Abrégé non exhaustif sur l'évaluation et la sélection de modèles et la sélection de variables

Matthieu Geist

2013 - 2014

Résumé

Pour un problème donné, le domaine de l'apprentissage machine (*machine learning*) peut proposer une grand nombre d'algorithmes différents (par exemple, machine à vecteur support ou régression logistique pour un problème de classification binaire). Il est alors naturel de vouloir comparer ces différents algorithmes. Cette comparaison se doit d'être quantitative, c'est ce que nous présentons sous le terme d'*évaluation de modèle*. Des problématiques connexes sont la *sélection de modèle* et la *sélection de variables*, nous les définissons et les discutons également (notamment leur lien au problème de l'évaluation, particulièrement les erreurs à ne pas faire). Ce document n'a pas pour ambition d'être exhaustif, il exposes quelques notions de base et fournit des pointeurs vers la littérature.

Table des matières

1 Préliminaires

Dans cette section, nous commençons par poser le cadre de l'apprentissage statistique tel qu'introduit par Vapnik [27], afin principalement de fixer les notations. C'est le cadre probabiliste usuel de l'apprentissage statistique, dont dérive naturellement la notion de risque qui servira à quantifier la qualité d'un modèle.

Dans ce document, nous resterons volontairement éloignés de l'aspect algorithmique de l'apprentissage supervisé, afin de conserver le caractère générique de nos propos. Toutefois, le lecteur intéressé peut se référer à [17], dont une partie du présent document s'inspire, qui est un ouvrage très complet et relativement exhaustif sur les techniques d'apprentissage supervisé (en mettant l'accent sur les fondements statistiques du domaine).

1.1 Paradigme de l'apprentissage supervisé

Pour formaliser le problème de l'apprentissage supervisé, nous nous donnons :

- un générateur aléatoire de vecteurs $\mathbf{x} \in \mathbb{X}$, échantillonnés (généralement indépendamment) selon une distribution $\mathbb{P}(\mathbf{x})$, fixe mais inconnues. Typiquement, $\mathbb{X}$ est ici un sous-espace de $\mathbb{R}^d$;
- un superviseur (ou oracle) qui pour chaque entrée $\mathbf{x}$ associe une sortie $y \in \mathcal{Y}$, échantillonnée selon la distribution conditionnelle $\mathbb{P}(y|\mathbf{x})$, également fixe mais inconnue. Typiquement, $\mathcal{Y}$ sera identifiable à un sous-ensemble fini de $\mathbb{N}$ pour la classification ou à un sous-ensemble compact de $\mathbb{R}$ pour la régression. Nous ne considérons pas de sorties vectorielles, mais les résultats présentés s'y adaptent ;
- une machine capable d'implémenter un ensemble de fonctions, c'est-à-dire que nous définissons un espace d'hypothèses $\mathcal{H} \subset \mathcal{Y}^{\mathbb{X}}$.

L'objectif est donc de trouver la fonction $f \in \mathcal{H}$ qui prédise "au mieux" les réponses de l'oracle.

Pour quantifier cela, nous introduisons une fonction de perte $L \in \mathbb{R}^{\mathcal{Y}\times\mathcal{Y}}$ qui mesure l'erreur $L(y, f(\mathbf{x}))$ entre la réponse y de l'oracle pour une entrée $\mathbf{x}$ et la prédiction $f(\mathbf{x})$ pour cette même entrée. Le risque réel est alors l'intégration de la fonction de perte sur l'espace des possibles :

$$R(f) = \int_{\mathbb{X}\times\mathcal{Y}} L(y, f(\mathbf{x}))d\mathbb{P}(\mathbf{x}, y) \tag{1}$$

$$= \mathbb{E}[L(y, f(\mathbf{x}))]. \tag{2}$$

L'objectif de l'apprentissage supervisé est alors de déterminer la fonction f de $\mathcal{H}$ qui minimise le risque réel associé, c'est-à-dire résoudre

$$f_0 \in \operatorname*{argmin}_{f \in \mathcal{H}} \mathbb{E}[L(y, f(\mathbf{x}))]. \tag{3}$$

Toutefois, même en faisant abstraction de la difficulté potentielle du problème d'optimisation sous-jacent, ce problème nécessite de connaître la distribution jointe $\mathbb{P}(\mathbf{x}, y) = \mathbb{P}(\mathbf{x})\mathbb{P}(y|\mathbf{x})$, qui est inconnue. La seule information disponible se trouve dans les données. On suppose disposer d'une base d'entraînement composée de couples entrée-sortie, $\{(\mathbf{x}_i, y_i)_{1 \leq i \leq N}\}$. Nous noterons parfois z_i le couple $(\mathbf{x}_i, y_i)$. On peut alors estimer f_0 par f_N, minimiseur du risque empirique associé à la base d'entraînement :

$$f_N \in \operatorname*{argmin}_{f \in \mathcal{H}} R_N(f) \text{ avec } R_N(f) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(\mathbf{x}_i)). \tag{4}$$

On peut remarquer que pour une fonction f donnée, le risque empirique est un estimateur non biaisé du risque réel, c'est-à-dire que $\mathbb{E}[R_N(f)] = R(f)$. Toutefois, dans le cas général on ne peut pas écrire que le risque empirique du minimiseur f_N est un estimateur sans biais du risque réel du minimiseur f_0, cela en raison de la dépendance implicite de f_N en les données (qui sont les quantités aléatoires).

Dès lors, on peut se demander si l'estimateur f_N converge vers la solution f_0, à quel point $R_N(f_N)$ est une bonne approximation de $R(f_N)$, etc. Vapnik [27] répond à ses questions, parmi d'autres. Sans trop entrer dans les détails, il montre par exemple qu'avec probabilité d'au moins $1 - \delta$ on a

$$\forall f \in \mathcal{H}, \quad R(f) \leq R_N(f) + O\Big(\sqrt{\frac{d_{\text{vc}}}{N} \ln \frac{1}{\delta}}\Big), \tag{5}$$

où d_{vc} est la dimension de Vapnik-Chervonenkis de l'espace d'hypothèse $\mathcal{H}$ (grossièrement, le nombre de degrés de liberté, soit le nombre de paramètres libres pour une architecture linéaire). On peut donc calculer une borne supérieure du risque réel, qui montre par ailleurs que le principe de minimisation du risque empirique est pertinent et donne même sa vitesse de convergence. Cela pourrait être utilisé pour évaluer un estimateur f, et même sélectionner le modèle (entendre l'espace d'hypothèse, voir le principe de minimisation structurelle du risque [1] [27]). Notons toutefois que nous avons ici accès à

1. Simplement, on considère des espaces d'hypothèses imbriqués les uns dans les autres, on résout autant de problèmes de minimisation empirique correspondant, puis on choisit l'estimateur minimisant la borne entière (ne considérer que le risque empirique entraînerait une préférence des modèles les plus riches, c'est la problématique du sur-apprentissage).

une majoration du risque, pas au risque lui-même. La borne est vraie pour toute distribution $\mathbb{P}(\mathbf{x}, y)$. C'est sa grande force, de par le caractère général qu'elle apporte, mais c'est aussi sa faiblesse, dans la mesure où la borne peut souvent être large. Pour cette raison notamment, nous ne développons pas plus avant cette approche, le lecteur intéressé pour se référer à [13] pour une introduction au sujet.

1.2 Objet de ce document et sémantique

Dans ce document nous traitons plusieurs problèmes, à savoir estimer la qualité d'un estimateur, déterminer quelles variables (quelles composantes de $\mathbf{x}$) utiliser pour construire l'estimateur et enfin sélectionner un estimateur parmi plusieurs, tout cela pouvant être allègrement combiné et mélangé.

(Evaluation de modèle.) Soit un estimateur $\hat{f}$ donné, un premier problème est d'estimer son adéquation au problème posé, via une mesure quantitative de sa qualité. Dans ce document, nous considérons que la qualité d'un estimateur se mesure par le risque réel associé [2], qui doit être estimé, ce que nous traitons section 2. Cela suppose le choix d'une fonction de perte L. Ce n'est pas forcément la fonction de perte sous-jacente à l'algorithme, qui peut d'ailleurs être motivé par d'autres arguments (par exemples, il n'y a pas de fonction de perte évidente *a priori* pour l'algorithme de classification des k plus proches voisins). On peut par exemple mesurer la qualité d'un classifieur binaire en utilisant une perte quadratique. Toutefois, la décision dépendant du signe de l'estimateur et non de l'amplitude, cela n'est pas forcément pertinent. Tout dépend de ce que l'on souhaite mesurer. Toute autre chose étant égale, la perte la plus naturelle est celle sous-jacente à l'algorithme (lorsqu'il y en a une).

(Sélection de variables.) Pour produire un estimateur $\hat{f}$, on peut être amené à considérer un grand nombre de variables. Par variable, nous entendons l'une des composantes d'un vecteur d'entrée $\mathbf{x}$, que nous noterons x_j. Notons que cette sémantique englobe plus généralement le problème de sélection de fonctions de base (comme le choix des gaussiennes dans un réseau de fonctions à base radiale), si l'architecture est linéaire. Typiquement, si l'on

2. On pourrait considérer nombre d'autres critères, comme les matrices d'adjacence, les courbes ROC, etc. Si le critère considéré est en fait un risque pour une certaine fonction de perte, le cas traité l'englobe. Si de façon générale ce critère est un moment d'une quantité aléatoire pouvant être estimé à partir des données, les notions de validation croisée et de bootstrapping présentées section 2 devraient s'appliquer très directement. Sinon, me consulter.

considère des fonctions de base, cela revient à transformer les entrées grâce à l'attribut vectoriel ϕ (concaténation des fonctions de base) défini par

$$\phi : \mathbf{x} \in \mathbb{R}^d \to \phi(\mathbf{x}) \in \mathbb{R}^p. \tag{6}$$

On peut donc considérer les couples entrée-sortie $(\phi(\mathbf{x}_i), y_i)$ pour la sélection de variables (où l'on sélectionne maintenant une des composantes de ϕ), il n'y a aucune perte de généralité. Cette problématique de sélection de variable est intrinsèquement liée à l'évaluation et à la sélection de modèle, nous la traitons donc section 3.

(Sélection de modèle.) Au delà de la sélection de variables, on peut se poser la question du choix des méta-paramètres, de l'espace d'hypothèse [3], etc. Nous traitons cet aspect section 4. Ce choix entre plusieurs modèles (*sélection de modèle*) se fait en utilisant l'estimation du risque (*évaluation de modèle*), mais les deux notions ne doivent pas être confondues et doivent être utilisées de concert correctement.

2 Estimation du risque

Cette section présente des méthodes d'estimation du risque réel. Il est clair que si l'on considère un estimateur $\hat{f}$ obtenu en minimisant un risque empirique, le minimum atteint est une très mauvaise estimation du risque réel (car trop optimiste). D'un point de vue probabiliste, cela est dû à la dépendance (évidente) de l'estimateur aux données. Pour s'en convaincre plus intuitivement, il suffit de considérer le cas du sur-apprentissage. Il faut donc utiliser d'autres données (validation croisée et à K plis [2, 24, 25]), ou les mêmes mais en faisant attention (bootstrapping [8, 11]) [4].

3. Le choix de l'espace d'hypothèses peut-être vu comme redondant avec le problème de sélection de variables dans le cas d'une dépendance linéaire de l'estimateur en ces dernières, si l'on considère un seul et même algorithme. En effet, la section de variables consiste à sélectionner les composantes de l'entrée $\mathbf{x}$ nécessaire à la prédiction. Si l'on considère un attribut vectoriel ϕ, la sélection de l'espace d'hypothèse consiste à choisir les composantes (les fonctions de base) de ϕ utiles au problèmes. En considérant des entrées $\phi(\mathbf{x})$, les deux problématiques se rejoignent. Toutefois, ce n'est généralement pas le cas pour les approches non-linéaires, considérer par exemple un réseau de neurones à une ou plusieurs couches cachées.

4. Notons qu'il existe d'autres critères que l'estimation du risque réel, comme les critères BIC (*Bayesian Information Criterion*), AIC (*Akaike Information Criterion*) ou encore la statistique C_p. Ils sont moins généraux (les trois estiment une quantité proportionnelle à l'erreur en *fixed-design*, c'est-à-dire l'erreur uniquement sur les échantillons d'entraînement, sous des contraintes plus ou moins fortes —C_p pour une perte quadratique et une para-

2.1 Validation croisée

La validation croisée est probablement la méthode la plus simple et la plus utilisée pour estimer le risque réel d'un estimateur. Supposons que l'on dispose d'un estimateur $\hat{f}$ calculé à partir de la base d'exemples $\{(\mathbf{x}_i, y_i)_{1\leq i\leq N}\}$. Supposons également disposer d'une base de test $\{(\mathbf{x}'_i, y'_i)_{1\leq i\leq M}\}$, constituée de couples entré-sortie échantillonnés selon la même distribution jointe $\mathbb{P}(\mathbf{x}, y)$, mais de façon indépendante (notamment aux échantillons d'entraînement). Un estimateur $\hat{R}$ du risque réel du modèle $\hat{f}$ est

$$\hat{R}(\hat{f}) = \frac{1}{M}\sum_{i=1}^{M} L(y'_i, \hat{f}(\mathbf{x}'_i)). \tag{7}$$

Si l'on veut un intervalle de confiance sur ce risque, il suffit d'appliquer une inégalité de concentration. Typiquement, en utilisant Hoeffding, on devrait pouvoir montrer qu'avec probabilité d'au moins $1-\delta$ on a

$$R(\hat{f}) \leq \hat{R}(\hat{f}) + O\Big(\sqrt{\frac{1}{M}\ln\frac{1}{\delta}}\Big). \tag{8}$$

Cette approche est légitime si l'on peut échantillonner des exemples de test à la demande ou si l'on dispose d'une grande base d'exemples, que l'on peut alors scinder (une partie pour l'apprentissage, une autre pour l'évaluation). Une alternative est la validation croisée à K-plis.

Avant cela, nous anticipons légèrement la section 4, qui traite de la sélection de modèle. Supposons que notre estimateur dépende d'un certain méta-paramètre α (par exemple facteur de régularisation pour la régression de Tikhonov, c'est-à-dire régularisation ℓ_2). A chaque valeur du méta-paramètre correspond un modèle $\hat{f}_\alpha$; il s'agit de choisir le meilleur (par rapport à notre critère). On peut par exemple choisir α qui minimise notre estimé du risque réel :

$$\hat{\alpha} \in \operatorname*{argmin}_{\alpha} \hat{R}(\hat{f}_\alpha) \text{ où } \hat{R}(\hat{f}_\alpha) = \frac{1}{M}\sum_{i=1}^{M} L(y'_i, \hat{f}_\alpha(\mathbf{x}'_i)). \tag{9}$$

Supposons que l'on souhaite ensuite évaluer le modèle $\hat{f}_{\hat{\alpha}}$, par exemple pour le comparer à un autre algorithme, produisant d'autres estimateurs. On ne peut pas utiliser le minimum de $\hat{R}(\hat{f}_\alpha)$, essentiellement pour la même raison que celle qui nous a fait introduire l'estimateur $\hat{R}$ au départ : la dépendance aux données. Il faut donc introduire un troisième ensemble de couples entrée-sortie $\{(\mathbf{x}''_i, y''_i)_{1\leq i\leq P}\}$, indépendant des précédents. Le risque est alors estimé

métrisation linéaire, AIC et BIC plus généralement pour une perte log-vraisemblance—), nous ne les considérons donc pas.

par $\tilde{R}$:

$$\tilde{R}(\hat{f}_{\hat{\alpha}}) = \frac{1}{P}\sum_{i=1}^{P} L(y_i'', \hat{f}_{\hat{\alpha}}(\mathbf{x}_i'')). \tag{10}$$

Ainsi, ils faut considérer en général trois ensembles d'exemples :

- un ensemble d'*entraînement* pour calculer les estimateurs ;
- un ensemble de *validation* pour la sélection de méta-paramètre (entre autre, il peut également servir pour la sélection de variables, voir section 3.3) ;
- un ensemble de *test* pour l'évaluation du modèle.

2.2 Validation croisée à K plis

En général, il n'y a pas suffisamment de données pour diviser la base d'exemples en plusieurs parties (entraînement, validation et test). Une solution consiste donc à utiliser les données à la fois pour l'entraînement et pour le test, ce qui est fait par la validation croisée à K plis. Formellement, la validation croisée estime le risque réel dans le cas d'une division apprentissage/test. Si l'on souhaite une division apprentissage/validation/test, il suffit de cascader.

Le principe est le suivant. On dispose d'un ensemble de N couples entrée-sortie. On le divises en K ensembles d'environ la même taille. On peut produire K estimateurs, de risque réel associé estimé sur le $k^{\text{ième}}$ ensemble pour un entraînement sur les $K-1$ autres. L'estimation finale du risque réel est la moyenne (éventuellement pondérée) des risques réels estimés de chacun des estimateurs.

Plus formellement, on introduit une fonction de partition

$$\kappa : \{1, \ldots, N\} \to \{1, \ldots, K\} \tag{11}$$

qui indique à quel ensemble k appartient le couple entrée-sortie d'indice i. Nous notons $\hat{f}^{-k}$ l'estimateur obtenu en enlevant le $k^{\text{ième}}$ ensemble des données, c'est-à-dire entraîné en utilisant la base d'exemples $\{(\mathbf{x}_i, y_i),\ i \notin \kappa^{-1}(k)\}$. L'estimation du risque donné par la validation croisée est alors

$$\hat{R}_{\text{cv}}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(\mathbf{x}_i)). \tag{12}$$

Cela revient à faire un apprentissage sur $K-1$ ensembles, estimer le risque sur le $k^{\text{ième}}$ ensemble, puis moyenner (avec les pondérations ad-hoc) les risques

obtenus pour chaque k. Une écriture équivalente est donc :

$$\hat{R}_{\text{cv}}(\hat{f}) = \sum_{k=1}^{K} \frac{|\kappa^{-1}(k)|}{N} R_k(\hat{f}) \text{ avec } R_k(\hat{f}) = \frac{1}{|\kappa^{-1}(k)|} \sum_{i \in \kappa^{-1}(k)} L(y_i, \hat{f}^{-k}(\mathbf{x}_i)). \tag{13}$$

Si l'on choisit des ensembles de la même taille, on a pour tout k l'égalité $|\kappa^{-1}(k)| = \frac{N}{K}$. Dans ce cas, on retrouve une expression peut-être plus usuelle du risque estimé :

$$\hat{R}_{\text{cv}}(\hat{f}) = \frac{1}{K} \sum_{k=1}^{K} R_k(\hat{f}). \tag{14}$$

Comme auparavant, la validation croisée peut servir à choisir la valeur d'un méta-paramètre :

$$\hat{\alpha} = \operatorname*{argmin}_{\alpha} \hat{R}_{\text{cv}}(\hat{f}_\alpha). \tag{15}$$

A nouveau, il faudra un ensemble de test pour évaluer le risque de l'estimateur $\hat{f}_{\hat{\alpha}}$.

Le choix du nombre de partitions est ouvert. Des choix typiques sont $K = 5$ ou $K = 10$. Le choix $K = N$ est appelé *leave-one-out cross-validation*. Ce dernier cas est approximativement non biaisé par rapport au vrai risque, mais il peut avoir une très forte variance (car les N ensembles d'apprentissage sont très semblables). Le coût computationnel peut aussi être très important (N phases d'apprentissage sur des ensembles de taille $N-1$). Toutefois, dans certains cas (par exemple moindres carrés linéaire), cette estimation du risque peut être calculée analytiquement. Soient

$$\mathbf{y} = \begin{pmatrix} y_1 & \dots & y_N \end{pmatrix}^T \in \mathbb{R}^N \text{ et } \mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \dots & x_N \end{bmatrix}^T \in \mathbb{R}^{N \times d} \tag{16}$$

le vecteur de sorties et la matrice d'entrée respectivement. Une méthode de prédiction linéaire est une méthode pour laquelle on cherche une prédiction $\hat{\mathbf{y}}$ des observations $\mathbf{y}$ sous la forme

$$\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}. \tag{17}$$

Par exemple, pour les moindres carrés linéaires, on a

$$\mathbf{S} = \mathbf{X} \left(\mathbf{X}^T \mathbf{X} \right)^{-1} \mathbf{X}^T. \tag{18}$$

Pour beaucoup de modèles linéaires, on peut montrer [5] que (en supposant κ égale à l'identité, étant donné le contexte) :

$$\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{f}^{-i}(\mathbf{x}_i))^2 = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{y_i - \hat{f}(\mathbf{x}_i)}{1 - \mathbf{S}_{ii}} \right)^2, \tag{19}$$

5. Par exemple, pour les moindres carrés, on peut passer par Sherman-Morrison.

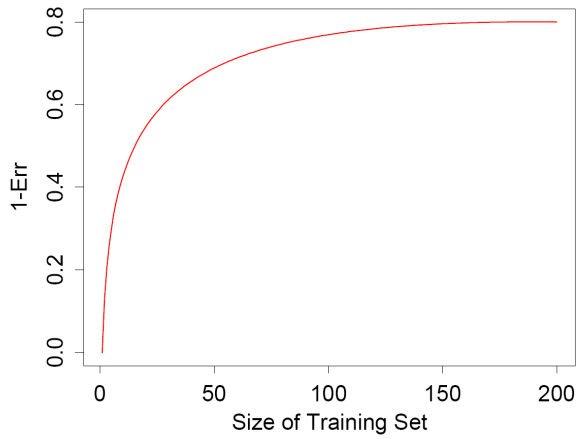

FIGURE 1 – Illustration pour la validation croisée [17]. L'abscisse est la taille de l'ensemble d'entraînement et l'ordonnée est 1 moins le risque réel.

où $\mathbf{S}_{ii}$ est le $i^{\text{ème}}$ élément diagonal de $\mathbf{S}$. Cela inspire l'estimateur de validation croisée généralisée [14, 28], donné par :

$$\hat{R}_{gcv}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \hat{f}(\mathbf{x}_i)}{1 - \frac{\text{trace}\,(\mathbf{S})}{N}}\right)^2. \tag{20}$$

La quantité trace $(\mathbf{S})$ peut s'interpréter comme étant le nombre effectif de paramètres. Notamment, si $\mathbf{S}$ est une projection orthogonale sur un espace d'hypothèses engendré par M fonctions de base, alors trace $(\mathbf{S}) = M$. Toutefois, on se restraint ici aux estimateurs linéaires (et à une perte quadratique), ce qui est moins général que la validation croisée.

Discutons un peu plus le choix du nombre d'ensembles K, hors du cas extrême $K = N$. Pour cela, considérons la figure 1, qui représente $1 - R(\hat{f}_N)$ en fonction de N, en supposant le risque réel majoré par 1 pour une certaine méthode d'apprentissage produisant des estimés $\hat{f}_N$. On voit que la pente de cette courbe est forte entre 0 et 100, faible après (les 100 premiers échantillons réduisent plus le risque que les 100 suivants). Considérons que l'on a 200 observation, et que l'on fait de la validation croisée avec $K = 5$. L'entraînement se fait donc sur des bases de taille 160, le comportement est donc quasiment le même que pour la base complète. Toutefois, si nous n'avons que 50 observations, choisir $K = 5$ impose un entraînement avec 40 échantillons, la figure montre bien que l'on va sous-estimer $1 - R(\hat{f})$. Le risque sera donc sur-évalué. Globalement, il est difficile de tenir ce genre de raisonnement en pratique (nous le présentons pour son intérêt pédagogique), et généralement $K = 5$ ou $K = 10$ sont considérés comme des bon compromis [7, 18].

La remarque que nous avons faite sur la séparation de la base d'exemples en une base d'apprentissage, une base de validation et une base de test reste vraie pour la validation croisée à K plis (où le résultat sera cascadé). On sépare la base d'exemples en K_1 plis. Un duo apprentissage et validation est toute la base privée du $k^{\text{ième}}$ pli (qui est un pli de test). On peut ensuite séparer l'ensemble de ces $K_1 - 1$ plis en K_2 plis, afin d'avoir une base d'entraînement et une autre de validation.

2.3 Du bon usage de la validation croisée

Supposons que l'algorithme d'apprentissage ne soit qu'un élément de la chaîne de traitement. La validation croisée doit se faire sur l'ensemble de la chaîne, c'est-à-dire que les données doivent être séparées dès le départ.

Illustrons cela par un exemple. Supposons avoir à traiter un problème de régression avec un grand nombre de variables, et disposer d'une heuristique permettant de choisir ces variables en fonction de leur corrélation avec les sorties associées (nous verrons un tel exemple section 3). Une stratégie expérimentale pourrait être :

1. trouver un bon sous-ensemble de variables en utilisant l'heuristique ;
2. en utilisant uniquement ces variables, construire un régresseur ;
3. utiliser la validation croisée pour choisir un méta-paramètre.

Ceci *n'est pas* la bonne approche pour faire de la validation croisée. En effet, l'étape 1 utilise tous les exemples de sortie. Si l'on sépare la base d'entraînement après avoir sélectionné un sous-ensemble de variables, le fait que ces variables aient été choisis en fonction des sorties associées fait que l'estimateur $\hat{f}$ *ne sera pas* indépendant des échantillons de test. En pratique, on aura tendance à sous-estimer le risque réel. La procédure correcte est la suivante :

1. séparer la base d'exemples en K plis ;
2. pour chaque plis $k = 1 \dots K$:
   (a) trouver un bon sous-ensemble de variables en utilisant l'heuristique appliquée à la base d'exemples privée du $k^{\text{ième}}$ plis,
   (b) utiliser ces variables et la base privée du $k^{\text{ième}}$ plis pour calculer le régresseur,
   (c) utiliser les $k^{\text{ième}}$ plis pour le choix du méta-paramètre.

Ainsi, la validation croisée doit s'effectuer sur l'ensemble de la chaîne de traitement. Une exception notable est les prétraitements non-supervisés : ils peuvent être effectués sans séparer les données au préalables (dans la mesure où ces traitements se font indépendamment des sorties y_i, on n'introduit

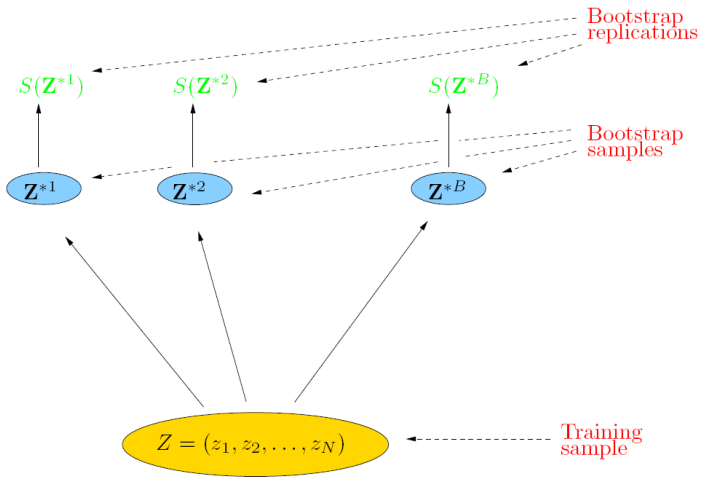

FIGURE 2 – Illustration pour le bootstrap [17].

pas de biais). Tout cela peut paraître évident, mais ce sont des erreurs qui apparaissent souvent dans la littérature, voir [3] pour une discussion.

2.4 Bootstrapping

En rappelant que nous notons $z_i = (\mathbf{x}_i, y_i)$, nous avons une base d'exemples $\mathcal{Z} = \{z_1, \ldots, z_N\}$. L'idée du bootstrap est de générer B ensembles $\mathcal{Z}^{*b}$ de taille N à partir de $\mathcal{Z}$, en faisant un tirage aléatoire uniforme avec remise. L'apprentissage est ensuite effectué sur chacun des B ensembles bootstrappés, et on examine le comportement de l'apprentissage obtenus sur les B répliques.

De façon générale, soit $g(\mathcal{Z})$ n'importe quelle quantité calculée à partir des données de $\mathcal{Z}$ (typiquement, le risque). On peut utiliser les répliques de bootstrap pour calculer n'importe quel moment de $g(\mathcal{Z})$, comme la moyenne par exemple :

$$\hat{\mathbb{E}}[S(\mathcal{Z})] = \frac{1}{B}\sum_{b=1}^{B} S(\mathcal{Z}^{*b}). \tag{21}$$

L'idée générale est illustrée sur la figure 2.

C'est le risque réel qui nous intéresse ici. Si l'on note $\hat{f}^{*b}$ l'estimateur calculé sur $\mathcal{Z}^{*b}$, une estimation naturelle du risque est :

$$\hat{R}_{\text{boot}}(\hat{f}) = \frac{1}{B}\sum_{b=1}^{B}\left(\frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}^{*b}(\mathbf{x}_i))\right). \tag{22}$$

Ainsi, l'ensemble bootstrappé $\mathcal{Z}^{*b}$ joue le rôle de base d'entraînement et l'ensemble d'exemples original $\mathcal{Z}$ joue le rôle de base de test. Il y un recouvrement important entre les deux ensembles, l'estimateur $\hat{R}_{\text{boot}}$ n'est donc a priori pas bon (il y a des problèmes de dépendance, c'est pour cela que les ensembles ne se recouvrent pas en validation croisée) : le risque réel sera sous-estimé.

Une possibilité est d'imiter la validation croisée. Soit $1 \leq i \leq N$, notons C^{-i} l'ensemble des indices b tels que $z_i \notin \mathcal{Z}^{*b}$. Un estimateur "*leave-one-out* bootstrappé" du risque est défini par :

$$\hat{R}^{(1)}(\hat{f}) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{|C^{-i}|} \sum_{b \in C^{-i}} L(y_i, \hat{f}^{*b}(\mathbf{x}_i)). \tag{23}$$

Cela suppose que pour tout i, on ait $C^{-i} \neq \emptyset$. Plus généralement, on peut introduire l'ensemble $D = \{i : 1 \leq i \leq N, C^{-i} \neq \emptyset\}$. On a alors :

$$\hat{R}^{(1)}(\hat{f}) = \frac{1}{|D|} \sum_{i \in D} \frac{1}{|C^{-i}|} \sum_{b \in C^{-i}} L(y_i, \hat{f}^{*b}(\mathbf{x}_i)). \tag{24}$$

Il n'y a plus de problème de sur-apprentissage comme pour l'estimateur R_{boot}, mais on peut montrer que cet estimateur $\hat{R}^{(1)}$ se comporte grossièrement comme la validation croisée à $K = 2$ plis [17]. En se rappelant la figure 1, on peut voir qu'à moins d'avoir beaucoup de données (cas où le bootstrap n'est a priori pas le plus intéressant), il y aura un biais dans l'estimation (la pente de la courbe étant typiquement forte en $\frac{N}{2}$, l'estimation du risque empirique sera sensiblement différente, en moyenne, si l'on considère N exemples ou $\frac{N}{2}$).

L'estimateur ".632" a été proposé pour réduire ce biais. Il est défini par :

$$\hat{R}^{(.632)}(\hat{f}) = 0,368 R_N(\hat{f}) + 0,632 \hat{R}^{(1)}(\hat{f}) \tag{25}$$

où nous rappelons que R_N est le risque empirique (sur la base d'entraînement). L'obtention de cet estimateur est complexe [9]. Nous notons simplement que le terme $0,632$ est environ la probabilité que l'observation i appartienne au $b^{\text{ième}}$ ensemble boostrappé [6].

Dans le cas où il y a du sur-apprentissage, le risque empirique va typiquement être nul et l'estimateur .632 sera à nouveau biaisé. Pour résoudre ce problème, l'estimateur ".632+" a été proposé [12]. Soit γ le taux d'erreur à information zéro, estimé par

$$\hat{\gamma}(\hat{f}) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} L(y_i, \hat{f}(\mathbf{x}_j)). \tag{26}$$

6. Nous avons $\mathbb{P}(z_i \in \mathcal{Z}^{*b}) = 1 - (1 - \frac{1}{N})^N \approx 1 - \exp^{-1} \approx 0,632$.

Le taux relatif de sur-apprentissage est alors défini par

$$\hat{r} = \frac{\hat{R}^{(1)} - R_N}{\hat{\gamma} - R_N} \in [0, 1]. \tag{27}$$

Ce coefficient varie de zéro dans le cas où il n'y a pas de sur-apprentissage ($\hat{R}^{(1)} = R_N$) à un lorsque le sur-apprentissage atteint la valeur de l'information zéro ($\hat{\gamma} - R_N$). Finalement, l'estimateur .632+ est défini par

$$\hat{R}^{(.632+)} = (1 - \hat{\omega})R_N + \hat{\omega}\hat{R}^{(1)} \text{ avec } \hat{\omega} = \frac{0,632}{1 - 0,368\hat{r}}. \tag{28}$$

C'est l'estimateur de risque le plus sophistiqué que nous verrons dans ce document.

3 Sélection de variables

Cette section fait une présentation rapide du problème de sélection de variables, tel que défini dans la section 1.2. Il existe quelques revues de l'état de l'art sur le sujet, notamment [15, 22]. Il existe également différentes librairies, notamment `fst3` [1] qui est en `c++` et basée sur des templates et la librairie `boost`.

3.1 Taxinomie

On distingue principalement trois approches pour la sélection de variables : les *filters*, les *wrappers* et les *embedded*. Pour les deux premières approches, le principe est d'ajouter et/ou de supprimer des variables (nous verrons un exemple de comment dans la section 3.2) selon un certain critère qui est une heuristique pour les *filters* (typiquement une mesure de corrélation ou d'information mutuelle) et une estimation du risque réel de l'estimateur pour les *wrappers* (ce qui suppose de lancer de nombreux apprentissages). Les *filters* sont généralement plus légers que les *wrappers* (car ne nécessitant pas d'apprentissage), mais moins fondés théoriquement car basés sur des heuristiques. Les deux approches ont leurs défenseurs, mais cela dépend sensiblement du domaine applicatif (et a fortiori du nombre de variables à traiter). Les *embeddeds*, quant à eux, ajoutent un terme de pénalisation de la complexité au risque empirique minimisé. LASSO (*Least Absolute Shrinkage and Selection Operator*) [26] en est peut-être l'un des représentants les plus célèbres. Il consiste à ajouter un terme de pénalisation ℓ_1 à un moindres carrés linéaire (pour lequel le modèle de prédiction est linéaire, $\hat{f}(\mathbf{x}_i) = \theta^T \mathbf{x}_i$) :

$$\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} (y_i - \theta^T \mathbf{x}_i)^2 + \lambda \|\theta\|_1. \tag{29}$$

La régularisation ℓ_1 va avoir tendance à promouvoir la parcimonie des solutions (c'est-à-dire à mettre exactement à 0 certains paramètres), et donc à faire de la sélection de variables. Il y a une vaste littérature sur le sujet, mais ces méthodes étant moins génériques, nous ne les développerons pas plus.

3.2 Recherche dans l'espace des variables

Nous présentons ici quelques méthodes classiques de sélection de variables, à savoir SFS (*Sequential Forward Selection*), SBS (*Sequential Backward Selection*) et SFFS (*Sequential Forward Floating Selection*). Un état de l'art d'approches plus récentes est fait par [22].

Rappelons et introduisons quelques notations. Les entrées sont de la forme $\mathbf{x} \in \mathbb{R}^d$. Une variable est l'une de ses d composantes, que l'on notera x_j :

$$\mathbf{x} = \begin{pmatrix} x_1 & \dots x_j & \dots x_d \end{pmatrix}^T. \tag{30}$$

Nous notons $\mathcal{X}_{\sigma,k}$ un ensemble de k variables parmi les d possibles :

$$\mathcal{X}_{\sigma,k} = \Big\{ x_j : j \in \sigma^{-1}(1), \sigma \in \{0,1\}^{\{1\dots d\}} \text{ et } |\sigma^{-1}(1)| = k \Big\}. \tag{31}$$

L'application σ associe donc à chaque variable initiale une valeur booléenne qui indique si elle appartient au nouveau sous-ensemble.

Il y a donc $\frac{d!}{k!(d-k)!}$ tels sous-ensembles possibles. Nous supposons disposer d'une fonction de score J qui évalue la qualité $J(\mathcal{X}_{\sigma,k})$ des k variables sélectionnées (typiquement, cette qualité sera évaluée en utilisant tout ou partie de la base d'entraînement, voir la section 2.3 ; le choix de cette fonction de score fait l'objet de la section 3.3). Définissons un certain nombre de quantités :

– l'*importance individuelle* S_0 définie par

$$S_0(x_i) = J(x_i),\ 1 \le i \le d, \tag{32}$$

c'est l'importance d'une seule variable ;

– l'*importance* $S_{\sigma,k-1}(x_j)$ de la variable $x_j \in \mathcal{X}_{\sigma,k}$ définie par :

$$S_{\sigma,k-1}(x_j) = J(\mathcal{X}_{\sigma,k}) - J(\mathcal{X}_{\sigma,k} \setminus \{x_j\}); \tag{33}$$

– l'*importance* $S_{\sigma,k+1}(x_j)$ de la variable $x_j \in \mathcal{X}_d \setminus \mathcal{X}_{\sigma,k}$ définie par [7] :

$$S_{\sigma,k+1}(x_j) = J(\mathcal{X}_{\sigma,k} \cup \{x_j\}) - J(\mathcal{X}_{\sigma,k}); \tag{34}$$

7. Nous notons $\mathcal{X}_{\sigma,d} = \mathcal{X}_d$ car il n'y a qu'un seul tel ensemble.

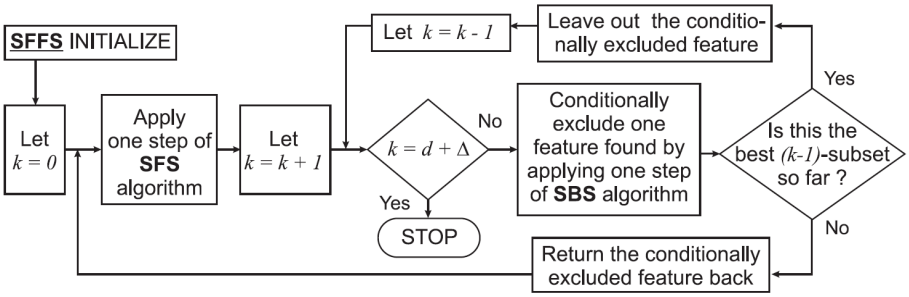

FIGURE 3 – Illustration schématique de SFFS [21, 22].

– une variable x_j de $\mathcal{X}_{\sigma,k}$ est :
  – la *variable la plus importante* de l'ensemble $\mathcal{X}_{\sigma,k}$ si

$$S_{\sigma,k-1}(x_j) = \max_{x_i \in \mathcal{X}_{\sigma,k}} S_{\sigma,k-1}(x_i), \tag{35}$$

  – la *variable la moins importante* de l'ensemble $\mathcal{X}_{\sigma,k}$ si :

$$S_{\sigma,k-1}(x_j) = \min_{x_i \in \mathcal{X}_{\sigma,k}} S_{\sigma,k-1}(x_i), \tag{36}$$

– une variable x_j de $\mathcal{X}_d \setminus \mathcal{X}_{\sigma,k}$ est :
  – la *variable la plus importante* respectivement à l'ensemble $\mathcal{X}_{\sigma,k}$ si

$$S_{\sigma,k+1}(x_j) = \max_{x_i \in \mathcal{X}_d \setminus \mathcal{X}_{\sigma,k}} S_{\sigma,k+1}(x_i), \tag{37}$$

  – la *variable la moins importante* respectivement à l'ensemble $\mathcal{X}_{\sigma,k}$ si :

$$S_{\sigma,k+1}(x_j) = \min_{x_i \in \mathcal{X}_d \setminus \mathcal{X}_{\sigma,k}} S_{\sigma,k+1}(x_i). \tag{38}$$

Ces notions étant posées, l'algorithme SFS (*Sequential Forward Search*) consiste à partir d'un ensemble vide puis à ajouter de façon gloutonne à chaque étape la variable la plus importante respectivement à l'ensemble de l'étape précédente, ce jusqu'à ce que le nombre désiré de variables soit atteint (par exemple, d'autres critères peuvent être considérés). L'algorithme SBS (*Sequential Backward Search*) part de l'ensemble $\mathcal{X}_d$ contenant toutes les variables et supprime de façon gloutonne à chaque étape la variable la moins importante par rapport à l'ensemble de l'étape précédent. Ce sont les approches les plus classiques.

Toutefois, si l'on considère SFS par exemple, une variable que l'on vient d'ajouter peut rendre obsolète une variable ajoutée dans les premiers pas.

De façon plus générale, il n'y aucune raison que le meilleur de tous les sous-ensembles possibles par rapport à la fonction de score ($\text{argmin}_{\sigma,k} J(\mathcal{X}_{\sigma,k})$, dont la recherche est un problème NP-complet) soit obtenu lors d'une des itérations. Pour palier à ce problème, l'algorithme SFFS (*Sequential Floating Forward Search*) a été introduit [21]. Nous ne le détaillons pas, il est résumé figure 3, mais l'idée générale est d'alterner les étapes où on ajoute des variables (en utilisant SFS) et ou on en enlève (en utilisant SBS). Cet algorithme a été étendu [23] et a donné lieu à de nombreux descendants [22].

3.3 *Filter, Wrapper*

La fonction de score J peut être définie selon deux approches : *filter* ou *wrapper*. Comme nous l'avons déjà dit, les *filters* sont des heuristiques tandis que les *wrappers* utilisent une estimation du risque empirique pour J.

Considérons un exemple usuel de *filter*, l'heuristique CSF (*Correlation-based feature selection*) [16]. L'idée sous-jacente est que les variables doivent être fortement corrélées avec les sorties, mais peu entre elles. Supposons disposer de la base d'apprentissage $\{(\mathbf{x}_i, y_i)_{1\leq i\leq N}\}$ (donc *après* séparation en bases d'apprentissage/validation/test). L'heuristique est donnée par :

$$J(\mathcal{X}_{\sigma,k}) = \frac{k\bar{r}(\mathcal{X}_{\sigma,k}, y)}{\sqrt{k(k-1)\bar{r}(\mathcal{X}_{\sigma,k}, \mathcal{X}_{\sigma,k})}}, \tag{39}$$

$$\text{avec } \bar{r}(\mathcal{X}_{\sigma,k}, y) = \frac{1}{k}\sum_{i\in\sigma^{-1}(1)} r(x_i, y) \tag{40}$$

$$\text{et } \bar{r}(\mathcal{X}_{\sigma,k}, \mathcal{X}_{\sigma,k}) = \frac{1}{k(k-1)}\sum_{i,j\in\sigma^{-1}(1), i\neq j} r(x_i, x_j), \tag{41}$$

où r est une mesure de corrélation, par exemple

$$r(x_i, y) = \frac{\frac{1}{N}\sum_{n=1}^{N}(x_i)_n y_n}{(\frac{1}{N}\sum_{n=1}^{N}(x_i)_n^2)(\frac{1}{N}\sum_{n=1}^{N} y_n^2)} \tag{42}$$

$$\text{et } r(x_i, x_j) = \frac{\frac{1}{N}\sum_{n=1}^{N}(x_i)_n (x_j)_n}{(\frac{1}{N}\sum_{n=1}^{N}(x_i)_n^2)(\frac{1}{N}\sum_{n=1}^{N}(x_j)_n^2)}, \tag{43}$$

où $(x_i)_n$ est la $i^{\text{ème}}$ composante de l'entrée $\mathbf{x}_n$. D'autres mesures de corrélation sont possibles, comme l'information mutuelle par exemple [16]. Il existe bien sûr d'autres heuristiques *filter*, dont un certain nombre est recensé par [30].

Les *wrappers* [19], quant à eux, consistent à utiliser une estimation du risque réel pour l'estimateur produit par un algorithme donné en n'utilisant que les variables données par $\mathcal{X}_{\sigma,k}$. Notons $\mathbf{x}^\sigma \in \mathbb{R}^k$ le vecteur dont l'on ne

retient que les composantes décrites par $\mathcal{X}_{\sigma,k}$. Supposons disposer d'une base d'entraînement $\{(\mathbf{x}_i, y_i)_{1\leq i\leq N}\}$ et d'une base de validation $\{(\mathbf{x}'_i, y'_i)_{1\leq i\leq M}\}$. Soit $\hat{f}$ l'estimateur obtenu par un algorithme d'apprentissage (typiquement celui pour lequel on cherche à sélectionner les variables) sur la base d'exemples $\{(\mathbf{x}_i^\sigma, y_i)_{1\leq i\leq N}\}$. Le score de l'ensemble $\mathcal{X}_{\sigma,k}$ peut alors être donné par :

$$J(\mathcal{X}_{\sigma,k}) = \frac{1}{M}\sum_{i=1}^{M} L(y'_i, \hat{f}(\mathbf{x}_i'^\sigma)), \tag{44}$$

pour une certaine perte L (généralement fortement liée à l'algorithme considéré). Bien sûr, tout autre estimateur du risque, comme la validation croisée à K plis ou le bootstrapping, peut être envisagé.

3.4 *Embedded*

Les méthodes dite *embedded* effectuent la sélection de variables durant l'apprentissage et sont généralement algorithme-spécifique. Par exemple, les machines à vecteur support (SVM pour *Support Vector Machine*) [27] peuvent être considérées comme une telle approche, dans la mesure où elles choisissent les vecteurs supports parmi la base d'exemples (dans ce cas, ce sont moins les variables que les fonctions de base que l'on choisit). LASSO [26] est un autre exemple que nous avons déjà mentionné, on pourrait citer tous ses cousins comme LARS [10] ou *elastic net* [31], entre autres. Il est possible de remonter plus loin : les arbres de décision comme CART [6] ont des mécanismes internes de sélection de variables.

On peut mentionner une autre approche, plutôt catégorisée dans la sélection de modèle (dans le sens où elle sélectionne l'espace d'hypothèse, ce qui peut se confondre avec la sélection de variables lorsque la paramétrisation est linéaire), parfois appelée *complexity regularization* (ce terme étant parfois utilisé dans un cadre plus général), qui consiste à ajouter un terme de pénalisation au risque empirique minimisé, la particularité étant que ce terme de pénalisation dépend également des données (voir par exemple [4, 5, 29, 20]). La motivation est d'obtenir des bornes du type Vapnik-Chervonenkis (voir l'équation (5)), mais dépendant de la distribution et donc plus fine (afin d'avoir une meilleure estimation du risque pour la sélection de modèle, avec des garanties théoriques). Ce type d'approche est assez complexe et mathématique (on peut grossièrement les lier à la minimisation structurelle du risque de Vapnik [27]), et ne présente a priori pas la généricité des approches précédemment mentionnées.

4 Sélection de modèle

On peut voir le modèle comme une chaîne de traitement avec en entrée des données, en sortie un estimateur, et composée de blocs qui sont :

- soit des traitements non-supervisés ;
- soit des procédures de sélection de variable (sous-entendu supervisées) ;
- soit un algorithme d'apprentissage.

Pour sélectionner un modèle parmi plusieurs en utilisant la validation croisée, il faut donc comme nous l'avons vu section 2 séparer de la base d'entraînement une base de test avant tout traitement supervisé (le reste de la base pouvant être à son tour séparer en une base d'entraînement et une autre de validation), puis évaluer le risque en bout de chaîne. Le même principe peut s'appliquer si l'on choisit la validation croisée à K plis ou le bootstrap (dans ce cas, on veillera à bootstrapper les nouvelles bases d'exemples préalablement à tout traitement supervisé). On veillera finalement à respecter les principes énoncés tout au long de ce document, notamment ceux de la section 2.3.

Rappelons que les approches de *complexity regularization* brièvement abordées fin de section 3.4 sont mentionnées dans la littérature comme étant des algorithmes de selection de modèle. C'est un peu différent de ce que nous considérons ici, dans la mesure où il s'agit de choisir un espace d'hypothèses (mais notons que la finalité est la même). Encore une fois, ces approches semblent moins génériques et assez complexes (offrant toutefois des garanties théoriques intéressantes).

Références

[1] Feature selection toolbox (fst3 library). `http://fst.utia.cz/`.

[2] D. Allen. The relationship between variable selection and data augmentation and a method of prediction. *Technometrics*, 1974.

[3] C. Ambroise and G. McLachlan. Selection bias in gene extraction on the basis of microarray gene-expression data. In *Proc. of the National Academy of Science*, volume 99, pages 6562–6566, 2002.

[4] A.R. Barron. Complexity regularization with application to artificial neural networks. In G. Roussas, editor, *Nonparametric Functional Estimation and Related Topics*, pages 561–576. Kluwer Academic Publishers, 1991.

[5] P.L. Bartlett, S. Boucheron, and G. Lugosi. Model selection and error estimation. *Machine Learning*, 48 :85–113, 2002.

[6] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, 1984.

[7] L. Breiman and P. Spector. Submodel selection and evaluation in regression : the X-random case. *International Statistical Review*, 60 :291–319, 1992.

[8] B. Efron. Bootstrap methods : another look at the jackknife. *Annals of Statistics*, 7(1-26), 1979.

[9] B. Efron. Estimating the error rate of a prediction rule : some improvements on cross-validation. *Journal of the American Statistical Association*, 78 :316–331, 1983.

[10] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least Angle Regression. *Annals of Statistics*, 32(2) :407–499, 2004.

[11] B. Efron and R. Tibshirani. *An Introduction to the Bootstrap*. Chapman and Hall, London, 1993.

[12] B. Efron and R. Tibshirani. Improvements on cross-validation : the .632+ bootstrap method. *Journal of the American Statistical Association*, 92 :548–560, 1997.

[13] M. Geist. Précis introductif à l'apprentissage statistique. Support de cours, Supélec, 2011. (disponible sur la page web de l'auteur).

[14] G. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21 :215–224, 1979.

[15] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3 :1157–1182, 2003.

[16] M. Hall. *Correlation-based Feature Selection for Machine Learning*. PhD thesis, University of Waikato (New Zealand), 1999.

[17] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[18] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In Morgan Kaufmann, editor, *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1137–1143, 1995.

[19] R. Kohavi and G. John. Wrappers for feature selection. *Artificial Intelligence*, 97(1-2) :273–324, 1997.

[20] G. Lugosi and M. Wegkamp. Complexity regularization via localized random penalties. *The Annals of Statistics*, 32(4) :1679–1697, 2004.

[21] P. Pudil, J. Novovicova, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15 :1119–1125, 1994.

[22] P. Somol, J. Novovicova, and P. Pudil. *Pattern Recognition Recent Advances*, chapter Efficient Feature Subset Selection and Subset Size Optimization. InTech, 2010.

[23] P. Somol, P. Pudil, J. Novovicova, and P. Paclik. Adaptive floating search methods in feature selection. *Pattern Recognition Letters*, 20 :1157–1163, 1999.

[24] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society Series B*, 36 :111–157, 1974.

[25] M. Stone. An asymptotic equivalence of choice of model by crossvalidation and akaike's criterion. *Journal of the Royal Statistical Society Series B*, 1977.

[26] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1) :267–288, 1996.

[27] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, September 1998.

[28] G. Wahba. Spline bases, regularization and generalized cross-validation for solving approximation problems with large quantities fo noisy data. In Academic Press, editor, *Proc. of the International Conference on Approximation theory in Honour of George Lorenz*, pages 905–912, 1980.

[29] M. Wegkamp. Model selection in nonparametric regression. *The Annals of Statistics*, 31(1) :252–273, 2003.

[30] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, and H. Liu. Advancing Feature Selection Research – ASU Feature Selection Repository. Technical report, Arizona State University, 2008.

[31] H. Zou and T. Hastie. Regularization and variable selection via the Elastic Net. *Journal of the Royal Statistical Society B*, 67(2) :301–320, 2005.