App. statistique : méthodes non linéaires pour la régression
Les modèles à base d'arbres

C. HELBERT

Contexte :

- $X_1, ..., X_p$ sont des variables explicatives (descripteurs, prédicteurs)
- Y est la variable à expliquer quantitative.

Contexte :

- $X_1, ..., X_p$ sont des variables explicatives (descripteurs, prédicteurs)
- Y est la variable à expliquer quantitative.

Modèles présentés dans la suite du cours :

- alternatives aux méthodes de régression (linéaires)
- hypothèses sous-jacentes à ces modèles permettent d'aborder la grande dimension $p >> n$

Plan

Contexte :

- $X_1, ..., X_p$ sont des variables explicatives (descripteurs, prédicteurs) à valeurs dans $[-1, 1]$
- Y est la variable à expliquer quantitative.

Contexte :

- $X_1, ..., X_p$ sont des variables explicatives (descripteurs, prédicteurs) à valeurs dans $[-1, 1]$
- Y est la variable à expliquer quantitative.

CART ("Classification and Regression Tree") :

- méthode simple, modèle interprétable
- valable également en grande dimension $p >> n$
- nombreuses extensions prenant en compte les défauts de la méthode.

Principe : Découpage de l'espace en régions , $R_1, ..., R_M$ sur lesquelles la réponse est constante.

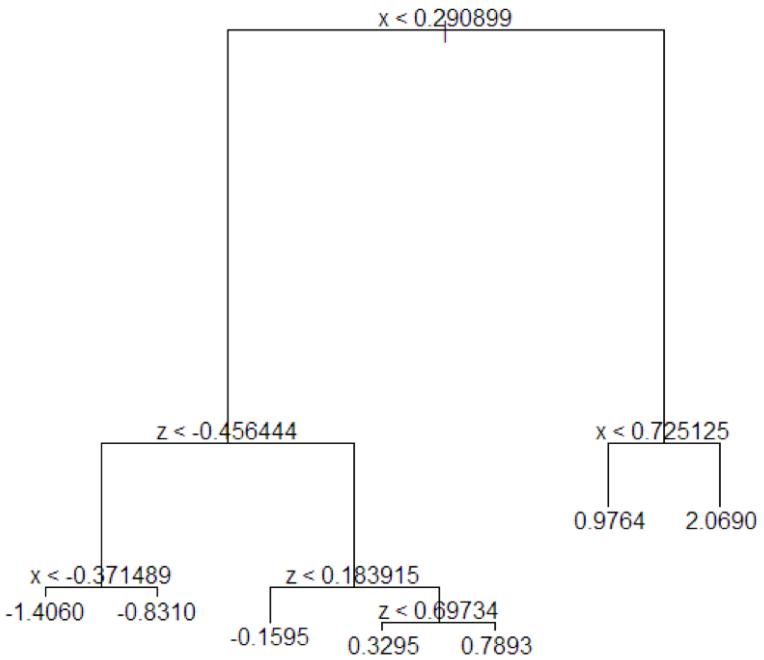
La prédiction a alors la forme suivante :

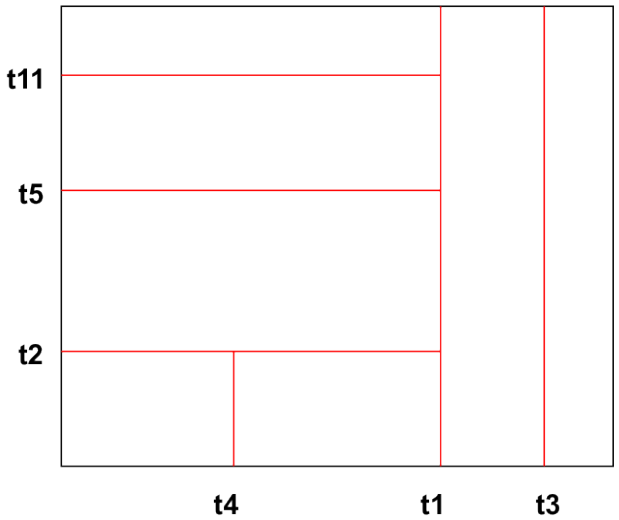
$$\hat{f}(X) = \sum_{m=1}^{M} c_m \mathbf{1}_{R_m}(X)$$

Simplification : découpage binaire récusif

- On découpe de façon récursive et parallèle aux axes
- Description simplifiée de la partition de l'espace (réunion de rectangles)
- Présentation sous forme d'un arbre (lecture facile du modèle)

CART - Illustration





Avantages :

- Modèle simple à interpréter
- Prise en compte des interactions

Inconvénients :

- Grande sensibilité au plan d'expériences
- Découpage pas forcément optimal car parallèle aux axes
- Estimation par une constante (surfaces discontinues)

CART - Estimation

1 Remarque : Pour une partition de l'espace $R_1, ..., R_M$ donnée, la meilleure estimation de $c_1, ..., c_M$ est donnée par la moyenne :

$$\hat{c}_j = \frac{1}{\sum_{i=1}^{n} \mathbf{1}_{R_j}(\mathbf{x_i})} \sum_{i=1}^{n} y_i \mathbf{1}_{R_j}(\mathbf{x_i})$$

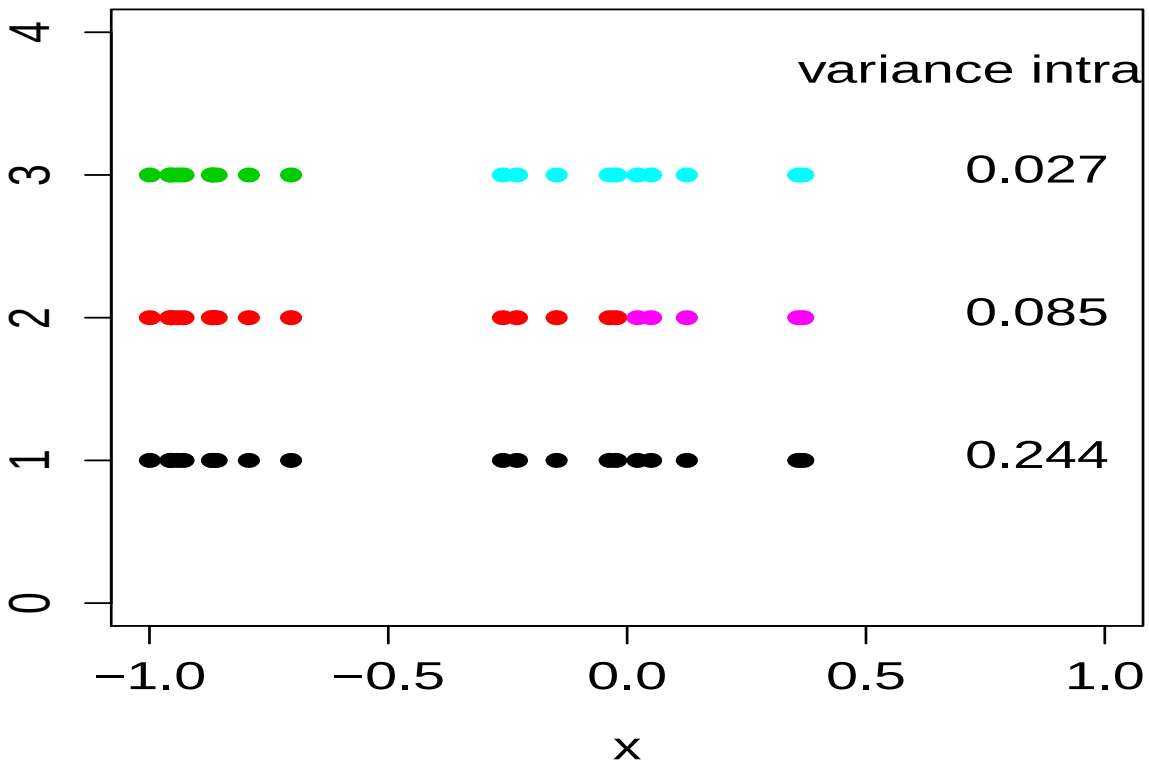
CART - Estimation

2 Estimation des régions : Le découpage se fait récursivement dans chaque branche de l'arbre.
Découper une fois : trouver $(j, t) \in \{1, ..., p\}x[-1, 1]$ tel que la quantité suivante soit minimale :

$$\sum_{i=1}^{n} \left((y_i - \hat{c}_j^-)^2 \mathbf{1}_{[-1,t[}(x_{ij}) + (y_i - \hat{c}_j^+)^2 \mathbf{1}_{[t,1]}(x_{ij}) \right)$$

où $\hat{c}_j^-$ et $\hat{c}_j^+$ sont estimées comme au point 1.

CART - Estimation



CART - Estimation

3 Arrêt de la procédure : attention au compromis biais - variance
   - Découpage trop fin : erreur d'apprentissage très faible (peu de biais), modèle mauvais en prédiction (variance élevée, trop proche des données)
   - Découpage trop grossier : pas assez informatif (biais important).

CART - Estimation

Idée :

- on construit l'arbre le plus fin possible ,$\mathcal{T}_0$, par exemple on impose un nombre minimum d'observations par feuille de 5,
- on l'élague a posteriori sur un critère type AIC (compromis bon apprentissage / faible complexité).

CART - Estimation

Idée :

- on construit l'arbre le plus fin possible ,$\mathcal{T}_0$, par exemple on impose un nombre minimum d'observations par feuille de 5,
- on l'élague a posteriori sur un critère type AIC (compromis bon apprentissage / faible complexité).

On cherche $\mathcal{T}_\alpha$ inclus dans $\mathcal{T}_0$ tel que $C_\alpha(\mathcal{T})$ soit minimum, où

$$C_\alpha(\mathcal{T}) = \sum_{i=1}^{n} \sum_{m=1}^{M} (y_i - \hat{c}_{Rm})^2 \mathbf{1}_{Rm}(x_i) + \alpha M$$

CART - Estimation

Idée :

- on construit l'arbre le plus fin possible ,$\mathcal{T}_0$, par exemple on impose un nombre minimum d'observations par feuille de 5,
- on l'élague a posteriori sur un critère type AIC (compromis bon apprentissage / faible complexité).

On cherche $\mathcal{T}_\alpha$ inclus dans $\mathcal{T}_0$ tel que $C_\alpha(\mathcal{T})$ soit minimum, où

$$C_\alpha(\mathcal{T}) = \sum_{i=1}^{n} \sum_{m=1}^{M} (y_i - \hat{c}_{Rm})^2 \mathbf{1}_{Rm}(x_i) + \alpha M$$

Remarque : α est choisi par validation croisée.

Avantages :

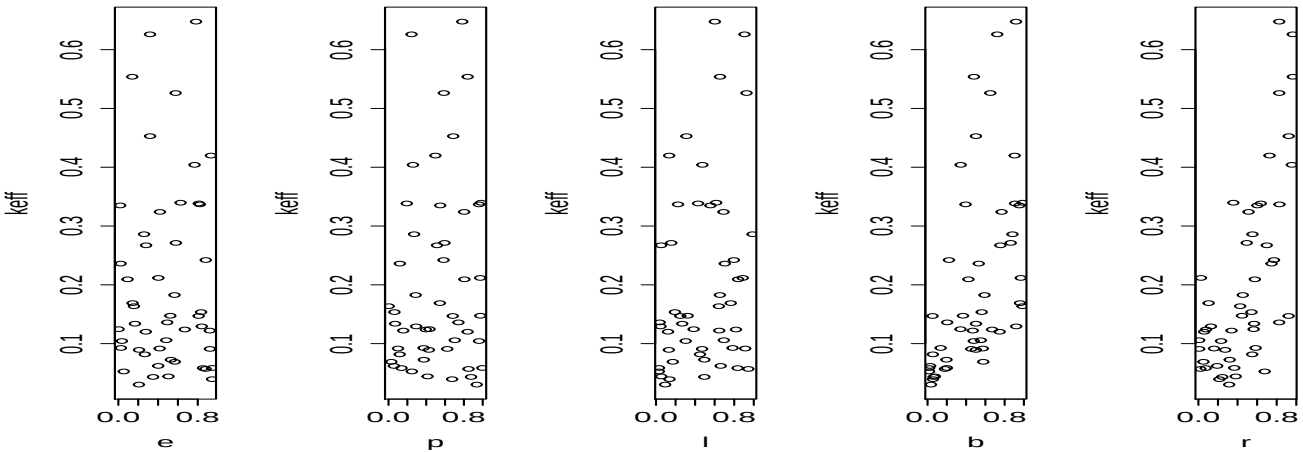
- Simplicité d'interprétation
- Prise en compte des non linéarités et des interactions

Inconvénients :

- Surface de prédiction discontinue (-> extensions vers des modèles de type MARS)
- Sensibilité aux points du plan d'expériences (-> techniques de bagging)

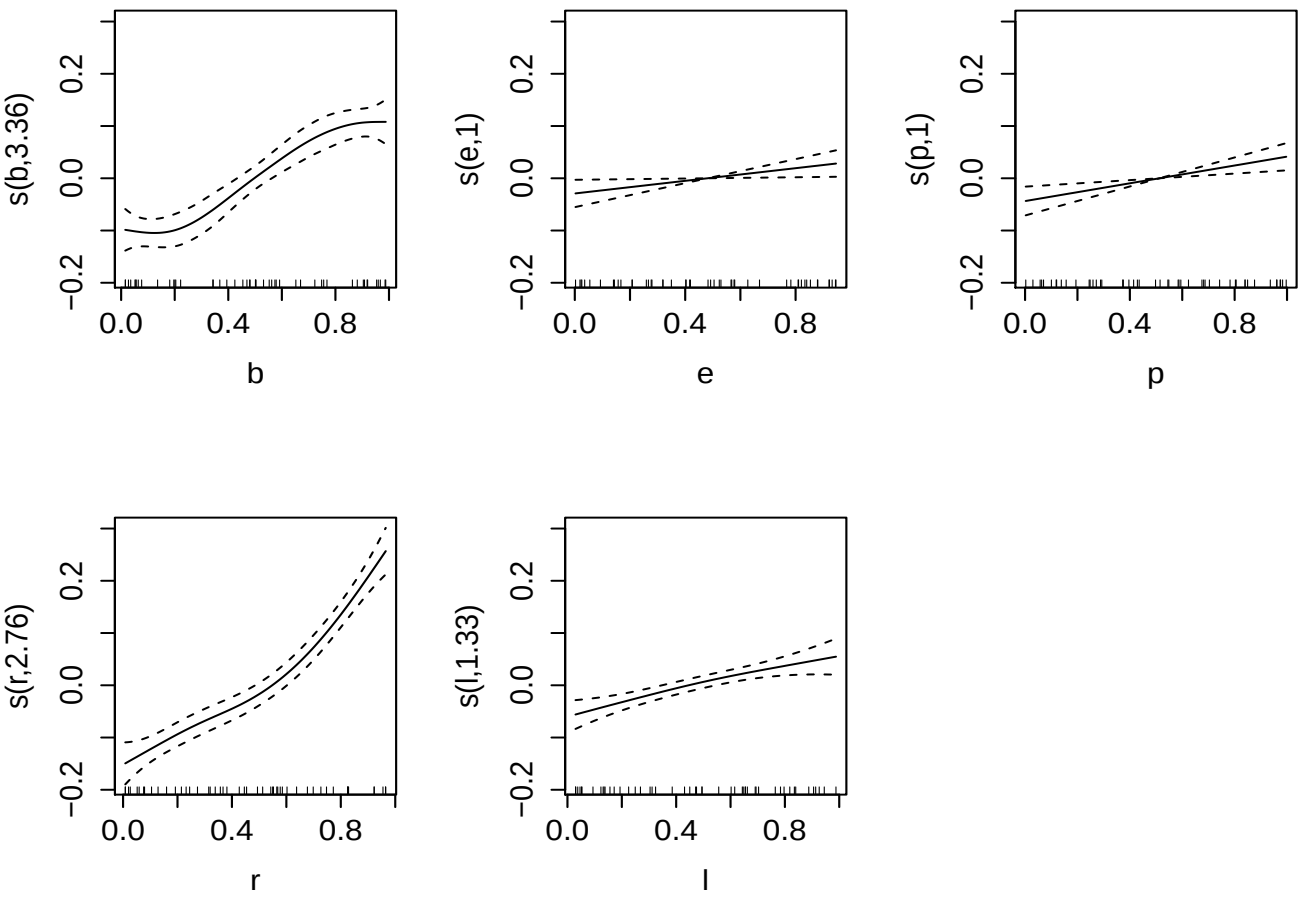
Exemple cas IRSN

- *keff* : criticité neutronique
- *e* : enrichissement
- *p* : pas de la grille
- *l* : longueur des crayons d'uranium
- *b* : densité du brouillard d'eau
- *r* : coefficient de reflexion de la gaine



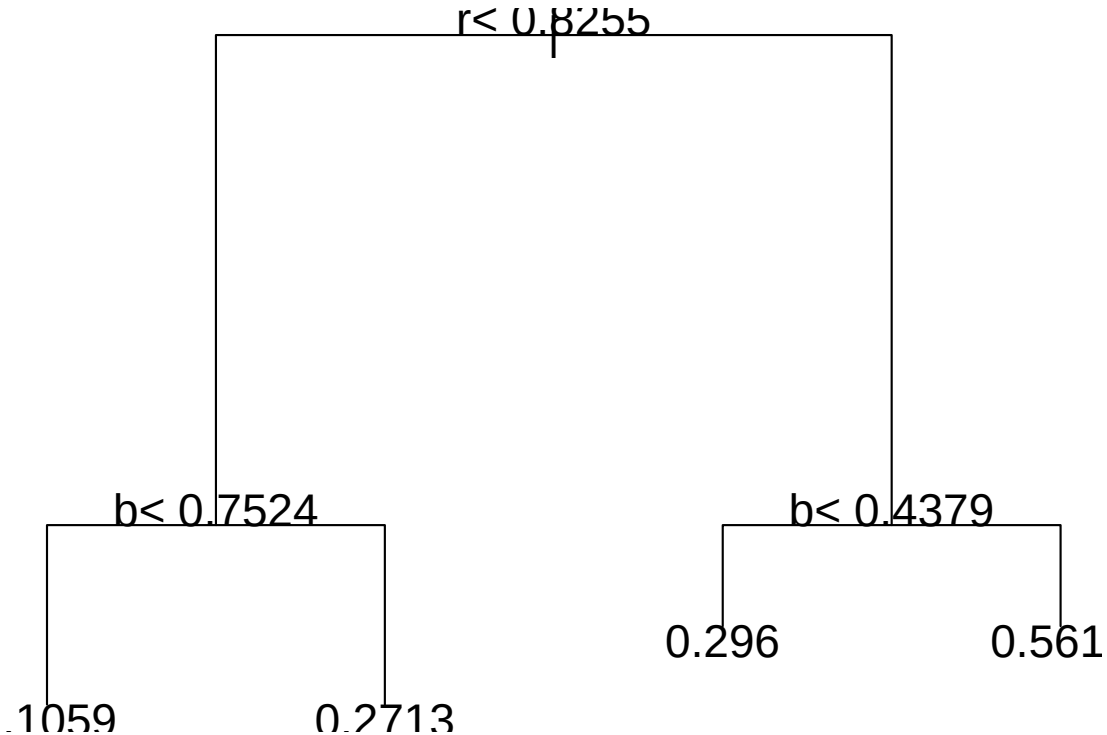
Exemple cas IRSN - modèle additif

Package **mgcv**. Le $R2$ sur l'ensemble test (324 observations) est égale à 0.77.



Exemple cas IRSN - modèle arbre

Package **mgcv**. Le $R2$ sur l'ensemble test (324 observations) est égale à 0.72.



Plan

Multivariate Adaptative Regression Splines

On peut voir les modèles MARS :

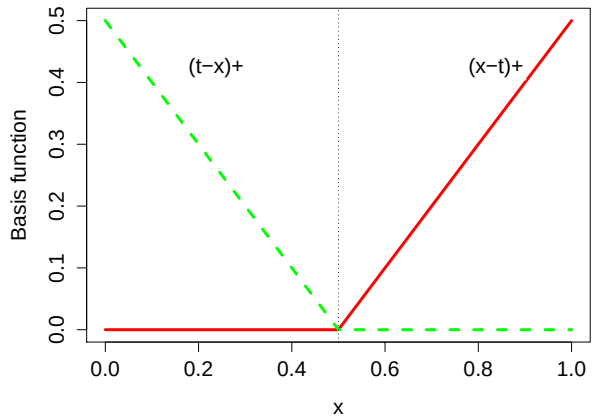
- Généralisation d'une approche de régression stepwise
- Extension de CART au cas linéaire sur chaque zone, surface finale continue

On va présenter MARS avec l'approche régression sur des fonctions de bases linéaires par morceaux.

Les fonctions de base sont de la forme :

$$(x-t)^+ = \begin{cases} x-t, si\ x > t \\ 0, otherwise \end{cases}$$

$$(t-x)^+ = \begin{cases} t-x, si\ x < t \\ 0, otherwise \end{cases}$$



De même que pour CART, on reconnait un paramètre de "seuil" (t) à estimer.
L'ensemble $\mathcal{C}$ des fonctions de bases est $\{(X_j - t)^+, (X_j - t)^+\}$, $t \in \{x_{1j}, ..., x_{nj}\}$, $j = 1, ..., p$.

MARS - le modèle

Le modèle a alors la forme suivante :

$$f(X) = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X)$$

où $h_m(X)$ est une fonction de $\mathcal{C}$, ou un produit de deux.

Les β_m sont estimés par moindres carrés.

MARS - estimation

Etape froward Les fonctions h_m sont introduites les unes après les autres dans une logique stepwise (diminution la plus importante de l'erreur)

- La première fonction ajoutée au modèle $\mathcal{M}$ est $h_0(X) = 1$
- Ensuite, on ajoute au modèle $\mathcal{M}$ les fonctions de bases résultat du produit entre une paire de fonctions "réflexives" et les fonctions déjà présentes dans le modèle, i.e. $\{h_l(X)(X_j - t)^+, h_l(X)(t - X_j)^+\}$ où h_l est présente dans $\mathcal{M}$.

Attention : le modèle prend bien en compte les interactions (produit de fonctions de deux variables différentes) mais il "sur-apprend" les observations.

MARS - estimation

Etape backward : on élimine un terme du modèle qui entraine la plus faible décroissance des moindres carrés. La question qui se pose est de choisir la taille λ du modèle final $\mathcal{M}_{final}$.

λ peut être choisi par validation croisée (très coûteux) ou par un critère type AIC, critère GCV tel que :

$$GCV = \frac{\sum_{i=1}^{n}(y_i - \hat{f}_\lambda(x_i))^2}{(1 - S(\lambda)/n)^2}$$

où $S(\lambda)$ correspond à la complexité du modèle (nombre de paramètres β + 3*nombre de noeuds t)

Avantages du modèle et de la procédure d'estimation :

- Les fonctions de bases n'agissent que sur une partie du support et sont nulles en dehors (propriété "locale")
- Caractère parcimonieux du modèle final, intéressant en grande dimension (information présente uniquement où il se passe des choses).
- Les interactions sont ajoutées de façon hiérarchique. Un produit de 4 variables ne peut exister que si tous les sous produits de 3 variables, de 2 variables et les termes seuls sont déjà dans le modèle (restriction possible à 2).
- Restriction usuelle : une variable ne peut être présente qu'une seule fois dans un produit (éviter la variance des fonctions de base à degré élevé).

Exemple cas IRSN - modèle MARS

$factor

	b	e	p	r	l
[1,]	0	0	0	0	0
[2,]	0	0	0	1	0
[4,]	1	0	0	1	0
[5,]	-1	0	0	1	0
[6,]	0	0	0	1	0
[10,]	1	0	1	0	0
[13,]	0	0	0	1	-1
[15,]	0	0	-1	1	0
[16,]	1	1	0	0	0
[18,]	-1	0	1	0	0

$cuts

	[,1]	[,2]	[,3]	[,4]	[,5
[1,]	0.000000	0.000000	0.000000	0.000000	0.00000
[2,]	0.000000	0.000000	0.000000	0.273415	0.00000
[4,]	0.342326	0.000000	0.000000	0.273415	0.00000
[5,]	0.342326	0.000000	0.000000	0.273415	0.00000
[6,]	0.000000	0.000000	0.000000	0.747490	0.00000
[10,]	0.531571	0.000000	0.498032	0.000000	0.00000
[13,]	0.000000	0.000000	0.000000	0.273415	0.64671
[15,]	0.000000	0.000000	0.742637	0.273415	0.00000
[16,]	0.531571	0.276803	0.000000	0.000000	0.00000
[18,]	0.531571	0.000000	0.244667	0.000000	0.00000

10 termes sélectionnés sur les 20 termes ajoutés dans le modèle.

$R2 = 0.87$

Plan

Pour un seul arbre, le modèle peut s'écrire :
$T(x, \theta) = \sum_{m=1}^{M} \gamma_m \mathbf{1}(x \in \mathcal{R}_m)$ où $\theta = \{\mathcal{R}_m, \gamma_m\}_{m=1,\dots,M}$.

Le "boosted tree" est une somme de plusieurs de ces arbres :

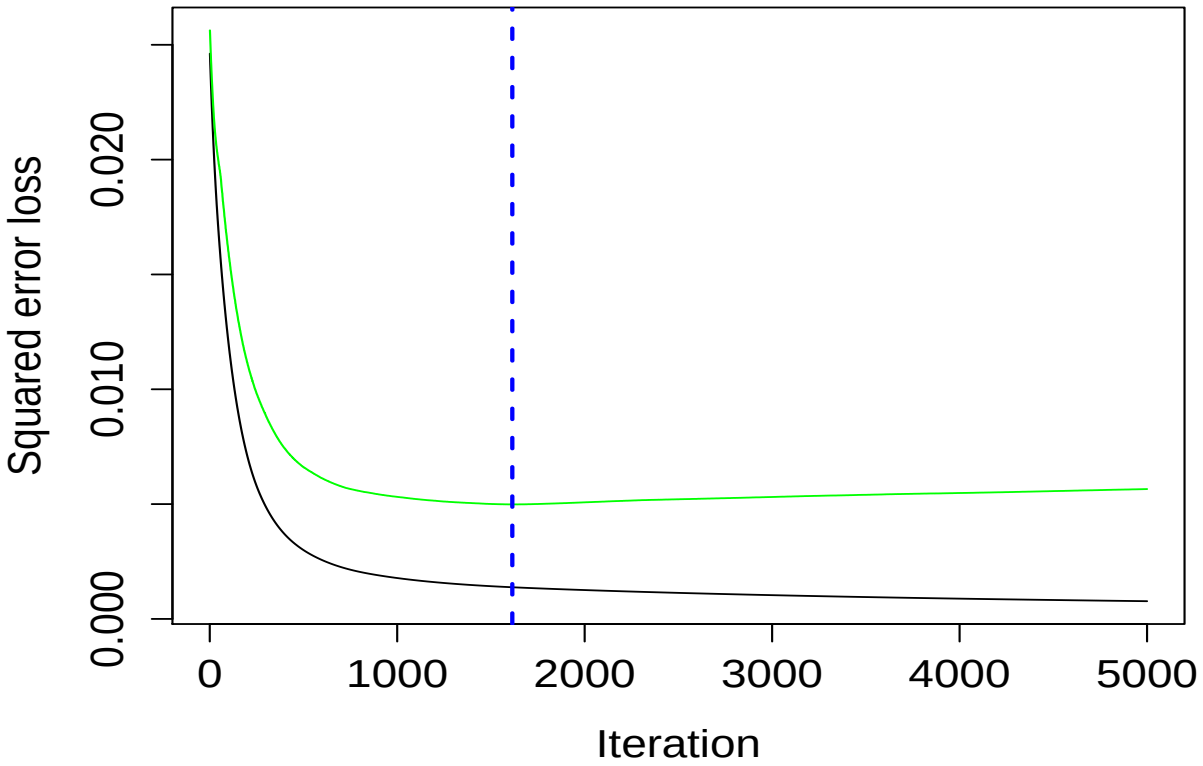
$$f_J(x) = \sum_{j=1}^{J} T(x, \theta_j)$$

introduits par une approche "forward" de façon à résoudre

$$\hat{\theta}_j = \underset{\theta_j}{argmin} \sum_{i=1}^{n} \mathcal{L}\left(y_i, f_{j-1}(x_i) + T(x_i, \theta_j)\right)$$

où la fonction $\mathcal{L}$ est la somme des carrés des résidus et où
$\theta_j = \{\mathcal{R}_{mj}, \gamma_{mj}\}_{m=1,\dots,M_j}$.
Remarque : γ_{mj} correspond à la moyenne des résidus $y_i - f_{j-1}(x_i)$ dans la région m.

Exemple IRSN

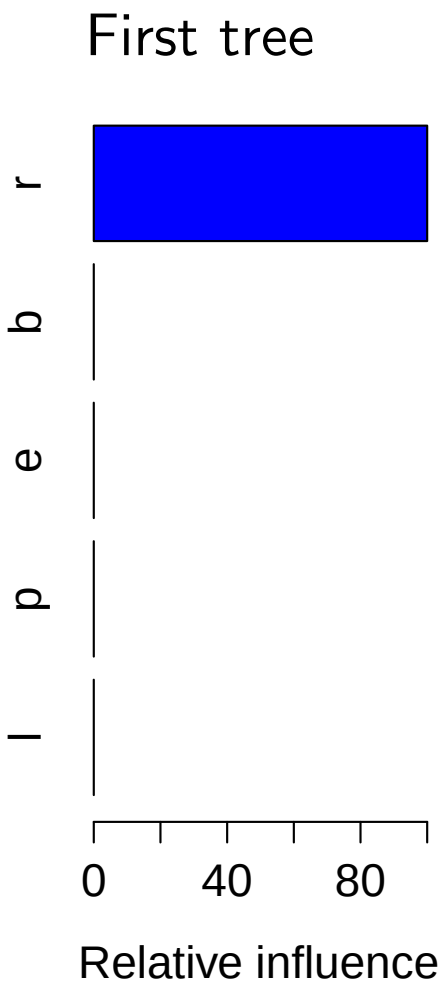


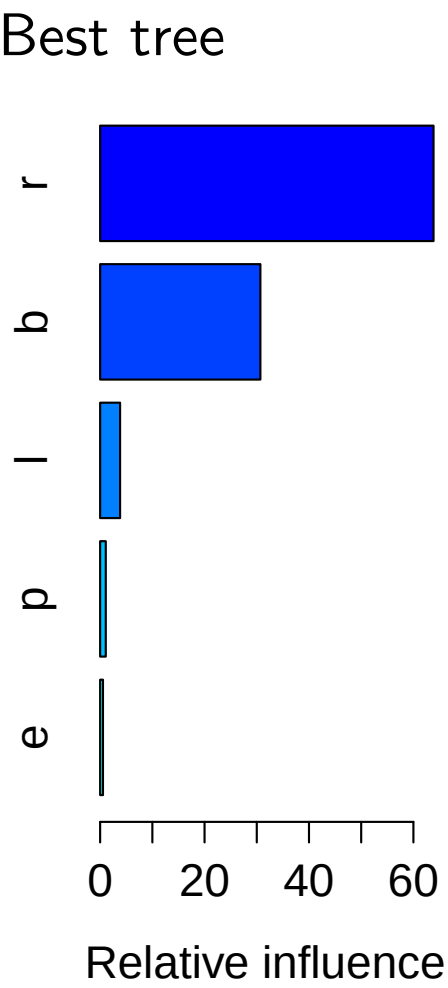
Par validation croisée, on détecte que le nombre optimal d'arbres

Importance relative

- A chaque fois qu'une variable est sélectionnée, elle fait baissée l'erreur résiduelle d'une certaine quantité
- on peut définir l'importance relative d'une variable par le nombre de fois où la variable est sélectionnée pondérée par la diminution de l'erreur entrainée quand la variable est sélectionnée et moyenné sur tous les arbres.

Exemple IRSN





Remarques

- Différentes fonctions de pertes existent pour la régression
  - $(y_i - f(x_i))^2$
  - $|y_i - f(x_i)|$
  - Huber :

$$\begin{cases} y_i - f(x_i), & si|y_i - f(x_i)| < \delta \\ \delta sign(y_i - f(x_i)) & sinon \end{cases}$$

    où $\delta =$ quantile ordre α de $|y_i - f(x_i)|$
  - Huber M-estimation :

$$\begin{cases} (y_i - f(x_i))^2, & si|y_i - f(x_i)| < \delta \\ 2\delta|y_i - f(x_i)| - \delta^2, & sinon \end{cases}$$

- Pour le boosting, de nombreux algorithmes ont été développés ces dernières années pour améliorer la qualité prédictive des modèles et la vitesse de convergence.

Principe :

- on construit différents arbres, i.e. autant de prédicteurs différents (experts) , puis le prédicteur final correspond à la moyenne des différents arbres (=> robuste)
- chaque arbre est construit à partir d'une sélection aléatoire d'observations au sein de l'ensemble d'apprentissage (tirage avec remise) (=> éviter la sensibilité aux points du plan d'expériences)

L'estimateur du **"bagging tree"** a l'expression suivante ($B \in \mathbb{N}$) :

$$\hat{f}_{bag}(x) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}^{*b}(x)$$

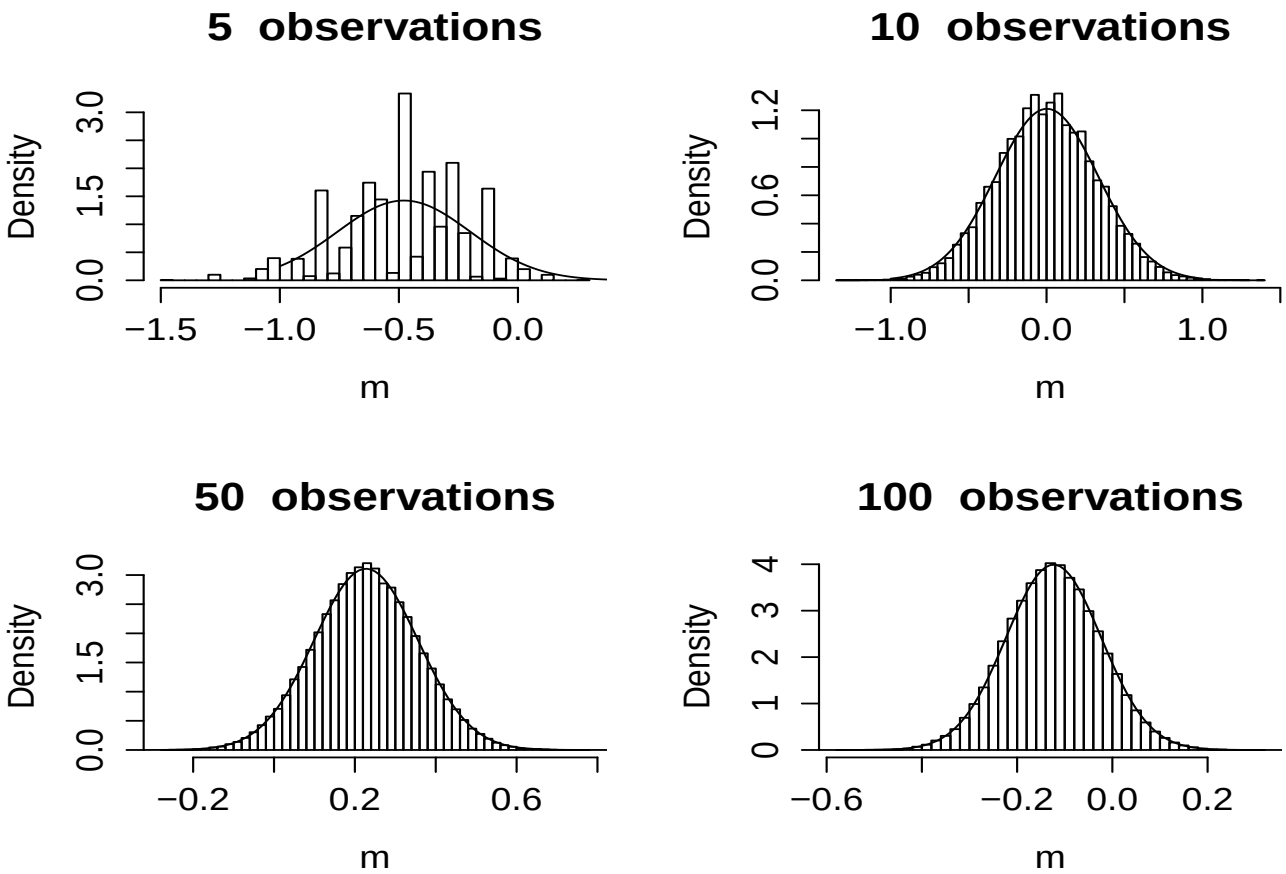
où

- $\hat{f}^{*b}$ correspond à l'arbre construit sur $\mathbf{Z}^{*b}$
- $\mathbf{Z}^{*b}, b = 1, ..., B$ sont des échantillons bootstrap de $\mathbf{Z}$, i.e. $\mathbf{Z}^* = \{(x_1^*, y_1^*), ..., (x_n^*, y_n^*)\}$ où $(x_i^*, y_i^*) \sim \hat{\mathcal{P}}$
- $\hat{\mathcal{P}}$ est la distribution empirique uniforme sur $\mathbf{Z} = \{(x_1, y_1), ..., (x_n, y_n)\}$, i.e. poids $\frac{1}{n}$ sur chaque observation.

Exemple IRSN

Le $R2$ sur l'ensemble test est égal à 0.75.

Illustration Bootstrap



100000 tirages bootstrap

Introduction

- **boosting** : prédicteur qui évolue dans le temps (somme d'arbres grossiers qui apprennent de plus en plus)
- **bagging** : technique pour réduire la variance de prédiction, adaptée à des modèles qui ont beaucoup de variance et peu de biais (arbres très développés), consiste à réaliser la moyenne d'arbres obtenus sur des échantillons bootstrap
- **random forests** : modification du bagging, moyenne d'une collection d'arbres décorrélés entre eux.

Définition des random forests

- **bagging** : moyenner des prédicteurs à haute variabilité mais non (ou peu) biaisés.
- Les arbres sont alors de très bons candidats puisqu'ils attrapent des interactions complexes entre variables et s'ils sont développés suffisamment ils sont non biaisés (cas limite = une région par observation).
- la moyenne de ces arbres a alors de bonnes propriétés : faible variance (propriété de la moyenne) + biais faible.

Définition des random forests

Considérons $T = \frac{1}{B}\sum_{b=1}^{B} X_b$ où $X_1, ..., X_b$ sont identiquement distribuées de variance σ^2 et de corrélation croisée positive ρ.

$$Var(T) = \rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$

Si $B \to +\infty$:

- le deuxième terme s'annule
- le premier terme ne s'annule pas

Particularité du cas corrélé : la variance de la moyenne ne pourra pas décroitre au dessous de $\rho\sigma^2$.

Définition des random forests

Idée des **random forests** : pour chaque échantillon bootstrap, i.e. pour chaque arbre de la forêt,

- à chaque division, on choisit aléatoirement un sous ensemble de variables $m \leq p$

Sur la collection d'arbres $\{T(x;\theta_b)\}_1^B$ ainsi construit, le prédicteur de la forêt aléatoire est :

$$\hat{f}_{rf}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T(x;\theta_b)$$

où le b^{ieme} arbre est caractérisé par θ_b qui contient l'information sur les variables dans les divisions successives, les seuils de coupure, les valeurs aux feuilles.

Out of Bag Samples

- Une notion important des forêts aléatoires est la notion d'échantillon "out of bag" : pour claque observation $z_i = (x_i, y_i)$ on construit son prédicteur "Forêt aléatoire" en faisant la moyenne de la prédiction en ce point de tous les arbres de la forêt dont l'échantillon bootstrap ne contient pas l'observation en question.
- l'erreur OOB peut alors être considérée comme une erreur de validation croisée. Sauf que l'erreur OOB est mise à jour au fur et à mesure de l'évaluation de l'algorithme. La stabilisation de cette erreur peut être un critère d'arrêt de l'algorithme.

- méthode très populaire parmi les méthodes d'apprentissage
- simple à construire et à interpréter
- dans la pratique les résultats sont très bons : RF fait mieux que le bagging et fait presque aussi bien que le boosting.

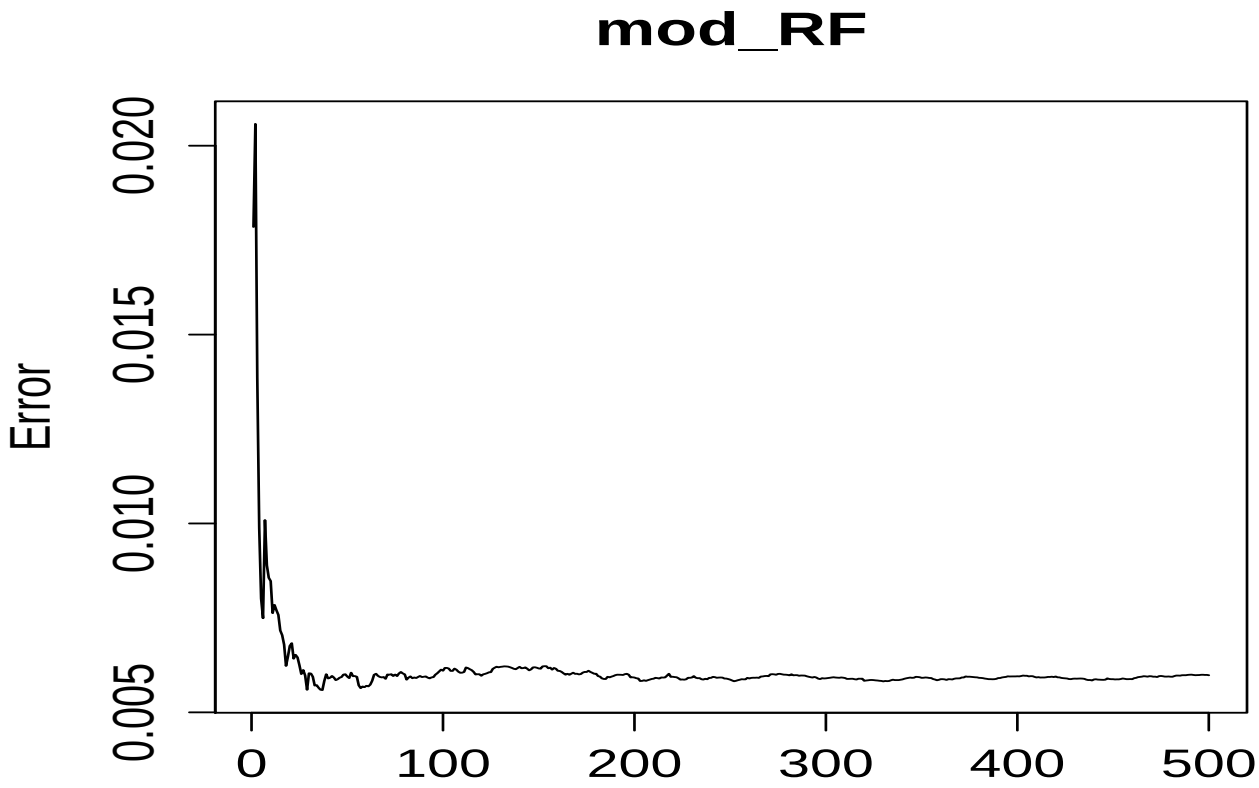
Dans la pratique le paramètre m est à choisir par validation croisée. Les valeurs par défaut sont

- pour la classification : $\lfloor \sqrt{p} \rfloor$, avec un nombre minimum d'observations aux noeuds terminaux égal à 1,
- pour la régression : $\lfloor \frac{p}{3} \rfloor$, avec un nombre minimum d'observations aux noeuds terminaux égal à 5.

Exemple IRSN

Le $R2$ sur l'ensemble test est égal à 0.74 (moins bien que le bagging mais il s'agit d'un exemple avec très peu de variables et très peu d'observations).

```
> mod_RF <- randomForest(keff~., data=dataIRSN5D,ntree = 500, mtry =4,sampsize = 50,nodesize =2)
```



Plan

Fonction de pertes en classification

Contexte : réponse binaire $\{0, 1\}$. Pour les modèles à base d'arbres, la règle de classement est la suivante, pour une région donnée $\mathcal{R}_m$: si $\hat{p}_m > 0.5$ (où $\hat{p}_m = \frac{1}{N_m} \sum_{x_i \in \mathcal{R}_m} \mathbf{1}(y_i = 1)$) alors $\hat{y}(x) = 1$ sinon $\hat{y}(x) = 0$.
Les différentes fonctions de perte en classification sont les suivantes :

- **erreur de classement** :

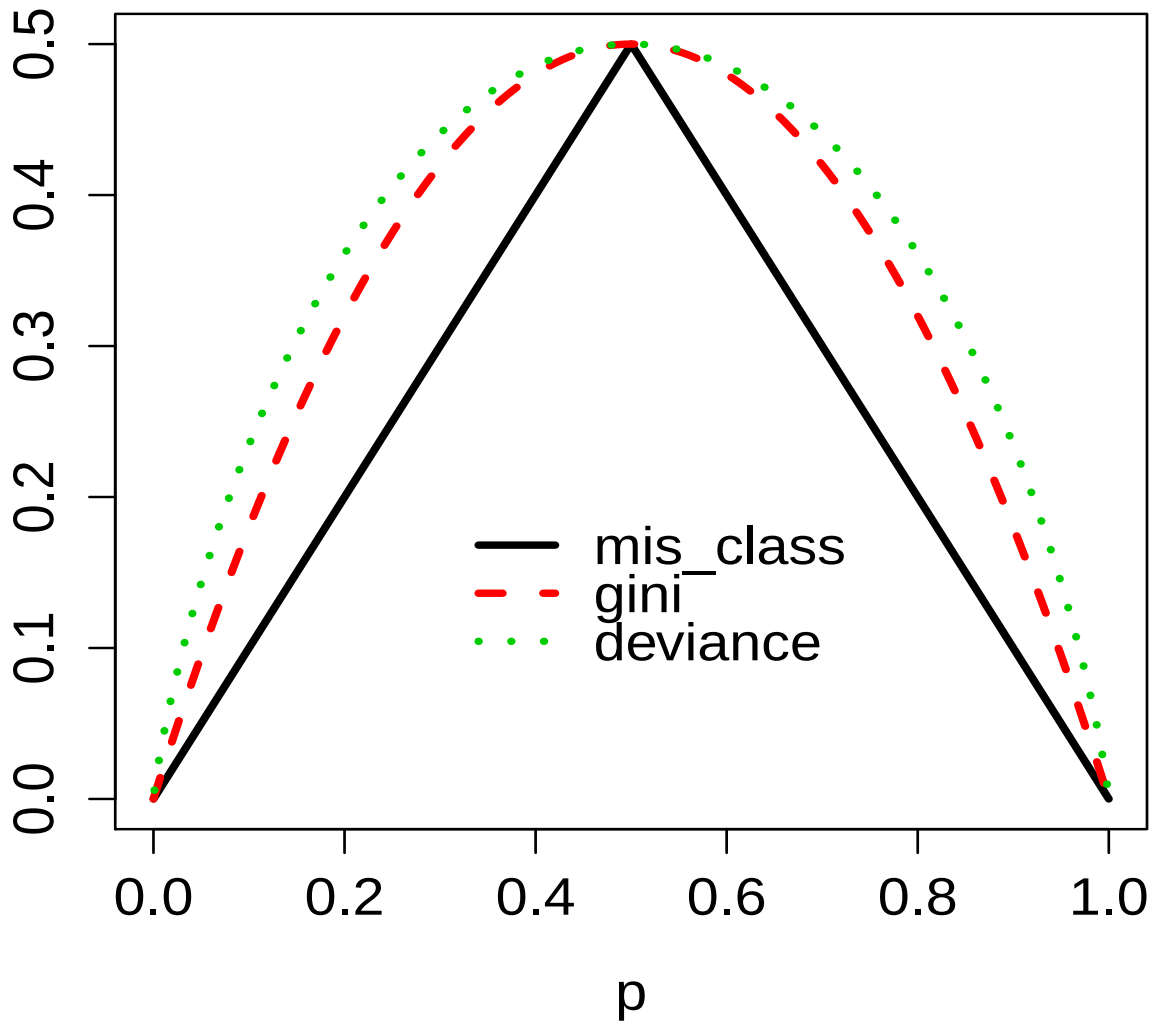
$$min(\hat{p}_m, 1 - \hat{p}_m)$$

- **indice de Gini** :

$$2\hat{p}_m(1 - \hat{p}_m)$$

- **Cross-entropy ou deviance** :

$$-(\hat{p}_m log(\hat{p}_m) + (1 - \hat{p}_m) log(1 - \hat{p}_m))$$


0.0
0.1
0.2
0.3
0.4
0.5
0.0
0.2
0.4
0.6
0.8
1.0
p
mis_class
gini
deviance

- Algorithmes de boosting très performants en classification => poids des observations mal classées qui évoluent en fonction du temps
- Autres méthodes très développées en classification aujourd'hui : SVM ("Support Vector Machine"), SVM à noyaux (extensions des méthodes de classifications à frontières linéaires)