L'apprentissage face à la malédiction de la grande dimension

Collège de France
Cours 2: **Dilemme Biais-Complexité**
de Stéphane Mallat

Notes de John Zarka et Stéphane Mallat

1 Compléments au cours précédent

1.1 Lien entre intelligence artificielle et sciences des données

L'*intelligence artificielle* ('IA') est un terme mal défini mais de nature essentiellement anthropomorphique. Il recouvre au sens large toutes les théories et techniques permettant à des machines d'imiter l'intelligence humaine, qui est elle-même mal définie. Ainsi on ne considère pas qu'une calculette fait de l'intelligence artificielle bien qu'elle puisse effectuer des opérations arithmétiques que seul l'homme est capable de faire.

Le fort développement de l'intelligence artificielle à partir des années 70 et jusqu'aux années 90 s'est ainsi principalement consacré à émuler le fonctionnement du cerveau *conscient*. La recherche s'articulait alors autour des notions de logique, à partir de représentations symbolique et de règles débouchant par exemple sur l'implémentation de *systèmes expert* avec des langages de programmation logique type Prolog.

L'essor de l'intelligence artificielle depuis 10 ans suit une démarche différente, visant à imiter des fonctions cognitives humaines beaucoup plus proche du cerveau *inconscient*, comme la perception. Elle se développe ainsi principalement autour des *sciences des données* au premier lieu desquelles l'apprentissage statistique et la prédiction. Cela met en jeu des outils mathématiques provenant des statistiques, des probabilités, de l'analyse et la géométrie, qui sont différents de ceux de la première période de l'IA. Il ne s'agit pas de comprendre ou d'expliquer les phénomènes que l'on prédit.

Un des enjeux de l'intelligence artificielle et un de ses axes de développement futur sera probablement de faire la correspondance entre ces deux approches, correspondant grossièrement au cerveaux *conscient* et *inconscient*, afin d'expliquer avec des modèles explicites les résultats obtenus par l'apprentissage statistique. Ces résultats sont difficiles à expliquer car ils résultent de l'agrégation de milliers d'indicateurs faibles, ce qui n'est pas compatible avec une explication qui puisse se résumer en quelques points. Pourtant, l'explication est nécessaire dans de nombreuses applications, par exemple afin de permettre un droit de recours. Ainsi, des plaintes ont déjà été déposées aux États Unis contre l'utilisation d'un outil d'apprentissage statistique dans des prises de décisions concernant la libération de prisonniers, sachant qu'aucune explication de l'indicateur n'est fournie.

Le cours va lui être essentiellement centré sur les sciences des données dont j'ai donné une cartographie lors du cours précédent et qui sont à l'origine du récent renouveau de l'intelligence artificielle.

1.2 Notes de cours

Les notes de cours seront disponibles à l'adresse suivante : www.di.ens.fr/~mallat/CoursCollege.html

2 Apprentissage supervisé

2.1 Algorithme d'apprentissage

Je vais commenccer en précisant le cadre formel et les objectifs de l'*apprentissage supervisé*. Les challenges du Challenge Data sont des exemples d'apprentissages supervisés. Ils sont disponibles à l'adresse : https://challengedata.ens.fr/.

On note $y \in \mathcal{A}$ (où $\mathcal{A}$ est continu en régression et discret en classification) la réponse ou *label* à une question posée sur $x \in \mathbb{R}^d$ (où x peut représenter un image, une série temporelle, etc.).

On suppose que l'on dispose de n données d'entrainement $\{(x_i, y_i)\}_{i \leq n}$ composées de n entrées $x_i \in \mathbb{R}^d$ et n réponses $y_i \in \mathcal{A}$. Ces données d'entraînement sont des réalisations de n couples aléatoires $\{(X_i, Y_i)\}_{i \leq n}$ d'échantillons indépendants et identiquement distribués des données de même lois que (X, Y). On notera (x, y) une réalisation particulière des données.

Un algorithme d'apprentissage prend en entrée une donnée x à partir duquel il prédit une approximation $\tilde{y}$ de la réponse y. Un tel algorithme inclue des paramètres internes qui sont optimisés grâce à n exemples d'entraînement (x_i, y_i) de façon à ce que la prédiction soit bonne sur ces exemples : $\tilde{y}_i \approx y_i$. Le but est que la précision de la prédiction se généralise à d'autres données x de "même type" que les exemples : on parle alors d'erreur de *généralisation*. L'algorithme d'apprentissage supervisé calcule $\tilde{y} = \tilde{f}(x)$ avec une fonction $\tilde{f}$ séléctionnée parmi une classe $\mathcal{H}$ de fonctions possibles. On suppose généralement que la réponse y associée à x est unique, auquel cas $y = f(x)$ pour une certaine fonction f. L'algorithme calcule donc une approximation $\tilde{f}$ de f.

2.2 Apprentissage linéaire

Plaçons nous maintenant dans un cas de classification binaire, c'est à dire $y \in \mathcal{A} = \{-1, 1\}$.

Un algorithme d'apprentissage séléctionne $\tilde{f}$ au sein d'une famille $\mathcal{H}$ de fonctions à valeur dans $\{-1, 1\}$. Un exemple de telle famille $\mathcal{H}$ est la famille des classificateurs *linéaires*

$$\mathcal{H} = \left\{ \tilde{f} : \mathbb{R}^d \to \{-1, 1\} \,|\, \tilde{f} : x \mapsto \text{signe}(\langle w, x \rangle - b) \right\}.$$

Elle est paramétrée par $w \in \mathbb{R}^d$ et $b \in \mathbb{R}$. Ces classificateurs linéaires définissent une frontière de décision entre les deux classes +1 et -1 qui est l'hyperplan $\left\{ x \in \mathbb{R}^d \,|\, \langle w, x \rangle = b \right\}$. Les entrées x sont représentées par des croix dans nos illustrations lorsqu'elles appartiennent à la classe 1 (i.e les réponses y associées valent 1), et par des ronds lorsque y lorsqu'elles appartiennent à la classe -1.

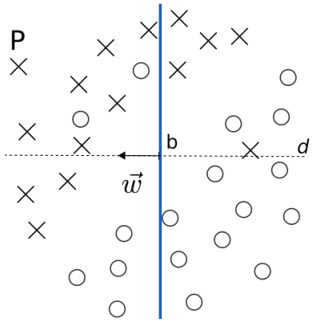

FIGURE 1 – Frontière linéaire entre deux classes

L'algorithme d'*apprentissage supervisé* consiste à déterminer une valeur "optimale" pour les paramètres w et b, où l'optimalité se définit au sens de la minimisation d'une *erreur* sur n données d'entrainement $\{(x_i, y_i)\}_{i \leq n}$. L'algorithme d'entrainement contient donc une procédure d'optimisation des paramètres afin de minimiser cette erreur.

Dans notre exemple en figure 1, les valeurs de w et b ne sont pas "optimales" pour les prédictions si l'on définit l'erreur à optimiser par le nombre de mauvaises classification des données fournies (représentées par les croix et les ronds). Il en existe en revanche plusieurs valeurs optimales pour ce risque comme illustré ci-dessous :

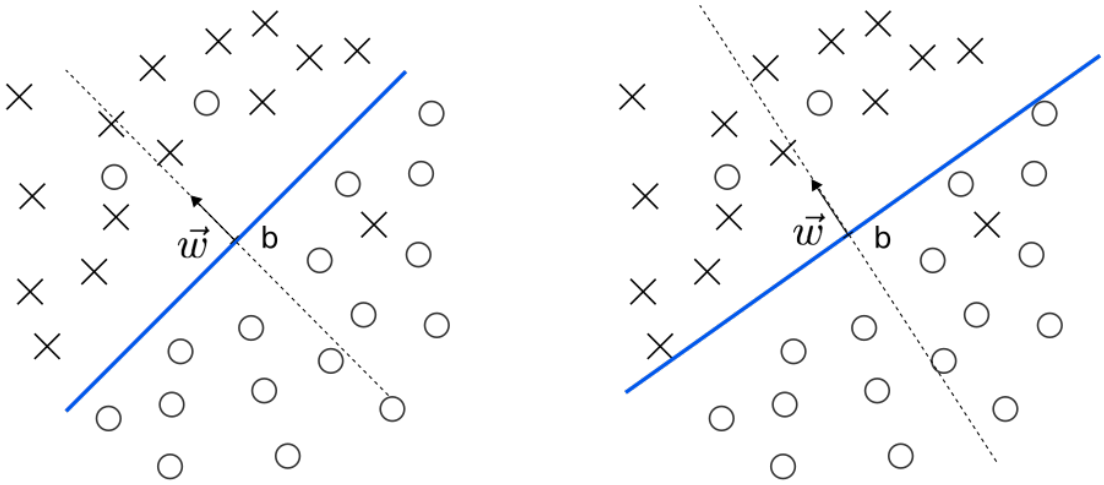

FIGURE 2 – Deux valeurs optimales pour les paramètres $\vec{w}$ et b

2.3 Changement de représentation

Nous avons décrit précédemment le fonctionnement d'un algorithme de classification *linéaire* dans le cadre d'un problème de classification binaire avec deux classes : 1 et -1. Il n'est pas toujours possible de séparer deux classes de données avec une frontière plane. La figure ci-dessous montre un exemple ou cela n'est pas possible.

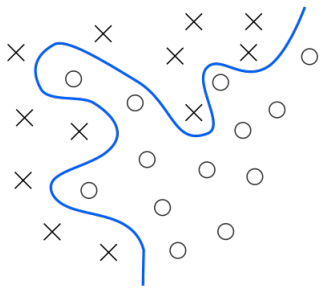

FIGURE 3 – Frontière courbe entre deux classes

Dans les algorithmes de classification (binaire), plutôt que d'utiliser une classe de fonctions $\mathcal{H}$ essayant d'adapter une frontière courbe directement sur les données, on va adapter les données avec un changement de variable $\phi(\cdot)$ menant x dans un espace d'arrivée de dimension m dans lequel on espère qu'on puisse séparer les classes linéairement par un hyperplan. On passe ainsi d'une frontière courbe dans l'espace de départ à une frontière "aplatie" entre les classes dans l'espace d'arrivée. La classe de fonctions $\mathcal{H}$ considérée par notre algorithme est donc l'ensemble des fonctions $\tilde{f}$ effectuant des prédictions de la forme

$$\tilde{y} = \tilde{f}(x) = \text{signe}(\langle w, \phi(x)\rangle - b) = \text{signe}(\sum_{k=1}^{m} w_k \phi(x)_k - b)$$

avec $w \in \mathbb{R}^m$, $b \in \mathbb{R}$. Le cas de la régression où $y \in \mathcal{A} = \mathbb{R}$ s'adapte simplement en considérant des fonctions $\tilde{f}$ effectuant des prédictions à partir d'une entrée x de la forme

$$\tilde{y} = \tilde{f}(x) = \langle w, \phi(x)\rangle - b = \sum_{k=1}^{m} w_k \phi(x)_k - b.$$

Restons ici dans le cadre de la classification binaire : tout le problème est d'optimiser ce changement de variable

$$\phi : x \in \mathbb{R}^d \mapsto \phi(x) = (\phi(x)_1, \ldots, \phi(x)_m) \in \mathbb{R}^m.$$

Si on connait le changement de variable, le classificateur est caractérisé par un hyperplan de vecteur normal w sur lequel $\phi(x)$ est projeté afin de déterminer la classe de x. Cet axe de projection est celui qui mène d'une classe à une autre classe. Chacun de ces $\phi(x)_k$ correspond à un *attribut* servant à construire un critère discriminant entre les classes à l'aide d'une combinaison linéaire pondérée par les w_k. Chaque attribut est un classificateur faible, comme par exemple la taille pour la classification homme / femme : la combinaison linéaire pondérée sert alors à obtenir un classificateur plus puissant. Les poids correspondent à la confiance que l'on accorde à chacun de ces critères et la classe prédite correspond à un vote majoritaire sur ces attributs pondérée. La difficulté principale consiste à trouver ces critères et donc le changement de variable ϕ. On peut soit le spécifier totalement *a priori*, notamment si l'on ne dispose pas de beaucoup de données, effectuer une sélection statistique des critères parmi un dictionnaire de critères (qui constitue également de l'information injectée *a priori*). Les algorithmes de boosting et les arbre de décision peuvent se voir dans ce cadre là.

Les paramètres de l'algorithme de classification linéaire sont w et b. L'optimisation de cet algorithme consiste alors à trouver les paramètres minimisant le risque ou erreur empirique afin d'obtenir une erreur de généralisation faible.

2.4 Risque

Un algorithme d'apprentissage supervisé est restreint à une classe de fonctions $\mathcal{H}$ que l'on choisit *a priori* - par exemple l'ensemble des classificateurs linéaires. L'objectif est de sélectionner $\tilde{f} \in \mathcal{H}$ avec les exemples entrainement $\{(x_i, y_i)\}_{i \leq n}$, de façon à ce que la prédiction $\tilde{y} = \tilde{f}(x)$ soit proche de la réponse y pour tout une catégorie de donnée $x \in \Omega$.

On quantifie l'erreur entre $\tilde{y}$ et y avec une mesure de risque $r(\tilde{y}, y) \in \mathbb{R}$. Dans le cas de la régression où $\mathcal{A} \in \mathbb{R}$, on peut prendre un risque quadratique :

$$r(y, \tilde{y}) = (y - \tilde{y})^2$$

Dans le cas d'un problème de classification on peut compter le nombre d'erreurs avec $r(y, \tilde{y}) = 0$ si $\tilde{y} = y$ et $r(y, \tilde{y}) = 1$ si $\tilde{y} \neq y$.

L'erreur / risque *empirique* sur les données $\{(x_i, y_i)\}_{i \leq n}$ de la fonction $\tilde{f}$ fournie par l'algorithme se définit alors par :

$$\tilde{R}_e(\tilde{f}) = \frac{1}{n} \sum_{i=1}^{n} r(\tilde{f}(x_i), y_i).$$

Les données d'entraînement étant des réalisations d'échantillons aléatoires i.i.d, l'erreur empirique sur les données d'entraînement correspond à une réalisation d'une erreur empirique aléatoire $\tilde{R}_e(\tilde{f})$ sur n échantillons i.i.d $\{(X_i, Y_i)\}_{i \leq n}$:

$$\tilde{R}_e(\tilde{f}) = \frac{1}{n} \sum_{i=1}^{n} r(\tilde{f}(X_i), Y_i).$$

L'erreur / risque moyen ou erreur / risque de *généralisation* est définie en moyenne sur la distribution jointe de (X, Y) :

$$R(\tilde{f}) = \mathbb{E}\left[r\left(\tilde{f}(X), Y\right)\right]$$

Le but est de minimiser cette erreur moyenne.

De nombreux algorithmes d'apprentissage optimisent le choix de $\tilde{f}$ dans $\mathcal{H}$ en minimisant l'erreur empirique $\tilde{R}_e$, qui est la seule à laquelle on a accès :

$$\tilde{f} = \arg\min_{h \in \mathcal{H}} \tilde{R}_e(h)$$

Pour un $h \in \mathcal{H}$, si les exemples d'entraînement $\{(X_i, Y_i)\}_{i \leq n}$ ont la même loi que (X, Y) alors l'erreur empirique est un estimateur non biaisé de l'erreur de généralisation :

$$\mathbb{E}\left[\tilde{R}_e(h)\right] = \mathbb{E}\left[\frac{1}{n} \sum_{i=1}^{n} r(h(X_i), Y_i)\right] = \mathbb{E}\left[r\left(h(X), Y\right)\right] = R(h).$$

Par contre, ce résultat n'est pas vrai pour le minimisateur du risque empirique $\tilde{f}$. En effet, $\tilde{f}$ dépend des (X_i, Y_i). Les valeurs de $r(\tilde{f}(X_i), Y_i)$ ne sont à priori pas i.i.d et l'estimateur $\tilde{R}_e(\tilde{f})$ est

potentiellement fortement biaisé.

Une question fondamentale est de comprendre comment mesurer et majorer l'écart entre l'erreur empirique $\tilde{R}_e(\tilde{f})$ et l'erreur de généralisation $R(\tilde{f})$ et ainsi déterminer les conditions pour que l'erreur empirique soit une bonne estimation de l'erreur de généralisation. Une autre question sera de s'assurer que l'erreur de généralisation $R(\tilde{f})$ est faible.

3 Dilemme biais-fluctuations

3.1 Majoration du risque de généralisation

Nous cherchons des bornes sur le risque de généralisation $R(\tilde{f})$ que l'on ne peut pas calculer. On note f_a la "meilleure" approximation au sein de $\mathcal{H}$ qui minimise le risque généralisation :

$$f_a = \arg\min_{h \in \mathcal{H}} R(h)$$

et

$$\tilde{f} = \arg\min_{h \in \mathcal{H}} \tilde{R}_e(h).$$

Proposition 1

$$R(f_a) \leq R(\tilde{f}) \leq R(f_a) + 2 \max_{h \in \mathcal{H}} |R(h) - \tilde{R}_e(h)|$$

Démonstration : La première inégalité est immédiate. Pour la seconde, on observe que

$$\begin{aligned} R(\tilde{f}) &= R(\tilde{f}) - \tilde{R}_e(\tilde{f}) + \underbrace{\tilde{R}_e(\tilde{f}) - \tilde{R}_e(f_a)}_{\leq 0} + \tilde{R}_e(f_a) - R(f_a) + R(f_a) \\ &\leq R(\tilde{f}) - \tilde{R}_e(\tilde{f}) + \tilde{R}_e(f_a) - R(f_a) + R(f_a) \\ &\leq R(f_a) + |R(\tilde{f}) - \tilde{R}_e(\tilde{f})| + |\tilde{R}_e(f_a) - R(f_a)| \\ &\leq R(f_a) + 2 \max_{h \in \mathcal{H}} |R(h) - \tilde{R}_e(h)|. \end{aligned}$$

□

On appelle $R(f_a)$ l'erreur d'*approximation* qui représente l'erreur de généralisation minimale au sein de la classe de fonctions $\mathcal{H}$ tandis que $\max_{h \in \mathcal{H}} |R(h) - \tilde{R}_e(h)|$ constitue l'erreur maximum de *fluctuation* entre le risque empirique et le risque moyen de n'importe quel prédicteur $h \in \mathcal{H}$.

On retrouve dans cette inégalité le *dilemme biais-variance* ou encore *dilemme biais-complexité* : l'erreur moyenne du minimisateur du risque empirique est majorée par un terme de biais -l'erreur d'approximation- plus un terme de variation ou de complexité -l'erreur de fluctuation-. On détaille ici ces différents termes :

Plus la classe de modélisation $\mathcal{H}$ est grande, meilleure sera la meilleure approximation f_a de f et plus l'erreur de biais sera faible. Le terme de biais décroit donc quand $\mathcal{H}$ augmente. A l'inverse, la fluctuation maximum augmente lorsque $\mathcal{H}$ augmente. Si $\mathcal{H}$ est trop grand, le minimisateur du risque empirique $\tilde{f}$ aura tendance à trop coller aux données d'entrainement et ne va pas capturer la régularité du phénomène sous-jacent, qui permet de généraliser le résultat. C'est le *surapprentissage*. L'erreur de fluctuation augmente donc quand la taille des modèles donc avec le cardinal de $\mathcal{H}$ pour des classes emboitées. L'optimum se situe donc pour un cardinal de $\mathcal{H}$ tel que ces deux erreurs sont du même ordre comme illustré dans la figure ci-dessous. Toute la difficulté va être de trouver des classes $\mathcal{H}$ qui ne sont pas trop grandes et qui produisent une petite erreur d'approximation.

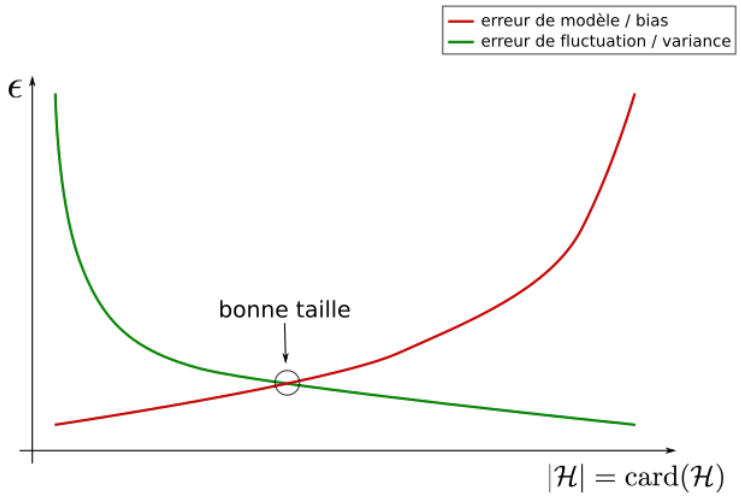

FIGURE 4 – Dilemme biais-fluctuation : le terme de biais décroit tandis que celui de fluctuation croit avec le cardinal de $\mathcal{H}$

3.2 PAC : Probablement Approximativement Correct

Nous démontrons ici un résultat PAC (Probablement Approximativement Correct), que l'erreur de fluctuation $\max_{h\in\mathcal{H}} |R(h) - \tilde{R}_e(h)|$ -le terme de fluctuation- peut être majorée en probabilité ($\tilde{R}_e(h)$ étant une variable aléatoire) par la taille (ou plus précisément la *complexité*) de la classe de fonctions $\mathcal{H}$ considérée. On va montrer que cette erreur de fluctuation peut être arbitrairement petite si n est suffisamment grand quelque soit la distribution des données (X, Y). La convergence en probabilité de l'erreur empirique $\tilde{R}_e(\tilde{f})$ vers l'erreur de généralisation $R(\tilde{f})$ est dans ce cas uniforme par rapport à la loi de (X, Y).

Pour cela, nous rappelons auparavant que pour un $h \in \mathcal{H}$ fixé, $\tilde{R}_e(h)$ est un estimateur sans biais de l'erreur de généralisation $R(h)$. L'erreur de fluctuation mesure donc de manière uniforme (car on ne sait pas quelle fonction $\tilde{f} \in \mathcal{H}$ sera le minimisateur empirique) l'écart absolu entre l'estimateur $\tilde{R}_e(h)$ et sa moyenne $R(h)$. On peut mesurer cet écart absolu presque sûrement, en probabilité, en moyenne ou en loi. Presque sûrement, nous savons que cet écart absolu tend asymptotiquement vers 0 par la loi forte des grands nombres, et en loi nous savons que cet écart tend asymptotiquement vers une gaussienne par le théorème central limite, et donc que cet écart absolu tend vers une valeur absolue de gaussienne. En probabilité ce type d'écart s'appelle une inégalité de concentration et mesure les probabilités des grandes déviations et en moyenne cet écart représente la variation de notre estimateur. On souhaiterait un résultat non asymptotique qui nous donne la vitesse de convergence pour une mesure de convergence plus forte que la convergence en loi.

Théorème 1 *Si $R(h) \in [0, 1] \forall h$ et le cardinal $|\mathcal{H}|$ de $\mathcal{H}$ est fini, alors :*

$$\mathbb{P}\left(\max_{h\in\mathcal{H}} |R(h) - \tilde{R}_e(h)| \leq \epsilon\right) \geq 1 - \delta \quad si \quad \epsilon^2 = \frac{\log(|\mathcal{H}|) + \log(2/\delta)}{2n}$$

En injectant le théorème 1 dans la proposition 1 on obtient le corollaire suivant :

Corollaire Avec probabilité $\geq 1 - \delta$ on a :

$$R(\tilde{f}) \leq R(f_a) + 2\sqrt{\frac{\log(|\mathcal{H}|) + \log(2/\delta)}{2n}}$$

La preuve est basée sur l'inégalité de concentration de Hoeffding qui contrôle en probabilité les grandes déviations, dont on énonce ici seulement le résultat :

Lemme 1 *(Inégalité de Hoeffding) Soient $(Z_i)_{1\leq i\leq n}$ n variables aléatoires i.i.d telles que $\forall i \leq n$, $Z_i \in [a, b]$ et $E(Z_i) = \mu$. On note $\tilde{\mu}$ la moyenne empirique $\tilde{\mu} = \frac{1}{n}\sum_{i=1}^{n} Z_i$.*

Alors, $\forall \epsilon > 0$

$$\mathbb{P}\left(|\tilde{\mu} - \mu| \geq \epsilon\right) \leq 2\exp\left(-\frac{2n\epsilon^2}{(b-a)^2}\right)$$

Ce lemme énonce une loi de décroissance gaussienne des grandes déviations d'une moyenne empirique par rapport à la vraie moyenne. Ce résultat, proche de ce qu'on obtiendrait de manière asymptotique par le théorème central limite, peut être vu comme une version non asymptotique de ce théorème.

Nous allons montrer le théorème 1 en contrôlant la terme de grande déviation uniforme

$$\mathbb{P}\left(\max_{h\in\mathcal{H}} |R(h) - \tilde{R}_e(h)| \geq \epsilon\right)$$

à l'aide du lemme d'Hoeffding et d'une majoration "brutale" d'une union par une somme.

On considère l'ensemble (où e désigne un évènement élémentaire) :

$$\begin{aligned}\mathcal{E} &= \{e \,|\, \exists h \in \mathcal{H} \,|\, |R(h) - \tilde{R}(h)| \geq \epsilon\} \\ &= \cup_{h\in\mathcal{H}}\{e \,|\, |R(h) - \tilde{R}(h)| \geq \epsilon\}\end{aligned}$$

Alors

$$\begin{aligned}\mathbb{P}(\mathcal{E}) &\underbrace{\leq}_{P(A\cup B)\leq P(A)+P(B)} \sum_{h\in\mathcal{H}} \mathbb{P}\left(e \,|\, |R(h) - \tilde{R}(h)| \geq \epsilon\right) \\ &\leq \sum_{h\in\mathcal{H}} 2\,\exp\left(-2n\epsilon^2\right) \quad \text{en utilisant Hoeffding avec [a,b] = [0,1]} \\ &\leq 2|\mathcal{H}|\,\exp\left(-2n\epsilon^2\right)\end{aligned}$$

On veut pour prouver le théorème que $\mathbb{P}(\mathcal{E}) \leq \delta$, ce qui est vrai si et seulement

$$\epsilon^2 \geq \frac{\log(2|\mathcal{H}|/\delta)}{2n}$$

ce qui donne le résultat souhaité.

Ce résultat amène à quelques commentaires :

— Si la taille de $\mathcal{H}$ augmente, alors comme on s'y attend la borne supérieure de l'erreur de fluctuation augmente.

— Le terme $\log(|\mathcal{H}|)$ correspond à l'entropie de la classe $\mathcal{H}$ des prédicteurs (lorsque la distribution de probabilité est uniforme sur la classe). Il s'agit du nombre de bits pour coder un élément de l'ensemble.

— Le résultat a été obtenu en effectuant une majoration brutale d'une union par une somme et peut donc être raffiné.

— En général $\mathcal{H}$ est infini mais peut être paramétré par des paramètres de dimension $m+1$ dans le cas de la régression / classification linéaire où les paramètres sont $w \in \mathbb{R}^m$ et $b \in \mathbb{R}$. On peut ainsi quantifier les valeurs de chaque dimension des paramètres sur N valeurs. Puisque $\mathcal{H}$ est paramétré par $m+1$ paramètres que l'on a quantifiés sur N valeurs, alors $|\mathcal{H}| \sim N^{m+1}$ et donc $\log(|\mathcal{H}|) \sim m\log N$. Pour que la quantification produise une erreur de l'ordre de ϵ il faut choisir $N \sim m/\epsilon$ avec un pas de quantification de l'ordre de ϵ/m. Pour que l'erreur de fluctuation soit faible et inférieure à ϵ, on voit par le théorème 1 que l'on doit avoir $\log(|\mathcal{H}|)/n \ll \epsilon^2$ petit et donc $m/n \ll \epsilon^2$. On voit ici que le nombre de paramètres de notre algorithme de classification / régression linéaire doit être petit devant le nombre d'exemples.

  Nous verrons que dans les réseaux de neurones, on utilise en général plus de paramètres que de nombre d'exemples, mais que l'on arrive malgré tout à de bons résultats de généralisation. Ce résultat est lié à un phénomène de contraction non linéaire incorporé dans le réseau.

3.3 Surapprentissage et données de test

On parle de surapprentissage lorsque l'erreur de généralisation est beaucoup plus importante que le risque empirique.

On peut ainsi facilement trouver une fonction $\tilde{f}$ ayant une erreur empirique faible mais une erreur de généralisation importante. Cela peut typiquement arriver lorsque la classe de fonctions $\mathcal{H}$ est trop large et n'incorpore pas les *a priori* de régularité que l'on a sur la fonction f générant les sorties que l'on cherche à estimer. Le terme de fluctuation est alors très élevé, et la fonction s'attache beaucoup trop aux données d'entrainement. C'est typiquement le cas des algorithmes des K plus proches voisins (*K-nearest neighbours*) et notamment du 1 plus proche voisin qui à partir des données d'entraînement $\{(x_i, y_i)\}_{i\leq n}$ calcule le prédicteur $\tilde{f}^{1NN}$ tel que :

$$\tilde{f}^{1NN}(x) = y_i \quad \text{si} \quad \|x - x_i\| \leq \|x - x'_i\|_{i'\neq i}$$

où $\|\cdot\|$ est n'importe quelle norme sur $\mathbb{R}^d$. On a immédiatement que $\forall i \leq n\ \tilde{f}^{1NN}(x_i) = y_i$ et donc l'erreur empirique $\tilde{R}_e(\tilde{f}^{1NN})$ vaut 0. En revanche l'erreur de généralisation $R(\tilde{f}^{1NN})$ peut-être très importante.

Une erreur de biais peut aussi apparaître si les échantillons aléatoires d'entrainement $(X_i, Y_i)_i$ ne suivent pas la loi des données (X, Y). Par exemple, si l'on veut prédire la consommation énergétique d'un bâtiment en fonction de certains paramètres pour des bâtiments se situant dans différents endroits du monde mais que l'on ne dispose que d'exemples en France, les prédictions sur des bâtiments de Sibérie peuvent être catastrophiques. Il faut donc s'assurer que les bases de données d'entrainement reflètent a la même variabilité que les X, Y.

Pour estimer l'erreur de généralisation, on a besoin de n_t nouveaux échantillons de données $\{(X_i, Y_i)\}_{i\leq n_t}$ i.i.d de même loi que (X, Y), qui n'ont pas été utilisés lors du calcul de $\tilde{f}$ par l'algorithme d'apprentissage. On a en effet bien dans ce cas là que les $r(\tilde{f}(X_i), Y_i)_{i\leq n_t}$ sont i.i.d et donc l'erreur empirique sur les données de test ou erreur de *test* notée $\tilde{R}_t(\tilde{f})$ est bien un estimateur non biaisé de l'erreur de généralisation. On la calcule à partir de n_t nouveaux exemples dits données de test et notées $\{(x_i^t, y_i^t)\}_{i\leq n_t}$ qui sont des réalisations de nouveaux échantillons :

$$\tilde{R}_t(\tilde{f}) = \frac{1}{n_t}\sum_{i=1}^{n_t} r(\tilde{f}(x_i^t), y_i^t) \tag{1}$$

3.4 L'apprentissage supervisé en pratique : le Challenge Data

La pratique du Challenge Data illustre les phénomènes décrits précédemment et notamment l'écueil du surapprentissage. Les challenges proposés dans le cadre du Challenge Data visent tous à proposer à chaque entrée x un estimateur $\tilde{y}$ de la réponse y à x. Ils fournissent ainsi des données d'entraînement $\{(x_i, y_i)\}_{i\leq n}$, des données de test $(x_i^t)_{i\leq n_t}$ pour lesquelles le participant devra calculer des prédictions $\tilde{y}_i = \tilde{f}(x_i^t)$. Ces prédictions sont comparées lors de chaque soumission aux "vraies" réponses $(y_i^t)_{i\leq n_t}$ avec un calcul de risque

$$\tilde{R}_t(\tilde{f}) = \tilde{R}\Big((\tilde{y}_i)_{i\leq n_t}, (y_i)_{i\leq n_t}\Big).$$

Les $(y_i^t)_{i\leq n_t}$ sont dans un fichier caché sur le serveur. Le calcul du risque $\tilde{R}_t$ est aussi fournie aux participants sous la forme d'une fonction Python. Elle s'écrit le le plus souvent comme une somme (1). Il existe néanmoins quelques exceptions comme l'AUC (*Area Under the ROC Curve*) qui mesure la performance d'un classificateur binaire associant une classe à un "score" (génériquement une probabilité d'appartenir à une classe) en calculant de manière relative l'erreur d'ordonnancement entre les scores prédits et les vraies classes de sorties.

Un des principaux écueils de la démarche d'apprentissage consiste à sur-apprendre et donc mal généraliser. Ce sur-apprentissage peut avoir lieu sur les données d'entraînement mais également sur les données de test à travers les soumissions sur le site du challenge. Afin d'éviter ce phénomène, les soumissions sont limitées à deux par jour et le score renvoyé lors des soumissions ne correspond qu'à une partie des prédictions, le score sur l'autre partie demeurant caché et faisant office de score

final. Les scores finaux seront communiqués début juin et en décembre.

Afin de vérifier sur les n données d'entrainement qu'un algorithme généralise bien, on utilise généralement la technique dite de *cross validation* pour laquelle les données d'entraînement sont segmentées en données d'entraînement à proprement parler sur lesquelles l'algorithme est entrainé et données de validation sur lesquelles l'algorithme est testé. On peut par exemple diviser les données d'entraînement en un bloc d'entraînement contenant 3/4 des données puis un bloc de validation contenant 1/4 des données.

Se pose alors la démarche de choix de l'algorithme : la première chose à faire consiste d'abord à essayer de regarder les données et leurs propriétés statistiques pour voir si des représentations simples (comme des moyennes ou des variances) portent une part importante du signal à l'aide d'algorithmes simples, puis si cela ne fonctionne pas d'évoluer vers des algorithmes sollicitant des classes de fonction d'estimation de plus en plus complexes dont on optimisera les paramètres sur les données d'entraînement en vérifiant qu'on a évité le surapprentissage. Il faut éviter de commencer avec des algorithmes complexes comme les réseaux de neurones profonds et plutôt essayer les algorithmes classiques suivants :

— Arbres de décision
— Boosting, XGboost,
— Apprentissage à noyaux, SVMs
— Puis éventuellement des réseaux de neurones profonds notamment pour de la vision

Les arbres de décision, les algorithmes de boosting et les algorithmes à noyaux type SVM nécessitent moins de données que les réseaux de neurones profonds car ils utilisent de l'information a priori. Les réseaux de neurones nécessitent eux beaucoup d'exemples.

Le fonctionnement de ces algorithmes est bien décrit dans un livre de référence sur le domaine que je conseille : *The Elements of Statistical Learning* de Hastie, Tibshirani et Fridman, disponbile gratuitement à l'adresse suivante : https://web.stanford.edu/~hastie/Papers/ESLII.pdf