

Arbres CART et Forêts aléatoires, Importance et sélection de variables

Robin Genuer, Jean-Michel Poggi

▶ **To cite this version:**

Robin Genuer, Jean-Michel Poggi. Arbres CART et Forêts aléatoires, Importance et sélection de variables. 2017. hal-01387654v2

HAL Id: hal-01387654
https://hal.science/hal-01387654v2

Preprint submitted on 16 Jan 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Arbres CART et Forêts aléatoires
Importance et sélection de variables

Robin Genuer[1] et Jean-Michel Poggi[2]

[1]ISPED, Univ. Bordeaux, INSERM U-1219 & INRIA, SISTM, France ,
robin.genuer@u-bordeaux.fr
[2] LMO, Univ. Paris-Sud Orsay & Univ. Paris Descartes, France,
jean-michel.poggi@math.u-psud.fr

16 janvier 2017

Résumé

Deux des algorithmes proposés par Leo Breiman : les arbres CART (pour Classification And Regression Trees) introduits dans la première moitié des années 80 et les forêts aléatoires apparues, quant à elles, au début des années 2000, font l'objet de cet article. L'objectif est de proposer sur chacun des thèmes abordés, un exposé, une garantie théorique, un exemple et signaler variantes et extensions. Après un préambule, l'introduction rappelle les objectifs des problèmes de classification et de régression avant de retracer quelques prédécesseurs des forêts aléatoires. Ensuite, une section est consacrée aux arbres CART puis les forêts aléatoires sont présentées. Ensuite, une procédure de sélection de variables basée sur la quantification de l'importance des variables est proposée. Enfin l'adaptation des forêts aléatoires au contexte du Big Data est esquissée.

Mots clés : CART, Forêts aléatoires, Importance des variables, Sélection de variables, Big data.

Abstract

Two algorithms proposed by Leo Breiman : CART trees (Classification And Regression Trees for) introduced in the first half of the 80s and random forests emerged, meanwhile, in the early 2000s, are the subject of this article. The goal is to provide each of the topics, a presentation, a theoretical guarantee, an example and some variants and extensions. After a preamble, introduction recalls objectives of classification and regression problems before retracing some predecessors of the Random Forests. Then, a section is devoted to CART trees then random forests are presented. Then, a variable selection procedure based on permutation variable importance is proposed. Finally the adaptation of random forests to the Big Data context is sketched.

Keywords : CART, Random Forests, Variable Importance, Variable selection, Big data.

Table des matières

1 Préambule **5**

2 Introduction **6**
2.1 Objectifs . . . 6
2.1.1 Régression . . . 6
2.1.2 Classification . . . 7
2.1.3 Une remarque sur l'espace d'entrée . . . 8
2.2 Un peu d'histoire . . . 8
2.2.1 Principe des méthodes d'ensemble . . . 8
2.2.2 Bagging . . . 9
2.2.3 Boosting . . . 9
2.2.4 Randomizing Outputs . . . 10
2.2.5 Random Subspace . . . 10
2.2.6 Les ingrédients clés . . . 10
2.3 Un jeu de données fil rouge . . . 11

3 Arbres CART **11**
3.1 Construction de l'arbre maximal . . . 12
3.2 Élagage . . . 13
3.3 Garantie théorique . . . 15
3.4 Interprétabilité et instabilité . . . 15
3.4.1 Découpes compétitives . . . 16
3.4.2 Valeurs manquantes . . . 16
3.4.3 Interprétabilité . . . 16
3.4.4 Valeurs aberrantes . . . 17
3.4.5 Complexité algorithmique . . . 17
3.5 Exemple . . . 17
3.6 Extensions . . . 19

4 Des arbres aux forêts aléatoires **20**
4.1 Définition générale des forêts aléatoires . . . 20
4.2 Bagging . . . 21
4.3 Forêts aléatoires "Random Inputs" . . . 22
4.4 Erreur OOB . . . 24
4.5 Garantie théorique . . . 25
4.6 Exemple . . . 26
4.7 Variantes et extensions . . . 26
4.7.1 Variantes . . . 26
4.7.2 Extensions . . . 27

5 Forêts aléatoires et sélection de variables **28**
5.1 Importance des variables . . . 29
5.2 Sélection de variables . . . 30
5.3 Garantie théorique . . . 32
5.4 Exemple . . . 33
5.5 Variantes et extensions . . . 34
5.5.1 Variantes des mesures d'importance . . . 34
5.5.2 Extension à la sélection de variables fonctionnelles . . . 35

6 Forêts aléatoires et Big Data **35**
6.1 Passage à l'échelle 35
6.2 Forêts aléatoires en ligne 37
6.3 Echantillonnage et BLB 37
6.4 Garantie théorique 38

7 Remerciements **38**

Bibliographie **38**

1 Préambule

Comme l'indique son titre, cet article traite de deux des algorithmes proposés par Leo Breiman : les arbres CART (pour Classification And Regression Trees) introduits dans la première moitié des années 80 et les forêts aléatoires apparues, quant à elles, au début des années 2000, marquant ainsi une période d'intense évolution conjointe de la statistique et de ce que l'on appelle aujourd'hui l'apprentissage statistique. La trajectoire des intérêts de Leo Breiman, dont la biographie scientifique est esquissée dans l'intéressante conversation Olshen and Breiman (2001) et dans Cutler (2010), constitue sans nul doute un trait magistral de ces disciplines. Il a en effet, après avoir développé son activité dans le domaine des probabilités sous un angle très proche des mathématiques pures, marqué de son empreinte la statistique appliquée et l'apprentissage.

Les arbres de décision donnent lieu à des méthodes très versatiles permettant de traiter semblablement le cas de la régression, de la classification bi-classe ou multi-classe ou encore de mélanger des variables explicatives quantitatives et qualitatives. CART est d'une certaine manière la première pierre de l'édifice des méthodes d'arbres qui, en général, héritent de ces propriétés. En effet, bien que connue depuis les années 60, la méthode des arbres de décision souffrait de fortes critiques justifiées et CART leur offre un cadre conceptuel de type sélection de modèles, qui leur confère ainsi à la fois une large applicabilité, une facilité d'interprétation et des garanties théoriques.

Mais l'un des défauts majeurs, intrinsèque, demeure : l'instabilité. Le remède, original et très profondément statistique, consiste à exploiter la variabilité naturelle des méthodes d'estimation en conjuguant deux mécanismes fondamentaux : la perturbation aléatoire des arbres et la combinaison d'un ensemble d'arbres plutôt que la sélection de l'un d'entre eux. Plusieurs algorithmes ont été proposés, la plupart par Breiman, comme le Bagging (Breiman (1996)) et diverses variantes de Arcing (Breiman (1998)) mais pas tous, en particulier le célèbre Adaboost (Freund and Schapire (1997)) qui fut développé sur des bases conceptuelles assez différentes (théorie des jeux). Cet ensemble de propositions sont surpassées, au moins sur le plan expérimental, par les forêts aléatoires (abrégées parfois RF pour Random Forests dans la suite).

En effet, les forêts aléatoires sont une méthode de statistique non-paramétrique aux performances exceptionnelles très tôt remarquées et sans cesse confirmées depuis. Même si ce type d'exercice est un peu artificiel, on peut le saisir au travers des multiples évaluations massives menées périodiquement dans les revues appliquées de Machine Learning (comme par exemple *Journal of Machine Learning Research* ou encore *Pattern Recognition Letters*) et dans lesquelles elles ressortent systématiquement dans les 2 ou 3 meilleures. La dernière et très remarquée peut être trouvée dans le papier de Fernández-Delgado et al. (2014) qui couronne les RF, alors que, moins d'une dizaine d'années auparavant, le papier Wu et al. (2008) mentionne CART mais pas encore les forêts aléatoires. Introduites par Breiman (2001), elles sont depuis de plus en plus utilisées pour traiter de nombreuses données réelles dans des domaines d'application variés, citons par exemple l'étude des biopuces (Díaz-Uriarte and Alvarez De Andres (2006)), l'écologie (Prasad et al. (2006)), la prévision de la pollution (Ghattas (1999)) ou encore la génomique (Goldstein et al. (2010) et Boulesteix et al. (2012)), et pour une revue plus large, voir Verikas et al. (2011).

Au delà des performances et du caractère automatique de la méthode avec très peu de paramètres à régler, l'un des aspects les plus importants sur le plan appliqué est sans nul doute la quantification de l'importance des variables. Cette notion peu examinée par les statisticiens (voir par exemple le papier de synthèse Grömping (2015) en régression) trouve dans le cadre des forêts aléatoires une définition commode, facile à évaluer et s'étendant naturellement au cas des groupes de variables (voir Gregorutti et al. (2015)).

Le plan de ce papier est le suivant. Ce préambule se poursuit par la section 2 qui rappelle les objectifs des problèmes de classification et de régression avant de retracer quelques prédécesseurs des forêts aléatoires. La section 3 est consacré aux arbres CART suivie de la section

4 présentant les forêts aléatoires. Ensuite, la section 5 propose une procédure de sélection de variables basée sur la quantification de l'importance des variables. Enfin la section 6 esquisse l'adaptation des forêts aléatoires au contexte du Big Data.

2 Introduction

Le problème de base consiste à s'intéresser au couple aléatoire (X, Y), où les variables explicatives $X \in \mathcal{X}$ (typiquement $\mathcal{X} = \mathbb{R}^p$ mais on peut aussi considérer $\mathcal{X}$ de la forme $\mathcal{X} = \mathbb{R}^{p'} \otimes \mathcal{Q}$ pour mélanger variables explicatives quantitatives et qualitatives) sont utilisées pour expliquer la variable réponse $Y \in \mathcal{Y}$. L'espace $\mathcal{Y} = \mathbb{R}$ pour la régression et $\mathcal{Y} = \{1, \ldots, L\}$ pour la classification. Le cadre général est donc celui de l'estimation ou de la prédiction à partir de données d'un échantillon $\mathcal{L}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, les X_i sont appelées entrées et les Y_i sorties, vues comme la réalisation de n variables aléatoires i.i.d. de même loi que (X, Y).

2.1 Objectifs

Nous parlons d'un problème de prédiction lorsque la prédiction $\hat{y}$ doit être la plus proche possible de la vraie réponse y, associée à x. Il existe une autre façon de voir le problème, c'est le point de vue de l'estimation. Il s'agit dans ce cas d'estimer la fonction (inconnue) qui à X associe Y. Bien entendu, ce problème est relié au précédent : si nous disposons d'une "bonne" estimation du lien entre X et Y, nous pourrons a fortiori "bien" prédire la sortie correspondant à une nouvelle entrée x. Néanmoins, *a contrario*, il est parfois possible de bien prédire alors que l'estimation de la fonction qui relie X et Y n'est pas très bonne.

Il existe deux principaux cadres en apprentissage statistique : la régression et la classification, qui diffèrent par la nature de la sortie Y.

2.1.1 Régression

Le cadre de la régression est celui où la réponse Y est continue, typiquement lorsque $\mathcal{Y} = \mathbb{R}$. Le modèle statistique s'écrit alors sous la forme suivante :

$$Y = s(X) + \varepsilon \tag{1}$$

La fonction de régression $s : \mathcal{X} \to \mathbb{R}$ est inconnue et nous cherchons à l'estimer à partir des mesures (X_i, Y_i) dont nous disposons dans l'échantillon $\mathcal{L}_n$. Ces mesures sont des observations de $s(X_i)$ bruitées par des variables aléatoires ε_i. Pour des raisons d'identifiabilité, nous supposons que la variable de bruit ε est centrée conditionnellement à X : $\mathrm{E}[\varepsilon|X] = 0$. Il existe alors une unique fonction s qui satisfait $s(X) = \mathrm{E}[Y|X]$.

Ce modèle statistique est appelé modèle de régression non-paramétrique puisqu'essentiellement aucune contrainte *a priori* n'est imposée à la fonction de régression s, contrairement aux modèles paramétriques comme par exemple le modèle de régression linéaire. Dans un tel modèle, on cherche en effet s sous la forme d'une combinaison linéaire des coordonnées des composantes de X et les coefficients de cette combinaison linéaire, appelés les paramètres du modèle, sont à estimer.

Dans le cadre du modèle (1), nous introduisons deux mesures de qualité : l'une pour le problème de prédiction, l'autre pour le problème d'estimation.

— Étant donné un prédicteur $\hat{h}$, c'est-à-dire une fonction de $\mathcal{X}$ dans $\mathbb{R}$, construite sur l'échantillon d'apprentissage $\mathcal{L}_n$. Le but de $\hat{h}$ est de prédire la sortie y associée à une entrée x. Nous mesurons la qualité de $\hat{h}$ par son erreur de généralisation, définie par :

$$\mathrm{E}[(\hat{h}(X) - Y)^2] \,.$$

— Pour le problème d'estimation, nous disposons d'un estimateur $\hat{s}$ de la fonction de régression s, c'est-à-dire une fonction de $\mathcal{X}$ dans $\mathbb{R}$, construite sur l'échantillon d'apprentissage $\mathcal{L}_n$. Le but de $\hat{s}$ est d'estimer au mieux la fonction s. Nous mesurons la qualité de $\hat{s}$ par son risque, défini par :

$$\mathrm{E}[(\hat{s}(X) - s(X))^2] \, .$$

Ces deux mesures de qualité dépendent donc du point de vue (prédiction ou estimation). De plus, comme nous avons supposé que $\mathrm{E}[\varepsilon|X] = 0$, ces deux mesures satisfont la relation suivante : pour un prédicteur $\hat{h}$,

$$\mathrm{E}[(\hat{h}(X) - Y)^2] = \mathrm{E}[(\hat{h}(X) - s(X))^2] + \mathrm{E}[\varepsilon^2] \, .$$

Ainsi, en régression, la différence entre prédiction et estimation est essentiellement une différence de point de vue et de vocabulaire. Comme nous allons maintenant le voir, ce n'est pas le cas en classification.

2.1.2 Classification

En classification (appelée souvent classification supervisée), la réponse Y est discrète et désigne la classe (ou le label, l'étiquette de la classe) à laquelle appartient l'entrée X associée. Ici, $\mathcal{Y} = \{1, \ldots, L\}$, où L désigne le nombre de classes. Nous codons l'ensemble des classes de façon ordonnée pour faciliter les notations, mais l'ensemble des classes n'a en général pas de structure, Y est nominale (ou catégorielle).

En régression, le but est d'estimer la fonction de régression, qui n'est autre que l'espérance conditionnelle de Y sachant X. En classification, nous ne pouvons pas écrire le modèle sous une forme équivalente au modèle (1). Mais le but est maintenant d'estimer les probabilités *a posteriori* définies, pour un $x \in \mathcal{X}$ fixé, par :

$$\forall c \in \{1, \ldots, L\} \quad \mathrm{P}(Y = c|X = x)$$

c'est-à-dire les probabilités pour Y d'appartenir à chacune des classes, conditionnellement à X.

Le fait que nous traitions un échantillon d'observations bruitées implique que pour un x fixé, il n'y a pas forcément une probabilité *a posteriori*, parmi les L, qui soit égale à 1 et les autres égales à 0. Donc, pour certaines observations, la classe correspondante à x devrait être c, mais se retrouve altérée en c' dans l'échantillon. En régression, le bruit vient du fait que nous n'observons pas exactement $s(X)$, mais $s(X)+\varepsilon$, alors qu'en classification, le bruit provient du fait que certaines étiquettes des classes sont altérées.

En classification, nous avons également deux mesures de qualité : l'une pour la prédiction, l'autre pour l'estimation.

— Nous mesurons la qualité d'un prédicteur $\hat{h}$ par son erreur de généralisation, définie par :

$$\mathrm{P}(\hat{h}(X) \neq Y).$$

— Le prédicteur qui minimise l'erreur de généralisation est appelé prédicteur de Bayes. Ce prédicteur prédit pour un x fixé la quantité suivante :

$$\underset{c \in \{1,\ldots,L\}}{\operatorname{argmax}} \mathrm{P}(Y = c|X = x).$$

Bien entendu, ce prédicteur n'est calculable que si l'on connaît la loi du couple (X, Y). C'est un estimateur idéal qu'on cherche à approcher.
Notons $\hat{p}(x, c)$ un estimateur de $\mathrm{P}(Y = c|X = x)$, la probabilité *a posteriori*. Une façon d'approcher le prédicteur de Bayes est alors de proposer un prédicteur $\hat{h}$ qui

prédit, pour un x donné, la quantité $\underset{c\in\{1,\dots,L\}}{\text{argmax}}\ \hat{p}(x,c)$. Nous pouvons alors mesurer la capacité du prédicteur $\hat{h}$ à bien estimer le prédicteur de Bayes, par exemple, par la quantité suivante :

$$\mathrm{E}\left[\sum_{c=1}^{L} |\,\hat{p}(X,c) - \mathrm{P}(Y=c|X)\,|\right].$$

Dans ce cadre, il n'est pas nécessaire de très bien estimer les probabilités *a posteriori* pour bien prédire. En effet, prenons un problème à deux classes, notées 1 et 2. Si $\mathrm{P}(Y = 1|X = x) = 0.99$, l'estimateur de Bayes prédit alors la classe 1 pour l'observation x. Mais alors, si $\hat{p}(x,1) = 0.51$, le prédicteur $\hat{h}$ prédit lui aussi la classe 1 pour l'observation x, alors que l'estimation de la probabilité *a posteriori* est assez mauvaise.

Une des particularités des forêts aléatoires est qu'elles peuvent être utilisées dans des cadres de régression et de classification, et seules quelques légères adaptations sont nécessaires pour passer d'un cadre à l'autre. De plus, elles présentent de très bonnes performances en prédiction (c'est-à-dire en terme d'erreur de généralisation) dans les deux cas.

2.1.3 Une remarque sur l'espace d'entrée

Souvent, on se limite à l'étude de problème d'apprentissage statistique lorsque l'espace d'entrée $\mathcal{X}$ est égal à $\mathbb{R}^p$, en particulier lorsqu'il est question de l'étude théorique des arbres ou des forêts aléatoires, accessible uniquement dans ce cas.

L'entier naturel p désigne le nombre de coordonnées de X, et nous appelons ces coordonnées les variables explicatives ou les covariables du modèle et nous noterons X^j la j-ième variable.

Le rapport entre le nombre d'observations n et le nombre de variables p est crucial en statistique, et peut mener à des problèmes très différents (voir paragraphe 5).

2.2 Un peu d'histoire

Cette section présente quelques méthodes d'ensemble apparues avant les forêts aléatoires (détaillées plus bas) et souvent définies pour des règles de base qui ne sont pas des arbres de décision.

2.2.1 Principe des méthodes d'ensemble

Les forêts aléatoires combinent des prédicteurs ou estimateurs de base que sont les arbres, donnant lieu à ce que l'on appelle aujourd'hui les méthodes d'arbres. Plus largement ce sont des méthodes d'ensemble dont le principe général (voir Dietterich (2000a)) est de construire une collection de prédicteurs, pour ensuite agréger l'ensemble de leurs prédictions. En régression, agréger les prédictions de q prédicteurs revient par exemple à en faire la moyenne : chaque prédicteur fournit un $\hat{y}_l$, et la prédiction finale est alors $\frac{1}{q}\sum_{l=1}^{q}\hat{y}_l$. En classification, l'agrégation consiste par exemple à faire un vote majoritaire parmi les labels des classes fournis par les prédicteurs.

Soulignons le fait que l'étape d'agrégation de ces méthodes est toujours très simple et n'est pas optimisée, contrairement aux méthodes dites d'agrégation de modèles, qui pour une famille de prédicteurs donnée, cherche la meilleure manière de les combiner pour obtenir un bon prédicteur agrégé (voir par exemple les travaux de Lecué (2007)).

Ainsi, au lieu d'essayer d'optimiser une méthode "en un coup", les méthodes d'ensemble génèrent plusieurs règles de prédiction et mettent ensuite en commun leurs différentes réponses. L'heuristique de ces méthodes est qu'en générant beaucoup de prédicteurs, on explore largement l'espace des solutions, et qu'en agrégeant toutes les prédictions, on dispose d'un

prédicteur qui rend compte de cette exploration. Ainsi, on s'attend à ce que le prédicteur final soit meilleur que chacun des prédicteurs individuels.
Illustrons sur un cas simple en nous plaçant dans un cadre de classification à deux classes. Pour que le classifieur agrégé commette une erreur pour un x donné, il faut qu'au moins la moitié des classifieurs individuels se soient également trompés pour ce même x. On peut espérer que ceci n'arrive pas très souvent, car même si les classifieurs individuels commettent des erreurs, il est peu probable qu'ils commettent les mêmes erreurs pour les mêmes entrées. D'où l'idée que les prédicteurs individuels doivent être différents les uns des autres : la majorité ne doit pas se tromper pour un x donné. Pour que cela soit possible, il faut également que les prédicteurs individuels soient relativement bons. Et là où un prédicteur se trompe, les autres doivent prendre le relais en ne se trompant pas.
L'explication heuristique du succès de ces méthodes d'ensemble s'appuie donc sur deux propriétés : en premier lieu, chaque prédicteur individuel doit être relativement bon et en outre, les prédicteurs individuels doivent être différents les uns des autres. Le premier point est nécessaire, car agréger des prédicteurs tous mauvais ne pourra vraisemblablement pas donner un bon. Le second point est également naturel, car agréger des prédicteurs trop peu variés donnera encore un prédicteur semblable et n'améliorera pas les prédictions individuelles.

2.2.2 Bagging

Le Bagging est une méthode introduite par Breiman (1996) pour les arbres, et directement issue de la remarque selon laquelle les arbres CART sont instables et sensibles aux fluctuations de l'ensemble des données de l'échantillon d'apprentissage $\mathcal{L}_n$. On peut en fait considérer plus généralement une méthode de prédiction (appelée règle de base), qui construit sur $\mathcal{L}_n$ un prédicteur $\hat{h}(.,\mathcal{L}_n)$. Le principe du Bagging est de tirer un grand nombre d'échantillons, indépendamment les uns des autres, et de construire, en appliquant à chacun d'eux la règle de base, un grand nombre de prédicteurs. La collection de prédicteurs est alors agrégée en faisant simplement une moyenne ou un vote majoritaire. Cette méthode est centrale et on y reviendra largement dans la section 4.2.

2.2.3 Boosting

Introduit par Freund et al. (1996), le Boosting est une des méthodes d'ensemble les plus performantes à ce jour. Étant donné un échantillon d'apprentissage $\mathcal{L}_n$ et une méthode de prédiction (ou règle de base), qui construit sur $\mathcal{L}_n$ un prédicteur $\hat{h}(.,\mathcal{L}_n)$. Le principe du Boosting est de tirer un premier échantillon bootstrap $\mathcal{L}_n^{\Theta_1}$, où chaque observation a une probabilité $1/n$ d'être tirée, puis d'appliquer la règle de base pour obtenir un premier prédicteur $\hat{h}(.,\mathcal{L}_n^{\Theta_1})$. Ensuite, l'erreur de $\hat{h}(.,\mathcal{L}_n^{\Theta_1})$ sur l'échantillon d'apprentissage $\mathcal{L}_n$ est calculée. Un deuxième échantillon boostrap $\mathcal{L}_n^{\Theta_2}$ est alors tiré mais la loi du tirage des observations n'est maintenant plus uniforme. La probabilité pour une observation d'être tirée dépend de la prédiction de $\hat{h}(.,\mathcal{L}_n^{\Theta_1})$ sur cette observation. Le principe est, par le biais d'une mise à jour exponentielle bien choisie, d'augmenter la probabilité de tirer une observation mal prédite et de diminuer celle de tirer une observation bien prédite. Une fois le nouvel échantillon $\mathcal{L}_n^{\Theta_2}$ obtenu, on applique à nouveau la règle de base $\hat{h}(.,\mathcal{L}_n^{\Theta_2})$. On tire alors un troisième échantillon $\mathcal{L}_n^{\Theta_3}$, qui dépend des prédictions de $\hat{h}(.,\mathcal{L}_n^{\Theta_2})$ sur $\mathcal{L}_n$ et ainsi de suite. La collection de prédicteurs obtenus est alors agrégée en faisant une moyenne pondérée, là encore via des poids exponentiels bien choisis.

Le Boosting est donc une méthode séquentielle, chaque échantillon étant tiré en fonction des performances de la règle de base appliquée sur l'échantillon précédent. En cela, le Boosting diffère de façon importante du Bagging, où les échantillons sont tirés indépendamment les uns des autres, et peuvent être obtenus en parallèle. L'idée du Boosting est de se concentrer de plus en plus sur les observations mal prédites par la règle de base, pour essayer d'apprendre au mieux cette partie difficile de l'échantillon en vue d'améliorer les performances globales.

Mentionnons que le Boosting, d'abord défini pour la classification, a été généralisé pour la régression par Drucker (1997). On trouve dans ce cadre, une étude de son instabilité dans Gey and Poggi (2006) ou une utilisation pour la détection de données aberrantes dans Cheze and Poggi (2006).

Pour dénommer les méthodes de type Boosting, Breiman (1998) parle d'algorithmes **Arc**ing, pour **A**daptively **R**esample and **C**ombine. L'idée est bien qu'au lieu de ré-échantillonner de façon indépendante comme dans le Bagging, on ré-échantillonne de façon adaptative dans le Boosting. Contrairement à nombre d'autres méthodes d'ensemble, le Boosting a beaucoup été étudié théoriquement, voir par exemple Bartlett and Traskin (2007) et les références de cet article.

2.2.4 Randomizing Outputs

Le Bagging et le Boosting construisent une collection de prédicteurs en ré-échantillonnant les observations de $\mathcal{L}_n$, Breiman (2000a) introduit la méthode Randomizing Outputs pour les problèmes de régression, qui est une méthode d'ensemble de nature différente. Le principe est, ici, de construire des échantillons indépendants dans lesquels on altère les sorties de l'échantillon d'apprentissage. La modification que subissent les sorties est obtenue en rajoutant une variable de bruit à chaque Y_i de $\mathcal{L}_n$. On obtient alors une collection d'échantillons "à sorties randomisées", puis on applique une règle de base sur chacun et on agrège enfin l'ensemble des prédicteurs obtenus.

L'idée de Randomizing Outputs est, encore, qu'en appliquant une règle de base sur des échantillons à sorties randomisées, on obtient une collection de prédicteurs différents les uns des autres.

2.2.5 Random Subspace

Un autre type de méthode d'ensemble est introduit dans Ho (1998). Il n'est plus ici question de jouer sur l'échantillon, mais plutôt d'agir sur l'ensemble des variables considérées. Le principe de la méthode Random Subspace est de tirer aléatoirement un sous-ensemble de variables et d'appliquer une règle de base sur $\mathcal{L}_n$ qui ne prend en compte que les variables sélectionnées. On génère alors une collection de prédicteurs chacun construit en utilisant des variables différentes, puis on agrège ces prédicteurs. Les sous-ensembles de variables sont tirés indépendamment pour chaque prédicteur.

L'idée de cette méthode est de construire plusieurs prédicteurs, chacun étant spécialisé et réputé bon dans un sous-espace de $\mathcal{X}$ particulier, pour ensuite en déduire un prédicteur sur l'espace d'entrée tout entier.

2.2.6 Les ingrédients clés

Les méthodes d'ensemble évoquées ici ont toutes un principe général commun. Il s'agit de partir d'une règle de prédiction de base, puis de perturber cette règle, pour construire une collection de prédicteurs issus de différentes perturbations, que l'on agrège. Les perturbations peuvent porter sur l'échantillon (ré-échantillonnage, sorties randomisées) ou le sous-espace d'entrée dans lequel on construit le prédicteur, et les différentes perturbations sont générées indépendamment, ou non, les unes des autres.

Pour chacune de ces méthodes, les auteurs montrent, souvent sur des simulations, que le prédicteur agrégé final fait systématiquement mieux (en terme d'erreur de généralisation) que la règle de prédiction de base. Donc, en pratique, il apparaît que "perturber puis agréger" améliore les performances d'une méthode de prédiction donnée.

La remarque précédente ne vaut que si dans la collection construite, les prédicteurs sont différents les uns des autres. C'est pourquoi ces stratégies sont appliquées à des règles de base qui sont des méthodes dites "instables". Une méthode est instable si de petites perturbations de l'échantillon d'apprentissage peuvent engendrer de grandes modifications du prédicteur

obtenu. Par exemple, les arbres de décision sont instables au contraire des méthodes linéaires (régression ou analyse discriminante) qui sont stables et donc jamais utilisées comme règle de base dans les méthodes d'ensemble.

2.3 Un jeu de données fil rouge

Tout au long de ce document, nous illustrerons l'application des différentes méthodes sur les très classiques données `spam` à des fins pédagogiques.

Ce jeu de données, bien connu et largement disponible, est dû à un ingénieur de HP, prénommé George, qui a analysé un échantillon de ses emails professionnels :

- Les individus sont les 4601 emails en question dont 2788 sont des emails souhaitables et 1813 (soit 40%) des emails indésirables, c'est-à-dire des spams.
- La variable réponse est donc binaire : *spam* ou *non-spam*.
- les variables explicatives sont au nombre de $p = 57$: 54 sont des proportions d'occurrences de mots ou de caractères, comme par exemple `$`, `!`, `free`, `money`, 2 sont liées aux longueurs des suites de lettres majuscules (la moyenne, la plus longue) et enfin la dernière est le nombre de lettres majuscules dans le mail. Ces variables sont classiques et définies grâce à des procédures usuelles en analyse textuelle, permettant de traiter statistiquement des individus caractérisés par des textes.

Le but est double. Premièrement, construire un "bon" filtre anti-spam : un nouveau mail arrive, il faut réussir à prédire si c'est un spam ou non. Deuxièmement, on s'intéresse aussi à savoir quelles sont les variables sur lesquelles se base le plus le filtre anti-spam (ici des mots ou des caractères).

Pour juger de la performance d'un filtre anti-spam, on découpe le jeu de données en deux : 2300 mails pour l'apprentissage, 2301 mails pour tester les prédictions.

Nous avons donc un problème de classification à 2 classes ($L = 2$) avec un nombre d'individus ($n = 2300$ pour l'apprentissage, la construction des modèles) largement plus important que le nombre de variables ($p = 57$). De plus, nous disposons d'un échantillon test de grande taille ($m = 2301$) pour évaluer une estimation de l'erreur de prédiction.

3 Arbres CART

L'acronyme CART signifie **C**lassification **A**nd **R**egression **T**rees. Il désigne une méthode statistique, introduite par Breiman et al. (1984) qui construit des prédicteurs par arbre aussi bien en régression qu'en classification.

Parfois introduites avant CART, d'autres méthodes pour construire des arbres de décision sont connues et largement disponibles, comme par exemple CHAID introduit par Kass (1980) ou encore l'algorithme $C4.5$ par Quinlan (1993). On peut saisir encore aujourd'hui l'actualité des arbres de décision dans deux récentes références : Patil and Bichkar (2012) en informatique et Loh (2014) en statistique, qui contiennent des synthèses utiles et les bibliographies associées.

Dans la suite, bien qu'en principe plusieurs façons de construire des arbres de décision CART sont possibles, par exemple en changeant la famille de coupures autorisées, la fonction de coût ou encore la règle d'arrêt, nous nous limitons à celle, couramment utilisée, présentée dans le livre de Breiman et al. (1984). Nous y renvoyons pour des compléments et des variantes qui n'ont pas vraiment eu la diffusion de la version la plus simple dont l'interprétabilité, en particulier, a fait le succès. Un exposé en français, concis et clair de la méthode CART en régression peut être trouvé dans le chapitre 2 de la thèse de Gey (2002).

Le principe général de CART est de partitionner récursivement l'espace d'entrée $\mathcal{X}$ de façon binaire, puis de déterminer une sous-partition optimale pour la prédiction. Bâtir un arbre CART se fait en deux étapes. Une première phase est la construction d'un arbre maximal, qui permet de définir la famille de modèles à l'intérieur de laquelle on cherchera

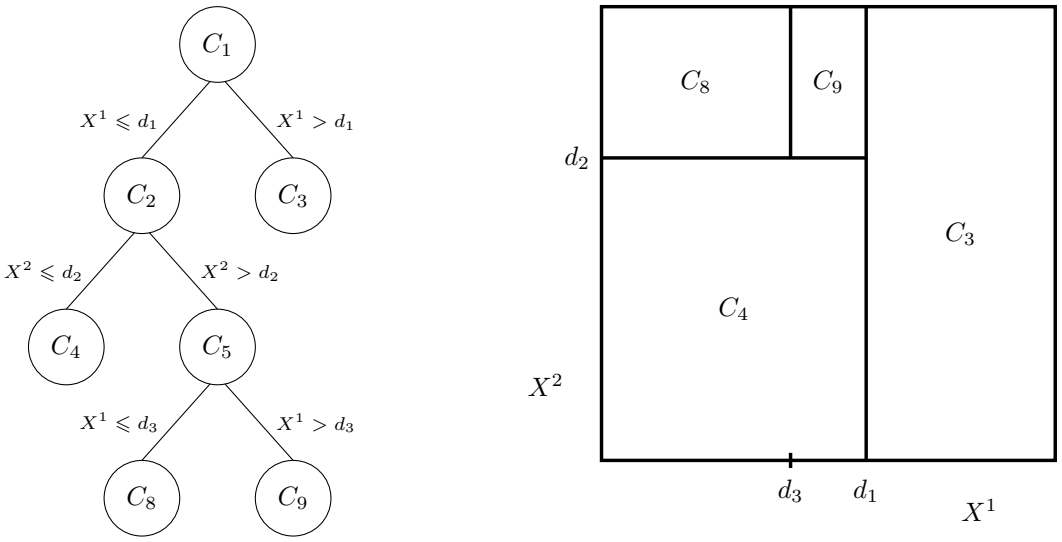


FIGURE 1 – À gauche : un arbre de classification permettant de prédire le label correspondant à un x donné. À droite : la partition associée dans l'espace des variables explicatives

à sélectionner le meilleur, et une seconde phase, dite d'élagage, qui construit une suite de sous-arbres optimaux élagués de l'arbre maximal. Détaillons chacune de ces étapes.

3.1 Construction de l'arbre maximal

A chaque pas du partitionnement, on découpe une partie de l'espace en deux sous-parties. On associe alors naturellement un arbre binaire à la partition construite. Les nœuds de l'arbre sont associés aux éléments de la partition. Par exemple, la racine de l'arbre est associé à l'espace d'entrée tout entier. Ses deux nœuds fils sont associés aux deux sous-parties obtenues par la première découpe du partitionnement, et ainsi de suite. La figure 1 illustre la correspondance entre un arbre binaire et la partition associée.

Détaillons maintenant la règle de découpe. Pour fixer les idées, le lecteur peut se restreindre à des variables explicatives continues (la cas qualitatif est néanmoins mentionné explicitement dans le texte chaque fois que cela est utile), l'espace d'entrée est alors $\mathbb{R}^p$, où p est le nombre de variables. Partons de la racine de l'arbre (associée à $\mathbb{R}^p$ tout entier), qui contient toutes les observations de l'échantillon d'apprentissage $\mathcal{L}_n$. La première étape de CART consiste à découper au mieux cette racine en deux nœuds fils. Nous appelons coupure (ou découpe ou même split) un élément de la forme

$$\{X^j \leqslant d\} \cup \{X^j > d\},$$

où $j \in \{1, \ldots, p\}$ et $d \in \mathbb{R}$. Découper suivant $\{X^j \leqslant d\} \cup \{X^j > d\}$ signifie que toutes les observations avec une valeur de la j-ième variable plus petite que d vont dans le nœud fils de gauche, et toutes celles avec une valeur plus grande que d vont dans le nœud fils de droite. La méthode sélectionne alors la meilleure découpe, c'est-à-dire le couple (j, d) qui minimise une certaine fonction de coût :

— En régression, on cherche à minimiser la variance intra-groupes résultant de la découpe d'un nœud t en 2 nœuds fils t_L et t_R. La variance d'un nœud t étant définie par $V(t) = \frac{1}{\#t} \sum_{i:x_i \in t} (y_i - \overline{y}_t)^2$ où $\overline{y}_t$ est la moyenne des y_i des observations présentes dans le nœud t et l'on est donc conduit à minimiser

$$\frac{1}{n} \sum_{(x_i,y_i) \in t_L} (y_i - \overline{y}_{t_L})^2 + \frac{1}{n} \sum_{(x_i,y_i) \in t_R} (y_i - \overline{y}_{t_R})^2 = \frac{\#t_L}{n} V(t_L) + \frac{\#t_R}{n} V(t_R).$$

— En classification (où l'ensemble des classes est $\{1, \ldots, L\}$), on définit l'impureté des nœuds fils, le plus souvent par le biais de l'indice de Gini. L'indice de Gini d'un nœud t est défini par $\Phi(t) = \sum_{c=1}^{L} \hat{p}_t^c(1-\hat{p}_t^c)$, où $\hat{p}_t^c$ est la proportion d'observations de classe c dans le nœud t. On est alors conduit à pour tout nœud t et tout split admissible à maximiser

$$\Phi(t) - \left(\frac{\#t_L}{\#t}\Phi(t_L) + \frac{\#t_R}{\#t}\Phi(t_R) \right).$$

En régression, on cherche donc des découpes qui tendent à diminuer la variance des nœuds obtenus. En classification, on cherche à diminuer la fonction de pureté de Gini, et donc à augmenter l'homogénéité des nœuds obtenus, un nœud étant parfaitement homogène s'il ne contient que des observations de la même classe. Au passage, on pourrait imaginer de mesurer l'homogénéité des nœuds par une autre fonction liée au taux de fausses classifications mais ce choix naturel ne conduit pas à une fonction de pureté strictement concave garantissant l'unicité de l'optimum à chaque découpe.

Mentionnons ici que dans le cas d'une variable explicative X^j catégorielle, rien ce qui précède ne change sauf que dans ce cas, une coupure est simplement un élément de la forme

$$\{X^j \in d\} \cup \{X^j \in \bar{d}\},$$

où d et $\bar{d}$ sont non vides et constituent une partition de l'ensemble des modalités de la variable X^j.

Une fois la racine de l'arbre découpée, on se restreint à chacun des nœuds fils et on recherche alors, suivant le même procédé, la meilleure façon de les découper en deux nouveaux nœuds, et ainsi de suite. Les arbres sont ainsi développés, jusqu'à atteindre une condition d'arrêt. Une règle d'arrêt classique consiste à ne pas découper des nœuds qui contiennent moins d'un certain nombre d'observations. Les nœuds terminaux, qui ne sont plus découpés, sont appelés les feuilles de l'arbre. A noter, que l'on ne découpe pas un nœud pur, c'est-à-dire un nœud ne contenant que des observations dont les sorties sont les mêmes (typiquement en classification). On appelle arbre maximal, noté T_{max}, l'arbre pleinement développé. Dans le même temps, on associe à chaque nœud t de l'arbre une valeur ($\overline{Y}_t$ en régression ou le label de la classe majoritaire des observations présentes dans le nœud t en classification). Donc, à un arbre est associée une partition (définie par ses feuilles) et également des valeurs qui sont attachées à chaque élément de cette partition. Le prédicteur par arbre est alors la fonction constante par morceaux, associée à l'arbre (voir figure 1).

3.2 Élagage

La deuxième étape de l'algorithme CART, s'appelle l'élagage et consiste à chercher le meilleur sous-arbre élagué de l'arbre maximal (meilleur au sens de l'erreur de généralisation). L'idée est que l'arbre maximal possède une très grande variance et un biais faible. *A contrario*, un arbre constitué uniquement de la racine (qui engendre alors un prédicteur constant) a une très petite variance mais un biais élevé. L'élagage est une procédure de sélection de modèles, où les modèles sont les sous-arbres élagués de l'arbre maximal, soit tous les sous-arbres binaires de T_{max} ayant la même racine que T_{max}. Cette procédure minimise un critère pénalisé où la pénalité est proportionnelle au nombre de feuilles de l'arbre.

Tous les sous-arbres binaires de T_{max} contenant la racine sont des modèles admissibles. Entre T_{max}, le modèle de complexité maximale, qui conduit au surajustement aux données de l'échantillon d'apprentissage et l'arbre restreint à la racine qui est fortement biaisé, il s'agit de trouver l'arbre optimal parmi les admissibles. En nombre fini, il suffirait donc au moins en principe, de construire la suite de tous les meilleurs arbres à k feuilles pour $1 \leqslant k \leqslant |T_{max}|$, où $|T|$ désigne le nombre de feuilles de l'arbre T, et de les comparer par exemple sur un échantillon test. Mais le nombre de modèles admissibles est exponentiel d'où une complexité algorithmique explosive. Fort heureusement, une énumération implicite et efficace suffit pour atteindre un résultat optimal. Le moyen consiste simplement dans l'algorithme d'élagage

qui assure l'extraction d'une suite de sous-arbres emboîtés (c'est-à-dire élagués les uns des autres) $T_1, \ldots, T_K$ tous élagués de T_{max}, où T_k minimise un critère des moindres carrés pénalisé en régression. Cette suite est obtenue de manière itérative en coupant des branches à chaque étape, ce qui ramène la complexité à un niveau très raisonnable. On se restreint sans inconvénient au cas de la régression dans les quelques lignes qui suivent, la situation étant identique en classification.

La clé est de pénaliser l'erreur d'ajustement d'un sous-arbre T élagué de T_{max} :

$$\overline{err}(T) = \frac{1}{n} \sum_{\{t \ feuille \ de \ T\}} \sum_{(x_i,y_i)\in t} (y_i - \overline{y}_t)^2$$

par une fonction linéaire du nombre de feuilles $|T|$ conduisant au critère des moindres carrés pénalisés :

$$crit_\alpha(T) = \overline{err}(T) + \alpha|T|.$$

Ainsi $\overline{err}(T)$ qui mesure l'ajustement du modèle T aux données, décroît avec le nombre de feuilles alors que $|T|$ qui quantifie la complexité du modèle T, croît avec le nombre de feuilles. Le paramètre α régle la pénalité : plus α est grand, plus les modèles complexes c'est-à-dire comptant beaucoup de feuilles, sont pénalisés.

L'algorithme d'élagage est donné dans la Table 1, où pour tout nœud interne t d'un arbre T, on note T_t la branche de T issue du nœud t (contenant tous les descendants du nœud t) et l'erreur correspondante est donnée par $\overline{err}(t) = n^{-1} \sum_{\{x_i \in t\}} (y_i - \bar{y}_t)^2$.

Entrée	Arbre maximal T_{max}.
Initialisation	$\alpha_1 = 0$, $T_1 = T_{\alpha_1} = \text{argmin}_T$ élagué de $_{T_{max}}$ $\overline{err}(T)$. `initialiser` $T = T_1$ et $k = 1$.
Iteration	`Tant que` $\lvert T\rvert > 1$, `Calculer` $\alpha_{k+1} = \min_{\{t \text{ nœud interne de } T\}} \dfrac{\overline{err}(t) - \overline{err}(T_t)}{\lvert T_t\rvert - 1}$. `Elaguer` toutes les branches T_t de T telles que $\overline{err}(T_t) + \alpha_{k+1}\lvert T_t\rvert = \overline{err}(t) + \alpha_{k+1}$ `Prendre` T_{k+1} le sous-arbre élagué ainsi obtenu. `Boucler sur` $T = T_{k+1}$ et $k = k + 1$.
Sortie	Arbres $T_1 \succ \ldots \succ T_K = \{t_1\}$, Paramètres $(0 = \alpha_1; \ldots; \alpha_K)$.

TABLE 1 – Algorithme d'élagage de CART

Le résultat principal du livre Breiman et al. (1984) établit que la suite de paramètres $(0 = \alpha_1; \ldots; \alpha_K)$ est strictement croissante, associée à la suite $T_1 \succ \ldots \succ T_K = \{t_1\}$ constituée de modèles emboités au sens de l'élagage et que, pour tout $1 \leqslant k \leqslant K$

$$\begin{aligned} \forall \alpha \in [\alpha_k, \alpha_{k+1}[\qquad T_k &= \underset{\{T \text{ sous-arbre de } T_{max}\}}{\text{argmin}} \ crit_\alpha(T) \\ &= \underset{\{T \text{ sous-arbre de } T_{max}\}}{\text{argmin}} \ crit_{\alpha_k}(T) \end{aligned}$$

en posant ici $\alpha_{K+1} = +\infty$.

Autrement dit, la suite $T_1, \ldots, T_K$ contient toute l'information statistique utile puisque pour tout $\alpha \geqslant 0$, le sous-arbre minimisant $crit_\alpha$ est un sous-arbre de la suite produite par l'algorithme d'élagage.

On peut la visualiser (voir l'exemple sur les données `spam` dans la Figure 2) par le biais de la suite des paramètres $(\alpha_k)_{1\leqslant k\leqslant K}$, avec le nombre de feuilles, les erreurs de validation (obtenues par validation croisée) de chaque arbre, et une estimation de l'écart-type de cette erreur. Chaque point représente ainsi un arbre, avec l'estimation de l'écart-type de l'erreur de validation sous forme de segment vertical. Le choix de l'arbre optimal peut se faire directement en minimisant l'erreur de validation ou en appliquant la règle du 1 écart-type consistant à choisir l'arbre le plus compact atteignant une erreur de validation inférieure à la valeur la valeur du minimum précédent augmenté de l'écart-type estimé de l'erreur de validation. Cette quantité est représentée par la ligne horizontale en pointillés sur l'exemple de la Figure 2.

Il faut remarquer que, bien entendu, si un arbre quelconque de cette suite comporte k feuilles, c'est le meilleur arbre à k feuilles. En revanche, cette suite ne contient pas tous les meilleurs arbres à k feuilles pour $1 \leqslant k \leqslant |T_{max}|$ mais seulement une partie d'entre eux. Les arbres manquants ne sont simplement pas compétitifs car de critère pénalisé plus grand.

Comme nous le verrons plus bas, les forêts aléatoires sont, la plupart du temps, des forêts d'arbres non élagués. Cependant, insistons sur le fait qu'un arbre CART, s'il est utilisé seul, doit être élagué.

3.3 Garantie théorique

Une première garantie théorique minimale est disponible dans Breiman et al. (1984), puisqu'en annexe figure un résultat assez général de consistance d'arbres de décision mais dont les conditions ne sont pas nécessairement vérifiées, en particulier par l'arbre élagué optimal. Il faut attendre Gey and Nedelec (2005) pour obtenir en régression un résultat non asymptotique justifiant la forme de la pénalité et Gey (2012) pour un résultat semblable dans le cas de la classification, et c'est celui-ci que nous proposons d'esquisser ci-dessous.

Il s'agit des bornes de risque sur l'étape d'élagage pour la classification binaire. Le critère pénalisé est alors de la forme :

$$\hat{R}_{pen}(T) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{\hat{f}_T(X_i)\neq Y_i} + \alpha|T|$$

avec, comme ci-dessus, $|T|$ le nombre de feuilles de T. Un résultat typique pour la classification binaire dit que lorsque le sous-arbre T_{opt} est choisi par la méthode du Hold-out avec un échantillon $\mathcal{L}_1$ pour construire et élaguer T_{max}, un échantillon $\mathcal{L}_2$ pour choisir l'arbre minimisant l'erreur de prédiction et sous une condition sur la marge h, il existe des constantes positives C_1, C_2, C_3 telles que :

$$\mathbb{E}\left[l(f^*, \hat{f}_{T_{opt}})|\mathcal{L}_1\right] \leqslant C_1 \inf_{T\preceq T_{max}} \left[\inf_{f\in S_T} l(f^*, f) + h^{-1}\frac{|T|}{n_1}\right] + \frac{C_2}{n_1} + C_3\frac{\ln n_1}{n_2}$$

où S_T est l'ensemble des classifieurs définis sur la partition $\widetilde{T}$ induite par l'ensemble des feuilles de T, et $l(f^*, f) = \mathrm{P}\left(f(X) \neq Y\right) - \mathrm{P}\left(f^*(X) \neq Y\right)$.

Ce résultat établit que la performance effective de l'arbre sélectionné est, au premier ordre, du même ordre de grandeur que la performance du meilleur classifieur augmentée de la pénalité, en justifiant ainsi la forme.

Un point à remarquer est que la qualité de la sélection de l'estimateur est appréciée conditionnellement à l'échantillon $\mathcal{L}_1$, puisque la famille de modèles à l'intérieur de laquelle on fouille est dépendante des données. Le risque de référence est évalué sur la famille S_T est l'ensemble des classifieurs définis sur la partition $\widetilde{T}$, issus de l'arbre maximal.

3.4 Interprétabilité et instabilité

On rassemble dans cette section des remarques dont l'utilité est manifeste pour le statisticien appliqué.

3.4.1 Découpes compétitives

On peut disposer en chaque nœud de l'arbre de la suite ordonnée par réduction décroissante de l'hétérogénéité de toutes les découpes (une par variable explicative). C'est ce que l'on appelle les découpes compétitives ou concurrentes (competing splits) et elles sont, en tout nœud, nécessairement calculées lors de la construction de l'arbre maximal. La possibilité du développement manuel de l'arbre maximal peut être précieuse et s'opère en choisissant en chacun des nœuds dans la liste ordonnée des splits, soit la découpe optimale, soit une découpe légèrement moins bonne mais portant sur des variables moins incertaines, plus faciles, moins chères à mesurer ou encore plus interprétables (voir par exemple Ghattas (2000)).

3.4.2 Valeurs manquantes

L'une des difficultés concrètes pour le calcul d'une prédiction est la présence de valeurs manquantes. CART en offre un traitement efficace et très élégant. En effet, tout d'abord remarquons que lorsque des variables sont manquantes pour un x donné, cela ne pose un problème que si l'on passe par un nœud dont la coupure est basée sur l'une de ces variables. Ensuite, en un nœud ou la variable sur laquelle porte le split est manquante, on peut penser utiliser l'une des autres variables, par exemple la seconde découpe compétitive. Mais cette idée n'est pas optimale, puisque la règle d'acheminement dans les fils droit et gauche respectivement peut être très éloignée de la règle d'acheminement induite par la découpe optimale. D'où l'idée de calculer en chaque nœud la liste des découpes de substitution (surrogate splits) définies par la découpe minimisant le nombre de désaccords avec la règle d'acheminement induite par la coupure optimale. On dispose ainsi d'une méthode de traitement des valeurs manquantes en prédiction locale et performante, qui ne passe pas par des méthodes d'imputation globales et souvent trop grossières.

Notons que si CART permet de bien gérer les valeurs manquantes en prédiction, les forêts aléatoires, qui sont des ensembles d'arbres non élagués, perdent essentiellement cette propriété. En effet, tout d'abord, pour un arbre non élagué donné, le chemin de la racine à une feuille est en général beaucoup plus long et, de plus, potentiellement de nombreux arbres sont touchés par une variable manquante particulière. Des solutions existent néanmoins dans ce cadre avec les forêts aléatoires floues comme par exemple dans Valdiviezo and Van Aelst (2015). Bien sûr, mais c'est une autre question, on peut utiliser les RF pour faire de l'imputation comme dans Stekhoven and Bühlmann (2012).

3.4.3 Interprétabilité

L'interprétabilité est un des aspects qui font le succès des arbres CART. On peut, en effet, très facilement répondre à la question du pourquoi, pour un x donné, on prévoit telle valeur de y en fournissant la suite des réponses aux questions constituées par les découpes successives rencontrés pour parcourir, pour le x en question, l'unique chemin de la racine à la feuille associée. Notons que les forêts aléatoires perdent cette propriété d'interprétabilité puisqu'il n'existe pas d'arbre associé à une forêt, même si l'on peut signaler des tentatives pour définir un arbre approchant en un certain sens une forêt, comme par exemple Hara and Hayashi (2016).

Mais de manière plus globale, au delà de l'interprétation d'une prédiction particulière, on peut, une fois l'arbre CART construit, considérer que les variables qui interviennent dans les découpes des nœuds de l'arbre (en particulier les nœuds les plus proches de la racine) sont les variables les plus utiles pour le problème considéré. En effet, plus une découpe est proche de la racine plus la décroissance de l'hétérogénéité est importante. C'est une façon heuristique et pratique d'avoir une information sur l'importance des variables. En réalité, cette première intuition donne des résultats biaisés et un indice d'importance des variables plus élaboré est fourni par les arbres CART. Il est basé à nouveau sur la notion de découpes de substitution. Suivant Breiman et al. (1984), on peut définir l'importance d'une variable en évaluant, en chaque nœud, la réduction d'hétérogénéité engendrée par l'usage du split

de substitution portant sur cette variable puis en les sommant sur tous les nœuds. Cet indice est, par exemple, utilisé par Poggi and Tuleau (2006) dans une procédure de sélection de variables, laquelle est examinée théoriquement par Sauve and Tuleau-Malot (2014). Pour intéressant qu'il soit, cet indice d'importance des variables n'est plus aujourd'hui très utilisé : il est en effet peu intuitif, très instable car dépendant d'un arbre donné et moins pertinent que l'importance des variables par permutation au sens des forêts aléatoires. En outre, son analogue, qui n'utilise pas les découpes de substitution, existe dans les forêts mais il a tendance à favoriser les variables catégorielles qui ont un grand nombre de modalités (nous y reviendrons en 5.5).

3.4.4 Valeurs aberrantes

Un autre avantage est la résistance naturelle aux valeurs aberrantes, la méthode étant purement non paramétrique, la présence d'une donnée aberrante dans l'ensemble d'apprentissage va contaminer essentiellement la feuille qui la contient, avec un faible impact pour les autres.

A propos des données aberrantes ou plus largement des données influentes, on peut utiliser l'instabilité de CART pour mesurer l'influence, non pas des variables, mais des individus sur l'analyse, cette approche classique en analyse des données est porteuse d'informations sur les individus qui ne sont pas seulement vus comme des répétitions uniquement utiles pour apprendre sur le modèle ou les variables. Ainsi dans Bar-Hen et al. (2015) sont proposées diverses mesures de l'influence des observations sur les résultats obtenus avec un arbre de classification CART. Ces mesures d'influence quantifient la sensibilité de l'analyse pour un arbre de classification de référence construit avec toutes les données par des mesures de la variabilité des prédictions fourni par les arbres jackknife construits avec toutes les observations moins une. L'analyse est étendue aux séquences d'arbres élagués pour produire une notion d'influence spécifique à la méthode CART.

3.4.5 Complexité algorithmique

D'autres qualités peuvent encore être notées et sans plus développer citons-en une dernière : le temps de calcul, qui en fait une méthode bien adaptée à l'analyse de vastes ensembles de données. En effet, l'algorithme de construction d'un arbre et l'algorithme d'élagage sont de complexité informatique faible : le nombre d'opérations requise est $\mathcal{O}(pnlog(n))$ au mieux (arbre parfaitement équilibré) ou de $\mathcal{O}(pn^2)$ au pire (arbre dit en peigne ou en arête de poisson, le plus déséquilibré).

3.5 Exemple

Examinons cette construction sur le jeu de données `spam`. Le package utilisé ici pour réaliser les arbres CART est le package `rpart` (Therneau et al., 2015) inclus dans la base de `R` (R Core Team, 2015).

La suite optimale des sous-arbres élagués T_{max} obtenue par la construction de l'arbre maximal et l'application de l'algorithme d'élagage est donné par la Figure 2.

Chaque point représente ainsi un arbre, avec l'estimation de l'écart-type de l'erreur de validation croisée sous forme de segment vertical. L'arbre minimisant ce critère est parfois encore un peu trop complexe (23 feuilles ici), et en relaxant un peu la condition de minimisation de l'erreur de prédiction par la règle du "1 écart-type" (1 s.e. rule de Breiman qui tient compte de l'incertitude de l'estimation de l'erreur des arbres de la suite), on obtient l'arbre de la Figure 3.

Le meilleur sous-arbre élagué de l'arbre maximal (à un écart-type près) comporte 17 feuilles et seules 14 variables, parmi les 57 initiales, figurent dans les découpes associées aux 16 nœuds internes : `charExclamation`, `charDollar`, `remove`, `capitalAve`, `money`, `george`, `hp`, `free`, `re`, `num000`, `our`, `edu`, `internet` et `meeting`.

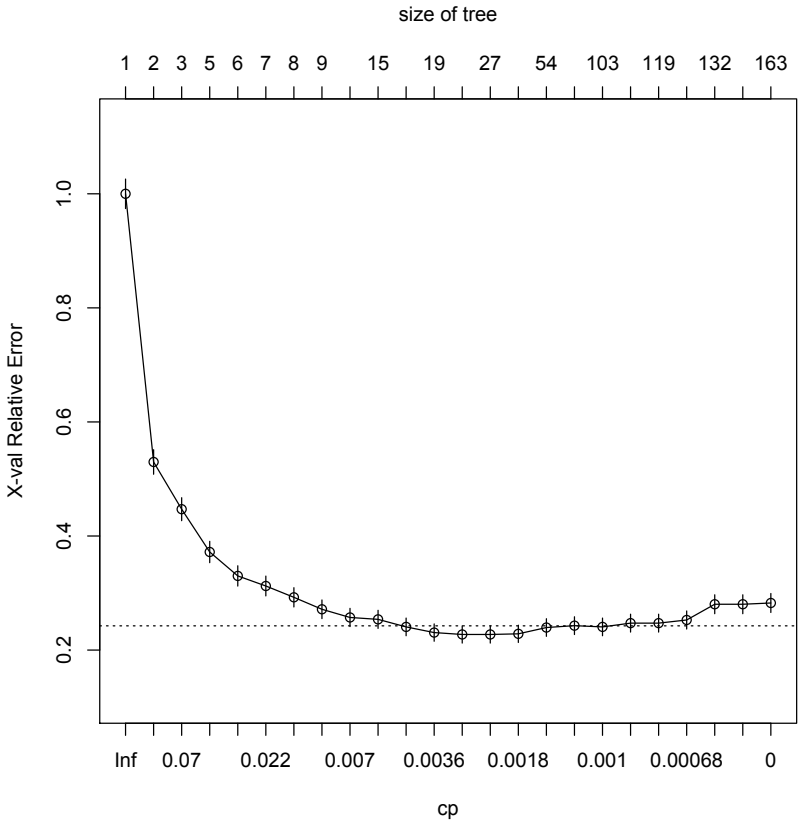


FIGURE 2 – La suite optimale des sous-arbres élagués de T_{max}

On peut illustrer la facilité d'interprétation en considérant par exemple le chemin de la racine à la feuille la plus à droite qui dit qu'un mail qui contient beaucoup de `$` et de `!` est presque toujours un spam. Inversement le chemin de la racine à la cinquième feuille la plus à droite exprime qu'un mail contenant beaucoup de `!`, de lettres capitales et de `hp` mais peu de `$` n'est presque jamais un spam. Pour cette dernière interprétation, il n'est pas inutile de rappeler que les mails examinés sont les mails professionnels d'un seul individu travaillant pour HP.

Enfin les erreurs empiriques et tests obtenues par différents arbres sont données par la Table 2.

Arbre	2 feuilles	1 s.e.	maximal	optimal
Erreur empirique	0.208	0.073	0.000	0.062
Erreur test	0.209	0.096	0.096	0.086

TABLE 2 – Erreurs empirique et test des 4 arbres pour les données `spam`

On remarque que, comme annoncé, l'arbre maximal (trop complexe) a une erreur empirique (*i.e.* sur l'échantillon d'apprentissage) nulle et que l'arbre à deux feuilles (trop simple) présente des erreurs test et empirique proches. L'arbre optimal, quant à lui, a la meilleure erreur test de 0.086, soit $8,6\%$.

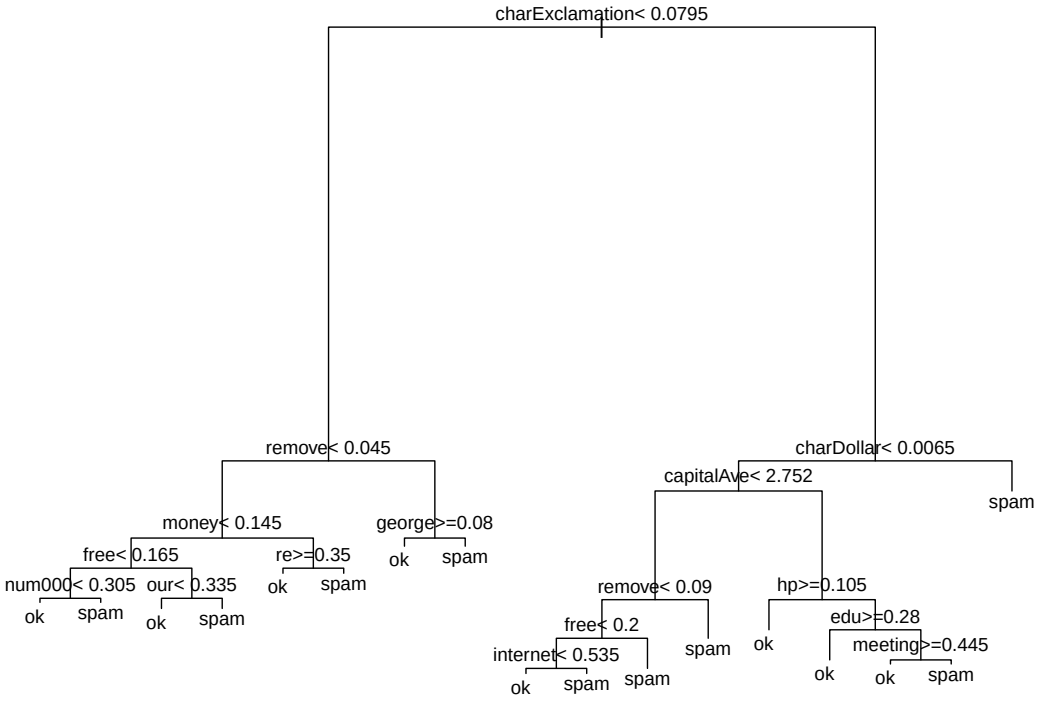


FIGURE 3 – Arbre CART élagué

3.6 Extensions

De nombreuses extensions ou variantes des arbres CART ont été proposées pour diverses fins.

On trouve tout d'abord des extensions, en régression, qui construisent des prédicteurs plus réguliers que les prédicteurs par arbres qui sont constants par morceaux, par exemple, l'algorithme MARS introduit par Friedman (1991).

D'autres idées pour déterminer les splits, en les choisissant déterministes, ce qui évite la dépendance de la famille de modèles aux données évoquée plus haut. Dans Donoho et al. (1997), pour des applications en traitement d'images, les découpes considérées sont dyadiques. Le découpage dyadique d'un rectangle du plan est déterministe et complet jusqu'à la résolution du pixel, c'est l'analogue de l'arbre maximal et l'élagage est opéré grâce à un algorithme classique de choix de la meilleure base de paquets d'ondelettes. Des idées semblables ont été généralisées dans Blanchard et al. (2007).

Citons aussi l'une des extensions les plus utilisées : les méthodes CART pour les données de survie par exemple LeBlanc and Crowley (1993) et Molinaro et al. (2004) ainsi que le plus récent article de synthèse Bou-Hamad et al. (2011).

Une extension aux données spatiales est à mentionner dans Bel et al. (2009) avec des applications dans le domaine de l'environnement.

On peut enfin renvoyer au livre récent Zhang and Singer (2013) portant plus largement sur les méthodes basées sur le partitionnement récursif, mais qui présente aussi des variantes de CART pour les données longitudinales ou pour les données fonctionnelles. Dans cette

ligne, signalons l'utilisation de CART en chimiométrie dans Questier et al. (2005).

4 Des arbres aux forêts aléatoires

4.1 Définition générale des forêts aléatoires

Après avoir examiné le socle des méthodes d'arbres, abordons le coeur de ce papier. Les forêts aléatoires ont été introduites par Breiman (2001) par la définition très générale suivante :

Définition 1. *Soit* $\left(\widehat{h}(.,\Theta_1),\ldots,\widehat{h}(.,\Theta_q)\right)$ *une collection de prédicteurs par arbres, avec* $\Theta_1,\ldots,\Theta_q$ *q variables aléatoires i.i.d. indépendantes de* $\mathcal{L}_n$*. Le prédicteur des forêts aléatoires* $\widehat{h}_{RF}$ *est obtenu est agrégeant cette collection d'arbres aléatoires de la façon suivante :*

- $\widehat{h}_{RF}(x) = \frac{1}{q}\sum_{l=1}^{q}\widehat{h}(x,\Theta_l)$ *(moyenne des prédictions individuelles des arbres) en régression,*
- $\widehat{h}_{RF}(x) = \underset{1\leqslant k\leqslant K}{\operatorname{argmax}} \sum_{l=1}^{q} \mathbb{1}_{\widehat{h}(x,\Theta_l)=k}$ *(vote majoritaire parmi les prédictions individuelles des arbres) en classification.*

En fait, cette définition n'est pas rigoureusement identique à celle de Breiman (2001), qui ne précise pas que les variables Θ_l sont indépendantes de $\mathcal{L}_n$. Nous adoptons néanmoins la Définition 1 car elle reflète mieux l'intuition que l'aléa supplémentaire apporté par les Θ_l est déconnecté des données d'apprentissage et de plus toutes les variantes de forêts aléatoires connues rentrent dans ce cadre.

Cette définition est illustrée par le schéma de la Figure 4. Nous en verrons plusieurs déclinaisons par la suite.

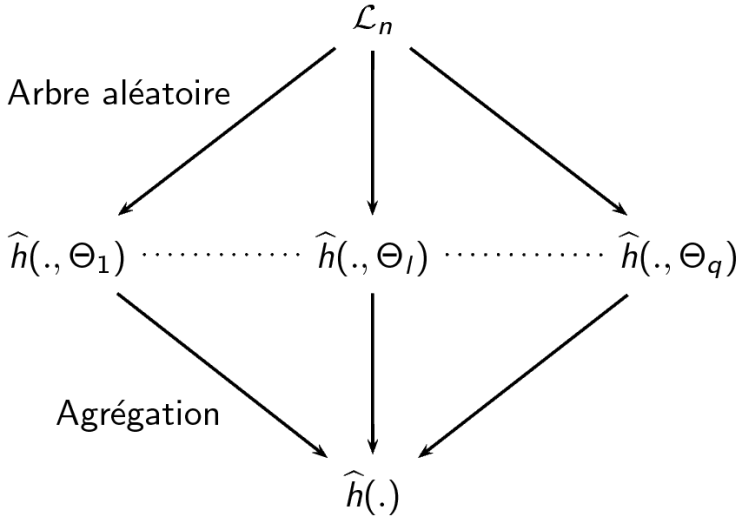


FIGURE 4 – Schéma général des forêts aléatoires

Le terme forêt aléatoire vient du fait que les prédicteurs individuels sont, ici, explicitement des prédicteurs par arbre, et du fait que chaque arbre dépend d'une variable aléatoire supplémentaire (c'est-à-dire en plus de $\mathcal{L}_n$). Une forêt aléatoire est l'agrégation d'une collection d'arbres aléatoires.

Les forêts aléatoires font bien partie de la famille des méthodes d'ensemble. Remarquons d'ailleurs que, parmi les méthodes d'ensemble précédemment citées (lorsque l'on choisit comme règle de base un arbre de décision), seul le Boosting ne rentre pas dans la définition

de forêts aléatoires. En effet, les arbres individuels du Boosting ne dépendent pas d'aléas indépendants les uns des autres. Le Bagging, Randomizing Outputs et Random Subspace sont alors des cas particuliers de forêts aléatoires, avec respectivement pour aléa supplémentaire le tirage de l'échantillon bootstrap, la modification aléatoire des sorties de $\mathcal{L}_n$ et le tirage des sous-ensembles de variables. En plus de ces trois méthodes, il existe de nombreux cas particuliers de forêts aléatoires dans la littérature.

Il existe une ambiguïté de vocabulaire dans la littérature. En effet, Leo Breiman, dans son article de 2001, définit les forêts aléatoires comme ci-dessus et sont donc pour lui une famille de méthodes. Or, dans le même article, il présente un cas particulier de forêts aléatoires, appelées Random Forests-RI, qu'il a implémentées (voir Breiman and Cutler (2004)). Par la suite, ce sont ces Random Forests-RI qui ont été quasi-systématiquement utilisées dans d'innombrables applications réelles. Et pour cause, le programme est accessible à tous, facile d'utilisation et la méthode atteint des performances empiriques exceptionnelles. Finalement, la dénomination "forêts aléatoires" désigne maintenant très souvent les Random Forests-RI. On trouve également le terme de "forêts aléatoires de Leo Breiman" pour désigner les Random Forests-RI.

Dans la suite de cette section, nous nous concentrons tout d'abord sur le Bagging qui est central dans l'analyse en évoquant quelques résultats avant de décrire en détails les Random Forests-RI et de définir l'erreur OOB qui en est une sortie très utile. Nous donnons ensuite quelques idées des garanties théoriques disponibles pour des variantes plus ou moins éloignées des RF-RI mais utiles. Enfin nous terminerons comme d'habitude par l'application aux données `spam` avant d'évoquer variantes et extensions.

4.2 Bagging

La méthode du Bagging a été introduite par Breiman (1996). Le mot Bagging est la contraction des mots **B**ootstrap et **Agg**regat**ing**. Étant donné un échantillon d'apprentissage $\mathcal{L}_n$ et une méthode de prédiction (appelée règle de base), qui construit sur $\mathcal{L}_n$ un prédicteur $\hat{h}(.,\mathcal{L}_n)$. Le principe du Bagging est de tirer indépendamment plusieurs échantillons bootstrap $(\mathcal{L}_n^{\Theta_1},\ldots,\mathcal{L}_n^{\Theta_q})$, d'appliquer la règle de base sur chacun d'eux pour obtenir une collection de prédicteurs $(\hat{h}(.,\mathcal{L}_n^{\Theta_1}),\ldots,\hat{h}(.,\mathcal{L}_n^{\Theta_q}))$, et enfin d'agréger ces prédicteurs de base.

L'idée du Bagging, et qu'en appliquant la règle de base sur différents échantillons bootstrap, on en modifie les prédictions, et donc on construit ainsi une collection variée de prédicteurs. L'étape d'agrégation permet alors d'obtenir un prédicteur performant.

Un échantillon bootstrap $\mathcal{L}_n^{\Theta_l}$ est, par exemple, obtenu en tirant aléatoirement n observations avec remise dans l'échantillon d'apprentissage $\mathcal{L}_n$, chaque observation ayant une probabilité $1/n$ d'être tirée. La variable aléatoire Θ_l représente alors ce tirage aléatoire. Une deuxième façon classique d'obtenir un échantillon bootstrap est de tirer aléatoirement k observations sans remise dans $\mathcal{L}_n$, avec $k < n$. La Figure 5 résume le principe de cette méthode.

Initialement, le Bagging a été introduit avec comme règle de base un arbre de décision. Cependant, le schéma est très général et peut-être appliqué à d'autres règles de base comme par exemple, la règle du plus proche voisin. Cette méthode du plus proche voisin "baggé" a été étudiée, dans un cadre de régression, par Biau and Devroye (2010), puis Biau et al. (2010) (voir également les références de cet article). Le premier article établit la consistance de la méthode du plus proche voisin "baggé" (l'estimateur obtenu converge vers la vraie fonction de régression quand n tend vers $+\infty$), à condition que le nombre d'observations k dans les échantillons bootstrap (avec ou sans remise) tende vers $+\infty$, mais moins vite que n : $\frac{k}{n} \to +\infty$. Ces conditions de convergence sur k et n sont des conditions que nous retrouverons plus bas. Le deuxième article va plus loin et montre que l'estimateur atteint la vitesse optimale de convergence pour la classe des fonctions Lipschitz, sous les mêmes conditions sur k et n.

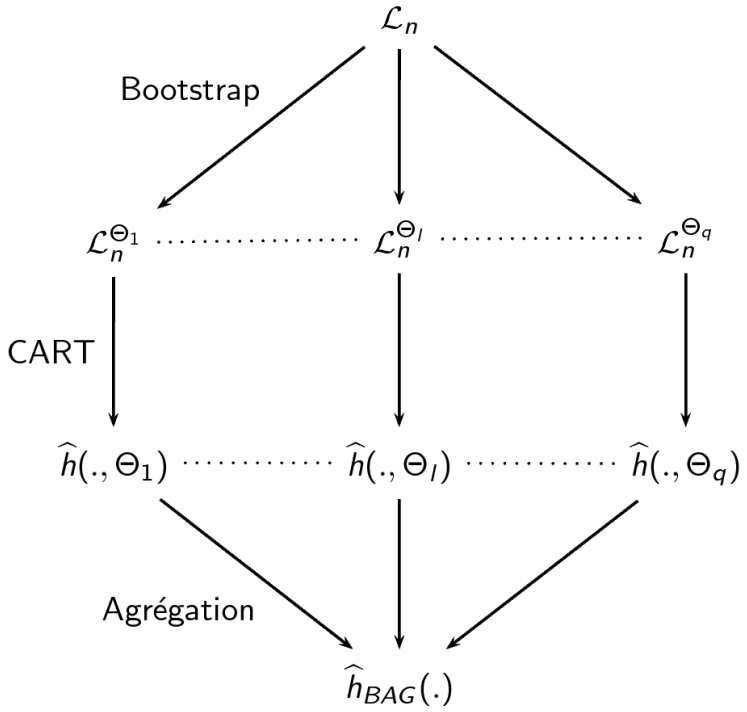


FIGURE 5 – Schéma du Bagging avec pour règle de base un arbre CART

Cette étude illustre les bienfaits des méthodes d'ensemble : partant d'une règle de base assez pauvre (la règle du plus proche voisin n'est pas consistante), le Bagging la transforme en une règle aux très bonnes propriétés asymptotiques (consistance et vitesse optimale de convergence). L'idée, ici, est que la méthode du plus proche voisin, n'explore pas assez l'espace : elle assigne à un x donné, le y correspondant à l'observation de $\mathcal{L}_n$ la plus proche de x. Le fait d'appliquer la méthode, sur un échantillon bootstrap permet de prendre en compte les sorties des observations plus éloignées de x (ce qui arrive lorsque les plus proches voisins de x ne sont pas présents dans l'échantillon bootstrap courant). Le plus proche voisin "baggé" met alors un poids sur chacune des données de $\mathcal{L}_n$ et le prédicteur agrégé est finalement une moyenne pondérée des Y_i de l'échantillon d'apprentissage. Les résultats théoriques nous assurent en fait que la méthode règle automatiquement et de façon optimale ces poids.

Nous évoquons maintenant un article sur l'analyse du Bagging. Bühlmann and Yu (2002) étudient, en dimension 1 ($\mathcal{X} = \mathbb{R}$), la convergence des trois paramètres qui définissent un arbre à deux feuilles (le point de coupure et les deux valeurs assignées à chacune des feuilles). Ces trois paramètres convergent en loi à la vitesse $n^{1/3}$ (Banerjee et al., 2007) et Bühlmann and Yu (2002) obtiennent alors un résultat montrant que le prédicteur Bagging d'arbres à deux feuilles (avec un sous-échantillonnage sans remise à la place du bootstrap) a une variance inférieure à celle de l'arbre à deux feuilles initial. C'est ainsi le premier résultat théorique illustrant l'intuition que les méthodes d'ensemble permettent une réduction de la variance.

4.3 Forêts aléatoires "Random Inputs"

Random Forests-RI signifie "forêts aléatoires à variables d'entrée aléatoires" (Random Forests with Random Inputs) et le principe de leur construction est tout d'abord de générer plusieurs échantillons bootstrap $\mathcal{L}_n^{\Theta_1}, \ldots, \mathcal{L}_n^{\Theta_q}$ (comme dans le Bagging). Ensuite, sur chaque

échantillon $\mathcal{L}_n^{\Theta_l}$, une variante de CART est appliquée. Plus précisément, un arbre est, ici, construit de la façon suivante. Pour découper un nœud, on tire aléatoirement un nombre m de variables, et on cherche la meilleure coupure uniquement suivant les m variables sélectionnées. De plus, l'arbre construit est complètement développé (arbre maximal) et n'est pas élagué. La collection d'arbres obtenus est enfin agrégée (moyenne en régression, vote majoritaire en classification) pour donner le prédicteur Random Forests-RI. La Figure 6 fournit le schéma récapitulatif de l'algorithme RF-RI, où Θ désigne le tirage bootstrap et Θ' désigne le tirage aléatoire des variables.

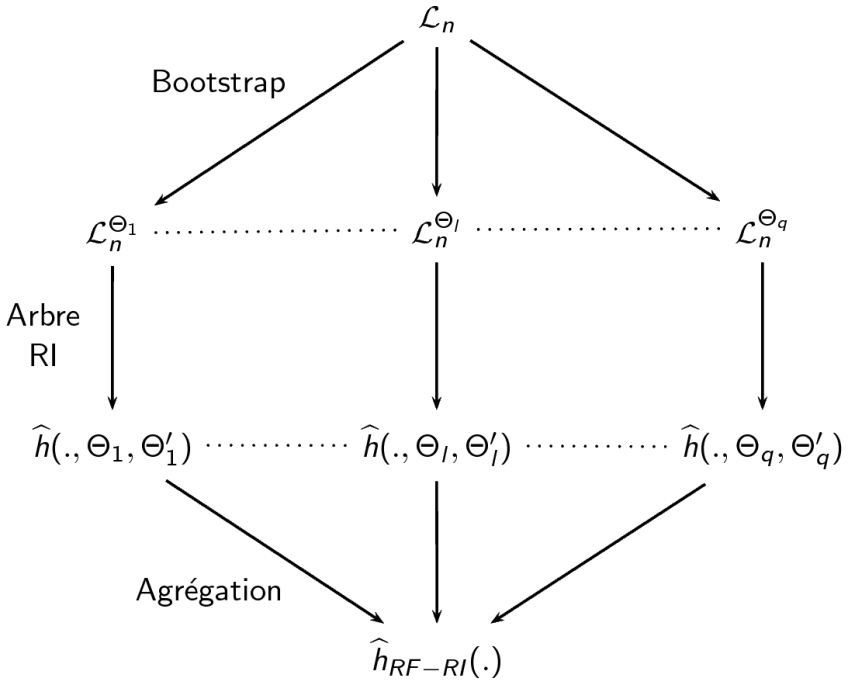


FIGURE 6 – Schéma des forêts aléatoires RF-RI

Ainsi, les Random Forests-RI peuvent être vues comme une variante du Bagging, où la différence intervient dans la construction des arbres individuels (les étapes de bootstrap et d'agrégation étant les mêmes). Le tirage, à chaque nœud, des m variables se fait, sans remise, et uniformément parmi toutes les variables (chaque variable a un probabilité $1/p$ d'être choisie). Le nombre m ($m \leqslant p$) est fixé au début de la construction de la forêt et est donc identique pour tous les arbres et pour tous les nœuds d'un même arbre mais naturellement les m variables impliquées dans deux nœuds distincts sont en général différentes. C'est un paramètre très important de la méthode. Une forêt construite avec $m = p$ revient à faire du Bagging d'arbres CART non élagués, alors qu'une forêt construite avec $m = 1$ est très différente du Bagging. En effet, lorsque $m = 1$, le choix de la variable, suivant laquelle est découpé un nœud, est complètement aléatoire (les coupures suivant cette variable ne sont pas mises en compétition avec des coupures utilisant d'autres variables).

Le tirage des m variables à chaque nœud représente un aléa supplémentaire, par rapport au Bagging. Pour les Random Forests-RI, il y a donc deux sources d'aléas pour générer la collection des prédicteurs individuels : l'aléa dû au bootstrap et l'aléa du choix des variables pour découper chaque nœud d'un arbre. Ainsi, on perturbe à la fois l'échantillon sur lequel on lance la règle de base et le coeur de la construction de la règle de base. Ce tirage aléatoire de variables pour découper un nœud avait déjà été utilisé par Amit and Geman (1997) dans des problèmes de reconnaissance d'image. Leur méthode a beaucoup influencé Leo Breiman dans sa mise au point de Random Forests-RI. Pour leur problème, le nombre de

coupures candidates était tellement gigantesque qu'ils étaient obligés de réduire le nombre de possibilités, par exemple en effectuant un choix aléatoire préliminaire à la découpe.

En pratique, les Random Forests-RI (avec le paramètre m bien choisi) améliorent les performances du Bagging (voir la comparaison des méthodes sur des données de référence dans Breiman (2001)). L'explication heuristique de ces améliorations est que le fait de rajouter un aléa supplémentaire pour construire les arbres, rend ces derniers encore plus différents les uns des autres, sans pour autant dégrader de façon significative leurs performances individuelles. Le prédicteur agrégé est alors meilleur. Nous avons vu que la plupart des méthodes d'ensemble construisent une collection de prédicteurs qui sont des versions perturbées d'une règle de base. La perturbation introduite doit alors réaliser un compromis : une trop grande perturbation dégrade les prédicteurs individuels et le prédicteur agrégé est alors mauvais alors qu'une trop petite perturbation induit des prédicteurs individuels trop similaires entre eux et le prédicteur agrégé n'apporte alors aucune amélioration. Les excellents résultats des Random Forests-RI en pratique laissent penser qu'elles (avec le paramètre m bien choisi) réalisent un bon compromis, en injectant la "bonne dose" d'aléa.

L'algorithme des Random Forests-RI a été codé par Breiman and Cutler (2004). Il a ensuite été importé dans le logiciel libre `R` par Liaw and Wiener (2002), via le package `randomForest`. Ce package est utilisé dans le traitement de très nombreuses applications réelles. Il existe deux principaux paramètres dans ce programme et qui sont aussi les deux seuls véritables paramètres de la méthode :

- Le paramètre le plus important est le nombre m de variables choisies aléatoirement à chacun des nœuds des arbres. Il est nommé `mtry` dans le package. Il peut varier de 1 à p et possède une valeur par défaut : $\sqrt{p}$ en classification, $p/3$ en régression.
  A cet égard on trouve dans Genuer et al. (2008)), une étude empirique qui précise les valeurs par défaut : pour les problèmes de régression, à l'exception du temps de calcul, il n'y a pas d'amélioration par rapport à Bagging non élagué (obtenu pour $mtry = p$). Pour les problèmes de classification standards, la valeur par défaut proposée dans le package est bon mais pour des problèmes de classification de grande dimension, des valeurs plus grandes pour $mtry$ donnent parfois des résultats bien meilleurs.
- Nous pouvons également jouer sur le nombre d'arbres q de la forêt. Ce paramètre est nommé `ntree` et sa valeur par défaut est 500.

Le programme permet également de régler d'autres aspects de la méthode : le nombre minimum d'observations (nommé `nodesize`) en dessous duquel on ne découpe pas un nœud, ou encore la façon d'obtenir les échantillons bootstrap (avec ou sans remise, ainsi que le nombre d'observations tirées). Nous laisserons dans les expériences numériques, les valeurs par défaut pour ces éléments, *i.e.* un `nodesize` de 1 en classification et 5 en régression, et les échantillons bootstrap considérés sont tous obtenus en tirant n observations avec remise dans l'échantillon d'apprentissage $\mathcal{L}_n$.

4.4 Erreur OOB

En plus de construire un prédicteur, l'algorithme des Random Forests-RI calcule une estimation de son erreur de généralisation : l'erreur Out-Of-Bag (OOB) où "Out-Of-Bag" signifie "en dehors du bootstrap". Cette erreur était déjà calculée par l'algorithme du Bagging, d'où la présence du mot "Bag". Le procédé de calcul de cette erreur est le suivant et il est très astucieusement intriqué dans l'algorithme de construction d'une RF.

Fixons une observation (X_i, Y_i) de l'échantillon d'apprentissage $\mathcal{L}_n$ et considérons l'ensemble des arbres construits sur les échantillons bootstrap ne contenant pas cette observation, c'est-à-dire pour lesquels cette observation est "Out-Of-Bag". Nous agrégeons alors uniquement les prédictions de ces arbres pour fabriquer notre prédiction $\hat{Y}_i$ de Y_i. Après avoir fait cette opération pour toutes les données de $\mathcal{L}_n$, nous calculons alors l'erreur commise : l'erreur quadratique moyenne en régression $\left(\frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2\right)$, et la proportion d'observations mal classées en classification $\left(\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_{\hat{Y}_i \neq Y_i}\right)$. Cette quantité est appelée erreur OOB

du prédicteur Random Forests-RI.

Cette estimation obéit aux mêmes contraintes que celles des estimateurs classiques de l'erreur de généralisation (par échantillon test ou par validation croisée), au sens où les données prédites sont des données qui n'ont pas été rencontrées au préalable par le prédicteur utilisé. Un avantage de l'erreur OOB par rapport aux estimateurs classiques est qu'elle ne nécessite pas de découpage de l'échantillon d'apprentissage. Ce découpage est en quelque sorte inclus dans la génération des différents échantillons bootstrap. Cependant, il faut bien noter que pour chaque observation ce n'est pas le même ensemble d'arbres qui est agrégé. En conséquence, cette erreur estime l'erreur de généralisation d'une forêt, mais elle n'utilise jamais les prédictions de la forêt elle-même, mais plutôt celles de prédicteurs qui sont des agrégations d'arbres de cette forêt. Par la suite, nous adopterons ici quasi exclusivement l'erreur OOB puisqu'elle est très liée à la définition de l'importance des variables et que nous l'utiliserons pour comparer des prédicteurs entre eux, et non pour obtenir une estimation précise de leurs erreurs de généralisation. Bien entendu, dans ce dernier cas, on procédera plus classiquement grâce à un échantillon test.

4.5 Garantie théorique

En dehors d'une majoration assez générale mais relativement grossière due à Breiman (2001), il n'y a pas de résultats théoriques disponibles pour les RF-RI à proprement parler. C'est l'une des rares, mais elle est cruciale, difficultés de cette méthode. On peut cependant citer des résultats théoriques, dus à un petit nombre d'auteurs, qui considèrent différentes variantes des forêts aléatoires et fournissent un premier pas vers des garanties théoriques tout en esquissant une compréhension plus profonde des extraordinaires performances pratiques des RF-RI, sans toutefois les élucider pour le moment.

En confondant ces différentes variantes (que l'on examinera en partie en 4.7), on peut trouver des résultats de consistance dans Breiman (2000b), Biau et al. (2008), Zhu et al. (2015), Ishwaran and Kogalur (2010) et Denil et al. (2014) ; un résultat de réduction de variance et de vitesse de convergence en dimension 1 dans Genuer (2012), un résultat de réduction du biais et de vitesse de convergence en dimension quelconque dans Arlot and Genuer (2014) ; et enfin un résultat de vitesse de convergence dans un contexte de réduction de dimension dans Biau (2012).

Signalons une contribution récente de Scornet et al. (2015) portant sur la consistance d'une variante assez réaliste des RF-RI dans le contexte des modèles additifs. Enfin, on peut trouver des résultats de normalité asymptotique dans Mentch and Hooker (2014) ou encore dans Wager (2014), ainsi que des résultats de vitesse de convergence pour des forêts en lien avec des estimateurs à noyaux dans Scornet (2016).

Sur ces questions, on ne saurait trop recommander la lecture du récent papier de revue Biau and Scornet (2016) qui est orienté vers les questions de l'analyse théorique des RF, ainsi que la discussion qui suit, à laquelle l'un d'entre nous a contribué (voir Arlot and Genuer (2016)). Un objectif de ce commentaire est de quantifier l'apport des différents ingrédients des forêts aléatoires, tout d'abord de manière théorique pour une variante très simple de forêt, puis par simulation sur une variante très proche des RF-RI. Il ressort de cette étude que c'est la randomisation des partitions (qu'elle soit obtenue grâce au bootstrap, au tirage des m variables à chaque nœud ou au tirage du point de coupure) associées aux arbres de la forêt qui serait la plus cruciale. Ceci explique (au moins partiellement) pourquoi les méthodes du Bagging (qui ne randomise pas sur la recherche de la coupure) et Extra-Trees de Geurts et al. (2006) (qui n'utilise pas de bootstrap, voir 4.7) donnent toutes deux des résultats très satisfaisants en pratique alors qu'elles sont très différentes dans le choix de l'aléa supplémentaire Θ.

4.6 Exemple

Les taux d'erreur test des prédicteurs Bagging et forêts aléatoires sont regroupés dans le tableau 3 et comparés à la performance de l'arbre CART optimal.

Prédicteur	arbre optimal	bagging	forêt aléatoire
Erreur test	0.086	0.060	0.052

TABLE 3 – Erreurs test du bagging et des forêts aléatoires, comparées à celles de l'arbre optimal pour les données `spam`

Pour appliquer le Bagging sur les données `spam`, nous utilisons le package `randomForest` et construisons donc un prédicteur Bagging avec comme règle de base un arbre CART non-élagué (le package ne permet pas d'élaguer les arbres d'une forêt).

Le Bagging est réputé stabiliser l'arbre CART classique en réduisant fortement sa variance et en améliorer sensiblement les performances. Sur notre jeu de données, l'erreur test atteinte par le Bagging (6%) est bien inférieure à celle atteinte par le meilleur des arbres précédents ($8,6\%$).

Une forêt aléatoire (construite à l'aide du package `randomForest` avec les paramètres par défaut) atteint, quant à elle, un taux d'erreur test de $5,2\%$, qui a encore diminué par rapport au Bagging. Rappelons que l'idée de cette amélioration qui est quasiment systématiquement observée, est que la collection d'arbres construite avec les forêts aléatoires est plus diversifiée que celle construite avec le Bagging (du fait des choix aléatoires de variables à chaque nœud). Cette diversité est bénéfique au moment de l'agrégation. On peut penser aussi que l'on "explore" plus l'espace des prédicteurs.

4.7 Variantes et extensions

4.7.1 Variantes

Comme mentionné ci-dessus, différentes versions des forêts aléatoires, de structure plus ou moins fortement dépendantes des données, ont été proposées. On parle souvent de forêts purement aléatoires lorsque les partitions associées aux arbres sont choisies aléatoirement indépendamment des observations de $\mathcal{L}_n$ (Breiman (2000b), Biau et al. (2008), Arlot and Genuer (2014)). Certaines variantes ont essentiellement été développées pour l'obtention de résultats théoriques, mais d'autres constituent des variantes assez compétitives sur le plan des performances avec les RF-RI (voir par exemple Cutler and Zhao (2001); Geurts et al. (2006); Duroux and Scornet (2016)). De surcroît, elles sont la plupart du temps beaucoup plus simples à définir, à calculer, à mettre à jour et s'adaptent par exemple plus facilement au contexte des flux de données (voir la section 6.2 dédiée aux RF en ligne).

Suivant Geurts et al. (2006), introduisons la méthode Extra-Trees (pour Extremly Randomized Trees). Le principe est, ici, de tirer aléatoirement m variables à chaque nœud, puis de choisir aléatoirement un point de coupure pour chaque variable : pour une variable continue X^j, on tire le seuil d de façon uniforme dans le segment délimité par la plus petite et la plus grande valeur de X^j des observations présentent dans le nœud courant. La coupure est alors $\{X^j \leqslant d\} \cup \{X^j > d\}$. Nous récupérons alors m coupures, et choisissons la meilleure coupure parmi celles-ci, c'est-à-dire celle qui minimise la fonction de coût considérée. Le choix des m variables à chaque nœud et le choix de la meilleure coupure sont les mêmes que dans les RF-RI. Cependant, ici un aléa supplémentaire est introduit au niveau du point de coupure. Là où les RF-RI optimisent le point de coupure sur $\mathcal{L}_n$, les Extra-Trees tire ce point de coupure aléatoirement. Geurts et al. (2006) illustrent dans leur article les très bonnes performances des Extra-Trees, qui apportent même parfois des améliorations par rapport à la méthode de référence RF-RI.

Nous présentons un résumé du type de randomisation (ou aléa supplémentaire) utilisée pour construire ces forêts aléatoires. Cette présentation est en grande partie basée sur celle

de Liu et al. (2005).

1. Randomisation préalable à la construction de l'arbre :
   (a) Ré-échantillonnage : *e.g.* Bagging, Random Forests-RI ;
   (b) Tirage d'un sous-ensemble de variables (fixe pour tout l'arbre) : *e.g.* Random Subspace ;
   (c) Perturbation des sorties : *e.g.* Randomizing Outputs.
2. Randomisation au coeur de la construction de l'arbre :
   (a) Tirage d'un sous-ensemble de variables à chaque nœud : *e.g.* Random Forets-RI, Extra-Trees ;
   (b) Tirage d'une coupure parmi de bonnes coupures candidates : *e.g.* Randomization (Dietterich (2000b)) ;
   (c) Tirage du point de coupure : *e.g.* Extra-Trees.

Bien entendu, il existe pour chaque type d'aléa, plusieurs randomisations possibles : ré-échantillonnage avec ou sans remise, sous-échantillonné ou non, tirage uniforme ou suivant une autre loi des sous-ensembles de variables, etc. Il est également possible de combiner plusieurs randomisations dans une même méthode, comme c'est déjà le cas dans Random Forets-RI ou Extra-Trees. Néanmoins, plusieurs associations de ces randomisations ont déjà été testées par les auteurs des méthodes citées ici. Finalement, seules les méthodes présentant les meilleurs résultats ont été retenus. Par exemple, Geurts et al. (2006) ont essayé de coupler le Bagging et Extra-Trees, mais ont montré qu'en pratique cela n'apporte pas d'amélioration, ceci étant sûrement dû au fait que la randomisation des partitions est déjà assez forte dans Extra-Trees (voir les commentaires en 4.5).

4.7.2 Extensions

Tout d'abord, on peut associer à toutes les extensions des arbres CART listées dans la section 3.6 des extensions naturelles des forêts.

Signalons quelques-unes des extensions des forêts développées pour des objectifs variés. Par exemple, pour des problèmes de classement Clémençon et al. (2013), pour l'analyse des données de survie Hothorn et al. (2006) et Ishwaran et al. (2008), ou encore pour la régression quantile Meinshausen (2006) pour se limiter aux plus connues.

On trouve aussi dans Yan et al. (2013) une extension, plus surprenante, des RF au contexte de la classification non supervisée pour définir les Cluster forests. Un article de revue sur ce type de sujet est récemment paru visant les applications de type chimiométrie (voir Afanador et al. (2016)).

Suivant la remarque de Breiman, les RF conduisent à des résultats d'autant meilleurs que la diversité des arbres de la forêt est grande. Ainsi, récemment, certaines extensions ont été définies par l'amélioration d'une RF initiale. Fawagreh et al. (2015) utilisent une technique d'apprentissage non supervisé (Local Outlier Factor, LOF) pour identifier les arbres divers dans la forêt puis, ils effectuent l'élagage de l'ensemble en sélectionnant les arbres avec les scores de LOF les plus élevés pour produire une extension de RF appelé LOFB-DRF, beaucoup plus petite en taille et souvent meilleure. Ce schéma peut être étendu en utilisant d'autres mesures de diversité, voir Tang et al. (2006) qui présentent une analyse théorique sur six mesures de diversité existantes.

Une autre extension possible est de pondérer a posteriori les arbres pour améliorer la performance prédictive, voir par exemple récemment Winham et al. (2013).

D'autres variantes ont été introduites pour sortir du cadre des arbres avec des coupures parallèles aux axes, déjà par Breiman (2001) avec les Random Forests-RC (RC pour "random combination"), voir aussi plus récemment Blaser and Fryzlewicz (2015); Menze et al. (2011).

Une très récente variante due à Biau et al. (2016), exploite une manière d'associer à un arbre un réseau de neurones à deux couches cachées (la première contient les nœuds non

terminaux et la seconde les feuilles de l'arbre, les entrées sont les variables explicatives et la sortie la variable à expliquer) et introduit ainsi une variante neuronale des forêts aléatoires qui hérite naturellement des techniques d'optimisation et de réglage des paramètres des réseaux de neurones. Dans ce contexte, des résultats de consistance sont obtenus et le gain sur le plan des performances est substantiel tant vis-à-vis des réseaux de neurones que des forêts aléatoires, au prix toutefois d'une plus grande difficulté de réglage des très nombreux paramètres.

5 Forêts aléatoires et sélection de variables

Par le passé, les problèmes considérés en statistique comportaient typiquement beaucoup d'observations (n de l'ordre de quelques centaines ou milliers) et peu de variables (p seulement de l'ordre de la ou de quelques dizaines). Les progrès technologiques font que l'acquisition de données est devenue de plus en plus facile techniquement et de nos jours des bases de données gigantesques sont collectées ou simplement disponibles, quasi-quotidiennement. Les techniques statistiques classiques ne suffisent plus pour traiter ces nouvelles données. Le nombre de variables p, peut désormais atteindre des dizaines voire des centaines de milliers. Dans le même temps, pour beaucoup d'applications, le nombre d'observations n, se trouve réduit à quelques dizaines. Le domaine typique de telles situations est le domaine biomédical où l'on peut maintenant faire énormément de mesures sur un individu donné (mesures d'expression de gènes par exemple), mais le nombre d'individus sur lequel l'expérience est menée est réduit (ainsi dans le cas d'étude d'une maladie, le nombre de porteurs de la maladie qui participent à une étude est souvent limité).

Dans la suite, nous dirons que les données considérées sont de grande dimension lorsque le nombre de variables est très grand devant le nombre d'observations $n << p$. Nous avons en tête des problèmes où n est de l'ordre de 100 et p de l'ordre de plusieurs milliers. Un des avantages des forêts aléatoires est qu'elles sont très performantes aussi bien pour des problèmes classiques (où $p \leqslant n$) que pour des problèmes de grande dimension.

Dans de nombreux problèmes, et en particulier dans le cas de données de grande dimension, en plus d'un bon prédicteur, les praticiens souhaitent également avoir des informations supplémentaires sur les variables du problème et connaître les variables effectivement utiles pour expliquer le lien entrée-sortie. Ils désirent donc que le statisticien leur propose une sélection des variables. De plus, dans ce cadre, il est naturel de penser que relativement peu de variables (disons au maximum de l'ordre de n) agissent réellement sur la sortie et il convient de faire des hypothèses supplémentaires (de sparsité) pour lui redonner un sens. On trouve dans Giraud (2014) une très complète présentation des problèmes et techniques mathématiques pour aborder ce type de questions.

On peut en citer quelques méthodes de sélection de variables pour des données de grande dimension.

Commençons par une étude empirique dans Poggi and Tuleau (2006) où l'on introduit une méthode basée sur le score d'importance des variables fourni par l'algorithme CART. Dans un état d'esprit proche, citons aussi Questier et al. (2005).

En considérant le problème plus généralement, Guyon et al. (2002); Rakotomamonjy (2003); Ghattas and Ben Ishak (2008) utilisent le score calculé par les Support Vector Machines (SVM : Vapnik (2013)). Díaz-Uriarte and Alvarez De Andres (2006) proposent une procédure de sélection de variables basée sur l'importance des variables des forêts aléatoires. Toutes ces méthodes calculent tout d'abord un score pour chacune des variables, puis procèdent à une introduction séquentielle de variables (méthodes "forward'), ou une élimination séquentielle de variables (méthodes "backward" ou RFE pour Recursive Feature Elimination), ou exécutent des méthodes pas-à-pas (méthodes "stepwise") mêlant introduction et élimination de variables.

On trouve dans Fan and Lv (2008) une méthode en deux temps : une première étape d'élimination de variables pour atteindre une situation raisonnable où p est de l'ordre de n,

puis une deuxième étape de type forward basée par exemple sur le Least Absolute Shrinkage and Selection Operator (Lasso, Tibshirani (1996)).

Dans cet esprit, un schéma général pour calculer un score d'importance pour les variables est proposé dans Lê Cao et al. (2007), puis les auteurs utilisent ce schéma avec comme méthode de base CART et SVM. Leur idée est d'apprendre un vecteur de poids sur toutes les variables (leur meta-algorithme se nomme Optimal Feature Weighting (OFW)) : une variable avec un poids fort est importante, tandis qu'une variable avec un poids faible est inutile.

Enfin, plus récemment, des méthodes d'amélioration du Lasso, pour la sélection de variables, ont été mises au point. Ces dernières ont des points communs avec les méthodes d'ensemble. En effet, au lieu de chercher à faire de la sélection "en un coup" avec un Lasso classique, elles cherchent à construire plusieurs sous-ensembles de variables et à les mettre ensuite en commun. Dans Bolasso (pour Bootstrap-enhanced Lasso), introduit par Bach (2008), on génère plusieurs échantillons bootstrap puis on lance sur chacun d'eux la méthode du Lasso. Bolasso est donc à mettre en parallèle avec la méthode du Bagging de Breiman (1996). Dans Randomized Lasso, Meinshausen and Bühlmann (2010) choisissent de générer plusieurs échantillons par sous-échantillonnage et rajoutent un aléa supplémentaire dans la construction même du Lasso. Randomized Lasso est donc lui à rapprocher des forêts aléatoires Random Forests-RI. Dans le même esprit, on peut mentionner aussi Fellinghauer et al. (2013) qui utilisent les RF pour l'estimation robuste des modèles graphiques.

L'intérêt pour le sujet reste vif, comme en témoigne la référence récente Hapfelmeier and Ulm (2012) qui propose une nouvelle approche de sélection utilisant les forêts aléatoires ou encore Cadenas et al. (2013) qui décrit et compare ces différentes approches, dans un papier de synthèse.

Dans la suite de cette section, on définit tout d'abord l'importance des variables par permutation avant de présenter une procédure de sélection des variables de combinant étroitement scores d'importances, forêts aléatoires et des stratégies usuelles de sélection. Ensuite on donne quelques liens vers des résultats théoriques partiels avant d'appliquer la procédure au données `spam` et de mentionner variantes et extensions.

5.1 Importance des variables

Lorsqu'il ne s'agit pas seulement d'éliminer des variables du modèle ou de retenir seulement des variables explicatives suffisantes pour obtenir une bonne prédiction, on peut être intéressé par construire une hiérarchie de toutes les variables fondée sur l'importance vis-à-vis de la variable réponse. Un tel indice d'importance fournit donc un classement des variables, de la plus importante à la moins importante.

On trouvera dans Azen and Budescu (2003) une discussion générale sur l'importance des variables. Il est clair que cette notion a été relativement peu examinée par les statisticiens et principalement dans le cadre des modèles linéaires, comme en atteste le papier de synthèse de Grömping (2015) ou la récente thèse de Wallard (2015). Il se trouve que les forêts aléatoires offrent un cadre idéal, alliant une méthode non-paramétrique, ne prescrivant pas de forme particulière à la relation entre Y et les composantes de X, au ré-échantillonnage, pour disposer d'une définition à la fois efficace et commode de tels indices.

Dans ce cadre, l'une des mesures largement utilisées de l'importance d'une variable donnée est l'accroissement moyen de l'erreur d'un arbre dans la forêt lorsque les valeurs observées de cette variable sont permutées au hasard dans les échantillons OOB.
Fixons $j \in \{1, \ldots, p\}$ et détaillons le calcul de l'indice d'importance de la variable X^j. Considérons un échantillon bootstrap $\mathcal{L}_n^{\Theta_l}$ et l'échantillon OOB_l associé, c'est-à-dire l'ensemble des observations qui n'apparaissent pas dans $\mathcal{L}_n^{\Theta_l}$. Calculons errOOB_l, l'erreur commise sur OOB_l par l'arbre construit sur $\mathcal{L}_n^{\Theta_l}$ (erreur quadratique moyenne en régression, proportion de mal classés en classification). Permutons alors aléatoirement les valeurs de la j-ième variable dans l'échantillon OOB_l. Ceci donne un échantillon perturbé, noté $\widetilde{\text{OOB}_l}^j$. Calculons enfin

$\widetilde{\text{err}\overline{\text{OOB}}_l}^j$, l'erreur sur l'échantillon $\widetilde{\overline{\text{OOB}}_l}^j$. Nous effectuons ces opérations pour tous les échantillons bootstrap. L'importance de la variable X^j, $\text{VI}(X^j)$, est définie par la différence entre l'erreur moyenne d'un arbre sur l'échantillon OOB perturbé et celle sur l'échantillon OOB :

$$\text{VI}(X^j) = \frac{1}{q}\sum_{l=1}^{q}\left(\widetilde{\text{err}\overline{\text{OOB}}_l}^j - \text{errOOB}_l\right) .$$

Ainsi, plus les permutations aléatoires de la j-ième variable engendrent une forte augmentation de l'erreur, plus la variable est importante. A l'inverse, si les permutations n'ont quasiment aucun effet sur l'erreur (voire même la diminuent ce qui fait que VI peut être légèrement négative), la variable est considérée comme très peu importante.

Précisons que ce calcul de l'importance est exactement celui qui est exécuté dans le paquet `randomForest`. La définition qui est donné de l'importance dans Breiman (2001) est légèrement différente. Les échantillons OOB perturbés sont obtenus de la même manière. Par contre, l'importance d'une variable est alors définie par la différence entre l'erreur OOB sur les échantillons OOB perturbés, et l'erreur OOB initiale. Nous pensons que la raison de ce changement est que dans le calcul de l'erreur OOB, il y a une étape d'agrégation. L'agrégation a tendance à amoindrir les erreurs des arbres individuels, donc l'utilisation de l'erreur OOB dans le calcul tend à atténuer les effets des permutations aléatoires des variables. Or, dans le calcul de l'importance, nous voulons au contraire, que les effets de ces permutations soient les plus visibles possible. C'est pourquoi l'erreur OOB est remplacée par l'erreur moyenne sur tous les arbres, dans laquelle aucune étape d'agrégation n'est effectuée.

Des études, à l'aide de simulations pour l'essentiel, ont été menées pour illustrer le comportement de l'importance des variables des forêts aléatoires, en particulier dans les travaux de Strobl et al. (2007, 2008); Archer and Kimes (2008); Genuer et al. (2008, 2010b); Gregorutti et al. (2013). Dans ce type d'études empiriques, il est souvent difficile de tirer des conclusions générales très assurées, on peut néanmoins noter quelques comportements intéressants de ce score de VI pour la sélection :

— Les variables hors modèle, ou non informatives, ont une VI très faible.
— La variabilité au travers des répétitions des VI des variables hors modèle est moins importante que celle des variables informatives.
— En présence d'un groupe de variables très corrélées des effets de masquage apparaissent mais demeurent limités sauf lorsque le nombre de variables du groupe devient dominant par rapport aux autres variables informatives.
— Une évaluation stable de ces VI requiert des forêts touffues (*ntree* grand) et quelques répétitions de telles forêts sont utiles pour en quantifier la variabilité.

Nous allons donc nous concentrer sur la performance de prédiction des RF en se restreignant à l'estimation par l'erreur out-of-bag (OOB) et sur la quantification de l'importance des variables par permutation qui sont les ingrédients clés pour notre stratégie de sélection des variables.

5.2 Sélection de variables

Nous proposons dans Genuer et al. (2010b) (voir aussi Genuer et al. (2015) pour le package correspondant) une méthode de sélection de variables qui est une procédure "automatique" au sens où il n'y a aucun *a priori* à apporter pour faire la sélection. Par exemple, il n'est pas nécessaire de préciser le nombre de variables que l'on souhaite obtenir : la procédure s'adapte aux données pour fournir le sous-ensemble de variables final. De plus, notre méthode procède en deux étapes : la première (assez grossière et descendante) consiste à seuiller sur l'importance des variables dans le but d'éliminer un grand nombre de variables inutiles, tandis que la seconde (plus fine et ascendante) consiste en une introduction de variables dans des modèles de forêts aléatoires.

Nous distinguons en outre deux objectifs de sélection de variables, que nous appellerons (bien que cette terminologie puisse prêter à confusion) interprétation et prédiction :

1. Pour un but d'interprétation, nous cherchons à sélectionner toutes les variables X^j fortement reliées à la variable réponse Y (même si les variables X^j sont corrélées entre elles).
2. Alors que pour un but de prédiction, nous cherchons à sélectionner un petit sous-ensemble de variables suffisant pour bien prédire la variable réponse.

Typiquement, un sous-ensemble construit pour satisfaire (1) pourra contenir beaucoup de variables, qui seront potentiellement très corrélées entre elles. Au contraire, un sous-ensemble de variables satisfaisant (2) contiendra peu de variables, avec très peu de corrélation entre elles.

Une situation typique illustre la distinction entre les deux types de sélection de variables. Considérons un problème de classification en grande dimension ($n << p$) pour lequel chaque variable explicative est associée à un pixel dans une image ou un voxel dans une image 3D comme dans les problèmes de classification en activité cérébrale (IRMf), voir par exemple Genuer et al. (2010a). Dans de telles situations, il est supposé que beaucoup de variables sont inutiles ou non informatives et qu'il existe des groupes inconnus de prédicteurs fortement corrélés correspondant à des régions du cerveau impliquées dans la réponse à une stimulation donnée. Même si les deux objectifs de sélection des variables peuvent être d'intérêt, il est manifeste que trouver toutes les variables importantes hautement reliée à la variable réponse est utile pour l'interprétation, puisqu'il correspond à la détermination des régions entières du cerveau ou d'une image. Bien sûr, la recherche d'un petit nombre de variables suffisantes pour une bonne prédiction, permet d'obtenir les variables les plus discriminantes dans les régions précédemment mises en évidence mais est moins prioritaire dans ce contexte. Pour une approche plus formelle de cette distinction, on peut se reporter à l'intéressant papier de Nilsson et al. (2007).

Notre méthode de sélection de variables tente donc de satisfaire les deux objectifs précédents. Décrivons un peu plus précisément notre procédure en deux étapes qui est basée sur des forêts aléatoires comportant un très grand nombre d'arbres (typiquement $ntree = 2000$).

— Etape 1. Elimination préliminaire et classement :

  — Classer les variables par importance décroissante (en fait par VI moyenne sur 50 forêts typiquement).

  — Eliminer les variables de faible importance (soit m le nombre de variables conservées).

    Plus précisément, en partant de cet ordre, on considère la suite ordonnée des écart-types des VI correspondants (sd) que l'on utilise pour estimer la valeur d'un seuil sur les VI. Comme la variabilité des VI plus grande pour les vraies variables du modèle que pour les variables non informatives, la valeur du seuil est donnée par l'estimation de l'écart-type de VI pour ces dernières variables. Ce seuil est fixé au minimum prédit par le modèle CART ajusté aux données (X, Y) où les Y sont les sd des VI et les X sont les rangs.

    Ensuite seules les variables dont l'importance VI moyenne est plus grande que ce seuil sont gardées.

— Etape 2. Sélection de variables :

  — Pour l'*interprétation* : on construit la collection de modèles emboîtés constituées par les forêts construites sur les k premières variables, pour $k = 1$ à m et on sélectionne les variables du modèle conduisant à la plus faible erreur OOB. Ceci conduit à considérer m' variables.

    Plus précisément, on calcule les moyennes (typiquement sur 25 forêts) des erreurs OOB des modèles emboîtés en commençant par celui ne comportant que la variable la plus importante

et se terminant par celle impliquant toutes les variables importantes retenues précédemment. Idéalement, les variables du modèle conduisant à l'erreur OOB la plus faible sont choisies. En fait, pour faire face à l'instabilité, nous utilisons une astuce classique : nous sélectionnons le plus petit modèle avec une erreur inférieure à la plus faible erreur OOB augmentée de l'estimation de l'écart-type de cette erreur (basée sur les 25 mêmes répétitions de forêts).

— Pour la *prédiction* : à partir des variables retenues pour l'interprétation, on construit une suite de modèles en introduisant séquentiellement, dans l'ordre d'importance croissant, et en testant itérativement les variables. Les variables du dernier modèle sont sélectionnées.

Plus précisément, l'introduction séquentielle des variables est basée sur le test suivant : une variable est ajoutée seulement si l'erreur OOB décroît plus qu'un seuil. L'idée est que l'erreur OOB doit décroître plus que la variation moyenne engendrée par l'ajout de variables non informatives. Le seuil est fixé à la moyenne des valeurs absolues des différences premières des erreurs OOB entre le le modèle à m' variables et celui à m variables :

$$\frac{1}{m-m'} \sum_{j=m'}^{m-1} |\, errOOB(j+1) - errOOB(j) \,| \tag{2}$$

où $errOOB(j)$ est l'erreur OOB de la forêt construite grâce aux j variables les plus importantes.

Il faut noter que l'importance utilisée ici n'est pas normalisée, comme cela est souvent le cas. En effet, au lieu de considérer comme mentionné dans Breiman and Cutler (2004), que les importances brutes sont indépendantes et de même loi, de les normaliser et supposer leur normalité asymptotique, on préfère ici une solution entièrement pilotée par les données. C'est une des clés de notre stratégie qui préfère estimer directement la variabilité au travers des répétitions de forêts plutôt que d'utiliser la normalité quand *ntree* est suffisamment grand, qui est valide dans des conditions spécifiques, en fait très difficiles à vérifier puisque dépendant lourdement des paramètres de réglage de la méthode et des spécificités du problème. Une normalisation dirigée par les données nous prévient contre un comportement asymptotique mal spécifié.

Naturellement, un tel schéma peut être modifié et adapté dans diverses directions, par exemple à la marge, en réévaluant l'ordre des variables à chacune des étapes ou encore plus profondément, en classant les variables d'origine et en enlevant la moins importante puis en réitérant pour procéder à la phase d'élimination ou encore, en explorant plus de configurations suivant la valeur de m.

5.3 Garantie théorique

Il est très difficile d'obtenir des résultats de consistance de la sélection pour ce type de méthode dans un contexte de grande dimension. En général, on dispose de résultats sur des méthodes de même inspiration, essentiellement sur le modèle linéaire (voir Chen et al. (2014), Hesterberg et al. (2008), Paul et al. (2008) ou le livre de Giraud (2014)) et l'on utilise des heuristiques variées pour fouiller très partiellement la famille sous-jacente de modèles.

Un élément utile, même s'il est modeste, pour asseoir la validité théorique de ce type de procédure de sélection de variables basée sur une méthode progressive classique combinée avec l'utilisation de la mesure d'importance VI, est qu'une variable non informative, c'est-à-dire non incluse dans le modèle sous-jacent, a une importance théorique nulle.

Ishwaran et al. (2007) présente une première approche pour tenter d'expliquer de façon théorique le comportement de l'importance des variables, en étudiant une version simplifiée cette importance.

Récemment, Gregorutti et al. (2013) établit théoriquement que, dans le cas des modèles additifs, les variables non pertinentes ont une importance VI théorique nulle. Plus précisément, on dispose dans ce cas particulier d'une expression explicite de l'importance d'une variable comme la variance de la composante associée, à un facteur 2 près. Une conséquence immédiate, à l'évidente portée pratique, est qu'une variable non informative ou hors modèle a une importance nulle. Toujours dans ce cadre, une étude théorique de l'impact de la corrélation entre prédicteurs sur l'importance est menée. Fort de ces deux aspects, une

variante de l'algorithme RFE (Recursive Feature Elimination) reposant sur des classements des variables par importance, est proposée, puis étudiée par simulation et enfin, appliquée sur des données réelles d'aviation.

Notons enfin qu'un résultat similaire pour l'indice d'importance basé sur la diminution moyenne de l'impureté (voir 5.5.1) a été prouvée dans Louppe et al. (2013) et utilisée pour définir une stratégie de sélection de variables dans une situation de variables dont les interactions dépendent du contexte (voir Sutera et al. (2016)).

5.4 Exemple

Sur l'exemple des données `spam`, le classement des variables par ordre décroissant d'importance conduit à la Figure 7.

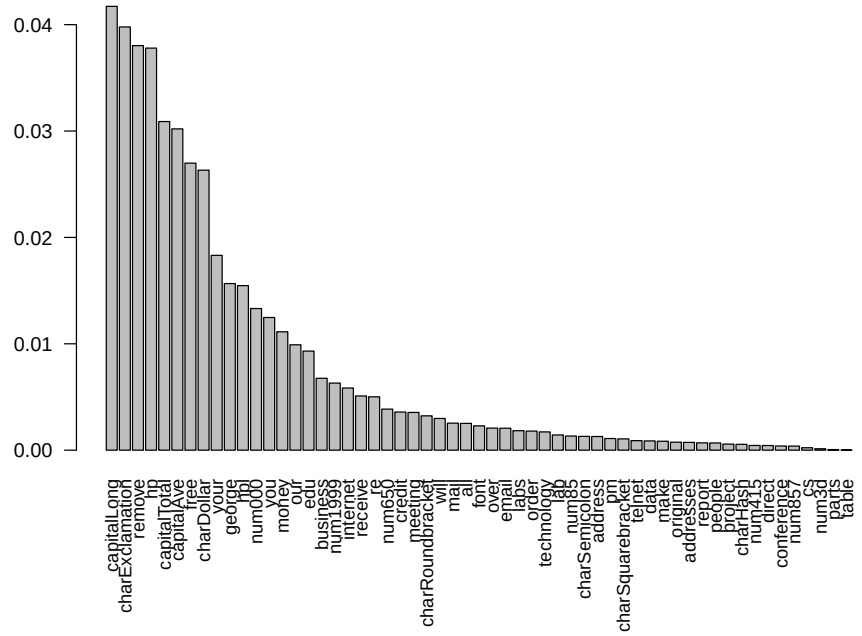


FIGURE 7 – Importance des variables en ordre décroissant

En se restreignant aux huit plus importantes qui semblent se dégager, on trouve les proportions d'occurrences des mots ou caractères `remove`, `hp`, `$`, `!`, `free` ainsi que les 3 variables liées aux longueurs des suites de lettres majuscules. Ce qui est assez conforme à l'intuition.

Remarquons ici que les variables impliquées dans les premières découpes de l'arbre optimal (voir la Figure 3) ne sont pas en tête du classement par importance des variables, même si elles apparaissent bien classées. Au contraire, certaines variables peuvent être importantes et ne pas apparaître dans l'arbre en question, tel est le cas ici de la variable la plus importante : `capitalLong`.

On peut ensuite s'intéresser au problème de la sélection des deux ensembles de variables pour l'interprétation et la prédiction dont l'objectif est de sélectionner deux sous-ensembles beaucoup plus petits que les 57 variables initiales sans trop dégrader les performances de la forêt initiale. Les résultats de l'application du package `VSURF` (voir Genuer et al. (2015)) sur les données `spam` sont rassemblés dans la Figure 8.

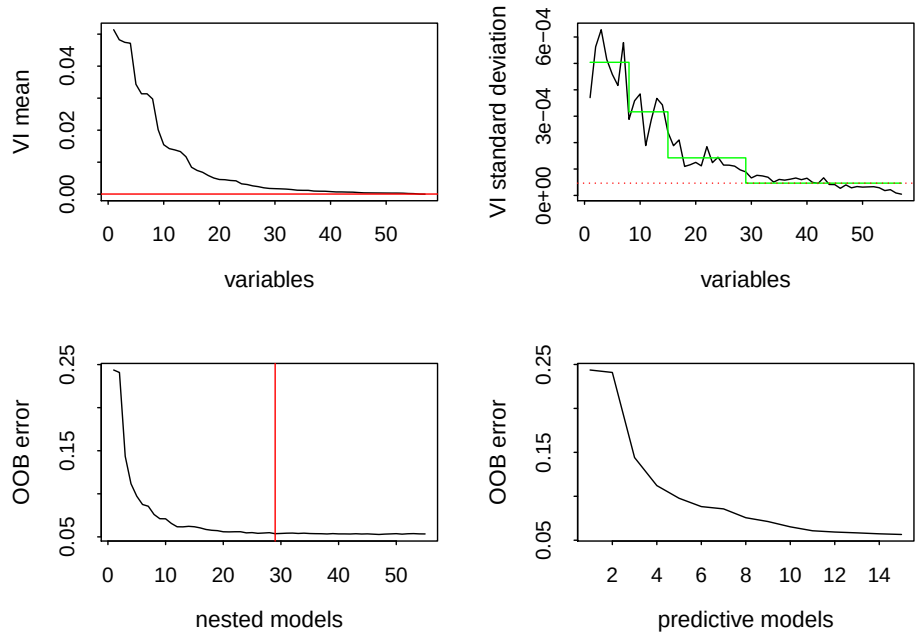


FIGURE 8 – La procédure `VSURF` appliquée aux données `spam`. Les graphiques du haut résument l'étape de seuillage, celui d'en bas à gauche illustre l'obtention de l'ensemble d'interprétation et en bas à droite, de prédiction.

Partant de 57 variables, la première étape d'élimination (voir les deux graphiques du haut de la Figure) les conserve quasiment toutes, l'étape d'interprétation (voir le graphique en bas à gauche) sélectionne environ la moitié (29) des variables et pour la prédiction (voir le graphique en bas à droite) une quinzaine.

Forêt	initiale	interprétation	prédiction
Erreur test	0.052	0.056	0.060

TABLE 4 – Erreurs test des RF utilisant respectivement les variables sélectionnées après l'étape d'interprétation et celle de prédiction, comparées à l'erreur d'une RF dite initiale, construite avec toutes les variables

Les performances sont rassemblées dans la Table 4. Il faut noter que, dans cet exemple, il y a une dégradation de l'erreur test vis-à-vis de la forêt initiale : de l'ordre de 8% pour l'ensemble d'interprétation et à nouveau 8% pour l'ensemble de prédiction. Mais elle est limitée si l'on remarque que l'erreur test finale est obtenue avec moins de 15 variables au lieu des 57 initialement considérées, et qu'elle est de l'ordre de grandeur de celle obtenue par le bagging.

5.5 Variantes et extensions

5.5.1 Variantes des mesures d'importance

Notons que Strobl et al. (2007) constatant que les VI sont biaisées en faveur des variables corrélées, propose dans Strobl et al. (2008) une variante du schéma de permutation appelé importance conditionnelle ne présentant pas ce défaut mais qui en augmente très notablement le coût de calcul. Il faut noter que ceci semble être particulièrement critique pour les

problèmes de grande dimension avec des prédicteurs fortement corrélés. Néanmoins, nos expériences précédentes Genuer et al. (2010b) sur la sélection de variables et, plus récemment, l'étude théorique de Gregorutti et al. (2013) montrent que, dans certaines situations, au contraire les scores de VI sont biaisés en faveur des variables non corrélées.

Signalons à nouveau qu'un indice basé sur la diminution moyenne de l'impureté (Mean Decrease Impurity importance (MDI)) et dont l'inspiration est dans le droit fil de l'importance des variables définie dans la méthode CART existe aussi. Il est défini, suivant Breiman (2001) ainsi : pour une variable donnée, son MDI est la somme des diminutions de l'impureté pour tous les nœuds où la variable est utilisée dans la découpe, pondérée par la proportion d'individus dans le nœud, en moyenne sur tous les arbres de la forêt. Même si son utilisation semble moins fréquente, elle a donné lieu à des résultats théoriques dans Louppe et al. (2013) et trouve des applications pratiques intéressantes malgré des limitations réelles comme la forte dépendance aux paramètres de la forêt et le fait qu'elle soit biaisée pour les arbres non élagués.

5.5.2 Extension à la sélection de variables fonctionnelles

Dans Gregorutti et al. (2015), on trouve une extension directe de l'importance des variables par permutation à un groupe de variables, il suffit pour cela de permuter, dans les échantillons OOB intervenant dans la définition de VI (voir 5.1), non pas seulement une variable mais de permuter solidairement toutes les variables d'un même groupe. Pour des modèles additifs fonctionnels, les résultats théoriques de Gregorutti et al. (2013) sont étendus. Ensuite une stratégie de sélection de variables dans un cadre de données fonctionnelles multivariées s'en déduit par le biais de la sélection de groupes de variables constitués des ensembles de coefficients d'ondelettes représentant les différentes variables fonctionnelles et l'utilisation d'une stratégie de type RFE.

6 Forêts aléatoires et Big Data

On ne saurait imaginer clore cet article sans évoquer le Big Data qui constitue l'un des prochains défis majeurs de la statistique. En effet, de récentes réflexions explorent déjà les nombreuses conséquences de ce nouveau contexte tant du point de vue algorithmique que sur le plan des implications théoriques (voir Fan et al. (2014); Hoerl et al. (2014); Jordan (2013)).

On se contente ici, en suivant Genuer et al. (2015)), de pointer quelques aspects spécifiques aux forêts aléatoires dans ce contexte, puisqu'elles y sont en principe fort bien adaptées.

On insiste toujours lorsque l'on parle du Big Data sur l'aspect données massives, mais il faut rappeler qu'il est souvent associé à deux autres traits importants : les flux de données (Aggarwal (2007)) et l'hétérogénéité des données (voir Besse et al. (2014) pour une introduction générale). Le problème posé par cette grande quantité de données est double : d'abord, la mise en oeuvre de nombreuses procédures statistiques peut prendre trop de temps pour fournir des résultats acceptables et d'autre part, elle requiert de distribuer les tâches entre les coeurs d'un même ordinateur ou bien encore de les distribuer au travers de plusieurs ordinateurs.

6.1 Passage à l'échelle

Une approche pour faire face à de telles données de grande taille est de "diviser pour régner". Le problème d'intérêt est divisé en sous-problèmes plus simples et les solutions sont ensuite combinées pour résoudre le problème d'origine. En particulier, le paradigme de programmation MapReduce (MR) (voir Figure 9) divise les données en petits sous-échantillons, tous traités indépendamment en parallèle. Plus précisément, MR se déroule en deux étapes :

dans une première étape, appelée l'étape Map, l'ensemble de données est divisé en plusieurs petits blocs de données, $(x_i, y_i)_{i\in\tau_k}$, avec $\cup_k \tau_k = \{1, \ldots, n\}$ et $\tau_k \cap \tau_{k'} = \emptyset$, chacun d'eux étant traité séparément par exemple par un des coeurs de l'une des machines du cluster le cas échéant. Ces différentes tâches Map sont indépendantes et produisent une liste de couples de la forme (clé, valeur), où "clé" est une clé indexant les données qui ont été utilisées pour construite "valeur". Puis, dans une seconde étape, appelée Reduce, chaque tâche Reduce traite toutes les sorties des tâches Map correspondant à une valeur de clé donnée.

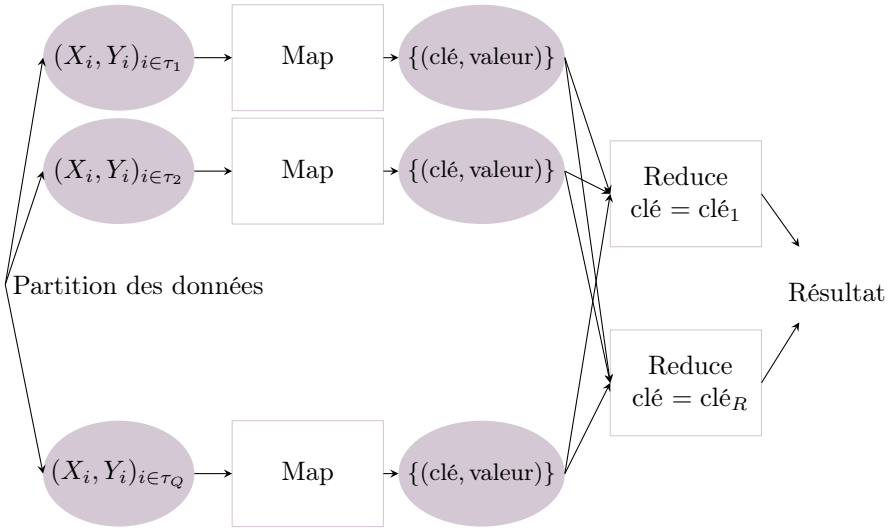


FIGURE 9 – Le paradigme de programmation MapReduce (MR)

On montre (voir Chu et al. (2010)) que nombre de méthodes statistiques peuvent facilement s'adapter à ce paradigme : tant que la méthode est basée sur le calcul d'une somme, des sommes partielles peuvent être calculées par les différentes tâches Map et l'étape Reduce consiste simplement dans le calcul de la somme finale comme somme des sommes partielles. Cette approche peut être appliquée directement à la régression linéaire, l'analyse discriminante, la classification naïve bayésienne et l'estimateur obtenu est exactement le même que celui qui aurait été obtenu directement par l'application globale de la méthode.

Cependant, la situation est un peu différente pour les RF, qui ne sont pas basées sur des sommes. Comme indiqué dans del Rio et al. (2014), la version standard MR des RF, noté MR-RF dans la suite, repose sur la construction parallèle des sous-forêts d'arbres obtenus sur des sous-échantillons bootstrap des sous-ensembles de données propres à chaque tâche Map : chaque bloc de données est ainsi envoyé à un travail Map dans lequel une RF (qui peut avoir un nombre modéré d'arbres) est construit (la clé de sortie est toujours égale à 1 et la valeur de sortie est la forêt). Il n'y a pas à proprement parler d'étape Reduce car la sortie des travaux Map est une collection de sous-forêts qui, fusionnées, donne la forêt finale. La prédiction elle-même, pour toutes les données, exige un autre passage de MR, comme expliqué dans del Rio et al. (2014). Cette méthode est celle mise en oeuvre dans la librairie Apache™ Mahout.

Mentionnons enfin que dans ce contexte, la moyenne des erreurs OOB des sous-forêts est directement accessible et est en général, de bonne qualité. Elle peut néanmoins ne pas être proche de l'erreur OOB de la forêt globale si l'échantillonnage est trop hétérogène entre les différentes tâches Map. Une remarque semblable quant à l'importance des variables s'en déduit. Dans cette situation on peut employer diverses stratégies pour se prévenir de ce risque (voir le paragraphe 6.3 ainsi que Genuer et al. (2015))).

6.2 Forêts aléatoires en ligne

L'idée générale des RF en ligne (ORF dans la suite), introduite par Saffari et al. (2009), est d'adapter la méthodologie RF au cas où les données arrivent séquentiellement, également connu sous le nom flux de données. Ce cadre suppose qu'à l'instant t on ne peut avoir accès à toutes les données du passé, mais seulement à l'observation courante ou quelques observations récentes. Les ORF ont été d'abord définies dans Saffari et al. (2009) et détaillées seulement pour la classification. Elles combinent l'idée du bagging en ligne de Oza (2005), les Extremely Randomized Trees (ERT) de Geurts et al. (2006) et un mécanisme pour mettre à jour la forêt à chaque instant d'arrivée d'une nouvelle observation.

Plus précisément, lorsqu'une nouvelle donnée arrive, le bagging en ligne met à jour k fois un arbre donné, où k est tiré au hasard suivant une loi de Poisson pour imiter un échantillon bootstrap par lots. Cela signifie que cette nouvelle donnée apparaîtra k fois dans l'arbre. Les forêts ERT sont utilisées au lieu des RF originales de Breiman, car elles permettent une mise à jour plus rapide : S découpes (définies par une variable et une valeur) sont tirées au hasard pour chaque nœud et le split final est optimisé uniquement parmi ces S splits candidats. En outre, toutes les décisions rendues par un arbre sont basées uniquement sur les proportions de chaque label dans les observations dans un nœud. ORF maintient à jour une mesure d'hétérogénéité sur la base de ces proportions, utilisées pour déterminer le label étiquetant un nœud. Donc quand un nœud est créé, S splits candidats (donc $2S$ nouveaux nœuds candidats) sont tirés au hasard et quand une nouvelle donnée arrive dans un nœud existant, cette mesure est mise à jour pour tous les $2S$ nœuds candidats. Ce mécanisme est répété jusqu'à ce qu'une condition d'arrêt soit réalisée et le split final minimise la mesure d'hétérogénéité parmi les S splits candidats. Ensuite, un nouveau nœud est créé et ainsi de suite.

L'erreur OOB d'un arbre t est aussi estimée en ligne : la donnée courante est OOB pour tous les arbres pour lesquels la variable aléatoire de Poisson utilisée pour répliquer l'observation dans l'arbre est égale à 0. La prédiction fournie pour un tel arbre t est utilisée pour mettre à jour la valeur courante de errTree_t l'erreur d'un arbre t. Cependant, la prédiction ne peut être réévaluée après que l'arbre ait été mis à jour avec la prochaine donnée, cette approche n'est donc qu'une approximation de errTree_t originale.

6.3 Echantillonnage et BLB

L'une des limites importantes de MR-RF est le biais d'échantillonnage possible provenant du découpage des données en plusieurs morceaux pour les travaux Map exécutés en parallèle. Cela pourrait être pris en compte en améliorant le plan de partition, en utilisant au moins une partition au hasard des données ou, probablement plus efficacement, grâce à une stratification selon la variable Y, ou encore de sous-échantillonner plus soigneusement en exploitant le fait que l'ensemble des données n'est probablement pas nécessaire pour obtenir des estimations précises.

Une autre alternative intéressante à un sous-échantillonnage et la stratification est le BLB (Bag of Little Bootstrap) décrite dans Kleiner et al. (2014). Cette méthode vise la construction d'échantillons bootstrap de taille n, chacun contenant seulement $m \ll n$ données différentes. La taille de l'échantillon est classique (n), évitant ainsi le problème du biais impliqué par la méthode du m parmi n bootstrap, conséquence directe de la stratégie de MR (chaque fragment contient une partie de l'ensemble de données et entraîne ainsi un échantillon bootstrap de taille m). Le traitement de cet échantillon est simplifié par une pondération intelligente et est donc gérable même pour n très grand puisqu'il ne contient qu'une petite fraction (m/n) d'observations différentes de l'ensemble de données d'origine. Cette approche est de surcroît, bien étayée par des résultats théoriques puisque les auteurs prouvent son équivalence avec la méthode bootstrap standard.

6.4 Garantie théorique

En dehors du résultat de consistance de Kleiner et al. (2014), mais qui porte sur de l'estimation par bootstrap et n'est pas directement applicable aux forêts BLB introduite précédemment, très peu de garanties théoriques sont disponibles pour les variante big data de forêts aléatoires.

Une exception notable est le résultat de consistance pour des forêts aléatoires en ligne, dû à Denil et al. (2013). Les auteurs introduisent une variante de ORF comportant deux principales différences. Tout d'abord, aucun bootstrap en ligne n'est effectué et d'autre part, le flux de données est divisé au hasard en deux flux : le flux de structure et le flux d'estimation. Les données du flux de structure sont utilisées seulement pour l'optimisation des découpes, tandis que les données du flux d'estimation ne sont utilisées que pour attribuer un label à un nœud. Rappelons que dans les ORF, et aussi dans la variante de Denil et al. (2013), les points de coupures sont choisis avec une loi uniforme. Ceci permet des mises à jour très rapides, nécessaires dans un cadre en ligne, mais permet également de simplifier l'analyse théorique de la méthode.

De plus, le découplage en données de structure et données d'estimation, qui permet d'avoir indépendance entre les labels associés aux feuilles d'un arbre et la partition de l'arbre, facilite l'obtention du résultat. Ce découplage, qui a déjà été introduit par Biau (2012) et utilisé dans des études de simulations par Arlot and Genuer (2014) et Arlot and Genuer (2016), a également permis à ces mêmes auteurs d'obtenir un résultat de consistance pour une variante (hors ligne) très proche des RF-RI dans un cadre de régression dans Denil et al. (2014).

7 Remerciements

Nous tenons à remercier les collègues qui ont accompagné nos réflexions sur ces sujets au travers de nombreuses collaborations, en particulier Sylvain Arlot, Servane Gey, Christine Tuleau-Malot et Nathalie Villa-Vialaneix.

Signalons enfin que ce document est une version préliminaire d'un chapitre d'un ouvrage en préparation édité par Myriam Maumy-Bertrand, Gilbert Saporta et Christine Thomas-Agnan, Apprentissage Statistique et Données Massives, Technip, 2017, issu des Journées d'Etudes en Statistique de la SFdS, 2-7 octobre 2017, Fréjus, France.

Bibliographie

Afanador, N. L., Smolinska, A., Tran, T. N., and Blanchet, L. (2016). Unsupervised random forest : a tutorial with case studies. *Journal of Chemometrics*, 30(5) :232–241.

Aggarwal, C. C. (2007). *Data streams : models and algorithms*, volume 31. Springer Science & Business Media.

Amit, Y. and Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural computation*, 9(7) :1545–1588.

Archer, K. J. and Kimes, R. V. (2008). Empirical characterization of random forest variable importance measures. *Computational Statistics & Data Analysis*, 52(4) :2249–2260.

Arlot, S. and Genuer, R. (2014). Analysis of purely random forests bias. *arXiv preprint arXiv :1407.3939*.

Arlot, S. and Genuer, R. (2016). Comments on : A random forest guided tour. *TEST*, 25(2) :228–238.

Azen, R. and Budescu, D. V. (2003). The dominance analysis approach for comparing predictors in multiple regression. *Psychological methods*, 8(2) :129.

Bach, F. R. (2008). Bolasso : model consistent lasso estimation through the bootstrap. In *Proceedings of the 25th international conference on Machine learning*, pages 33–40. ACM.

Banerjee, M., McKeague, I. W., et al. (2007). Confidence sets for split points in decision trees. *The Annals of Statistics*, 35(2) :543–574.

Bar-Hen, A., Gey, S., and Poggi, J.-M. (2015). Influence measures for cart classification trees. *Journal of Classification*, 32(1) :21–45.

Bartlett, P. L. and Traskin, M. (2007). Adaboost is consistent. *Journal of Machine Learning Research*, 8(Oct) :2347–2368.

Bel, L., Allard, D., Laurent, J., Cheddadi, R., and Bar-Hen, A. (2009). Cart algorithm for spatial data : Application to environmental and ecological data. *Computational Statistics & Data Analysis*, 53(8) :3082–3093.

Besse, P., Garivier, A., and Loubes, J. (2014). Big data - Retour vers le futur 3. De statisticien à data scientist. arXiv preprint arXiv :1403.3758.

Biau, G. (2012). Analysis of a random forests model. *Journal of Machine Learning Research*, 13(Apr) :1063–1095.

Biau, G., Cérou, F., and Guyader, A. (2010). On the rate of convergence of the bagged nearest neighbor estimate. *Journal of Machine Learning Research*, 11(Feb) :687–712.

Biau, G. and Devroye, L. (2010). On the layered nearest neighbour estimate, the bagged nearest neighbour estimate and the random forest method in regression and classification. *Journal of Multivariate Analysis*, 101(10) :2499–2518.

Biau, G., Devroye, L., and Lugosi, G. (2008). Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, 9(Sep) :2015–2033.

Biau, G. and Scornet, E. (2016). A random forest guided tour. *Test*, 25(2) :197–227.

Biau, G., Scornet, E., and Welbl, J. (2016). Neural random forests. *arXiv preprint arXiv :1604.07143*.

Blanchard, G., Schäfer, C., Rozenholc, Y., and Müller, K.-R. (2007). Optimal dyadic decision trees. *Machine Learning*, 66(2-3) :209–241.

Blaser, R. and Fryzlewicz, P. (2015). Random rotation ensembles. *J Mach Learning Res*, 2 :1–15.

Bou-Hamad, I., Larocque, D., Ben-Ameur, H., et al. (2011). A review of survival trees. *Statistics Surveys*, 5 :44–71.

Boulesteix, A.-L., Janitza, S., Kruppa, J., and König, I. R. (2012). Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics. *Wiley Interdisciplinary Reviews : Data Mining and Knowledge Discovery*, 2(6) :493–507.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2) :123–140.

Breiman, L. (1998). Arcing classifier. *The annals of statistics*, 26(3) :801–849.

Breiman, L. (2000a). Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3) :229–242.

Breiman, L. (2000b). Some infinity theory for predictor ensembles. Technical report, Technical Report 579, Statistics Dept. UCB.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1) :5–32.

Breiman, L. and Cutler, A. (2004). Random forest-manual.

Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984). *Classification and Regression Trees*. Chapman & Hall, New York.

Bühlmann, P. and Yu, B. (2002). Analyzing bagging. *Annals of Statistics*, pages 927–961.

Cadenas, J. M., Carmen Garrido, M., and Martínez, R. (2013). Feature subset selection filter-wrapper based on low quality data. *Expert Systems with Applications*, 40(16) :6241–6252.

Chen, Y., Du, P., and Wang, Y. (2014). Variable selection in linear models. *Wiley Interdisciplinary Reviews : Computational Statistics*, 6(1) :1–9.

Cheze, N. and Poggi, J.-M. (2006). Iterated boosting for outlier detection. In *Data Science and Classification*, pages 213–220. Springer.

Chu, C., Kim, S., Lin, Y., Yu, Y., Bradski, G., Ng, A., and Olukotun, K. (2010). Map-Reduce for machine learning on multicore. In Lafferty, J., Williams, C., Shawe-Taylor, J., Zemel, R., and Culotta, A., editors, *Advances in Neural Information Processing Systems (NIPS 2010)*, volume 23, pages 281–288, Hyatt Regency, Vancouver, Canada.

Clémençon, S., Depecker, M., and Vayatis, N. (2013). Ranking forests. *Journal of Machine Learning Research*, 14(Jan) :39–73.

Cutler, A. (2010). Remembering leo breiman. *The Annals of Applied Statistics*, 4(4) :1621–1633.

Cutler, A. and Zhao, G. (2001). Pert-perfect random tree ensembles. *Computing Science and Statistics*, 33 :490–497.

del Rio, S., López, V., Beniítez, J., and Herrera, F. (2014). On the use of MapReduce for imbalanced big data using random forest. *Information Sciences*, 285 :112–137.

Denil, M., Matheson, D., and de Freitas, N. (2013). Consistency of online random forests. In *Proceedings of the 30th International Conference on Machine Learning (ICML 2013)*, pages 1256–1264.

Denil, M., Matheson, D., and De Freitas, N. (2014). Narrowing the gap : Random forests in theory and in practice. In *ICML*, pages 665–673.

Díaz-Uriarte, R. and Alvarez De Andres, S. (2006). Gene selection and classification of microarray data using random forest. *BMC Bioinformatics*, 7(1) :3.

Dietterich, T. G. (2000a). Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer.

Dietterich, T. G. (2000b). An experimental comparison of three methods for constructing ensembles of decision trees : Bagging, boosting, and randomization. *Machine learning*, 40(2) :139–157.

Donoho, D. L. et al. (1997). Cart and best-ortho-basis : a connection. *The Annals of Statistics*, 25(5) :1870–1911.

Drucker, H. (1997). Improving regressors using boosting techniques. In *ICML*, volume 97, pages 107–115.

Duroux, R. and Scornet, E. (2016). Impact of subsampling and pruning on random forests. *arXiv preprint arXiv :1603.04261*.

Fan, J., Han, F., and Liu, H. (2014). Challenges of big data analysis. *National Science Review*, 1(2) :293–314.

Fan, J. and Lv, J. (2008). Sure independence screening for ultrahigh dimensional feature space. *Journal of the Royal Statistical Society B*, 70(5) :849–911.

Fawagreh, K., Gaber, M. M., and Elyan, E. (2015). An outlier detection-based tree selection approach to extreme pruning of random forests. *arXiv preprint arXiv :1503.05187*.

Fellinghauer, B., Bühlmann, P., Ryffel, M., Von Rhein, M., and Reinhardt, J. D. (2013). Stable graphical model estimation with random forests for discrete, continuous, and mixed variables. *Computational Statistics & Data Analysis*, 64 :132–152.

Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems. *J. Mach. Learn. Res*, 15(1) :3133–3181.

Freund, Y. and Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1) :119 – 139.

Freund, Y., Schapire, R. E., et al. (1996). Experiments with a new boosting algorithm. In *13th International Conference on Machine Learning*, volume 96, pages 148–156.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67.

Genuer, R. (2012). Variance reduction in purely random forests. *Journal of Nonparametric Statistics*, 24(3) :543–562.

Genuer, R., Michel, V., Eger, E., and Thirion, B. (2010a). Random forests based feature selection for decoding fmri data. In *Proceedings Compstat*, number 267, pages 1–8.

Genuer, R., Poggi, J.-M., and Tuleau, C. (2008). Random forests : some methodological insights. arXiv :0811.3619.

Genuer, R., Poggi, J.-M., and Tuleau-Malot, C. (2010b). Variable selection using random forests. *Pattern Recognition Letters*, 31(14) :2225–2236.

Genuer, R., Poggi, J.-M., and Tuleau-Malot, C. (2015). Vsurf : An r package for variable selection using random forests. *The R Journal*, 7(2) :19–33.

Genuer, R., Poggi, J.-M., Tuleau-Malot, C., and Villa-Vialaneix, N. (2015). Random Forests for Big Data. *ArXiv e-prints*.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, 63(1) :3–42.

Gey, S. (2002). *Bornes de risque, détection de ruptures, boosting : trois thèmes statistiques autour de CART en régression*. PhD thesis, Paris 11, Orsay.

Gey, S. (2012). Risk bounds for cart classifiers under a margin condition. *Pattern Recognition*, 45(9) :3523–3534.

Gey, S. and Nedelec, E. (2005). Model selection for cart regression trees. *IEEE Transactions on Information Theory*, 51(2) :658–670.

Gey, S. and Poggi, J.-M. (2006). Boosting and instability for regression trees. *Computational statistics & data analysis*, 50(2) :533–550.

Ghattas, B. (1999). Prévisions des pics d'ozone par arbres de régression, simples et agrégés par bootstrap. *Revue de statistique appliquée*, 47(2) :61–80.

Ghattas, B. (2000). Agrégation d'arbres de classification. *Revue de statistique appliquee*, 48(2) :85–98.

Ghattas, B. and Ben Ishak, A. (2008). Sélection de variables pour la classification binaire en grande dimension : comparaisons et application aux données de biopuces. *Journal de la société française de statistique*, 149(3) :43–66.

Giraud, C. (2014). *Introduction to High-Dimensional Statistics*, volume 139 of *Monographs on Statistics and Applied Probability*. Chapman and Hall/CRC, Boca Raton, FL.

Goldstein, B. A., Hubbard, A. E., Cutler, A., and Barcellos, L. F. (2010). An application of random forests to a genome-wide association dataset : methodological considerations & new findings. *BMC genetics*, 11(1) :1.

Gregorutti, B., Michel, B., and Saint-Pierre, P. (2013). Correlation and variable importance in random forests. *Statistics and Computing*, pages 1–20.

Gregorutti, B., Michel, B., and Saint-Pierre, P. (2015). Grouped variable importance with random forests and application to multiple functional data analysis. *Computational Statistics & Data Analysis*, 90 :15–35.

Grömping, U. (2015). Variable importance in regression models. *Wiley Interdisciplinary Reviews : Computational Statistics*, 7(2) :137–152.

Guyon, I., Weston, J., Barnhill, S., and Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3) :389–422.

Hapfelmeier, A. and Ulm, K. (2012). A new variable selection approach using random forests. *Computational Statistics & Data Analysis*, 60 :50–69.

Hara, S. and Hayashi, K. (2016). Making tree ensembles interpretable. *arXiv preprint arXiv :1606.05390*.

Hesterberg, T., Choi, N. H., Meier, L., Fraley, C., et al. (2008). Least angle and l1 penalized regression : A review. *Statistics Surveys*, 2 :61–93.

Ho, T. K. (1998). The random subspace method for constructing decision forests. *IEEE transactions on pattern analysis and machine intelligence*, 20(8) :832–844.

Hoerl, R., Snee, R., and De Veaux, R. (2014). Applying statistical thinking to 'Big Data' problems. *Wiley Interdisciplinary Reviews : Computational Statistics*, 6(4) :222–232.

Hothorn, T., Bühlmann, P., Dudoit, S., Molinaro, A., and Van Der Laan, M. J. (2006). Survival ensembles. *Biostatistics*, 7(3) :355–373.

Ishwaran, H. et al. (2007). Variable importance in binary regression trees and forests. *Electronic Journal of Statistics*, 1 :519–537.

Ishwaran, H. and Kogalur, U. B. (2010). Consistency of random survival forests. *Statistics & probability letters*, 80(13) :1056–1064.

Ishwaran, H., Kogalur, U. B., Blackstone, E. H., and Lauer, M. S. (2008). Random survival forests. *The annals of applied statistics*, pages 841–860.

Jordan, M. (2013). On statistics, computation and scalability. *Bernoulli*, 19(4) :1378–1390.

Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied statistics*, pages 119–127.

Kleiner, A., Talwalkar, A., Sarkar, P., and Jordan, M. (2014). A scalable bootstrap for massive data. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 76(4) :795–816.

Lê Cao, K.-A., Gonçalves, O., Besse, P., and Gadat, S. (2007). Selection of biologically relevant genes with a wrapper stochastic algorithm. *Statistical Applications in Genetics and Molecular Biology*, 6(1).

LeBlanc, M. and Crowley, J. (1993). Survival trees by goodness of split. *Journal of the American Statistical Association*, 88(422) :457–467.

Lecué, G. (2007). *Méthodes d'agrégation : optimalité et vitesses rapides*. PhD thesis, Paris 6.

Liaw, A. and Wiener, M. (2002). Classification and regression by randomforest. *R News*, 2(3) :18–22.

Liu, F. T., Ting, K. M., and Fan, W. (2005). Maximizing tree diversity by building complete-random decision trees. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 605–610. Springer.

Loh, W.-Y. (2014). Fifty years of classification and regression trees. *International Statistical Review*, 82(3) :329–348.

Louppe, G., Wehenkel, L., Sutera, A., and Geurts, P. (2013). Understanding variable importances in forests of randomized trees. In *Advances in neural information processing systems*, pages 431–439.

Meinshausen, N. (2006). Quantile regression forests. *Journal of Machine Learning Research*, 7(Jun) :983–999.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society : Series B (Statistical Methodology)*, 72(4) :417–473.

Mentch, L. and Hooker, G. (2014). Quantifying uncertainty in random forests via confidence intervals and hypothesis tests. *arXiv preprint arXiv :1404.6473*.

Menze, B. H., Kelm, B. M., Splitthoff, D. N., Koethe, U., and Hamprecht, F. A. (2011). On oblique random forests. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 453–469. Springer.

Molinaro, A. M., Dudoit, S., and Van der Laan, M. J. (2004). Tree-based multivariate regression and density estimation with right-censored data. *Journal of Multivariate Analysis*, 90(1) :154–177.

Nilsson, R., Peña, J. M., Björkegren, J., and Tegnér, J. (2007). Consistent feature selection for pattern recognition in polynomial time. *The Journal of Machine Learning Research*, 8 :589–612.

Olshen, R. and Breiman, L. (2001). A conversation with leo breiman. *Statistical Science*, pages 184–198.

Oza, N. (2005). Online bagging and boosting. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, volume 3, pages 2340–2345. IEEE.

Patil, D. V. and Bichkar, R. (2012). Issues in optimization of decision tree learning : A survey. *International Journal of Applied Information Systems (IJAIS), New York, USA*, 3(5).

Paul, D., Bair, E., Hastie, T., and Tibshirani, R. (2008). " preconditioning" for feature selection and regression in high-dimensional problems. *The Annals of Statistics*, pages 1595–1618.

Poggi, J.-M. and Tuleau, C. (2006). Classification supervisée en grande dimension. application à l'agrément de conduite automobile. *Revue de Statistique Appliquée*, 54(4) :41–60.

Prasad, A. M., Iverson, L. R., and Liaw, A. (2006). Newer classification and regression tree techniques : bagging and random forests for ecological prediction. *Ecosystems*, 9(2) :181–199.

Questier, F., Put, R., Coomans, D., Walczak, B., and Vander Heyden, Y. (2005). The use of cart and multivariate regression trees for supervised and unsupervised feature selection. *Chemometrics and Intelligent Laboratory Systems*, 76(1) :45–54.

Quinlan, J. R. (1993). *C4. 5 : Programming for machine learning*. Morgan Kauffmann.

R Core Team (2015). *R : A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rakotomamonjy, A. (2003). Variable selection using svm-based criteria. *Journal of machine learning research*, 3(Mar) :1357–1370.

Saffari, A., Leistner, C., Santner, J., Godec, M., and Bischof, H. (2009). On-line random forests. In *Proceedings of IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1393–1400. IEEE.

Sauve, M. and Tuleau-Malot, C. (2014). Variable selection through cart. *ESAIM : Probability and Statistics*, 18 :770–798.

Scornet, E. (2016). Random forests and kernel methods. *IEEE Transactions on Information Theory*, 62(3) :1485–1500.

Scornet, E., Biau, G., Vert, J.-P., et al. (2015). Consistency of random forests. *The Annals of Statistics*, 43(4) :1716–1741.

Stekhoven, D. J. and Bühlmann, P. (2012). Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1) :112–118.

Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., and Zeileis, A. (2008). Conditional variable importance for random forests. *BMC Bioinformatics*, 9(1) :307.

Strobl, C., Boulesteix, A.-L., Zeileis, A., and Hothorn, T. (2007). Bias in random forest variable importance measures : Illustrations, sources and a solution. *BMC bioinformatics*, 8(1) :25.

Sutera, A., Louppe, G., Huynh-Thu, V. A., Wehenkel, L., and Geurts, P. (2016). Context-dependent feature analysis with random forests. *arXiv preprint arXiv :1605.03848*.

Tang, E. K., Suganthan, P. N., and Yao, X. (2006). An analysis of diversity measures. *Machine Learning*, 65(1) :247–271.

Therneau, T., Atkinson, B., and Ripley, B. (2015). *rpart : Recursive Partitioning and Regression Trees*. R package version 4.1-9.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, series B*, 58(1) :267–2 88.

Valdiviezo, H. C. and Van Aelst, S. (2015). Tree-based prediction on incomplete data using imputation or surrogate decisions. *Information Sciences*, 311 :163–181.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer Science & Business Media.

Verikas, A., Gelzinis, A., and Bacauskiene, M. (2011). Mining data with random forests : A survey and results of new tests. *Pattern Recognition*, 44(2) :330–349.

Wager, S. (2014). Asymptotic theory for random forests. *arXiv preprint arXiv :1405.0352*.

Wallard, H. (2015). *Analyse des leviers. Effets de colinéarité et hiérarchisation des impacts dans les études de marché et sociales.* Ph.D. thesis, CNAM, Paris.

Winham, S. J., Freimuth, R. R., and Biernacka, J. M. (2013). A weighted random forests approach to improve predictive performance. *Statistical analysis and data mining*, 6(6) :496–505.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, G. J., Ng, A., Liu, B., Philip, S. Y., et al. (2008). Top 10 algorithms in data mining. *Knowledge and information systems*, 14(1) :1–37.

Yan, D., Chen, A., and Jordan, M. I. (2013). Cluster forests. *Computational Statistics & Data Analysis*, 66 :178–192.

Zhang, H. and Singer, B. (2013). *Recursive partitioning in the health sciences*. Springer Science & Business Media.

Zhu, R., Zeng, D., and Kosorok, M. R. (2015). Reinforcement learning trees. *Journal of the American Statistical Association*, 110(512) :1770–1784.