Introduction au domaine de recherche

Arbres de classification et forêts aléatoires

Jacques FOURNIER
sous la direction de Stéphane BOUCHERON
30 mai 2016

Préambule

Le développement des capacités de calcul et de gestion de grands volumes de données de ces dernières décénnies a permis l'essor de l'apprentissage statistique. Ce champ, à la frontière des mathématiques et de l'informatique, a pour but de concevoir des procédures automatiques permettant de mettre en évidence des règles générales à partir d'exemples. Les applications couvrent un large éventail de domaines allant de la biologie à la finance en passant par le traitement du language.

L'algorithme des forêts aléatoires, proposé par Leo Breiman (2001) figure parmi les méthodes d'apprentissage supervisé les plus performantes utilisées à ce jour (adaboost, réseaux de neurones, SVM,...).

Cependant si de nombreux résultats permettent de mieux comprendre certains mécanismes intervennant dans le fonctionnement de versions simplifiées de cet algorithme, beaucoup reste à faire pour montrer l'éfficacité théorique de la version originelle de Breiman.

J'introduirai dans un premier temps quelques résultats et concepts de l'apprentissage statistique. Ensuite, nous étudierons quelques résultats généraux des méthodes de partitionnement, puis nous étudierons les arbres de classifications qui sont à la base des forêts aléatoires. Enfin nous nous pencherons sur les forêts aléatoires. Je me limiterai dans ce mémoire au cas particulier de la classification binaire sur $\mathbb{R}^d$ pour plus de concision et de clarté.

Table des matières

1 Cadre d'étude, et rappel de concepts centraux

1.1 Cadre

On se donne deux espaces $\mathcal{X}$ (l'espace des features), et $\mathcal{Y}$ (l'espace des étiquettes). On dispose de n couples d'observations $(X_1, Y_1), ..., (X_n, Y_n)$ que l'on suppose être des réalisations indépendantes d'une même loi $\mathbb{P}_{X,Y}$. On notera $\mathcal{D}_n = \{(X_1, Y_1), ..., (X_n, Y_n)\}$. Le but est alors de prévoir l'étiquette Y associée à X pour un couple (X, Y) généré selon la loi $\mathbb{P}_{X,Y}$ de manière indépendante de $\mathcal{D}_n$. On cherche ainsi à déterminer une fonction de prédiction $g : \mathcal{X} \to \mathcal{Y}$ généralisant les relations $g(X_i) \sim Y_i$, observées sur $\mathcal{D}_n$. On distingue traditionnellement deux cas :

— $\mathcal{Y} \in \mathbb{R}^d$. On parle alors de **régression**.

— $\mathcal{Y}$ est un ensemble fini (non nécessairement ordonné). On parle alors de **classification**.

On dispose également d'une fonction de perte $l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$. Cette fonction de perte pénalise les mauvaises décisions. On peut donc par exemple souhaiter qu'elle soit minimale sur $\{(y, y) \ / y \in \mathcal{Y}\}$.

Définition 1 (fonction de risque)**.** *On définit une* ***fonction de risque*** r *comme étant* $r : g \to E_{\mathbb{P}_{(X,Y)}}[l(Y, g(X))]$. *Soit* $g \in \mathcal{Y}^{\mathcal{X}}$ *mesurable,* $r(g)$ *est appelé risque associé à la* ***fonction de prédiction*** g

Nous nous trouvons ainsi dans un cadre analogue à la théorie de la décision, ou encore des statistiques bayésiennes. Un risque défini une notion de qualité d'une fonction de prédiction (que l'on désire le plus faible possible).

Nous nous limiterons dans la suite au cas de la classification binaire, i.e. nous supposerons que $\mathcal{Y} = \{0, 1\}$. Nous faisons de plus l'hypothèse que $\mathcal{X} = \mathbb{R}^p$, et nous considérerons exclusivement la fonction de perte $l(y, z) = \mathbf{1}_{y \neq z}$. Cependant, il faut garder en tête que de nombreux résultats énoncés ici sont généralisables.

Le théorème suivant permet de borner le risque, et la preuve exhibe une fonction de décision optimale.

Théorème 1. *Soit* r *le risque associé à* $l(y, z) = \mathbf{1}_{y \neq z}$.
Alors il existe $g^* \in \underset{g \in \mathcal{Y}^{\mathcal{X}}}{Argmin}\, r(g)$. g^* *est appelé classifieur bayesien.*

Démonstration. $E[l(Y, z)|X = x] = \mathbf{1}_{z=0}\mathbb{P}(Y = 1|X = x) + \mathbf{1}_{z=1}\mathbb{P}(Y = 0|X = x)$. En posant $g^* : x \to \mathbf{1}_{\mathbb{P}(Y=1|X=x) > \frac{1}{2}}$ On a pour toute fonction de décision g mesurable :

$$E[l(Y, g^*(X))|X] = \inf_{z \in \{0,1\}} E[l(Y, z)|X] \leq E[l(Y, g(X))|X]$$
$$\text{et donc, } E[l(Y, g^*(X))] = E[\inf_{z \in \{0,1\}} E[l(Y, z)|X]] \leq E[l(Y, g(X))]$$

ce qui implique l'optimalité de g^* □

On notera $r^* = r(g^*)$ la borne inferieure obtenue. En général, $\mathbb{P}_{X,Y}$ est inconnu (donc g^* est également inconnu). Ainsi, on a recours à des **algorithmes d'apprentissage**, c'est à dire des fonctions $m : \bigcup_{n \in \mathbb{N}} (\mathcal{X} \times \mathcal{Y})^n \to \mathcal{Y}^{\mathcal{X}}$ pour tenter d'obtenir des fonctions de décision à faible risque. Autrement dit, $m(\mathcal{D}_n)$ est une fonction de prédiction. On utilisera parfois l'abus de notation $m(\mathcal{D}_n, x)$ pour désigner $m(\mathcal{D}_n)(x)$. La pertinence d'un algorithme peut être établie en montrant qu'il vérifie la propriété suivante.

Définition 2 (Consistance d'un algorithme)**.** *Un algorithme* m *est dit consistant (pour le risque de classification) si et seulement si* $E_{\mathbb{P}_{X,Y}^{\otimes n}}[r(m(\mathcal{D}_n))] \underset{n \to \infty}{\to} r^* = r(g^*)$

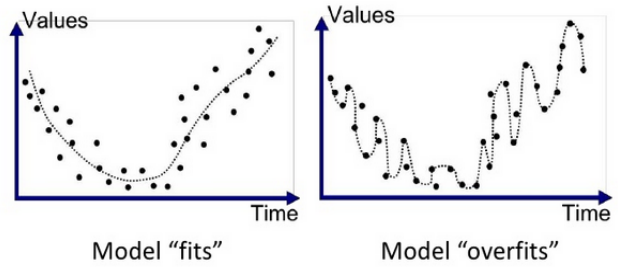
Ceci équivaut à une convergence L^1 de $r(m(\mathcal{D}_n))$ vers r^* car $E_{X,Y}[l(Y, m(\mathcal{D}_n, X))] \geq r^*$.

1.2 Minimisation du risque empirique

Une stratégie possible pour concevoir un algorithme m peut être de chercher à minimiser un estimateur connu de la fonctionnelle r. Le risque empirique $\hat{r}_n : g \to \frac{1}{n}\sum_{i=1}^n l(Y_i, g(X_i))$ semble être un bon candidat car pour tout $g \in \mathcal{Y}^{\mathcal{X}}$ mesurable,

$$\sqrt{n}(\hat{r}_n(g) - r(g)) \underset{n \to \infty}{\overset{(loi)}{\to}} \mathcal{N}(0, Var(l(Y, g(X)))) \text{ (par le TCL) et } \hat{r}_n(g) \underset{n \to \infty}{\overset{p.s.}{\to}} r(g) \text{ (par LLN)}$$

Remarque : Cet estimateur est simplement un **estimateur plug-in** de r dans le sens où dans l'expression $E_{\mathbb{P}_{X,Y}}[l(Y,g(X))]$ on peut remplacer $\mathbb{P}_{X,Y}$ par la **mesure empirique** $\hat{\mathbb{P}}_n = \frac{1}{n}\sum_{i=1}^n \delta_{Y_i,X_i}$ (les δ_{Y_i,X_i} étant des mesures dirac en (X_i,Y_i), qui est aléatoire) et constater ainsi que $E_{\hat{\mathbb{P}}_n}[l(Y,g(X))] = \hat{r}_n(g)$.
Cependant, une minimisation brutale du risque empirique peut conduire à des décisions catastrophiques (au sens du risque). En effet, on peut par exemple considérer le cas où $g(X_i) = Y_i$ pour tout i, et $g = 0$ sur $\mathcal{X} - \mathcal{D}_n$ (qui minimise $\hat{r}_n$). Le modèle est alors trop souple (*i.e.* l'espace de recherche $\mathcal{Y}^{\mathcal{X}}$ est trop grand), ce qui permet à l'algorithme d'apprendre par coeur les bonnes réponses. On dit aussi que l'algorithme sur-apprend (overfitting en anglais).



Ainsi, il faut restreindre cet espace de recherche pour que notre algorithme ait une faculté de généralisation. Soit $\mathcal{G}$, un sous ensemble de $\mathcal{Y}^{\mathcal{X}}$. Sous l'hypothèse que $g^*_{\mathcal{G}} \in \underset{g\in\mathcal{G}}{Argmin}\, r(g)$ est bien défini, on peut alors décomposer

$$r(m(\mathcal{D}_n)) - r^* = \underbrace{r(m(\mathcal{D}_n)) - r(g^*_{\mathcal{G}})}_{\text{excès de risque(stochastique)}} + \underbrace{r(g^*_{\mathcal{G}}) - r*}_{\text{biais}}$$

Il est alors possible de controler le terme d'erreur stochastique.

Lemme. *Si $m_{\mathcal{G}} \in \underset{g\in\mathcal{G}}{Argmin}\, \hat{r}_n(g)$ est bien défini alors,*

$$r(m_{\mathcal{G}}(\mathcal{D}_n)) - r(g^*) \leq 2 \sup_{g\in\mathcal{G}} |r(g) - \hat{r}_n(g)|$$

Démonstration.

$$r(m_{\mathcal{G}}) - r(g^*_{\mathcal{G}}) = r(m_{\mathcal{G}}) - \hat{r}_n(m_{\mathcal{G}}) + \hat{r}_n(m_{\mathcal{G}}) - \hat{r}_n(g^*) + \hat{r}_n(g^*) - r(g^*) \leq 2 \sup_{g\in\mathcal{G}} |r(g) - \hat{r}_n(g)|$$

□

Ainsi, on aimerait trouver un ensemble de fonctions $\mathcal{G}$ de taille raisonnable tels le terme $\sup_{g\in\mathcal{G}} |r(g) - \hat{r}_n(g)|$ tende vers 0. Montrer ce genre de convergence en probabilité dans le cadre de la classification binaire nous amène à aborder la question des inégalités maximales de concentration dans le cadre de la théorie de Vapnik-Chervonenkis.

1.3 Théorie de Vapnik-Chervonenkis

Les fonction de décision étant à valeur dans $\mathcal{Y} = \{0,1\}$, elles s'écrivent nécessairement sous la forme : $g_C : x \in \mathcal{X} \to \mathbf{1}_{x\in C}$ pour $C \subset \mathcal{X}$. Ainsi, sans pertes de généralité, on peut considérer qu'il existe un ensemble $\mathcal{C} \in 2^{\mathcal{X}}$ tel que : $\mathcal{G} = \{: x \to \mathbf{1}_{x\in C}, C \in \mathcal{C}\}$.

Définition 3 (trace). *Soit $\mathcal{C} \subset 2^{\mathcal{X}}$ et $\mathcal{E} = x_1, ..., x_n$ un ensemble fini de points de $\mathcal{X}$. On définit :*
- *la trace de $\mathcal{C}$ (ou $\mathcal{G}$) sur $\mathcal{E}$:$S(\mathcal{C},\mathcal{E}) = \#\{C \cap \mathcal{E} \,/\, C \in \mathcal{C}\}$*
- *le **coefficient d'éclatement** de $\mathcal{C}$: $S_n(\mathcal{C}) = \sup_{\mathcal{E}\,/\,\#\mathcal{E}=n} S(\mathcal{C},\mathcal{E})$*

Autrement dit, la trace de $\mathcal{E}$ est le nombre de parties de $\mathcal{E}$ que l'on peut construire en intersectant $\mathcal{E}$ avec un ensemble arbitaire $C \in \mathcal{C}$. On remarque que nécessairement, $S(\mathcal{C},\mathcal{E}) < 2^{\#\mathcal{E}}$.

Définition 4 (pulvérisation). *On dit que $\mathcal{C}$ pulvérise $\mathcal{E}$ si $S(\mathcal{C},\mathcal{E}) = 2^{\#\mathcal{E}}$*

Définition 5 (Dimension de Vapnik-Chervonenkis).

$$V_{\mathcal{C}} = \max\{k \in \mathbb{N} : \exists \mathcal{E} \subset \mathcal{X} \text{ tel que } \#\mathcal{E} = k \text{ et } \mathcal{E} \text{ est pulvérisé par } \mathcal{C}\}$$

exemple :
— pour $\mathcal{X} = \mathbb{R}$, $\mathcal{C} = \{\{x > a\}, a \in \mathbb{R}\}$, $V_{\mathcal{C}} = 2$
— pour $\mathcal{X} = \mathbb{R}$, $\mathcal{C} = \{\{x > a\}, a \in \mathbb{R}\} \cap \{\{x < a\}, a \in \mathbb{R}\}$, $V_{\mathcal{C}} = 2$
— pour $\mathcal{X} = \mathbb{R}^2$, $\mathcal{C} = \{\{\langle a, x\rangle > b\}, a \in \mathbb{R}^2, b \in \mathbb{R}\}$, $V_{\mathcal{C}} = 3$
— pour $\mathcal{X} = \mathbb{R}^d$, $\mathcal{C}$ l'ensemble des parties localement convexes et connexes de $\mathcal{X}$ alors $V_{\mathcal{C}} = \infty$.
Ainsi, la notion de dimension de Vapnik-Chervonenkis peut être vue comme une mesure de la complexité d'un ensemble d'ensemble (ou du pouvoir séparateur).

Théorème 2. *Vapnik-Chervonenkis-Sauer Si $V_{\mathcal{G}} < \infty$, alors en classification binaire ,*

$$\mathbb{P}_{\mathcal{D}_n}(\sup_{g\in\mathcal{G}} |r(g) - r_n(g)| > \epsilon) \leq 8S_n(\mathcal{C})e^{-\frac{n\epsilon^2}{32}} \leq 8(n+1)^{V_{\mathcal{G}}}e^{-\frac{n\epsilon^2}{32}}$$

Voir [9], chapitre 12 page 197 pour une preuve de la première inégalité, utiliser le théorème 13.2 du même ouvrage pour la seconde.

2 Méthodes de partitionnement

La théorie de Vapnik Chervonenkis nous indique que la démarche de minimisation du risque empirique sur un ensemble de fonctions constantes par morceau sur des ensembles simples (i.e. tel que S_n croisse de manière sous exponentielle) pourrait donner des résultats annulant assympotiquement l'erreur stochastique. Ainsi, une bonne façon obtenir un algorithme d'apprentissage en classification pourrait être la suivante :

1. partitionner l'espace $\mathcal{X}$ par un nombre fini de motifs simple (*i.e.* choisir une partition $\mathcal{P}$), pour assurer une VC-dimension finie.
2. fixer la décision renvoyée au sein de chacun des ensembles de la partition $\mathcal{P}$ obtenue (i.e. déterminer une fonction $g_{\mathcal{P}}$ constante sur l'ensemble des éléments de la partition).

On parle alors de méthodes d'apprentissage par partitionnement. On notera $Cst_{\mathcal{P}}$ l'ensemble des fonctions constantes sur les éléments de $\mathcal{P}$. Ainsi, on considèrera des fonction du type

$$g_{\mathcal{P}} : x \in \mathcal{X} \to \sum_{P\in\mathcal{P}} g_P \mathbf{1}_{x\in P} \qquad \text{avec } g_P \in \{0,1\} \text{ pour tout } P \in \mathcal{P}.$$

2.1 Décision optimale à partition fixée

Ayant fixé une partition $\mathcal{P}$, en définissant $g_P^* \overset{def}{\in} \underset{z\in\mathcal{Y}}{Argmin}\ E_{\mathbb{P}_{X,Y}}[l(Y,z)|X \in P]$, (qui est bien définit dans notre cadre de classification binaire), on obtient :

$$r(g_{\mathcal{P}}) = \sum_{P\in\mathcal{P}} E[l(Y \neq g_P)|P]\ \mathbb{P}_X(P) \geq r(g_{\mathcal{P}}^*) \qquad \text{avec } g_{\mathcal{P}}^*(x) = \sum_{P\in\mathcal{P}} g_P^* \mathbf{1}_{x\in P}$$

Définition 6 (risque d'une partition, d'un ensemble)**.** *On définit :*
— *le risque d'un ensemble $C \in \mathcal{X}$, $r(C) = min_{z\in\mathcal{Y}}(E[l(Y,z)\mathbf{1}_{X\in C}])$*
— *le risque d'une partition $\mathcal{P}$, $r(\mathcal{P}) = \sum_{P\in\mathcal{P}} r(P)$*

Le risque d'une partition $\mathcal{P}$ est une mesure du risque minimal que l'on peut atteindre avec une fonction de $Cst_{\mathcal{P}}$, et on souhaite qu'il soit la plus faible possible. De plus, le risque d'une partition se décompose en une somme de contributions indépendante chacun des ensembles de la partition. De manière analogue, on aimerait définir un risque empirique d'une partition (en remplaçant par $\mathbb{P}$ par $\hat{\mathbb{P}}_n$ ce qui est justifié par des argument TCL et LNN) par :

Définition 7 (risque empirique d'une partition, d'un ensemble)**.** *On définit :*
— *le risque d'un ensemble C de $\mathcal{X}$ $\hat{r}_n(C) = \min_{z\in\mathcal{Y}} E_{\hat{\mathbb{P}}_n}[l(Y,z)\mathbf{1}_{X\in C}] = \min_{z\in\mathcal{Y}} \frac{\sum_{i=1}^n l(Y_i,z)\mathbf{1}_{X_i\in C}}{n}$*
— *le risque d'une partition $\mathcal{P}$ par $\hat{r}_n(\mathcal{P}) = \sum_{P\in\mathcal{P}} \hat{r}_n(P)$*

Notons $\hat{g}_{\mathcal{P}}^* : x \to \sum_{P\in\mathcal{P}} \hat{g}_{n,\mathcal{P}}^* \mathbf{1}_{X\in P}$ avec $\hat{g}_{n,\mathcal{P}}^* \in \underset{z\in\mathcal{Y}}{Argmax} \sum_{j=1}^n \mathbf{1}_{Y_j=1 \text{ et } X_j\in P}$. Alors $\hat{g}_{\mathcal{P}}^*$ est un minimiseur du risque empirique sur $Cst_{\mathcal{P}}$ en classification binaire (avec pour perte $l(y,z) = \mathbf{1}_{y\neq z}$). La validité théorique de cette démarche est justifié par le résultat suivant :

Théorème 3 (Consistance des décisions minimisant le risque empirique en classification binaire)**.** *Soit $(\mathcal{P}art_n)$ une suite croissante de famille de partitions, et i_n une suite réelle positive vérifiant $i_n = o(\sqrt{\frac{n}{\log(S_n(\{P \ / \ P \in \mathcal{P} \in \mathcal{P}art_n\}))}})$ alors,* $\sup_{\mathcal{P} \in \mathcal{P}art_n \ / \ \#\mathcal{P} \leq i_n} |r(\hat{g}_{\mathcal{P}}) - r(\mathcal{P})| \underset{n\to\infty}{\to} 0$

Ce qui montre également le résultat plus faible :

$$\mathcal{P}_n \in \mathcal{P}art_n \forall n \in \mathbb{N} \text{ et } \#\mathcal{P}_n = o(\sqrt{\frac{n}{\log(S_n(\{P \ / \ P \in \mathcal{P} \in \mathcal{P}art_n\}))}}) \Rightarrow r(\hat{g}^*_{\mathcal{P}_n}) - \ r(\mathcal{P}_n) \underset{n\to\infty}{\to} 0$$

La preuve utilise des techniques d'inégalités maximales.

2.2 Partitionnement par minimisation sous contrainte du risque empirique

En l'absence de connaissance sur r, on cherchera en pratique à minimiser $\hat{r}_n$ sur un ensemble de partition de forme simple. Cette démarche peut être justifiée par le résultat suivant :

Théorème 4 (Lugosi-Nobel)**.** *Soit $(\mathcal{P}art_n)_{n\in\mathbb{N}}$, et notons $\mathcal{B}(\mathcal{P}art_n) = \{\bigcup_{P\in\mathcal{I}} P \ / \ \exists \mathcal{I} \subset \mathcal{P} \in \mathcal{P}art_n\}$. Alors, en classification binaire (avec $l(y,z) = \mathbf{1}_{y\neq z}$), pour tout $\epsilon > \sqrt{\frac{2}{n}}$*

$$\mathbb{P}(\sup_{\mathcal{P}\in\mathcal{P}art_n} |r(\mathcal{P}) - \hat{r}_n(\mathcal{P})| > \epsilon) \leq 2^6 \ S_n(\mathcal{B}(\mathcal{P}art_n)) \ e^{-n\frac{\epsilon^2}{2^9}})$$

La démonstration de ce théorème utilise notamment les indégalités de Hoeffding et de Vapnik-Chervonenkis.
Du théorème de Lugosi-Nobel, on peut déduire le résultat suivant :

Corollaire. *Si* $\frac{\log(S_n(\mathcal{B}(\mathcal{P}art_n)))}{n} \underset{n\to\infty}{\to} 0$*, alors* $\sup_{\mathcal{P}\in\mathcal{P}art_n} |r(\mathcal{P}) - \hat{r}_n(\mathcal{P})| \overset{p.s.}{\underset{n\to\infty}{\to}} 0$

Démonstration. En montrant que $\mathbb{P}(\sup_{\mathcal{P}\in\mathcal{P}art_n} |r(\mathcal{P}) - \hat{r}_n(\mathcal{P})| > \epsilon)$ est alors sommable, on peut alors déduire le résultat d'une application directe du théorème de Borel-Cantelli. □

Condition pour une consistance globale :
En combinant les résultats précédents (théorèmes 3 et 4) on parvient finalement à montrer que :

Théorème 5 (Consistance de la minimisation du risque empirique)**.** *Si :*
- $\frac{\log(S_n(\mathcal{B}(\mathcal{P}art_n)))}{n} \underset{n\to\infty}{\to} 0$
- $\mathcal{P}_n$ *est tel que* $\mathcal{P}_n \in \mathcal{P}art_n \ \forall n \in \mathbb{N}$ *et* $\#\mathcal{P}_n \leq i_n$ *avec* $i_n = o(\sqrt{\frac{n}{\log(S_n(\{P\in\mathcal{P}\in\mathcal{P}art_n\}))}})$
- $\hat{r}_n(\mathcal{P}_n) - \inf_{\mathcal{P}\in\mathcal{P}art_n \ / \ \#\mathcal{P}_n\leq i_n} \hat{r}_n(\mathcal{P}) \overset{proba}{\underset{n\to\infty}{\to}} 0$ *(respectivement p.s.)*

Alors, $r(\mathcal{P}_n) - \inf_{\mathcal{P}\in\mathcal{P}art_n \ / \ \#\mathcal{P}\leq i_n} r(\mathcal{P}) \overset{proba}{\underset{n\to\infty}{\to}} 0$ *(respectivement p.s.)*

3 Arbre de classifications

3.1 Partition en hyperrectangles

Ainsi, il est nécessaire que $\frac{\log(S_n(\mathcal{B}(\mathcal{P}art_n)))}{n} \underset{n\to\infty}{\to} 0$ pour obtenir la consistance de méthode de minimisation du risque empirique pour des méthodes de partitionnement. Nous étudions ici une famille de partitions nous permettant de construire les $\mathcal{P}art_n$.

Définition 8 (hyperrectangle)**.** *Un* ***hyperrectangle*** *de $\mathbb{R}^p$ est un volume connexe de mesure de Lebesgue non nul dont les frontières sont des morceau d'hyperplans, chacun normal à au moins un des vecteurs de la base canonique orthonormée. Nous noterons $\mathcal{H}$ l'ensemble des hyperrectangles.*

Ainsi, sur $\mathcal{X}$ les hyperrectangles sont les volumes de la forme $\prod_{j=1}^p I_j$ où I_j est un interval connexe de la forme $]\infty, a],]a, b],]b, +\infty[$, où $\mathbb{R}$, $a, b \in \mathbb{R}$. En notant e_i le $i^{\text{ième}}$ vecteur de la base canonique orthonormée, on remarque que pour tout hyperrectangle H et pour tout $b \in \mathbb{R}$, $H \cup \{x \in \mathcal{X} \ / \ \langle e_i, x\rangle > b\}$ est toujours un hyperrectangle. Plus généralement, $H_1, H_2 \in \mathcal{H} \Rightarrow H_1 \cap H_2 = \emptyset$

ou $H_1 \cap H_2 \in \mathcal{H}$.

Définissons $\mathcal{P}art_{\mathcal{H}}$ l'ensemble des partitions en hyperrectangles et $\mathcal{P}art_{\mathcal{H},n}$ l'ensembles des partions en au plus n hyperrectangles.

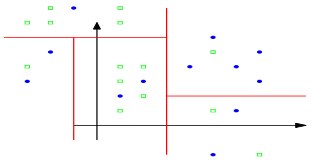


FIGURE 1 – exemple de partition en hyperrectangles

LEMME. *$S_n(\mathcal{B}(\mathcal{P}art_{\mathcal{H},t_n})) \leq (n+1)^{2pt_n}2^{t_n}$. Donc $t_n = o(\frac{\log(n)}{n}) \Rightarrow \frac{\log(S_n(\mathcal{B}(\mathcal{P}art_{\mathcal{H},t_n})))}{n} \underset{n\to\infty}{\to} 0$.*

Démonstration. fixons $\mathcal{E} = \{z_1, ..., z_n\}$. $\#\{P \cap \mathcal{E} \ / \ P \in \mathcal{P} \in \mathcal{P}art_{\mathcal{H},t_n}\} \leq (n+1)^{2p}$ car suivant chaque dimension (pour chaque dimension, il y a au plus $n+1$ possibilités de fixer un hyperplan séparateur orthogonal à cette dimension, et au plus deux hyperplans orthogonaux à une dimension). Or, il y a t_n hyperrectangles dans $\mathcal{P}$, et donc il y a au plus $(n+1)^{2pt_n}$ possibilités pour former chaque partition (en raisonnant par équivalence suivant la trace de chacun des éléments de la partition). L'ensemble des parties de $\mathcal{P}$ est borné par 2^{t_n} et donc, $S(\mathcal{B}(\mathcal{P}art_n), \mathcal{E}) \leq 2^{t_n}(n+1)^{2pt_n}$ Ceci étant valable pour tout $\mathcal{E}$, $S_n(\mathcal{B}(\mathcal{P}art_n)) \leq 2^{t_n}(n+1)^{2pt_n}$ □

COROLLAIRE. *Si $(\mathcal{P}_n)$ est telle que $\mathcal{P}_n \in \mathcal{P}art_{\mathcal{H},t_n}\ \forall\, n \in \mathbb{N}$ avec $t_n = o(\sqrt{\frac{n}{\log(n)}})$ et si $\hat{r}_n(\mathcal{P}_n) - \inf_{\mathcal{P}\in\mathcal{P}art_{\mathcal{H},t_n}} \hat{r}_n(\mathcal{P}) \overset{(proba)}{\underset{n\to\infty}{\to}} 0$, alors*

$$r(\hat{g}_{\mathcal{P}_n}) \overset{(proba)}{\underset{n\to\infty}{\to}} r^*$$

Démonstration. Il suffit de remarquer que $\inf_{\mathcal{P}\in\mathcal{P}art_{\mathcal{H}}} r(\mathcal{P}) = r^*$, que $n \leq S_n(\{P \in \mathcal{P} \in \mathcal{P}art_n\}) \leq (n+1)^{2p}$ En utilisant alors le lemme précédent, toutes les hypothèses du théorème 5 sont alors vérifiées et le résultat découle de son application □

Ce genre de résultat nous permet d'affirmer qu'il est nécessaire de contraindre la taille de la partition pour avoir des résultats de consistance. En effet, l'intuition nous dit que pour avoir $\hat{g}_{n,P} \underset{n\to\infty}{\to} g^*_P$, il est nécessaire que $N_n(P) \overset{(def)}{=} \sum_{j=1}^n \mathbf{1}_{X_j\in P} \underset{n\to\infty}{\to} \infty$ (on doit avoir un nombre d'observation assymptotiquement infini dans chaque élément de la partition).

3.2 Arbre de partitionnement

La partie précédente permettait de justifier une approche de minimisation du risque empirique pour déterminer une partition convenable. Cependant, une telle recherche est trop couteuse en pratique. En effet, l'étude du nombre de partitions de n éléments en k classe (aussi appelés nombre de Stirling de deuxième espèce). Ainsi, sur quelques exemples :

n	6	9	15
k	3	6	7
nombre de partitions en k classes	90	7770	408741333

On préfère donc en pratique adopter une approche récursive qui raffine progressivement la partition en choisissant une sur-partition minimisant la perte de risque. Le lecteur intéressé par ce genre approches gloutonnes pourra consulter [8] chapitre "Greedy algorithms".
Afin de modéliser cette idée de partition récursive, nous introduisons la notion de sur-partition.

Définition 9. *Soit $\mathcal{P}$ et $\mathcal{Q}$ deux partitions de $\mathcal{X}$. On dit que $\mathcal{Q}$ est une sur-partition de $\mathcal{P}$ si et seulement si, pour tout $A \in \mathcal{P}$ et pour tout $B \in \mathcal{Q}$, on a $B \subset A$ ou $B \cap A = \emptyset$. On dira que $\mathcal{Q}$ est plus fine que $\mathcal{P}$ et on notera $\mathcal{P} \preceq \mathcal{Q}$.*

PROPOSITION 1. *r et $\hat{r}_n$ sont décroissantes pour $\preceq$*

Démonstration. Soit $\mathcal{P}$ et $\mathcal{Q}$, deux partitions de $\mathcal{X}$ telles que $\mathcal{P} \preceq \mathcal{Q}$

$$\begin{aligned} r(\mathcal{P}) &= \sum_{P\in\mathcal{P}} min_{z\in\mathcal{Y}}(E[l(Y,z)\mathbf{1}_{X\in P}]) = \sum_{P\in\mathcal{P}} min_{z\in\mathcal{Y}}(\sum_{Q\subset P} E[l(Y,z)\mathbf{1}_{X\in Q}]) \\ &\geq \sum_{Q\in\mathcal{Q}} min_{z\in\mathcal{Y}}(E[l(Y,z)\mathbf{1}_{X\in Q}]) = r(\mathcal{Q}) \end{aligned}$$

un raisonnement similaire nous donne un résultat identique pour $\hat{r}_n$ □

Pour toute partition $\mathcal{P}$ et tout élément $P \in \mathcal{P}$, il est possible de raffiner la partition $\mathcal{P}$ en coupant P par un hyperplan séparateur (on parle alors de **splitting**). Ainsi, on considère une surpartition du type (pour $c \in [|1,p|]$ et $b \in \mathbb{R}$)

$$\mathcal{P}_{P,c,b} = (\mathcal{P} - \{P\})\ U\ \{P \cap \{x \in \mathcal{X}\ /\ \langle e_c, x\rangle > b\}, P \cap \{x \in \mathcal{X}\ /\ \langle e_c, x\rangle \leq b\}\}$$

On peut montrer qu'exprimer P en partition de deux hyperrectangles est équivalent à choisir des hyperplans séparateur du type $\langle e_c, x\rangle = b$.

Considérons la perte de risque associé au raffinement de $\mathcal{P}$ en $\mathcal{P}_{P,c,b}$:

$$r(\mathcal{P}) - r(\mathcal{P}_{P,c,b}) = r(P) - r(P \cap \{\langle e_c, x\rangle > b\}) - r(P \cap \{\langle e_c, x\rangle \leq b\})$$

Une approche de minimisation pas à pas reviens ainsi à **maximiser** $r(\mathcal{P}) - r(\mathcal{P}_{P,c,b})$ pour chaque cellule P de la partition. Cependant, r est toujours inconnu, et on préfère ainsi maximiser

$$\hat{r}_n(\mathcal{P}) - \hat{r}_n(\mathcal{P}_{P,i,b}) = \hat{r}_n(P) - \hat{r}_n(P \cap \{\langle e_c, x\rangle > b\}) - \hat{r}_n(P \cap \{\langle e_c, x\rangle \leq b\})$$

Remarque : en supposant P fixé, cela revient à considérer le programme d'optimisation suivant :

$$\min_{c\in[|1,p|],\ b\in\mathbb{R}} \hat{r}_n(P \cap \{\langle e_c, x\rangle > b\}) + \hat{r}_n(P \cap \{\langle e_c, x\rangle \leq b\}) \quad (Prog1)$$

Exemples :
Pour la perte $l(y,z) = \mathbf{1}_{y\neq z}$, le programme d'optimisation $(Prog1)$ est équivalent au suivant (en divisant la fonction objectif par $\mathbb{P}_n(P) = \frac{\sum_{i=1}^n \mathbf{1}_{X_i\in P}}{n}$ et en utilisant la probabilité empirique sur P : $\hat{\mathbb{P}}_{n,P} \overset{def}{=} \frac{\sum_{i=1}^n \delta_{(X_i,Y_i)}\mathbf{1}_{X_i\in P}}{N_n(P)}$)

$$\begin{aligned} \min_{c\in[|1,p|],\ b\in\mathbb{R}} & \hat{\mathbb{P}}_{n,P}(\langle e_c, X\rangle > b) \min_{j=0,1} \hat{\mathbb{P}}_{n,P}(Y = j|\langle e_c, X\rangle > b) \\ & + \hat{\mathbb{P}}_{n,P}(\langle e_c, X\rangle \leq b) \min_{j=0,1} \hat{\mathbb{P}}_{n,P}(Y = j|\langle e_c, X\rangle \leq b) \end{aligned}$$

Mesure d'impureté

Intuitivement, un algorithme d'apprentissage par partitionnement sera performant en classification binaire si sa partition parvien à séparer les $Y = 1$ ou $Y = 0$. Cela nous conduit à formuler une quantification de l'hétérogénéité d'une partition. On définit alors la notion de fonction d'impureté :

Définition 10 (fonction d'impureté)**.** $\Lambda : [0,1]^2 \to \mathbb{R}^+$ *est une fonction d'impureté si et seulement si elle vérifie les points suivants :*

— Λ *est invariante par permutation de ses arguments*
— Λ *est minimum* $(0,1)$ *et* $(1,0)$
— Λ *restreinte à* $\{(p_0,p_1)\ /\ p_0 \geq 0, p_1 \geq 0, p_1 + p_0 = 1\}$ *est maximum en* $(1/2, 1/2)$

Ainsi, si on a une proportion p_1 de 1 et p_0 de 0, alors $\Lambda(p_1,p_0)$ représente une mesure de l'inhomogénéité de la population. $\Lambda(p_1,p_0)$ est minimale lorsque $p_1 = 1$ (l'échantillon ne contient que des 1) ou $p_0 = 1$, $\Lambda(p_1,p_0) = 0$, et est maximale lorsque $p_1 = p_0 = \frac{1}{2}$.
Exemples de fonctions d'impureté : $p_1, p_0 \to p_1 p_0$ (fonction d'impureté de classification), $p_1, p_0 \to min(p_1,p_0)$ (fonction d'impureté de Gini), $p_1, p_0 \to p_0 \log(p_0) + p_1 \log(p_1)$.

grâce à cette formulation, on peut heuristiquement choisir b, et c de la manière suivante :

$$\begin{aligned} \min_{c\in[|1,p|]\ b\in\mathbb{R}} \quad & \hat{\mathbb{P}}_{n,P}(\langle e_c, x\rangle > b)\ \Lambda(\ \hat{\mathbb{P}}_{n,P}(Y = j|\langle e_c, X\rangle > b), \hat{\mathbb{P}}_{n,P}(Y = 1|\langle e_c, X\rangle > b)\ ) \\ & + \hat{\mathbb{P}}_{n,P}(\langle e_c, x\rangle \leq b)\ \Lambda(\ \hat{\mathbb{P}}_{n,P}(Y = j|\langle e_c, X\rangle \leq b), \hat{\mathbb{P}}_{n,P}(Y = j|\langle e_c, X\rangle \leq b)\ ) \end{aligned} \quad (Prog2)$$

On cherche ainsi à minimiser une mesure d'impureté agrégée sur une partition de P. Pour $l(y,z) = \mathbf{1}_{y \neq z}$, et $\Lambda : p_1, p_0 \to min(p_1, p_0)$, les programmes d'optimisation $(Prog1)$ et $(Prog2)$ sont équivalents. Ce résultat n'est pas un cas isolé, et on peut trouver de nombreux exemples pour lesquels ces deux raisonnements peuvent avoir des formulations identiques.

De manière générale, partant d'une fonction d'impureté Λ, il est possible de construire une mesure d'impureté (ou de l'hétérogénéité) d'une partition en agrégeant (proportionnellement au poids de chaque ensemble) des mesures d'impureté. Ces mesures d'impureté jouent l'analogue des fonction de risque.

Définition 11 (mesure d'impureté associé à Λ)**.**

On définit la fonction d'impureté r_Λ et la fonction d'impureté empirique $\hat{r}_{n,\Lambda}$ par :

$$r_\Lambda : \mathcal{P} \to \sum_{P \in \mathcal{P}} \mathbb{P}(P \cap \{\langle e_c, x\rangle \leq b\})\ \Lambda(\mathbb{P}(Y = j | \langle e_c, X\rangle \leq b\}), \mathbb{P}(Y = j | \langle e_c, X\rangle \leq b\}))$$
$$\hat{r}_{n,\Lambda} : \mathcal{P} \to \sum_{P \in \mathcal{P}} \hat{\mathbb{P}}_n(P \cap \{\langle e_c, x\rangle \leq b\})\ \Lambda(\hat{\mathbb{P}}_{n,P}(Y = j | \langle e_c, X\rangle \leq b\}), \hat{\mathbb{P}}_{n,P}(Y = j | \langle e_c, X\rangle \leq b\}))$$

Dans la suite, r_{Gini} désignera la mesure d'impureté de Gini. On vérifie que r_{Gini} et $\hat{r}_{n,Gini}$ se comportent comme des fonctions de risque (croissance par raffinement de partition et sur-additivité).

Théorème 6 (Consistance du partitionnement par la mesure d'impureté de Gini)**.** *r_{Gini} possède une borne inferieure $r^*_{Gini} = \inf_{\mathcal{P} \in \mathcal{P}art_{\mathcal{H}}} r^*_{Gini}(\mathcal{P})$ sur l'ensemble des partition en hyperrectangles. Deplus, $r_{Gini}(\mathcal{P}_n) \underset{n\to\infty}{\to} r^*_{Gini} \Rightarrow r(\mathcal{P}_n) \underset{n\to\infty}{\to} r^*$*

Ce théorème montre ainsi que partitionner par rapport à des mesures d'impuretés peut conduire à la consistance des partitions.

Arbre de partitionnement .

Définition 12 (arbre de partitionnement)**.** *Un arbre de partitionnement de $\mathcal{X}$ est un arbre étiqueté par des ensembles de $\mathcal{X}$ tels que :*

- *la racine soit étiquetée par $\mathcal{X}$*
- *si un noeud n'est pas terminal et est étiqueté par C, si $C_1, ..., C_k$ désignent l'ensemble des étiquettes des fils de ce noeud, alors $C_1, ..., C_k$ forment une partition de C.*

On vérifie sans peine que l'ensemble des étiquettes des feuilles d'un arbre de partitionnement forme une partition de $\mathcal{X}$. Pour un arbre de partitionnement $\mathcal{P}$, on notera $\mathcal{P}_\mathcal{T}$ la partition associée.

Dans toute la suite, on se restreindra à des arbres binaires (i.e. $k = 2$), et on confondra noeuds et étiquettes. Le programme d'optimisation précédent (Progr2) nous donne une manière de choisir un bon raffinement de partition pour un noeud P. Dans l'algorithme suivant, on suppose avoir une structure d'arbre de partitionnement autorisant l'insertion de fils, ainsi que d'une méthode $leafs$ qui renvoie l'ensemble des feuilles d'un arbre.

Algorithm 1: algorithme arbre de partitionnement :

input: $\mathcal{D}_n$, t_n nombres de noeuds de l'arbre
$\mathcal{U}nsplitable = \emptyset$ (noeuds que l'on ne peut plus splitter) $\mathcal{S}plitable = \mathcal{X}$ (noeuds à splitter)
$nbrNoeuds = 1$.
$\Lambda = p_0, p_1 \to p_0 p_1$
while $nbrNoeuds \leq t_n$ *et* $\mathcal{S}plitable \neq \emptyset$ **do**
  prendre $P \in \mathcal{S}plitable$ **if** $\#(P \times \{0\} \cap \mathcal{D}_n) > 0$ *et* $\#(P \times \{1\} \cap \mathcal{D}_n) > 0$ **then**
    //**remarque :** ici on est sur d'avoir une population homogène
    $\mathcal{U}nsplitable := \mathcal{U}nsplitable \cup \{P\}$
    $\mathcal{S}plitable := \mathcal{S}plitable - \{P\}$
  else
    déterminer i^*, b^* solution de

$$\min_{i \in [|1,p|],\ b \in \mathbb{R}} \quad \hat{\mathbb{P}}_n(P \cap \{\langle e_i, x\rangle > b\})\ \hat{\mathbb{P}}_{n,P}(Y = 1 | \langle e_i, X\rangle > b\})\hat{\mathbb{P}}_{n,P}(Y = 0 | \langle e_i, X\rangle > b\})$$
$$+\hat{\mathbb{P}}_n(P \cap \{\langle e_i, x\rangle \leq b\})\ \hat{\mathbb{P}}_{n,P}(Y = 1 | \langle e_i, X\rangle \leq b\}), \hat{\mathbb{P}}_{n,P}(Y = 0 | \langle e_i, X\rangle \leq b\}))$$

    $nbrNoeuds := nbrNoeuds + 1$
    Indiquer P $P \cap \{x\ \langle e_{i^*}, x\rangle \leq b^*\}$ et $P \cap \{x\ \langle e_{i^*}, x\rangle \leq b^*\}$ comme fils de P dans $\mathcal{T}$
  if $\mathcal{S}plitable = \emptyset$ **then**
    $\mathcal{S}plitable := leafs(\mathcal{T}) - \mathcal{U}nsplitable$

Remarque : la partition donnée par l'arbre de décision est bien controlée en taille, mais le risque de cette partition peut être très loin de $\inf_{\mathcal{P} \in \mathcal{P}art_{H,t_n}} \hat{r}_n(\mathcal{P})$. Ainsi, il n'est pas possible de conclure sur la consistance à ce stade.

3.3 Prunning at algorithme CART

L'algorithme CART (initiales de Classification and Regression Tree, voir [7]) prend le parti de faire grandir des arbres jusqu'à obtenir des noeuds purs. Cela revient à appliquer l'algorithme 1 en remplaçant t_n par ∞. L'échantillon d'apprentissage étant fini, la partition ainsi obtenue minimise le risque empirique. En effet lorsqu'on aboutit à un nœud pur (i.e ne possédant qu'une seule modalité de Y), l'algorithme ne peut plus minimiser la fonction de risque empirique ($\hat{r}_n$) en raffinant la partition au niveau de ce nœud car ce dernier est homogène.

Afin de se prémunir contre le sur-apprentissage, l'algorithme CART réduit donc la taille de la partition en sélectionnant un arbre parmi tous les sous arbres de $\mathcal{T}_{max}$, le sous arbre sélectionné était celui minimisant le critère $r(\mathcal{P}_\mathcal{T}) + \alpha\ \#(\mathcal{P}_\mathcal{T})$ avec α, un paramètre choisi de façon adaptative. On notera $\preceq$ la relation *être le sous arbre de* parmi les arbres de partitionnement. Cependant, r est inconnu, on adopte en pratique des stratégies d'approximation par des estimateurs empirique, ou de cross-validation. Une telle fonction de risque a l'avantage d'avoir quelques propriétés d'existence et d'unicité du sous arbre minimisant (c.f. [7]) :

Théorème 7 (Breiman). *Pour toute fonction de risque r (ou $\hat{r}_n$, r_{Gini} , $r_{n,Gini}$), pour tout $\alpha > 0$, et pour tout arbre de partitionnement $\mathcal{T}$, il existe un unique sous arbre $\mathcal{T}(\alpha)$ de $\mathcal{T}$ tel que :* $\mathcal{T}(\alpha) \in \underset{\mathcal{T}' \preceq \mathcal{T}}{Argmin}\ r(\mathcal{P}_{\mathcal{T}'}) + \alpha\ \#(\mathcal{P}_{\mathcal{T}'})$ *et* $\mathcal{T}^\# \in \underset{\mathcal{T}' \preceq \mathcal{T}}{Argmin}\ r(\mathcal{P}_{\mathcal{T}'}) + \alpha\ \#(\mathcal{P}_{\mathcal{T}'}) \Rightarrow \mathcal{T}(\alpha) \preceq \mathcal{T}^\#$.

De plus, pour tout $\alpha_2, \alpha_1 > 0$, on a :

- $\alpha_2 > \alpha_1 \Rightarrow \mathcal{T}(\alpha_2) \preceq \mathcal{T}(\alpha_1)$
- $\mathcal{T}(\alpha_2) \prec \mathcal{T}(\alpha_1) \Rightarrow \alpha_2 > \alpha_1$
- $(\mathcal{T}(\alpha_1))(\alpha_2) = \mathcal{T}(\alpha_2)$

Comme le nombre de sous arbres de $\mathcal{T}_{max}$ est fini, on peux en fait montrer :

Théorème 8. *Pour toute fonction de risque r (ou $\hat{r}_n$, r_{Gini} , $r_{n,Gini}$), $\exists 0 \leq \alpha_1 < ... < \alpha_K < +\infty$ et $\{\mathcal{X}\} = \mathcal{T}_K \prec ... \prec \mathcal{T}_0 = \mathcal{T}_{max}$ tel que*

$$\forall \alpha > 0\ \alpha \in [\alpha_j, \alpha_{j+1}[\Rightarrow \mathcal{T}(\alpha) = \mathcal{T}(\alpha_k)$$

(avec la convention que $\alpha_{K+1} = +\infty$)

Ces théorèmes sont à la base de méthodes efficaces permettant de calculer rapidement les α_i et les $\mathcal{T}(\alpha_i)$. Ces méthodes interviennent dans l'algorithme CART rapellé ci-dessous.

Algorithm 2: algorithme CART

input: $\mathcal{D}_n$. $N_c \in \mathbb{N}$ de classes pour la cross validation.

1. Répartir aléatoirement $\mathcal{D}_n$ en N_c échantillons $\mathcal{D}^{(1)}, ..., \mathcal{D}^{(N_c)}$
2. **for** $v = 1, ..., N_c$ **do**
   - Construire $\mathcal{T}^v_{max}$ à partir de $\mathcal{D}^{(-v)} = \mathcal{D}_n - \mathcal{D}^{(v)}$ suivant l'aglorithme 1.
   - Calcler $\alpha^{(v)}_0, ..., \alpha^{(v)}_{k(v)}$ et $\mathcal{T}^{(v)}(\alpha^{(v)}_0), ..., \mathcal{T}^{(v)}(\alpha^{(v)}_{k(v)})$ à partir de $r_{\mathcal{D}^{(v)}}$.
   - Poser $\mathcal{T}^{(v)} : \alpha \to \sum_{i=0}^{k(v)} \mathbf{1}_{\alpha \in [\alpha^{(v)}_i, \alpha^{(v)}_{i+1}[} \mathcal{T}^{(v)}(\alpha^{(v)}_i)$,
   - Calculer la fonction de risque empirique de classification $L_{\mathcal{D}^{(v)}}$ avec l'échantillon $\mathcal{D}^{(v)}$.
3. Poser $L^{cv}_n(\alpha) = \frac{1}{N_c} \sum_{i=1}^{N_c} L_{\mathcal{D}^{(v)}}(\mathcal{T}^{(v)}(\alpha))$
4. Faire grandir l'arbre $\mathcal{T}_{max}$ à partir de $\mathcal{D}_n$ suivant l'algorithme 1.
5. Calculer $\alpha_0, ..., \alpha_k$, et $\mathcal{T}_0, ..., \mathcal{T}_k$ à partir de $r_{\mathcal{D}_n}$)
6. Poser $\alpha'_i = \sqrt{\alpha_i \alpha_{i+1}}$ pour $i = 0, ..., k$, et choisir l'arbre $\mathcal{T}_{i'}$ avec $i' = \inf\{i \; L^{cv}_n(\alpha'_i) \leq \epsilon + \min_j L^{cv}_n(\alpha'_j)\}$, et renvoyer $\hat{g}^*_{\mathcal{P}_{mathcalT_{i'}}}$

Nous considérerons l'algorithme simplifié suivant. Cette version à l'avantage de ne pas s'embarrasser à selectionner le poids de pénalisation de manière adaptative (les α_i), et est ainsi plus simple à manipuler.

Algorithm 3: CART simplifié (classification)

input: $\mathcal{D}_n$, α_n

1. Fait grandir un arbre $\mathcal{T}_n$ suivant l'algorithme 1 avec $t_n = \infty$.
2. Choisir un arbre $\mathcal{T}^*_n$ dans $\underset{\mathcal{T} \preceq \mathcal{T}_n}{Argmin} \hat{r}_n(\mathcal{P}_\mathcal{T}) + \alpha_n |\mathcal{P}_\mathcal{T}|$
3. Calculer la fonction de décision $g^*_{\mathcal{P}_{\mathcal{T}^*_n}}$ défini par $g^*(P) \in \underset{z \in \{0,1\}}{Argmax} \sum_{i=1}^n \mathbf{1}_{X_i \in P \text{ et } Y_i = z} \quad \forall P \in \mathcal{P}_{\mathcal{T}^*_n}$

On peut alors montrer le résultat suivant :

Théorème 9 (consistance de CART). *Pour toute suite réelle $(\alpha_n)_{n \in \mathbb{N}}$ telle que $\frac{\sqrt{n}}{\sqrt{\log(n)}\alpha_n}$ soit bornée et $\alpha_n \underset{n \to \infty}{\to} \infty$. Si il existe une suite $(\mathcal{T}'_n)$ de sous arbres de $(\mathcal{T}_n)$ (i.e. $\mathcal{T}'_n \preceq \mathcal{T}_n \forall n \in \mathbb{N}$) consistante pour r_{Gini} et telle que $\#\mathcal{P}_{\mathcal{T}'_n} = o(\sqrt{(\frac{n}{\log(n)})})$, alors l'algorithme 3 décrit ci-dessus est consistant.*

Remarque : La condition d'existence d'une suite $(\mathcal{T}'_n)$ de sous arbres de $(\mathcal{T}_n)$ (*i.e.* $\mathcal{T}'_n \preceq \mathcal{T}_n \forall n \in \mathbb{N}$) consistante pour L_{Gini} et telle que $|\tilde{\mathcal{T}}'_n| = o(\sqrt{(\frac{n}{\log(n)})})$ est très restrictive.

Il est possible de tels résultats de consistance de suite de sous arbres. Voir par exemple [11] où la condition est démontrée lorsque $(\mathcal{T}_n)$ est α-équilibrée pour tout n et $\max_{P \in \tilde{\mathcal{T}}_n} (diam(P)) \log(n) \underset{n \to \infty}{\to} 0$.

4 Forêts Aléatoires

De nombreuses techniques d'apprentissage statistique se basent sur la combinaison de plusieurs classifieurs instables (par exemple, des réseau de neurones, boosting, ...). Des résultats ont montrés que ces versions améliorés peuvent être plus performantes (i.e. un taux d'erreur plus faible). Les forêts aléatoires appartiennent cette famille de techniques. L'idée est d'agréger des arbres de décisions, chacun ayant été généré à partir d'un sous échantillon de $\mathcal{D}_n$ tiré aléatoirement (on parle aussi d'échantillon bootstrap). Ce genre d'algorithme (qui consiste à aggreger des résultats d'autres algorithmes générés sur des échantillons bootstrap) est aussi appelé bagging (racourcis de bootstrap-aggregating).

En toute généralité, le bootstrap est un ensemble de méthodes statistiques qui permettent d'estimer des assympototiquement des quantités de la forme $\mathbb{P}_{Z_1,...,Z_k}(f_n(Z_1,...,Z_k) \in C)$ à partir d'un échantillon $z_1,...,z_n$. Nous renvoyons à [1] pour une introduction à la matière pour le cadre i.i.d. . Dans le cadre des forêts aléatoire, le terme bootstrap est simplement utilisé pour mentionner que $\hat{\mathcal{D}}_{a_n}$ est un échantillon tiré aléatoirement (avec ou sans remise) parmi les éléments de $\mathcal{D}_n$.

Algorithm 4: Bagging pour un algorithme de classification binaire

input: $\mathcal{D}_n$, un algorithme $m : \bigcup_{n\in\mathcal{N}} (\mathcal{X}\times\mathcal{Y})^n \to \mathcal{Y}^{\mathcal{X}}$, a_n taille des échantillons bootstrap, B taille du bagging

1. On génère B échantillons bootstrap à a_n observations, construits à partir de $\mathcal{D}_n$: $\hat{\mathcal{D}}^1_{a_n}$,...,$\hat{\mathcal{D}}^B_{a_n}$
2. On construit $(m(\hat{\mathcal{D}}^i_{a_n}))_{i\in\mathcal{B}}$
3. On agrège les prédicteurs dans le sens suivant :
   Dans le cas de la classification, la forêt est définie comme la fonction renvoyant le vote majoritaire *i.e* $\mathcal{B}agging : x \in \mathcal{X} \to \underset{z\in\{0,1\}}{Argmax}\big(\sum_{i=1}^{B} \mathbf{1}_z(m(\hat{\mathcal{D}}^i_{a_n})(x))\big)$

Il est possible de montrer le résultat suivant.

Théorème 10. *Consistance du bagging Soit m, un algorithme de classification binaire. Si :*

- $\mathbb{P}_{\mathcal{D}_n}(m(\mathcal{D}_n,x)=z) \underset{n\to\infty}{\to} \mathbb{P}(Y=z|X=x)\ \forall z\in\{0,1\}$ *et* $\forall x\in\mathcal{X}\in\mathcal{X}$
- $\max_{z\in\{0,1\}} \big(\mathbb{P}(Y=z|X) - \max_{z'\in\{0,1\}\ z'\neq z} \mathbb{P}(Y=z|X)\big)\ X\mathbb{P}_X p.s.$
- $\frac{a_n}{n} \underset{n\to\infty}{\to} 0$

Alors l'algorithme du bagging décrit par l'algorithme précédent est consistant pour le risque de classification.

Cependant, dans la version introduite par Breiman en 2001 (voir [6]) les algorithmes de base considérés (les m) sont des arbres qui ont été grandi jusqu'au bout (cf algorithme 1 ave $t_n = +\infty$) pour lesquels la condition de split (séparation d'un noeud en ses fils) revient à résoudre le programme d'optimisation ($Prog2$) pour j appartenant à un sous échantillon de coordonnées tirées au hasard (i.e. un sous ensemble de $[|1,p|]$), et en ne faissant pas de pruning.
Ainsi, les arbres de la forêt aléatoire comportent un élément d'aléa que l'on représente par une variable aléatoire Θ. Cette variable aléatoire (indépendante de l'échantillon $\mathcal{D}_n$) regroupe également l'aléa du au tirage d'échantillons bootstrap.

Algorithm 5: Forêt aléatoire en classification binaire

input: $\mathcal{D}_n$, un algorithme, a_n taille des échantillons bootstrap, B taille du bagging, $mtry \in [|1,p|]$ le nombre de coordonnées à tirer pour splitter chaque noeuds

1. On génère B échantillons bootstrap à a_n observations, construits à partir de $\mathcal{D}_n$: $\hat{\mathcal{D}}^1$,...,$\hat{\mathcal{D}}^B$
2. On construit des arbres de partitionnement : **for** k=0,...,B **do**

$\mathcal{U}nsplitable = \emptyset$ (noeuds que l'on ne peut plus splitter) $\mathcal{S}plitable = c$ (noeuds à splitter)
$\mathcal{T}_\| = root(\mathcal{X})$ **while** $\mathcal{S}plitable \neq \emptyset$ **do**
prendre $P \in \mathcal{S}plitable$ **if** $\#(P \times \{0\} \cap \hat{\mathcal{D}}^k) > 0$ *et* $\#(P \times \{1\} \cap \hat{\mathcal{D}}^k) > 0$ **then**
//**remarque :** ici on est sur d'avoir une population homogène
$\mathcal{U}nsplitable := \mathcal{U}nsplitable \cup \{P\}$
$\mathcal{S}plitable := \mathcal{S}plitable - \{P\}$
else
$\mathcal{M}try :=$ résultat du tirage de mtry éléments (sans remise) de $[|1,p|]$
déterminer i^*, b^* solution de

$$\min_{i \in \mathcal{M}try,\ b \in \mathbb{R}} \quad \hat{\mathbb{P}}_{n,P}(\langle e_i, X\rangle > b)\, \hat{\mathbb{P}}_{n,P}(Y = 1 | \langle e_i, X\rangle > b) \hat{\mathbb{P}}_{n,P}(Y = 0 | \langle e_i, X\rangle > b\}) + \hat{\mathbb{P}}_{n,P}(\langle e_i, x\rangle \leq b)\, \hat{\mathbb{P}}_{n,P}(Y = 1 | \langle e_i, X\rangle \leq b\})\, \hat{\mathbb{P}}_{n,P}(Y = 0 | \langle e_i, X\rangle \leq b\}))$$

Indiquer P $P \cap \{x\ \langle e_{i^*}, x\rangle \leq b^*\}$ et $P \cap \{x\ \langle e_{i^*}, x\rangle \leq b^*\}$ comme fils de P dans $\mathcal{T}$
if $\mathcal{S}plitable = \emptyset$ **then**
$\mathcal{S}plitable := leafs(\mathcal{T}) - \mathcal{U}nsplitable$
definir $m_n(\Theta_k) : x \in \mathcal{X} \to \sum_{P \in \mathcal{T}_k} \mathbf{1}_{x \in \mathcal{P}} Y_P$.

3. On agrège les prédicteurs en définissant :
   — la forêt d'estimation des probabilités qui définit en chaque point les pourcentages de vote des arbres. $m_{n,B} : x \in \mathcal{X} \to \frac{1}{B} \sum_{i=1}^{B} m_n(\Theta_k, x)$
   — La forêt de classification, qui est définie comme la fonction renvoyant le vote majoritaire des arbres *i.e* $\mathcal{F}orest_{n,B} : x \to \mathbf{1}_{m_{n,B}(\Theta_1,...,\Theta_B,x) > \frac{1}{2}}$

Tous les tirages (échantillon bootstrap et $\mathcal{M}try$) étant réalisés de manière indépendante, $\Theta_1, ..., \Theta_k$ sont également indépendants.

Par ailleurs, les arbres de la forêt (les $m_n(\Theta_k, .)$) étant grandis jusqu'à atteindre une condition de pureté du noeud, la consistance de tels arbres semble peu probable (intuitivement, l'estimateur local de $\mathbb{P}(Y = 1|X = x)$ ne semble pas consistant). Or, cet algorithme de forêts aléatoire réalise des résultats comparables aux meilleurs algorithmes de classification.
Ainsi, il semblerait que l'agrégation de ces classifieurs non consistants permette de produire une approche efficace.

4.1 Forêt infinie et moyennage

Définissons la forêt infinie (de l'estimation de probabilité) par $m_{\infty,n} : x \to E_\Theta[m_n(\Theta, x)]$ (qui est une fonction de $\mathcal{D}_n$). Pour tout $x \in \mathcal{X}$, par la loi forte des grands nombres :

$$m_{B,n}(x) \underset{n \to \infty}{\overset{p.s.}{\to}} m_{\infty,n}(x)$$

On remarque par ailleurs que les probabilités empirique dans le programme d'optimisation partitionnement d'un noeud (Progr2) sont des fonctions de (i, b) constantes par morceau, et ne prennant qu'un nombre fini de valeurs. Ainsi, on peut montrer que l'on peut choisir sans perte de généralité $b \in \{\langle e_i, X_k\rangle \ / \ k \in [|1,n|]\ i \in [|i,p|]\}$.
Sous cette condition, on peux montrer que :

Théorème 11. *Si on suppose que* $b \in \{\langle e_i, X_k\rangle \ / \ k \in [|1,n|]\ i \in [|i,p|]\}$, *alors conditionnellement à* $\mathcal{D}_n$, $x - p.s.$,

$$m_{B,n}(x) \underset{n \to \infty}{\overset{p.s.}{\to}} m_{\infty,n}(x)$$

La preuve du théorème passe par la définission d'une famille approprié de fonctions appartenant à une même classe de Donsker. Voir [12] pour plus de detail.

Ainsi, il suffirait de montrer que $r(m_{\infty,n}) \underset{n\to\infty}{\to} r^*$ pour obtenir la consistance des forêts aléatoires

4.2 Forêts aléatoires infinies

A toute partition $\mathcal{P}$ de $\mathcal{X}$ est associée une relation d'équivalence :
$x \overset{\mathcal{P}}{\leftrightarrow} z \Leftrightarrow$ x et z appartiennent au même ensemble pour la partition $\mathcal{P}$. On préfèrera utiliser la notation $x \overset{\Theta_k}{\leftrightarrow} z$ pour désigner $x \overset{\mathcal{P}_{\mathcal{T}_k}}{\leftrightarrow} z$. On notera $N_n(x,\Theta_k) = \#\{X_i \ / \ X_i \overset{\Theta_k}{\leftrightarrow} x \text{ et } i \in [|1,n|]\}$ (nombre d'éléments de $\mathcal{D}_n$ tombant dans la même classe que x).
En remarquant les partitions générées par les arbres de la forêt de Breiman sont homogènes,

$$m_n(\Theta_k, x) = \sum_{i=1}^{n} Y_i \frac{\mathbf{1}_{X_i \overset{\Theta_k}{\leftrightarrow} x}}{N(x,\Theta)}$$

$$m_{\infty,n}(x) = \sum_{i=1}^{n} Y_i E_\Theta[\frac{\mathbf{1}_{X_i \overset{\Theta_k}{\leftrightarrow} x}}{N(x,\Theta)}] = \sum_{i=1}^{n} Y_i W_{n,i}^{\infty}(x)$$

avec $W_{n,i}^{\infty}(x) = E_\Theta[\frac{\mathbf{1}_{X_i \overset{\Theta_k}{\leftrightarrow} x}}{N(x,\Theta)}]$

Les $W_{n,i}^{\infty}$ peuvent être vus comme des poids car ils sont positifs et $\sum_{i=1}^{n} W_{n,i}^{\infty}(x) = 1$ pour tout $x \in \mathcal{X}$. Cette formulation de moyennage local nous fait fortement penser au théorème de Stone que nous énonçons ici une version affaiblie

Théorème 12. *Stone pour l'estimation de probabilité Soit $W_{n,i} : \mathcal{X} \to \mathbb{R}^+$ $(1 \le i \le n)$, des poids fonction uniquement des $(X_i)_{i\in[|1,n|]}$ vérifiant :*
- *$\exists c > 0$ tel que $\forall f : \mathcal{X} \to \mathbb{R}^+$ $E[\sum_{i=1}^{n} W_{n,i}(X) f(X_i)] \le c\, E[f(X)]$*
- *$\sum_{i=1}^{n} W_{n,i}^{\infty}(x) = 1$ sur $\mathcal{X}$*
- *$\forall a > 0 E[\sum_{i=1}^{n} W_{n,i} \mathbf{1}_{||X_i - X|| > a}] \underset{n\to\infty}{\to} 0$*
- *$E[\sum_{i=1}^{n} W_{i,n}(X)^2] \underset{n\to\infty}{\to} 0$*

Si $\mathcal{Y} = \{0,1\}$ alors,

$$E[(\sum_{i=1}^{n} W_{n,i}(X) Y_i - \mathbb{P}(Y = 1|X))^2] \underset{n\to\infty}{\to} 0$$

Voir [9], chapitre 5 pour une preuve.

Cependant, la condition d'indépendance des poids $(W_{n,i})$ vis à vis des (Y_i) est capitale dans la démonstration (cette hypothèse permet d'annuler un terme en
$E[(\sum_{j=1}^{n}\sum_{i=1}^{n} W_{n,i}(X) W_{n,j}(Y_i - \mathbb{P}_{Y_i}(Y_i = 1|X_i))(Y_j - \mathbb{P}_{Y_j}(Y = 1|X_j))]$ en conditionnant par rapport à $(X, X_1, ..., X_n)$)).

Or, les arbres de la forêt étant construits de manière à maximiser l'homogénéité suivant $\mathcal{Y}$, les Y_i interviennent dans les poids $W_{n,i}^{\infty}$.

Ainsi, ce théorème n'est pas applicable tel quel. Certains articles parviennent à prouver des résultats de consistance sous certaines hypothèses fortes sur la probabilité ayant généré les observations (par exemple, dans [3], on se restreint à des modèles quasi-linéaires auquel on rajoute des termes d'erreurs gaussien).
Aucune preuve de consistance générale n'a été établie à ce jour.

5 Références

[1] Bertail. Bootstrap in the i.i.d case.

[2] Lugosi Biau, Devroye. Consistency of random forest and other averaging classifiers. *Machine Learning*, 2015.

[3] Vert Biau, Scornet. Consistency of random forest. *Machine Learning*, 2015.

[4] Breiman. Arcing classifiers. *Machine Learning*, 1998.

[5] Breiman. Consistency for a simple model of random forest. *Machine Learning*, 2001.

[6] Breiman. Random forests. *Machine Learning*, 2001.

[7] Olshen et Stone Breiman, Friedman. *Classification and Regression Trees*. 1984.

[8] Dasgupta, Papadimitriou, and Vazirani. *Algorithms*. 2009.

[9] Lugosi Devroye, Gyorfi. *A Probabilistic Theory of Pattern Recognition*. 1991.

[10] Friedman Hastie, Tibshirani. *The Elements of Statistical Learning*. 2009.

[11] Nobel. Analysis of a complexity based pruning scheme. *Machine Learning*, 2015.

[12] Scornet. On the asymptotics of random forests. *Journal of Multivariate Analysis*, 2016.

[13] van der Vaart. *Asymptotic Statistics*. 2000.