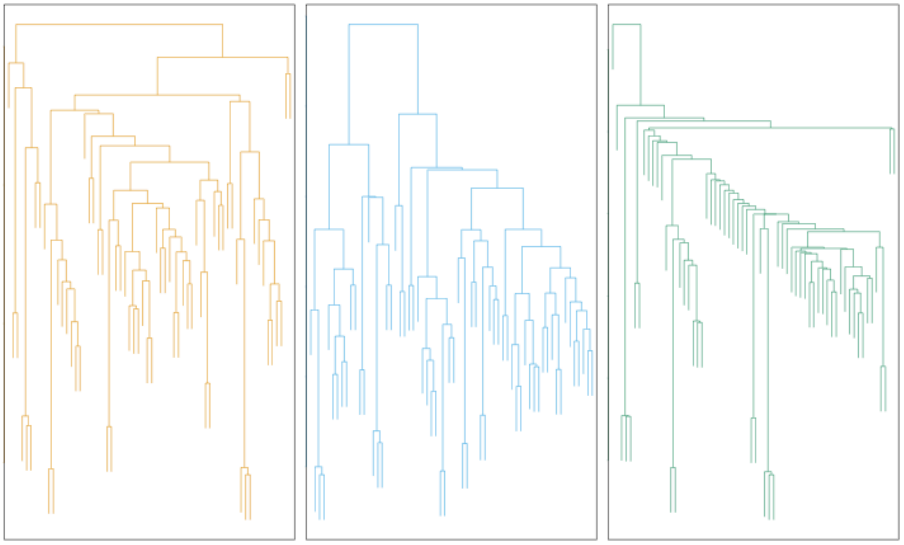


Figure 8.4: Différents dendrogrammes issus de différents «sauts».

Les grandes lignes d'un algorithme de CAH sont les suivantes :

1. **Initialisation**. Les classes initiales sont les n singletons de donnée. Calculer la matrice $n \times n$ de distances (ou dissemblances) deux à deux.
2. **Itération des deux étapes suivantes jusqu'à la racine**.

- **o** regrouper les deux classes les plus proches au sens du saut choisi
- **o** mettre à jour la matrice de distances en remplaçant les deux classes regroupées par la nouvelle et en calculant sa distance avec chacune des autres classes.

Le nombre retenu de classes est finalement déterminé a posteriori, sur la base d'un dendrogramme ou d'un graphique représentant les écarts de distance opérés à chaque regroupement.

8.3 Estimation d'un mélange de lois par EM

Une autre approche du *clustering* à K classes, notées $\omega_1 \dots \omega_K$, consiste à estimer une densité choisie comme un mélange de lois. On suppose que chaque classe ω_j est décrite par une loi conditionnelle notée :

$$p(\mathbf{x}|\theta_j) \tag{8.8}$$

Les paramètres θ_j suffisent à décrire la loi de chaque classe ω_j. Par exemple, les deux premiers moments $\theta_j = \mu_j, \Sigma_j$ suffisent à décrire une loi gaussienne pour $\mathbf{x} \in \mathbb{R}^p$. Le mélange des K classes est alors complètement défini si on connait la proportion p_j de chaque classe ω_j au sein du mélange (c'est-à-dire les probabilités a priori $p_j = P(\omega_j)$ de chaque classe). Cela conduit à la densité :

$$p(\mathbf{x}|\theta) = \sum_{j=1}^{K} p_j \;\; p(\mathbf{x}|\mu_j, \Sigma_j) \tag{8.9}$$

Dans cette équation, les paramètres $\theta = (p_1, \dots p_K, \mu_1, \Sigma_1, \dots \mu_K, \Sigma_K)$ du mélange forment les inconnues du problème de *clustering* ramené à un problème d'estimation de densité.

Pour expliquer les données d'apprentissage $D = \{\mathbf{x_i}\}_{i=1\dots n}$, il suffirait de connaître l'étiquette cachée $y_i \in \omega_1 \dots \omega_K$ qui associe la donnée $\mathbf{x_i}$ à la classe correcte y_i (par exemple $y_i = \omega_j$). La fonction indicatrice précédemment introduite est une variable cachée équivalente : $\delta(j|i)$ alloue la donnée i au cluster j si $\delta(j|i) = 1$ ou ne l'alloue pas à j si $\delta(j|i) = 0$. Connaissant ces variables y_i ou $\delta(j|i)$ cachées, l'estimation des composantes j du mélange serait immédiate :

- $\widehat{p_j} \leftarrow \frac{\widehat{n_j}}{n}$ avec $\widehat{n_j} = \sum_{i=1}^{n} \delta(j|i)$
- $\widehat{\mu_\mathbf{j}} \leftarrow \frac{1}{\widehat{n_j}} \sum_{i=1}^{n} \delta(j|i)\mathbf{x_i}$
- $\widehat{\Sigma_j} \leftarrow \frac{1}{\widehat{n_j}} \sum_{i=1}^{n} \delta(j|i)(\mathbf{x_i} - \widehat{\mu_\mathbf{j}})(\mathbf{x_i} - \widehat{\mu_\mathbf{j}})^T$

Comme ces variables sont cachées, on peut seulement choisir une valeur initiale pour θ : $\theta^0 = (p_1^0, \dots p_K^0, \mu_1^0, \Sigma_1^0, \dots \mu_K^0, \Sigma_K^0)$ et évaluer la probabilité a posteriori que y_i vale ω_j ayant en main la donnée $\mathbf{x_i}$:

$$P(y_i = \omega_j|\mathbf{x_i}, \theta^0) = \frac{p(\mathbf{x_i}|y_i = \omega_j, \theta^0)P(\omega_j)}{\sum_j p(\mathbf{x_i}|y_i = \omega_j, \theta^0)P(\omega_j)} \tag{8.10}$$

Que l'on peut réécrire en :

$$P(y_i = \omega_j | \mathbf{x_i}, \theta^0) = \frac{p_j^0 p(\mathbf{x_i} | \mu_j^0, \Sigma_j^0)}{\sum_j p_j^0 p(\mathbf{x_i} | \mu_j^0, \Sigma_j^0)} \tag{8.11}$$

On obtient ainsi au lieu d'une allocation stricte de la donnée $\mathbf{x_i}$ à la classe ω_j via $\delta(j|i) = 1$ ou 0, une allocation souple/probabiliste selon :

$$\widehat{p}(j|i) = P(y_i = \omega_j | \mathbf{x_i}, \theta^0) \tag{8.12}$$

Ce terme est simplement la probabilité a posteriori que la donnée $\mathbf{x_i}$ soit allouée à la classe ω_j ayant une estimation courante du mélange à notre disposition. On a bien entendu $\sum_{j=1}^{K} \widehat{p}(j|i) = 1$. Un algorithme itératif de type EM (mis pour *Expectation-Maximization*) similaire à celui décrit pour les k-moyennes en découle de manière assez immédiate :

1. **Initialisation**. Choix de $\theta^{(k=0)}$ à opérer aléatoirement ou stratégiquement.
2. **Itération selon k des deux étapes E et M suivantes**.

   (E) Assignation souple $\forall i$ de la donnée $\mathbf{x_i}$ d'apprentissage aux composantes du mélange

   $$\widehat{p}(j|i) \leftarrow P(y_i = \omega_j | \mathbf{x_i}, \theta^{(k)}) \tag{8.13}$$

   (M) Estimation affinée des $\theta^{(k+1)}$ selon :

   - $\widehat{p_j} \leftarrow \frac{\widehat{n_j}}{n}$ avec $\widehat{n_j} = \sum_{i=1}^{n} \widehat{p}(j|i)$
   - $\widehat{\mu_\mathbf{j}} \leftarrow \frac{1}{\widehat{n_j}} \sum_{i=1}^{n} \widehat{p}(j|i) \mathbf{x_i}$
   - $\widehat{\Sigma_j} \leftarrow \frac{1}{\widehat{n_j}} \sum_{i=1}^{n} \widehat{p}(j|i)(\mathbf{x_i} - \widehat{\mu_\mathbf{j}})(\mathbf{x_i} - \widehat{\mu_\mathbf{j}})^T$

Un critère d'arrêt de cet algorithme peut consister à observer une stagnation de la valeur de la log-vraisemblance :

$$lnL(D|\theta^{(k)}) = \sum_{i=1}^{n} ln \left(\sum_{j=1}^{K} \widehat{p_j} p(\mathbf{x_i} | \widehat{\mu_\mathbf{j}}, \widehat{\Sigma_j}) \right) \tag{8.14}$$

On montre en effet que la log-vraisemblance est maximisée localement par EM et ne peut donc qu'augmenter au fil des itérations. La figure 8.5 suivante illustre le fonctionnement de l'estimation d'un mélange de gaussiennes par EM.

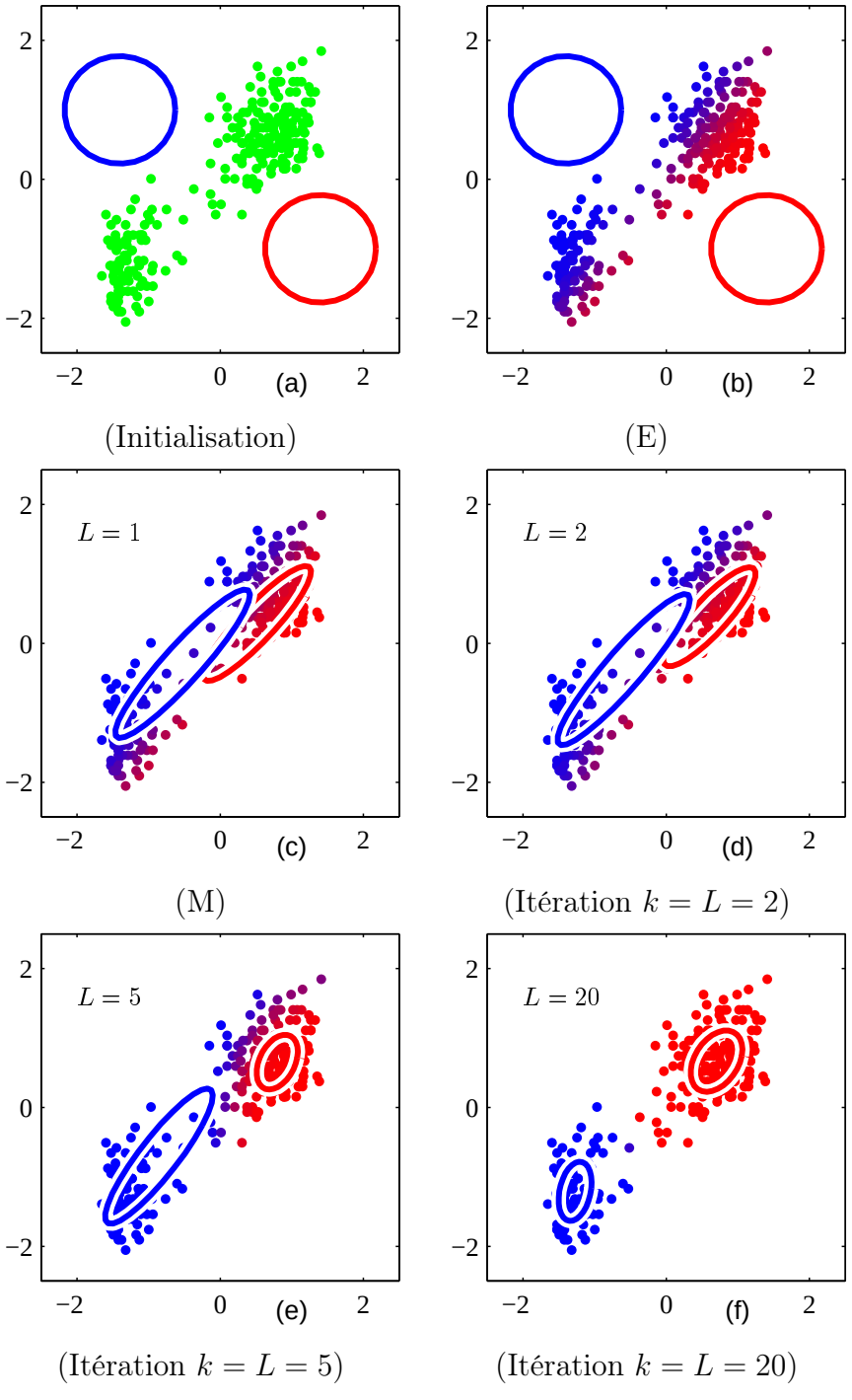


Figure 8.5: Illustration de l'estimation d'un mélange de gaussiennes par EM.

Thème 9

Apprentissage par renforcement

Vincent Charvillat
Mars 2010

Un bébé qui apprend à manier sa cuillère ou un enfant qui apprend à faire du vélo ignorent les équations différentielles qui gouvernent les systèmes mécaniques en présence. Ils procèdent par essai-erreur selon une stratégie d'apprentissage par renforcement. Ils ne reçoivent pas d'instructions précises (comme dans le cas de l'apprentissage supervisé) sur la meilleure conduite à adopter mais perçoivent des signaux d'encouragement ou de pénalisation suite aux actions qu'ils entreprennent[1]. On se place dans le cadre de l'apprentissage par renforcement lorsqu'un «agent» essaie de prédire séquentiellement les meilleures actions de contrôle ou qu'il cherche les meilleures décisions séquentielles dans un environnement incertain.

9.1 Décision séquentielle dans l'incertain

La figure 9.1 présente le cadre général de la décision séquentielle dans l'incertain. Nous considérons un agent dont l'objectif est de prédire la meilleure action à effectuer dans un état donné de l'environnement au sein duquel il doit apprendre à se comporter de manière optimale.

L'agent apprend de son interaction avec l'environnement en observant la dynamique de ce dernier et en recevant des récompenses. Il apprend par «essai-erreur» en renforcant les décisions d'actions qui conduisent à de bons cumuls de récompenses et, inversement, évite de renouveler des décisions infructueuses. Cela se traduit par une amélioration itérative de sa «politique décisionnelle».

Afin d'exprimer mathématiquement ce type de problèmes, on utilise une approche formelle où :

[1] On parle alors d'apprentissage par évaluation et on appelle de tels signaux des «récompenses» s'il faut les maximiser.

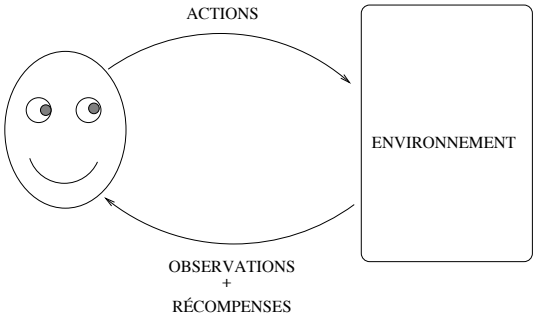


Figure 9.1: L'agent et l'environnement

1. **L'état** résume la situation de l'agent et de l'environnement à chaque instant. Il peut décrire la relation de l'agent à l'environnement (position, etc.), l'état propre de l'environnement, et l'état interne de l'agent (ses ressources, sa mémoire, etc.). La dynamique de l'état résulte des actions de l'agent sur l'environnement, de la dynamique propre de l'environnement, voire de la dynamique propre de l'agent. Ces dynamiques sont la plupart du temps non-déterministes, ce qui signifie que les mêmes actions n'entraînent pas toujours les mêmes effets.

2. **Les actions** sont choisies et exécutées par l'agent à chaque instant. Suite à cela, il reçoit une récompense instantanée, et perçoit le nouvel état courant. On parle de «trajectoire» lorsqu'on énumère la liste des paires (états, actions) suivies depuis un état initial donné.

3. **Les récompenses** sont réelles, positives ou négatives. On cherchera à maximiser ces récompenses ou minimiser les pertes pour améliorer la stratégie ou la politique de l'agent.

4. **La politique** modélise le comportement décisionnel de l'agent. Il s'agit dans le cas général d'une fonction, déterministe (ou aléatoire), associant à l'état courant et éventuellement à la trajectoire passée, l'action courante à exécuter. Compte tenu des incertitudes, une même politique peut donner suite à des trajectoires très variables selon les aléas.

Du point de vue de l'optimisation, nous chercherons à maximiser la qualité d'une politique décisionnelle. Il faudra donc être en mesure de comparer diverses politiques et d'améliorer une politique donnée. Naturellement un critère d'évaluation de la qualité d'une politique sera stochastique. Nous travaillerons dans le cadre général de l'optimisation stochastique.

9.2 Définition d'un Processus décisionnels de Markov

Un processus décisionnel de Markov (PDM) est un processus stochastique contrôlé satisfaisant la propriété de Markov, assignant des récompenses aux transitions d'états. On le définit par un quintuplet $(S; A; T; p_t; r_t)$ où S est l'ensemble fini (ici) d'états, A est celui des actions, T est l'axe temporel discret décrivant les instants

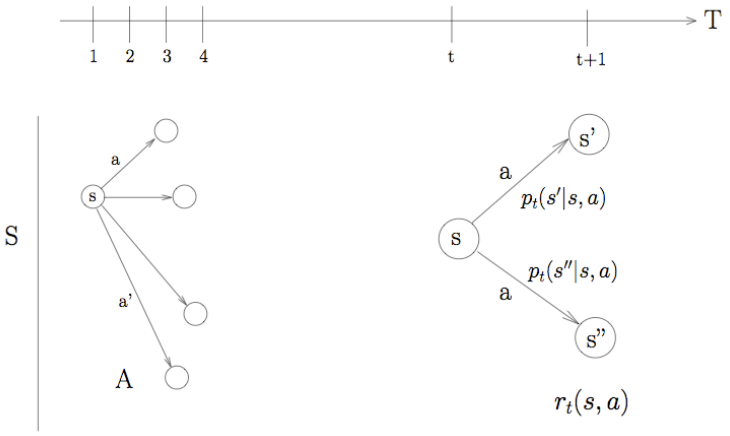


Figure 9.2: L'agent et l'environnement

$t \in T$ d'action et de transition entre états, $p_t()$ sont les probabilités de transitions entre états, $r_t()$ est une fonction de récompense sur les transitions. On retrouve donc formellement les ingrédients utiles à la figure 9.1.

A chaque instant t de T (figure 9.2), l'agent observe l'état courant $s \in S$, applique sur le système une action $a \in A$ qui l'amène, aléatoirement selon $p_t()$, dans le nouvel état s', et reçoit une récompense $r_t()$.

Le domaine T des étapes de décision est dans le cas le plus général un sous-ensemble de la demi-droite $\mathbb{R}^+$. Toutefois, lorsque T est continu, les problèmes de décision séquentielle correspondants sont mieux traités par des méthodes de contrôle optimal, basées sur les équations de la dynamique du système ; nous considérons ici uniquement le cas discret. Pour T discret, l'ensemble des étapes de décision peut être fini ou infini (on parle d'horizon fini ou infini), et ces étapes correspondent pour nous à des instants prédéterminés sur $\mathbb{R}^+$.

Les ensembles S et A seront supposés finis. Les distributions de probabilités de transition p_t identifient la dynamique de l'état Markovien du système. Pour une action a fixée, $p_t(\sigma'|\sigma, a)$ est la probabilité que le système dans l'état σ à la date t passe dans l'état σ' après avoir exécuté l'action a.

Ayant décidé l'action a dans l'état σ à l'instant t, l'agent reçoit une récompense, ou revenu, $r_t(\sigma, a) \in \mathbb{R}$. Les valeurs de r_t positives peuvent être considérées comme des gains, et les valeurs négatives comme des coûts. Cette récompense peut être instantanément perçue à la date t, ou accumulée de la date t à la date $t+1$, l'important étant qu'elle ne dépende que de l'état à l'instant courant et de l'action choisie.

Nous utiliserons par la suite des processus stationnaires, où les probabilités de transitions p_t et les récompenses r_t ne dépendent pas du temps : $\forall t \in T, p_t = p, r_t = r$.

Les politiques d'actions et les critères de performance Une stratégie associée à une politique est une fonction qui permet à l'agent, étant données les observations passées et présentes, de choisir à chaque instant t l'action à exécuter dans l'environnement afin de maximiser une récompense ou plus fonctionnellement de contrôler les trajectoires. Plusieurs remarques peuvent être faites à ce niveau. D'abord,

une politique peut déterminer précisément l'action à effectuer, ou simplement définir une distribution de probabilité selon laquelle cette action doit être sélectionnée. Nous nous intéresserons dans ce travail au cas des politiques stationnaires, markoviennes et déterministes qui associent de manière déterministe et indépendamment du temps une unique action à chaque état.

Résoudre un problème décisionnel de Markov consiste à rechercher dans l'espace des politiques considéré, celle (ou une de celles) qui optimise un critère de performance choisi pour le PDM. Dans le cadre des PDM, ce critère a pour ambition de caractériser les politiques qui permettront de générer des séquences de récompenses les plus importantes possibles. Mathématiquement, on cherchera à évaluer une politique sur la base d'une mesure du cumul espéré des récompenses instantanées le long d'une trajectoire. En ce qui nous concerne, nous utiliserons le critère γ-pondéré avec $\gamma \leq 1$:

$$E\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$$

En remarquant le caractère additif et séparable (en temps) du critère précédent, on peut exploiter le principe d'optimalité de la programmation dynamique. Nous utilisons le critère γ-pondéré pour lequel les résultats sont les plus généraux, en particulier pour la résolution des PDM par des méthodes d'apprentissage par renforcement. Le facteur γ utile en particulier en horizon infini, permet de réduire l'influence des récompenses « infiniment » éloignées vis-à-vis de la stratégie décisionnelle à adopter en un état donné.

9.3 Résolution et Apprentissage par renforcement

Le cadre théorique des PDM fait correspondre à chaque politique π une *fonction de valeur* V_π qui associe à un état $\sigma \in S$ une récompense globale $V_\pi(\sigma)$, obtenue en appliquant π à partir de σ. Une telle fonction de valeur permet de comparer des politiques. Une politique π domine une politique π' si

$$\forall \sigma \in S, \quad V_\pi(\sigma) \geq V_{\pi'}(\sigma)$$

Pour le critère γ-pondéré, on définit ainsi :

$$\forall \sigma \in S, \quad V_\pi(\sigma) = E\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = \sigma\right]$$

Cette fonction de valeur donne donc, pour un état, l'espérance des revenus futurs qui nous reviendront si on applique la politique π depuis cet état. Elle permet de formaliser la recherche d'une politique optimale π^* : c'est celle associée à la meilleure fonction de valeur $V^* = V_{\pi^*}$.

Les *équations d'optimalité de Bellman* caractérisent la fonction de valeur optimale unique V^* et une politique optimale π^* qu'on peut en déduire. Dans le cas du critère γ-pondéré, elles s'écrivent :

$$V^*(\sigma) = \max_{a \in A} \left(r(\sigma, a) + \gamma \sum_{\sigma' \in S} p(\sigma' | \sigma, a) V^*(\sigma') \right) \tag{9.1}$$

$$\forall \sigma \in S, \quad \pi^*(\sigma) = \underset{a \in A}{\operatorname{argmax}} \left(r(\sigma, a) + \gamma \sum_{\sigma' \in S} p(\sigma'|\sigma, a) V^*(\sigma') \right)$$

Pour les PDMs dont les nombres d'états et d'actions sont modestes (S et A sont assez petits), *l'algorithme d'itération de la valeur* ou son correspondant dans l'espace des politiques (*l'itération de politique*) sont des algorithmes de programmation dynamique stochastique utilisables. Il faut noter d'emblée que les recherches sont actives dans ce domaine.

Pour l'iteration de la valeur, on veut résoudre les équations de Bellman en $V^*(\sigma)$ en utilisant une méthode itérative de type point fixe qui calcule une suite $(V_n)_n$. Nous choisissons une V_0 aléatoirement, puis utilisons des itérations :

$$\forall \sigma, \quad V_{n+1}(\sigma) = \max_a \left(r(\sigma, a) + \gamma \sum_{\sigma'} p(\sigma'|\sigma, a) V_n(\sigma') \right)$$

Chaque itération améliore la politique courante associée à V_n. Si les récompenses sont bornées, la suite converge vers V^*, dont nous pouvons déduire π^*. La communauté de l'intelligence artificielle a développé récemment les méthodes de *l'apprentissage par renforcement* qui rendent l'optimisation des politiques pour les larges PDMs possible, par simulations et approximations pour la fonction de valeur et/ou la politique.

L'approche de l'apprentissage par renforcement consiste à apprendre une politique optimale au travers des estimations itératives de la fonction de valeur optimale en se basant sur des simulations. Aujourd'hui, l'apprentissage par renforcement est une des principales approches capables de résoudre les problèmes de décision séquentielle pour lesquels on ne dispose pas des probabilités de transition (absence de « modèle ») ou pour les PDMs avec un très large espace d'états. Il existe plusieurs algorithmes pour l'apprentissage par renforcement, tels que *Q-learning*, *R-learning*, *Sarsa*, *TD(λ)*.

Dans notre travail on va utiliser l'algorithme le plus célèbre (et aussi un des plus simples) à savoir le *Q-learning*. Le *Q-learning* est une méthode d'apprentissage par renforcement permettant de résoudre l'équation de Bellman pour le critére γ-pondéré. Son principe consiste à utiliser des simulations pour estimer itérativement les valeurs de la fonction V^* recherchée, sur la base de l'observation des transitions instantanées et de leur revenu associé. Pour cela, Watkins a introduit la fonction Q, dont la donnée est équivalente à celle de V mais qui permet un accès simple à la politique associée. Cet accès est surtout indépendant des probabilités de transitions propres au modèle Markovien sous-jacent.

A une politique π fixée de fonction de valeur V_π, on associe la nouvelle fonction dite de « Q-valeur » :

$$\forall \sigma \in S, a \in A, \quad Q_\pi(\sigma, a) = r(\sigma, a) + \gamma \sum_{\sigma'} p(\sigma'|\sigma, a) V_\pi(\sigma')$$

L'interprétation de la valeur Q_π est la suivante : c'est la valeur espérée du critère pour le processus partant de σ, exécutant l'action a, puis suivant la politique π par la suite. Il est clair que $V_\pi(x) = Q_\pi(x, \pi(x))$, et l'équation de Bellman vérifiée par la fonction Q^* devient :

Initialize Q_0
for $n = 0$ **to** $N_{\text{tot}} - 1$ **do**
σ_n =`choseState`
a_n =`choseAction`
(σ'_n, r_n) =`simulate`(σ_n, a_n)
`/*` update Q_{n+1} `*/`
$Q_{n+1} \leftarrow Q_n$
$d_n = r_n + \gamma \max_b Q_n(\sigma'_n, b) - Q_n(\sigma_n, a_n)$
$Q_{n+1}(\sigma_n, a_n) \leftarrow Q_n(\sigma_n, a_n) + \alpha_n d_n$
end for
return $Q_{N_{\text{tot}}}$

Figure 9.3: L'algorithme *Q-learning*.

$$\forall \sigma \in S, a \in A, \quad Q^*(\sigma, a) = r(\sigma, a) + \gamma \sum_{\sigma'} p(\sigma'|\sigma, a) \max_b Q^*(\sigma', b)$$

Ensuite, on voit qu'il devient élémentaire malgré l'absence de connaissance des probabilités de transition de « remonter » à la politique optimale :

$$\forall \sigma \in S, \quad V^*(\sigma) = \max_a Q^*(\sigma, a) \quad \pi^*(\sigma) = \operatorname*{argmax}_a Q^*(\sigma, a)$$

Le principe de l'algorithme *Q-learning* (figure 9.3) est de mettre à jour, à la suite de chaque transition observée $(\sigma_n, a_n, \sigma_{n+1}, r_n)$, la fonction de valeur courante Q_n pour le couple (σ_n, a_n), où σ_n représente l'état courant, a_n l'action sélectionnée et réalisée, σ_{n+1} l'état résultant et r_n la récompense immédiate.

Dans cet algorithme, N_{tot} est un paramètre initial fixant le nombre d'itérations. Le *taux d'apprentissage* $\alpha_n(\sigma, a)$ est propre à chaque paire état-action, et décroit vers 0 à chaque passage. La fonction `simulate` retourne un nouvel état et la récompense associée selon la dynamique du système. Le choix de l'état courant et de l'action à exécuter est effectué par les fonctions `choseState` et `choseAction`. La fonction `initialize` revient la plupart du temps à initialiser les composantes de Q_0 à 0.

La convergence de cet algorithme a été bien étudiée et est maintenant établie. Sous les hypothèses de finitude de S et A, si l'on suppose que chaque paire (s, a) est visitée un nombre infini de fois, si $\sum_n \alpha_n(s, a) = \infty$ et $\sum_n \alpha_n(s, a)^2 < \infty$, et enfin si $\gamma < 1$ ou si $\gamma = 1$ et pour toute politique il existe un état absorbant de revenu nul, alors la fonction Q_n converge presque sûrement vers Q^*.

9.4 Exercice sur les PDM.

Ronan est un concessionnaire automobile et cherche à maximiser ses profits. Chaque lundi Ronan achète des voitures de prix unitaire d en vue de les revendre. Ces voitures lui sont livrées instantanément (hypothèse simplificatrice !) et se rajoutent aux s voitures déjà en stock. Ronan ne peut stocker que S voitures au total dans ses installations. Pendant la semaine, Ronan vend un nombre aléatoire de voitures (noté

k). La v.a.r discrète k est uniformément distribuée entre 0 et le nombre maximal de voitures disponibles à la vente pour la semaine considérée. Chaque voiture est vendue au coût unitaire c. Le stockage induisant des frais, chaque voiture non-vendue coûte (par semaine) un montant unitaire u.

1. On vous demande de modéliser le problème d'achat optimal comme un Processus Décisionnel de Markov (PDM). C'est-à-dire :

   - (i) Préciser d'abord l'axe temporel des décisions. En d'autres termes répondre à la question : quand sont prises les décisions ?
   - (ii) Choisir un espace d'états (on notera σ un état dont la structure interne fera appel aux variables définies plus haut).
   - (iii) Expliciter l'espace des actions (une action sera notée a).
   - (iv) Donner les équations des probabilités de transitions $P(\sigma'|\sigma, a)$ en fonction des variables définies plus haut.
   - (v) Donner enfin l'équation des récompenses $r(\sigma, a, \sigma')$ en intégrant l'ensemble des éléments financiers (achat, vente, coût du stockage).

2. Si l'hypothèse sur la loi de k est exacte quel algorithme d'apprentissage de la politique optimale peut-on utiliser ? Si cette loi est inconnue a priori quel type d'algorithme est préférable ?

3. On considère maintenant que les décisions du lundi prises par Ronan sont des commandes de voitures à livrer le lundi suivant. Quelles modifications pourriez-vous apporter à votre modélisation ?

10

Travaux Pratiques

Vincent Charvillat
Janvier 2010

La séquence de TP proposée cette année est la suivante :

- Premier pas en apprentissage artificiel,
- Biais, corrélation et apprentissage actif,
- Erreur de généralisation et validation croisée,
- Validation croisée, GCV, AIC,
- Contrôle de complexité par régularisation,
- Eigenfaces,
- Décision et classification Bayésiennes (2 séances).