

Introduction au machine learning

L. Rouvière

laurent.rouviere@univ-rennes2.fr

Juin 2021

Présentation

- **Objectifs** : comprendre les aspects **théoriques** et **pratiques** des algorithmes machine learning de référence.
- **Pré-requis** : théorie des probabilités, modélisation statistique, régression (linéaire et logistique). R, niveau avancé.
- **Enseignant** : Laurent Rouvière laurent.rouviere@univ-rennes2.fr
 - **Recherche** : statistique non paramétrique, apprentissage statistique
 - **Enseignements** : statistique et probabilités (Université, école d'ingénieur et de commerce, formation continue).
 - **Consulting** : énergie, finance, marketing, sport.

- **Matériel** :
 - slides : https://lrouviere.github.io/machine_learning/
 - Tutoriel long : https://lrouviere.github.io/TUTO_ML/
 - Tutoriel court : https://lrouviere.github.io/machine_learning/tuto_court_intro_ml.html
- **3 parties** :
 1. **Machine Learning** : cadre, objectif, risque...
 2. **Algorithmes linéaires** : MCO, régularisation (ridge, lasso)
 3. **Algorithmes non linéaires** : arbres et forêts aléatoires

Objectifs/questions

- **Buzzword** : machine learning, big data, data mining, intelligence artificielle...
- **Machine learning** versus **statistique** (traditionnelle)
- **Risque** \implies calcul ou estimation : ré-échantillonnage, validation croisée...
- Algorithmes versus estimateurs...
- **Classification** des algorithmes. Tous équivalents? Cadre propice...
- ...

Première partie I

Machine learning

Motivations

Quelques exemples

Cadre statistique pour l'apprentissage supervisé

Estimation du risque

Annexe 1 : le package tidymodels

Annexe 2 : le package caret

Bibliographie

Motivations

Quelques exemples

Cadre statistique pour l'apprentissage supervisé

Estimation du risque

Annexe 1 : le package tidymodels

Annexe 2 : le package caret

Bibliographie

Apprentissage statistique ?

Plusieurs "définitions"

1. "... explores way of **estimating functional dependency** from a given collection of data" [Vapnik, 2000].
2. "...vast set of tools for modelling and understanding **complex data**" [James et al., 2015]

Apprentissage statistique ?

Plusieurs "définitions"

1. "... explores way of **estimating functional dependency** from a given collection of data" [Vapnik, 2000].
2. "...vast set of tools for modelling and understanding **complex data**" [James et al., 2015]

Wikipedia

L'**apprentissage automatique** (en anglais : machine learning), **apprentissage artificiel** ou **apprentissage statistique** est un champ d'étude de l'**intelligence artificielle** qui se fonde sur des approches **mathématiques et statistiques** pour donner aux **ordinateurs** la capacité d'apprendre à partir de donnée...

Apprentissage statistique ?

Plusieurs "définitions"

1. "... explores way of **estimating functional dependency** from a given collection of data" [Vapnik, 2000].
2. "...vast set of tools for modelling and understanding **complex data**" [James et al., 2015]

Wikipedia

L'**apprentissage automatique** (en anglais : machine learning), **apprentissage artificiel** ou **apprentissage statistique** est un champ d'étude de l'**intelligence artificielle** qui se fonde sur des approches **mathématiques et statistiques** pour donner aux **ordinateurs** la capacité d'apprendre à partir de donnée...

⇒ **Interface** : Mathématiques-statistique/informatique.

Constat

- Le **développement des moyens informatiques** fait que l'on est confronté à des données de plus en plus **complexes**.
- Les méthodes **traditionnelles** se révèlent souvent **peu efficaces** face à ce type de données.
- Nécessité de proposer des **algorithmes/modèles statistiques** qui apprennent directement à partir des données.

Un peu d'histoire - voir [Besse, 2018]

Période	Mémoire	Ordre de grandeur
1940-70	Octet	$n = 30, p \leq 10$
1970	kO	$n = 500, p \leq 10$
1980	MO	Machine Learning
1990	GO	Data-Mining
2000	TO	$p > n$, apprentissage statistique
2010	PO	n explose, cloud, cluster...
2013	serveurs	Big data
2017	??	Intelligence artificielle...

Un peu d'histoire - voir [Besse, 2018]

Période	Mémoire	Ordre de grandeur
1940-70	Octet	$n = 30, p \leq 10$
1970	kO	$n = 500, p \leq 10$
1980	MO	Machine Learning
1990	GO	Data-Mining
2000	TO	$p > n$, apprentissage statistique
2010	PO	n explose, cloud, cluster...
2013	serveurs	Big data
2017	??	Intelligence artificielle...

Conclusion

Capacités informatiques \implies Data Mining \implies Apprentissage statistique
 \implies Big Data \implies Intelligence artificielle...

Objectif \implies expliquer

- notion de **modèle** ;
- retrouver des lois de probabilités ;
- décisions prises à l'aide de **tests statistiques, intervalles de confiance**.

Approche statistique

Objectif \implies expliquer

- notion de **modèle** ;
- retrouver des lois de probabilités ;
- décisions prises à l'aide de **tests statistiques, intervalles de confiance**.

Exemples

- Tests indépendance/adéquation...
- Modèle linéaire : estimation, sélection de variables, analyse des résidus...
- Régression logistique...
- Séries temporelles...

Objectif \implies prédire

- notion d'**algorithmes de prévision** ;
- critères d'**erreur de prévision** ;
- **calibration** de paramètres (tuning).

Objectif \implies prédire

- notion d'**algorithmes de prévision** ;
- critères d'**erreur de prévision** ;
- **calibration** de paramètres (tuning).

Exemples

- Algorithmes linéaires (moindres carrés, régularisation, "SVM") ;
- Arbres, réseaux de neurones ;
- **Agrégation** : boosting, bagging (forêts aléatoires) ;
- Deep learning (apprentissage profond).

Statistique vs apprentissage

- Les objectifs **diffèrent** :
 - recherche de **complexité minimale** en statistique \implies le modèle doit être **interprétable** !
 - **complexité moins importante** en machine learning \implies on veut "juste bien prédire".

Statistique vs apprentissage

- Les objectifs **diffèrent** :
 - recherche de **complexité minimale** en statistique \implies le modèle doit être **interprétable** !
 - **complexité moins importante** en machine learning \implies on veut "juste bien prédire".
- Approches néanmoins **complémentaires** :
 - bien expliquer \implies bien prédire ;
 - "récentes" évolutions d'aide à l'**interprétation des algorithmes ML** \implies **scores d'importance** des variables...
 - un bon algorithme doit posséder des **bonnes propriétés statistiques** (convergence, biais, variance...).

Statistique vs apprentissage

- Les objectifs **diffèrent** :
 - recherche de **complexité minimale** en statistique \implies le modèle doit être **interprétable** !
 - **complexité moins importante** en machine learning \implies on veut "juste bien prédire".
- Approches néanmoins **complémentaires** :
 - bien expliquer \implies bien prédire ;
 - "récentes" évolutions d'aide à l'**interprétation des algorithmes ML** \implies **scores d'importance** des variables...
 - un bon algorithme doit posséder des **bonnes propriétés statistiques** (convergence, biais, variance...).

Conclusion

Ne **pas dissocier** les deux approches.

Problématiques associées à l'apprentissage

- **Apprentissage supervisé** : prédire une sortie $y \in \mathcal{Y}$ à partir d'entrées $x \in \mathcal{X}$;
- **Apprentissage non supervisé** : établir une typologie des observations ;
- **Règles d'association** : identifier des liens entre différents produits ;
- **Systemes de recommandation** : identifier les produits susceptibles d'intéresser des consommateurs.

Problématiques associées à l'apprentissage

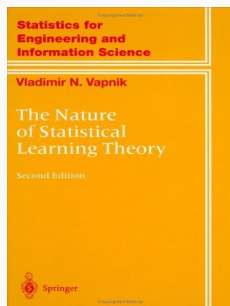
- **Apprentissage supervisé** : prédire une sortie $y \in \mathcal{Y}$ à partir d'entrées $x \in \mathcal{X}$;
- **Apprentissage non supervisé** : établir une typologie des observations ;
- **Règles d'association** : identifier des liens entre différents produits ;
- **Systemes de recommandation** : identifier les produits susceptibles d'intéresser des consommateurs.

Nombreuses applications

finance, économie, marketing, biologie, médecine...

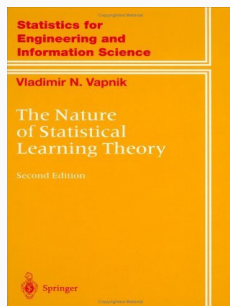
Approche mathématique

- Ouvrage fondateur : [Vapnik, 2000]

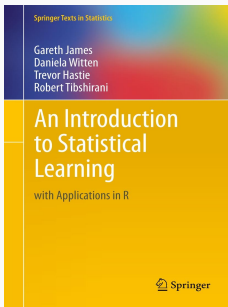
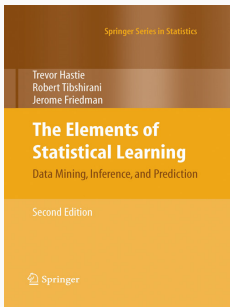


Approche mathématique

- Ouvrage fondateur : [Vapnik, 2000]
- voir aussi [Bousquet et al., 2003].



The Elements of Statistical Learning [[Hastie et al., 2009](#), [James et al., 2015](#)]



- Disponibles (avec jeux de données, codes...) aux url :

<https://web.stanford.edu/~hastie/ElemStatLearn/>

<http://www-bcf.usc.edu/~gareth/ISL/>

- Page de cours et tutoriels très bien faits sur la **statistique classique et moderne**.
- On pourra notamment regarder les **vignettes** sur la partie **apprentissage** :
 - [Wikistat, 2020a]
 - [Wikistat, 2020b]
 - ...

- Page de cours et tutoriels très bien faits sur la **statistique classique et moderne**.
- On pourra notamment regarder les **vignettes** sur la partie **apprentissage** :
 - [Wikistat, 2020a]
 - [Wikistat, 2020b]
 - ...
- Plusieurs parties de ce cours sont **inspirées de ces vignettes**.

Motivations

Quelques exemples

Cadre statistique pour l'apprentissage supervisé

Estimation du risque

Annexe 1 : le package tidymodels

Annexe 2 : le package caret

Bibliographie

Apprentissage statistique

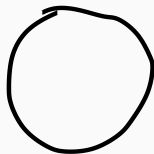
Comprendre et apprendre un comportement à partir d'exemples.

0	0	0	0	0
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4
5	5	5	5	5
6	6	6	6	6
7	7	7	7	7
8	8	8	8	8
9	9	9	9	9

Apprentissage statistique

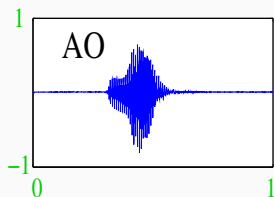
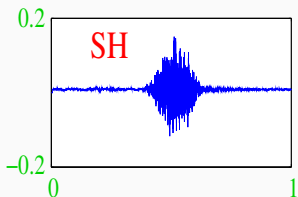
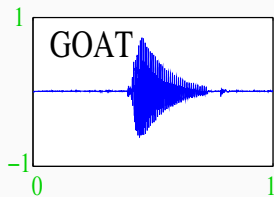
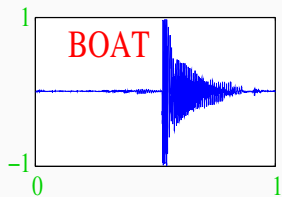
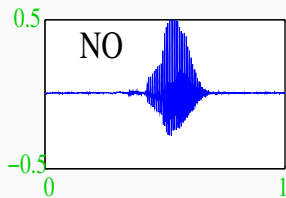
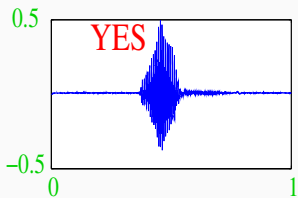
Comprendre et apprendre un comportement à partir d'exemples.

0	0	0	0	0
1	1	1	1	1
2	2	2	2	2
3	3	3	3	3
4	4	4	4	4
5	5	5	5	5
6	6	6	6	6
7	7	7	7	7
8	8	8	8	8
9	9	9	9	9

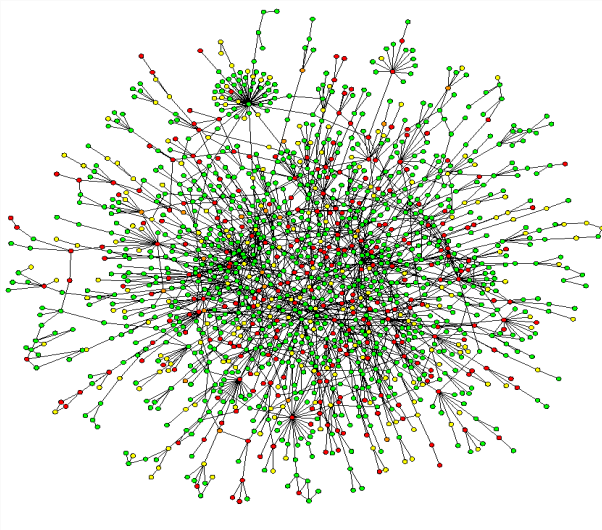


Qu'est-ce qui est écrit ? 0, 1, 2... ?

Reconnaissance de la parole



Apprentissage sur les réseaux



Prévision de pics d'ozone

- On a mesuré pendant 366 jours la **concentration maximale** en ozone (V4);
- On dispose également d'autres variables météorologiques (température, nébulosité, vent...).

```
> head(Ozone)
```

```
##      V1 V2 V3 V4   V5 V6 V7 V8   V9  V10 V11   V12 V13
## 1    1  1  4  3 5480  8 20 NA   NA 5000 -15 30.56 200
## 2    1  2  5  3 5660  6 NA 38   NA   NA -14   NA 300
## 3    1  3  6  3 5710  4 28 40   NA 2693 -25 47.66 250
## 4    1  4  7  5 5700  3 37 45   NA  590 -24 55.04 100
## 5    1  5  1  5 5760  3 51 54 45.32 1450  25 57.02  60
## 6    1  6  2  6 5720  4 69 35 49.64 1568  15 53.78  60
```

Prévision de pics d'ozone

- On a mesuré pendant 366 jours la **concentration maximale** en ozone (V4);
- On dispose également d'autres variables météorologiques (température, nébulosité, vent...).

```
> head(Ozone)
##   V1 V2 V3 V4   V5 V6 V7 V8   V9  V10 V11   V12 V13
## 1  1  1  4  3 5480  8 20 NA   NA 5000 -15 30.56 200
## 2  1  2  5  3 5660  6 NA 38   NA  NA -14   NA 300
## 3  1  3  6  3 5710  4 28 40   NA 2693 -25 47.66 250
## 4  1  4  7  5 5700  3 37 45   NA  590 -24 55.04 100
## 5  1  5  1  5 5760  3 51 54 45.32 1450  25 57.02  60
## 6  1  6  2  6 5720  4 69 35 49.64 1568  15 53.78  60
```

Question

Peut-on **prédire** la concentration maximale en ozone du **lendemain** à partir des prévisions météorologiques ?

Détection de spam

- Sur 4 601 mails, on a pu identifier **1813 spams**.
- On a également mesuré sur chacun de ces mails la présence ou absence de **57 mots**.

```
> spam %>% select(c(1:8,58)) %>% head()
##   make address  all num3d  our over remove internet type
## 1 0.00   0.64 0.64    0 0.32 0.00   0.00    0.00 spam
## 2 0.21   0.28 0.50    0 0.14 0.28   0.21    0.07 spam
## 3 0.06   0.00 0.71    0 1.23 0.19   0.19    0.12 spam
## 4 0.00   0.00 0.00    0 0.63 0.00   0.31    0.63 spam
## 5 0.00   0.00 0.00    0 0.63 0.00   0.31    0.63 spam
## 6 0.00   0.00 0.00    0 1.85 0.00   0.00    1.85 spam
```

Détection de spam

- Sur 4 601 mails, on a pu identifier **1813 spams**.
- On a également mesuré sur chacun de ces mails la présence ou absence de **57 mots**.

```
> spam %>% select(c(1:8,58)) %>% head()
##   make address  all num3d  our over remove internet type
## 1 0.00    0.64 0.64    0 0.32 0.00    0.00    0.00 spam
## 2 0.21    0.28 0.50    0 0.14 0.28    0.21    0.07 spam
## 3 0.06    0.00 0.71    0 1.23 0.19    0.19    0.12 spam
## 4 0.00    0.00 0.00    0 0.63 0.00    0.31    0.63 spam
## 5 0.00    0.00 0.00    0 0.63 0.00    0.31    0.63 spam
## 6 0.00    0.00 0.00    0 1.85 0.00    0.00    1.85 spam
```

Question

Peut-on construire à partir de ces données une méthode de **détection automatique** de spam ?

Motivations

Quelques exemples

Cadre statistique pour l'apprentissage supervisé

Estimation du risque

Annexe 1 : le package tidymodels

Annexe 2 : le package caret

Bibliographie

Régression vs classification

- **Données** de type **entrée-sortie** : $d_n = (x_1, y_1), \dots, (x_n, y_n)$ où $x_i \in \mathcal{X}$ représente l'entrée et $y_i \in \mathcal{Y}$ la sortie.

Objectifs

1. **Expliquer** le(s) mécanisme(s) liant les entrée x_i aux sorties y_i ;
2. **Prédire** « au mieux » la sortie y associée à une nouvelle entrée $x \in \mathcal{X}$.

Régression vs classification

- **Données** de type **entrée-sortie** : $d_n = (x_1, y_1), \dots, (x_n, y_n)$ où $x_i \in \mathcal{X}$ représente l'entrée et $y_i \in \mathcal{Y}$ la sortie.

Objectifs

1. **Expliquer** le(s) mécanisme(s) liant les entrée x_i aux sorties y_i ;
2. **Prédire** « au mieux » la sortie y associée à une nouvelle entrée $x \in \mathcal{X}$.

Vocabulaire

- Lorsque la variable à expliquer est quantitative ($\mathcal{Y} \subseteq \mathbb{R}$), on parle de **régression**.
- Lorsqu'elle est qualitative ($\text{Card}(\mathcal{Y})$ fini), on parle de **classification (supervisée)**.

Exemples

- La plupart des problèmes présentés précédemment peuvent être appréhendés dans un contexte d'**apprentissage supervisé** : on cherche à expliquer une sortie y par des entrées x :

y_i	x_i	
Chiffre	image	Classification
Mot	courbe	Classification
Spam	présence/absence de mots	Classification
C. en O_3	données météo.	Régression

Exemples

- La plupart des problèmes présentés précédemment peuvent être appréhendés dans un contexte d'**apprentissage supervisé** : on cherche à expliquer une sortie y par des entrées x :

y_i	x_i	
Chiffre	image	Classification
Mot	courbe	Classification
Spam	présence/absence de mots	Classification
C. en O_3	données météo.	Régression

Remarque

La nature des variables associées aux **entrées** x_i est **variée** (quanti, quali, fonctionnelle...).

Un début de formalisation mathématique

- Etant données des observations $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ on cherche à **expliquer/prédire** les sorties $y_i \in \mathcal{Y}$ à partir des entrées $x_i \in \mathcal{X}$.

Un début de formalisation mathématique

- Etant données des observations $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ on cherche à **expliquer/prédire** les sorties $y_i \in \mathcal{Y}$ à partir des entrées $x_i \in \mathcal{X}$.
- Il s'agit donc de trouver **une fonction de prévision** $f : \mathcal{X} \rightarrow \mathcal{Y}$ telle que

$$f(x_i) \approx y_i, i = 1, \dots, n.$$

Un début de formalisation mathématique

- Etant données des observations $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ on cherche à **expliquer/prédire** les sorties $y_i \in \mathcal{Y}$ à partir des entrées $x_i \in \mathcal{X}$.
- Il s'agit donc de trouver **une fonction de prévision** $f : \mathcal{X} \rightarrow \mathcal{Y}$ telle que

$$f(x_i) \approx y_i, i = 1, \dots, n.$$

- Nécessité de se donner un **critère** qui permette de mesurer la qualité des fonctions de prévision f .

Un début de formalisation mathématique

- Etant données des observations $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ on cherche à **expliquer/prédire** les sorties $y_i \in \mathcal{Y}$ à partir des entrées $x_i \in \mathcal{X}$.
- Il s'agit donc de trouver **une fonction de prévision** $f : \mathcal{X} \rightarrow \mathcal{Y}$ telle que

$$f(x_i) \approx y_i, i = 1, \dots, n.$$

- Nécessité de se donner un **critère** qui permette de mesurer la qualité des fonctions de prévision f .
- Le plus souvent, on utilise une **fonction de perte** $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+$ telle que

$$\begin{cases} \ell(y, y') = 0 & \text{si } y = y' \\ \ell(y, y') > 0 & \text{si } y \neq y'. \end{cases}$$

- Données $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ i.i.d. de loi inconnue P .

- **Données** $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ i.i.d. de **loi inconnue** P .
- **Prédicteur** : une fonction $f : \mathcal{X} \rightarrow \mathcal{Y}$.

- **Données** $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ i.i.d. de **loi inconnue** P .
- **Prédicteur** : une fonction $f : \mathcal{X} \rightarrow \mathcal{Y}$.
- **Coût** : $\ell(Y, f(X)) \implies$ ensemble des erreurs de prévision.

- **Données** $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ i.i.d. de **loi inconnue** P .
- **Prédicteur** : une fonction $f : \mathcal{X} \rightarrow \mathcal{Y}$.
- **Coût** : $\ell(Y, f(X)) \implies$ ensemble des erreurs de prévision.
- **Risque** : $\mathcal{R}(f) = E[\ell(Y, f(X))] \implies$ coût "moyen".

- **Données** $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ i.i.d. de **loi inconnue** P .
- **Prédicteur** : une fonction $f : \mathcal{X} \rightarrow \mathcal{Y}$.
- **Coût** : $\ell(Y, f(X)) \implies$ ensemble des erreurs de prévision.
- **Risque** : $\mathcal{R}(f) = E[\ell(Y, f(X))] \implies$ coût "moyen".

Oracle

f^* qui vérifie $\mathcal{R}(f^*) \leq \mathcal{R}(f)$ pour tout $f : \mathcal{X} \rightarrow \mathcal{Y}$

Approche statistique

- **Données** $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ i.i.d. de **loi inconnue** P .
- **Prédicteur** : une fonction $f : \mathcal{X} \rightarrow \mathcal{Y}$.
- **Coût** : $\ell(Y, f(X)) \implies$ ensemble des erreurs de prévision.
- **Risque** : $\mathcal{R}(f) = E[\ell(Y, f(X))] \implies$ coût "moyen".

Oracle

f^* qui vérifie $\mathcal{R}(f^*) \leq \mathcal{R}(f)$ pour tout $f : \mathcal{X} \rightarrow \mathcal{Y} \implies$ dépend de P , donc **inconnu**.

Approche statistique

- **Données** $d_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ i.i.d. de **loi inconnue** P .
- **Prédicteur** : une fonction $f : \mathcal{X} \rightarrow \mathcal{Y}$.
- **Coût** : $\ell(Y, f(X)) \implies$ ensemble des erreurs de prévision.
- **Risque** : $\mathcal{R}(f) = E[\ell(Y, f(X))] \implies$ coût "moyen".

Oracle

f^* qui vérifie $\mathcal{R}(f^*) \leq \mathcal{R}(f)$ pour tout $f : \mathcal{X} \rightarrow \mathcal{Y} \implies$ dépend de P , donc **inconnu**.

Objectif

Construire un **algorithme de prévision** $f_n(\cdot) = f_n(\cdot, d_n)$ tel que $\mathcal{R}(f_n) \approx \mathcal{R}(f^*)$.

Choix de la fonction de perte

- Le cadre mathématique développé précédemment sous-entend qu'une fonction est **performante** (voire **optimale**) vis-à-vis d'un **critère** (représenté par la **fonction de perte ℓ**).
- Un algorithme de prévision performant pour un critère ne sera **pas forcément performant pour un autre**.

Choix de la fonction de perte

- Le cadre mathématique développé précédemment sous-entend qu'une fonction est **performante** (voire **optimale**) vis-à-vis d'un **critère** (représenté par la **fonction de perte** ℓ).
- Un algorithme de prévision performant pour un critère ne sera **pas forcément performant pour un autre**.

Conséquence pratique

Avant de s'attacher à construire un algorithme de prévision, il est **capital** de savoir **mesurer la performance** d'un algorithme de prévision.

Régression versus classification

Régression $\implies \mathcal{Y} = \mathbb{R}$

- **Perte** : $\ell(y, y') = (y - y')^2$;
- **Risque** : $\mathcal{R}(m) = E[(Y - m(X))^2]$.
- **Champion** : $m^*(x) = E[Y|X = x]$ (fonction de régression).

Régression versus classification

Régression $\implies \mathcal{Y} = \mathbb{R}$

- **Perte** : $\ell(y, y') = (y - y')^2$;
- **Risque** : $\mathcal{R}(m) = E[(Y - m(X))^2]$.
- **Champion** : $m^*(x) = E[Y|X = x]$ (fonction de régression).

Classification $\implies \mathcal{Y} = \{1, \dots, K\}$

- **Perte** : $\ell(y, y') = 1_{y \neq y'}$;
- **Risque** : $\mathcal{R}(g) = P(g(X) \neq Y)$.
- **Champion** : $g^*(x) = \operatorname{argmax}_k P(Y = k|X = x)$ (règle de Bayes).

1. Restreindre la classe des candidats à \mathcal{F} (modèle);

Démarche

1. Restreindre la classe des candidats à \mathcal{F} (modèle);
2. Choisir (à partir des données) le meilleur candidat dans $\mathcal{F} \implies f_n$.

Démarche

1. Restreindre la classe des candidats à \mathcal{F} (modèle);
2. Choisir (à partir des données) le meilleur candidat dans $\mathcal{F} \implies f_n$.

Deux types d'erreur

$$\mathcal{R}(f_n) - \mathcal{R}(f^*) = \mathcal{R}(f_n) - \inf_{f \in \mathcal{F}} \mathcal{R}(f) + \inf_{f \in \mathcal{F}} \mathcal{R}(f) - \mathcal{R}(f^*).$$

- Erreur d'approximation ou terme de biais.
- Erreur d'estimation ou terme de variance.

\implies ces deux termes varient en sens inverse et dépendent de la complexité du modèle.

Complexité \implies compromis biais/variance

- \mathcal{F} de complexité faible \implies modèle peu flexible \implies mauvaise adéquation sur les données \implies biais \nearrow , variance \searrow .

Complexité \implies compromis biais/variance

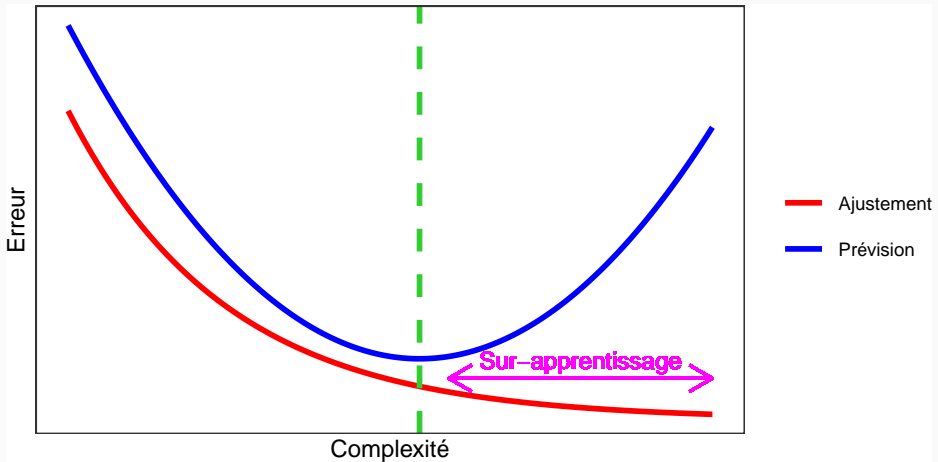
- \mathcal{F} de complexité faible \implies modèle peu flexible \implies mauvaise adéquation sur les données \implies biais \nearrow , variance \searrow .
- \mathcal{F} de complexité élevée \implies modèle trop flexible \implies sur-ajustement \implies biais \searrow , variance \nearrow .

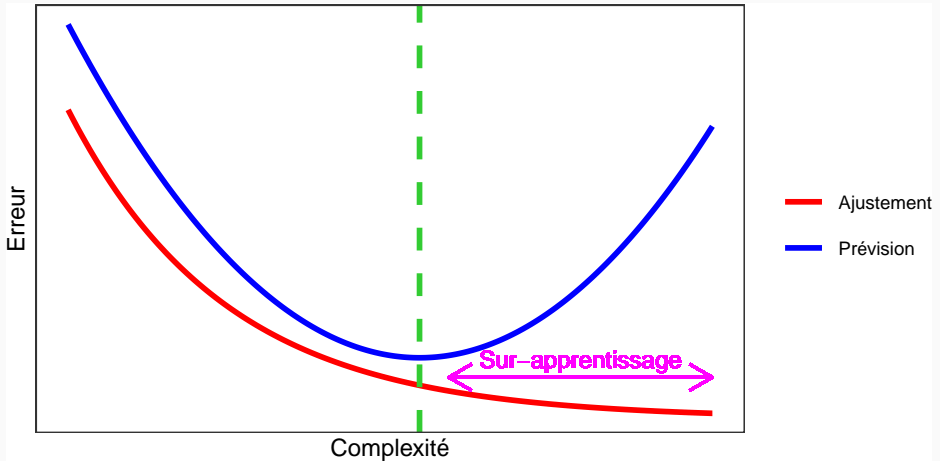
Complexité \implies compromis biais/variance

- \mathcal{F} de complexité faible \implies modèle peu flexible \implies mauvaise adéquation sur les données \implies biais \nearrow , variance \searrow .
- \mathcal{F} de complexité élevée \implies modèle trop flexible \implies sur-ajustement \implies biais \searrow , variance \nearrow .

Overfitting

Sur-ajuster signifie que le modèle va (trop) bien ajuster les données d'apprentissage, il aura du mal à s'adapter à de nouveaux individus.





Conclusion

Nécessaire de savoir **calculer** (ou plutôt **estimer**) le **risque de prévision**.

Motivations

Quelques exemples

Cadre statistique pour l'apprentissage supervisé

Estimation du risque

Annexe 1 : le package tidymodels

Annexe 2 : le package caret

Bibliographie

Rappels

- n observations $(X_1, Y_1), \dots, (X_n, Y_n)$ i.i.d à valeurs dans $\mathcal{X} \times \mathcal{Y}$.

Objectif

Etant donnée une fonction de perte $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+$, on cherche un **algorithme de prévision** $f_n(x) = f_n(x, \mathcal{D}_n)$ qui soit "proche" de l'oracle f^* défini par

$$f^* \in \underset{f}{\operatorname{argmin}} \mathcal{R}(f)$$

où $\mathcal{R}(f) = \mathbb{E}[\ell(Y, f(X))]$.

Rappels

- n observations $(X_1, Y_1), \dots, (X_n, Y_n)$ i.i.d à valeurs dans $\mathcal{X} \times \mathcal{Y}$.

Objectif

Etant donnée une fonction de perte $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+$, on cherche un **algorithme de prévision** $f_n(x) = f_n(x, \mathcal{D}_n)$ qui soit "proche" de l'oracle f^* défini par

$$f^* \in \underset{f}{\operatorname{argmin}} \mathcal{R}(f)$$

où $\mathcal{R}(f) = \mathbb{E}[\ell(Y, f(X))]$.

Question

Etant donné un algorithme f_n , **que vaut son risque** $\mathcal{R}(f_n)$?

Risque empirique

- La loi de (X, Y) étant **inconnue** en pratique, il est **impossible de calculer** $\mathcal{R}(f_n) = E[\ell(Y, f_n(X))]$.

Risque empirique

- La loi de (X, Y) étant **inconnue** en pratique, il est **impossible de calculer** $\mathcal{R}(f_n) = E[\ell(Y, f_n(X))]$.
- **Première approche** : $\mathcal{R}(f_n)$ étant une espérance, on peut l'estimer (LGN) par sa **version empirique**

$$\mathcal{R}_n(f_n) = \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f_n(X_i)).$$

Risque empirique

- La loi de (X, Y) étant **inconnue** en pratique, il est **impossible de calculer** $\mathcal{R}(f_n) = E[\ell(Y, f_n(X))]$.
- **Première approche** : $\mathcal{R}(f_n)$ étant une espérance, on peut l'estimer (LGN) par sa **version empirique**

$$\mathcal{R}_n(f_n) = \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f_n(X_i)).$$

Problème

- L'échantillon \mathcal{D}_n a **déjà été utilisé** pour construire l'algorithme de prévision $f_n \implies$ La LGN ne peut donc s'appliquer !
- **Conséquence** : $\mathcal{R}_n(f_n)$ conduit souvent à une **sous-estimation** de $\mathcal{R}(f_n)$.

Risque empirique

- La loi de (X, Y) étant **inconnue** en pratique, il est **impossible de calculer** $\mathcal{R}(f_n) = E[\ell(Y, f_n(X))]$.
- **Première approche** : $\mathcal{R}(f_n)$ étant une espérance, on peut l'estimer (LGN) par sa **version empirique**

$$\mathcal{R}_n(f_n) = \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f_n(X_i)).$$

Problème

- L'échantillon \mathcal{D}_n a **déjà été utilisé** pour construire l'algorithme de prévision $f_n \implies$ La LGN ne peut donc s'appliquer !
- **Conséquence** : $\mathcal{R}_n(f_n)$ conduit souvent à une **sous-estimation** de $\mathcal{R}(f_n)$.

Une solution

Utiliser des méthodes de type **validation croisée** ou **bootstrap**.

Apprentissage - Validation ou Validation hold out

- Elle consiste à séparer l'échantillon \mathcal{D}_n en :
 1. un **échantillon d'apprentissage** $\mathcal{D}_{n,app}$ pour construire f_n ;
 2. un **échantillon de validation** $\mathcal{D}_{n,test}$ utilisé pour estimer le risque de f_n .

Apprentissage - Validation ou Validation hold out

- Elle consiste à séparer l'échantillon \mathcal{D}_n en :
 1. un **échantillon d'apprentissage** $\mathcal{D}_{n,app}$ pour construire f_n ;
 2. un **échantillon de validation** $\mathcal{D}_{n,test}$ utilisé pour estimer le risque de f_n .

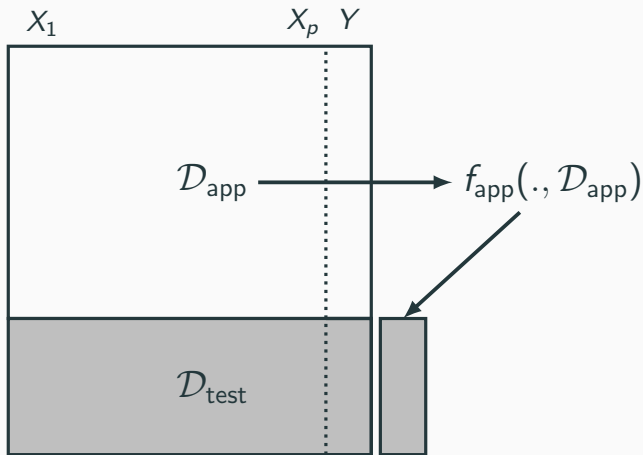
Algorithme

Entrée : $\{\mathcal{A}, \mathcal{T}\}$ une partition de $\{1, \dots, n\}$ en deux parties.

1. Ajuster l'algorithme de prévision en utilisant **uniquement les données d'apprentissage** $\mathcal{D}_{app} = \{(x_i, y_i) : i \in \mathcal{A}\}$. On désigne par $f_{app}(\cdot, \mathcal{D}_{app})$ l'algorithme obtenu.
2. Calculer les valeurs prédites $f_{app}(x_i, \mathcal{D}_{app})$ par l'algorithme pour chaque observation de l'échantillon test $\mathcal{D}_{test} = \{(x_i, y_i) : i \in \mathcal{T}\}$

Retourner :

$$\frac{1}{|\mathcal{T}|} \sum_{i \in \mathcal{T}} \ell(y_i, f_{app}(x_i, \mathcal{D}_{app})).$$



Commentaires

Nécessite d'avoir un **nombre suffisant d'observations** dans

1. \mathcal{D}_{app} pour bien ajuster l'algorithme de prévision ;
2. $\mathcal{D}_{\text{test}}$ pour bien estimer l'erreur de l'algorithme.

Validation croisée K -blocs

- **Principe** : répéter la hold out sur **différentes partitions**.

Algorithme - CV

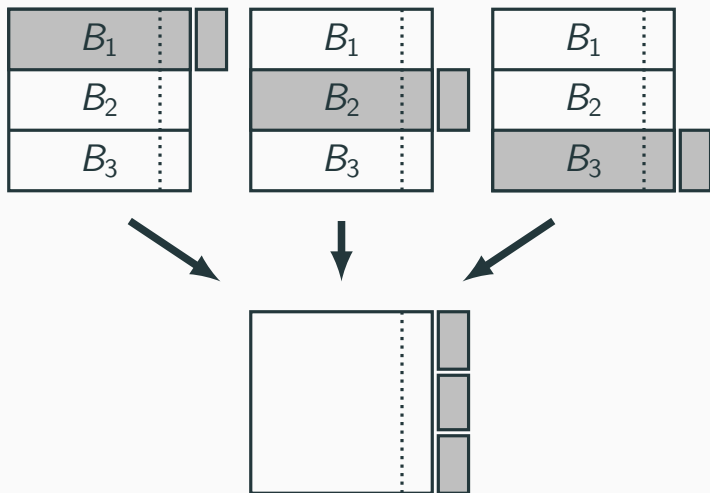
Entrée : $\{B_1, \dots, B_K\}$ une partition de $\{1, \dots, n\}$ en K blocs.

Pour $k = 1, \dots, K$:

1. Ajuster l'algorithme de prévision en utilisant **l'ensemble des données privé du k^e bloc**, c'est-à-dire $B_k = \{(x_i, y_i) : i \in \{1, \dots, n\} \setminus B_k\}$. On désigne par $f_k(\cdot) = f_k(\cdot, B_k)$ l'algorithme obtenu.
2. Calculer la valeur prédite par l'algorithme pour chaque observation du bloc k : $f_k(x_i)$, $i \in B_k$ et en déduire le **risque sur le bloc k** :

$$\widehat{\mathcal{R}}(f_k) = \frac{1}{|B_k|} \sum_{i \in B_k} \ell(y_i, f_k(x_i)).$$

Retourner : $\frac{1}{K} \sum_{k=1}^K \widehat{\mathcal{R}}(f_k)$.



Commentaires

- Le **choix de K** doit être fait par l'utilisateur (souvent $K = 10$).
- **Avantage** : plus adapté que la technique apprentissage/validation \implies **plus stable et précis**.
- **Inconvénient** : plus couteux en **temps de calcul**.

Commentaires

- Le **choix de K** doit être fait par l'utilisateur (souvent $K = 10$).
- **Avantage** : plus adapté que la technique apprentissage/validation \implies **plus stable et précis**.
- **Inconvénient** : plus couteux en **temps de calcul**.

Leave one out

- Lorsque $K = n$, on parle de validation croisée **leave one out** ;
- Le risque est alors estimé par

$$\widehat{\mathcal{R}}_n(f_n) = \frac{1}{n} \sum_{i=1}^n \ell(Y_i, f_n^i(X_i))$$

où f_n^i désigne l'algorithme de prévision construit sur \mathcal{D}_n **amputé de la i -ème observation**.

\implies recommandé uniquement lorsque **n est petit**.

- Estimation par pénalisation : critère ajustement/complexité, C_p de Mallows, AIC-BIC...
- Validation croisée Monte-Carlo : répéter plusieurs fois la validation hold out ;
- Bootstrap : notamment Out Of Bag ;
- voir [[Wikistat, 2020b](#)].

Motivations

Quelques exemples

Cadre statistique pour l'apprentissage supervisé

Estimation du risque

Annexe 1 : le package tidymodels

Annexe 2 : le package caret

Bibliographie

Présentation du package

- Successeur de `caret` pour conduire des projets machine learning sur R.
- Meta package qui inclut
 - `rsample` : pour ré-échantillonner
 - `yardstick` : pour les fonctions de perte
 - `recipe` : pour les recettes de préparation... des données
 - `tune` : pour calibrer les algorithmes
 - ...
- Tutoriel : <https://www.tidymodels.org>

Calibrer des paramètres

- Tous les algorithmes dépendent de **paramètres** θ que l'utilisateur doit sélectionner.
- Le procédé est **toujours le même** et peut se résumer dans l'algorithme suivant.

Choix de paramètres par minimisation du risque (grid search)

Entrées :

- Une grille `grille.theta` de valeurs pour θ ;
- Un risque de prévision \mathcal{R} ;
- un algorithme d'estimation du risque.

Pour chaque θ dans `grille.theta` :

- Estimer $\mathcal{R}(f_{n,\theta})$ par l'algorithme choisi $\implies \widehat{\mathcal{R}}(f_{n,\theta})$

Retourner : $\widehat{\theta}$ une valeur de θ qui minimise $\widehat{\mathcal{R}}(f_{n,\theta})$.

- Ce procédé est automatisé dans tidymodels.
- Il faut spécifier les différents paramètres :
 - la méthode (logistique, ppv, arbre, randomForest...)
 - Une grille pour les paramètres (nombre de ppv...)
 - Le critère de performance (erreur de classification, AUC, risque quadratique...)
 - La méthode d'estimation du critère (apprentissage validation, validation croisée, bootstrap...)

- Ce procédé est **automatisé** dans **tidymodels**.
- Il faut spécifier les différents paramètres :
 - la **méthode** (logistique, ppv, arbre, randomForest...)
 - Une grille pour les **paramètres** (nombre de ppv...)
 - Le **critère de performance** (erreur de classification, AUC, risque quadratique...)
 - La méthode d'**estimation du critère** (apprentissage validation, validation croisée, bootstrap...)
- Nous l'illustrons à travers le **choix du nombre de voisins** de l'algorithme des k -ppv.

- Une variable binaire à expliquer par 2 variables continues

```
> head(don.2D.500)
## # A tibble: 6 x 3
##       X1     X2 Y
##   <dbl> <dbl> <fct>
## 1 0.721 0.209 0
## 2 0.876 0.766 1
## 3 0.761 0.842 1
## 4 0.886 0.934 0
## 5 0.456 0.676 0
## 6 0.166 0.859 1
```

- On commence par renseigner l'**algorithme** et la manière dont on va **choisir les paramètres**.

```
> library(tidymodels)
> tune_spec <-
+   nearest_neighbor(neighbors=tune(),weight_func="rectangular") %>%
+   set_mode("classification") %>%
+   set_engine("kkn")
```

Le workflow

- On commence par renseigner l'**algorithme** et la manière dont on va **choisir les paramètres**.

```
> library(tidymodels)
> tune_spec <-
+   nearest_neighbor(neighbors=tune(),weight_func="rectangular") %>%
+   set_mode("classification") %>%
+   set_engine("kkn")
```

- On crée ensuite la **workflow** :

```
> ppv_wf <- workflow() %>%
+   add_model(tune_spec) %>%
+   add_formula(Y ~ .)
```

Ré-échantillonnage et grille de paramètres

- On spécifie ensuite la **méthode de ré-échantillonnage**, ici une **validation croisée 10 blocs**

```
> set.seed(12345)
> re_ech_cv <- vfold_cv(don.2D.500, v=10)
> re_ech_cv %>% head()
## # A tibble: 6 x 2
##   splits          id
##   <list>         <chr>
## 1 <split [450/50]> Fold01
## 2 <split [450/50]> Fold02
## 3 <split [450/50]> Fold03
## 4 <split [450/50]> Fold04
## 5 <split [450/50]> Fold05
## 6 <split [450/50]> Fold06
```


Ré-échantillonnage et grille de paramètres

- On spécifie ensuite la **méthode de ré-échantillonnage**, ici une **validation croisée 10 blocs**

```
> set.seed(12345)
> re_ech_cv <- vfold_cv(don.2D.500, v=10)
> re_ech_cv %>% head()
## # A tibble: 6 x 2
##   splits          id
##   <list>         <chr>
## 1 <split [450/50]> Fold01
## 2 <split [450/50]> Fold02
## 3 <split [450/50]> Fold03
## 4 <split [450/50]> Fold04
## 5 <split [450/50]> Fold05
## 6 <split [450/50]> Fold06
```

- Puis vient la **grille de paramètres**

```
> grille_k <- tibble(neighbors=1:100)
```

⇒ consulter <https://www.tidymodels.org/find/parsnip/> pour trouver les **identifiants** des algorithmes et de leurs paramètres.

Estimation du risque

- Fonction `tune_grid`

```
> tune_grid(..., resamples=..., grid=..., metrics=...)
```

Estimation du risque

- Fonction `tune_grid`

```
> tune_grid(..., resamples=..., grid=..., metrics=...)
```

- Calcul du **risque** pour chaque valeur de la grille :

```
> ppv.cv <- ppv_wf %>%  
+   tune_grid(  
+     resamples = re_ech_cv,  
+     grid = grille_k,  
+     metrics=metric_set(accuracy))
```

Estimation du risque

- Fonction `tune_grid`

```
> tune_grid(..., resamples=..., grid=..., metrics=...)
```

- Calcul du **risque** pour chaque valeur de la grille :

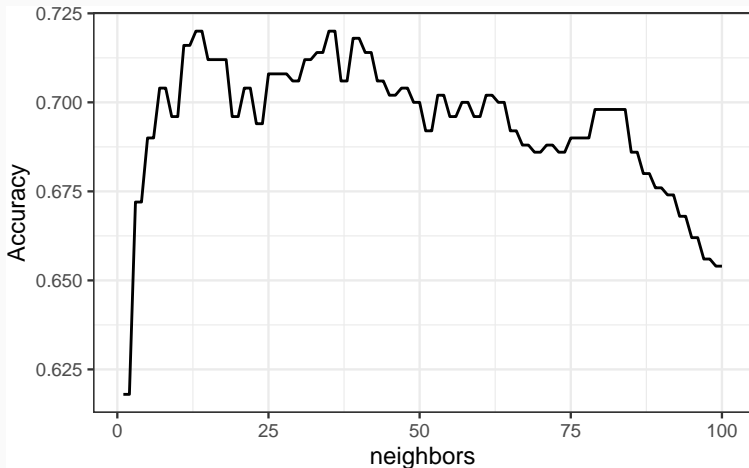
```
> ppv.cv <- ppv_wf %>%  
+   tune_grid(  
+     resamples = re_ech_cv,  
+     grid = grille_k,  
+     metrics=metric_set(accuracy))
```

- On lit les résultats avec `collect_metrics` :

```
> ppv.cv %>% collect_metrics() %>% select(1:5) %>% head()  
## # A tibble: 6 x 5  
##   neighbors .metric .estimator mean     n  
##     <int> <chr>      <chr>    <dbl> <int>  
## 1         1 accuracy binary    0.618    10  
## 2         2 accuracy binary    0.618    10  
## 3         3 accuracy binary    0.672    10  
## 4         4 accuracy binary    0.672    10  
## 5         5 accuracy binary    0.69     10  
## 6         6 accuracy binary    0.69     10
```

Visualisation des erreurs

```
> tbl <- ppv.cv %>% collect_metrics()  
> ggplot(tbl)+aes(x=neighbors,y=mean)+geom_line()+ylab("Accuracy")
```



Sélection du meilleur paramètre

- On visualise les **meilleures** valeurs de paramètres :

```
> ppv.cv %>% show_best() %>% select(1:6)
## # A tibble: 5 x 6
##   neighbors .metric .estimator mean     n std_err
##   <int> <chr>      <chr>    <dbl> <int> <dbl>
## 1      13 accuracy binary    0.72    10 0.0255
## 2      14 accuracy binary    0.72    10 0.0255
## 3      35 accuracy binary    0.72    10 0.0207
## 4      36 accuracy binary    0.72    10 0.0207
## 5      39 accuracy binary    0.718   10 0.0199
```

Sélection du meilleur paramètre

- On visualise les **meilleures** valeurs de paramètres :

```
> ppv.cv %>% show_best() %>% select(1:6)
## # A tibble: 5 x 6
##   neighbors .metric .estimator mean     n std_err
##   <int> <chr>      <chr>    <dbl> <int> <dbl>
## 1      13 accuracy binary    0.72    10 0.0255
## 2      14 accuracy binary    0.72    10 0.0255
## 3      35 accuracy binary    0.72    10 0.0207
## 4      36 accuracy binary    0.72    10 0.0207
## 5      39 accuracy binary    0.718   10 0.0199
```

- et on choisit celle qui **maximise l'accuracy** :

```
> best_k <- ppv.cv %>% select_best()
> best_k
## # A tibble: 1 x 2
##   neighbors .config
##   <int> <chr>
## 1      13 Preprocessor1_Model013
```

Algorithme final et prévision

- L'**algorithme final** s'obtient en entraînant la méthode sur **toutes les données** pour la **valeur de paramètre sélectionné** :

```
> final_ppv <-  
+   ppv_wf %>%  
+   finalize_workflow(best_k) %>%  
+   fit(data = don.2D.500)
```


Algorithme final et prévision

- L'**algorithme final** s'obtient en entraînant la méthode sur **toutes les données** pour la **valeur de paramètre sélectionné** :

```
> final_ppv <-  
+   ppv_wf %>%  
+   finalize_workflow(best_k) %>%  
+   fit(data = don.2D.500)
```

- On peut maintenant prédire de nouveaux individus :

```
> newx <- tibble(X1=0.3,X2=0.8)  
> predict(final_ppv,new_data=newx)  
## # A tibble: 1 x 1  
##   .pred_class  
##   <fct>  
## 1 0
```

Conclusion

- Les choix de l'utilisateur sont des paramètres de la procédure.
- \implies facilement personnalisable.
- Aisé de changer le critère, la méthode de ré-échantillonnage...

Motivations

Quelques exemples

Cadre statistique pour l'apprentissage supervisé

Estimation du risque

Annexe 1 : le package tidymodels

Annexe 2 : le package caret

Bibliographie

Le package caret

- Il permet d'évaluer la performance de **plus de 230 méthodes** :
<http://topepo.github.io/caret/index.html>

Le package caret

- Il permet d'évaluer la performance de **plus de 230 méthodes** :
<http://topepo.github.io/caret/index.html>
- Il suffit d'indiquer :
 - la **méthode** (logistique, ppv, arbre, randomForest...)
 - Une grille pour les **paramètres** (nombre de ppv...)
 - Le **critère de performance** (erreur de classification, AUC, risque quadratique...)
 - La méthode d'**estimation du critère** (apprentissage validation, validation croisée, bootstrap...)

Apprentissage-validation

```
> library(caret)
> K_cand <- data.frame(k=seq(1,500,by=20))
> library(caret)
> ctrl1 <- trainControl(method="LGOCV",number=1,index=list(1:1500))
> e1 <- train(Y~.,data=donnees,method="knn",trControl=ctrl1,tuneGrid=K_cand)
> e1

## k-Nearest Neighbors
##
## 2000 samples
##    2 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Repeated Train/Test Splits Estimated (1 reps, 75%)
## Summary of sample sizes: 1500
## Resampling results across tuning parameters:
##
##    k    Accuracy  Kappa
##    1  0.620      0.2382571
##   21  0.718      0.4342076
##   41  0.722      0.4418388
```

```
## 61 0.718 0.4344073
## 81 0.720 0.4383195
## 101 0.714 0.4263847
## 121 0.716 0.4304965
## 141 0.718 0.4348063
## 161 0.718 0.4348063
## 181 0.718 0.4348063
## 201 0.720 0.4387158
## 221 0.718 0.4350056
## 241 0.718 0.4350056
## 261 0.722 0.4428232
## 281 0.714 0.4267894
## 301 0.714 0.4269915
## 321 0.710 0.4183621
## 341 0.696 0.3893130
## 361 0.696 0.3893130
## 381 0.688 0.3727988
## 401 0.684 0.3645329
## 421 0.686 0.3686666
## 441 0.686 0.3679956
## 461 0.684 0.3638574
## 481 0.680 0.3558050
```

```
## Accuracy was used to select the optimal model using the largest value.
```

```
## The final value used for the model was k = 261.
```

Validation croisée

```
> library(doMC)
> registerDoMC(cores = 3)
> ctrl12 <- trainControl(method="cv",number=10)
> e2 <- train(Y~.,data=dapp,method="knn",trControl=ctrl12,tuneGrid=K_cand)
> e2
## k-Nearest Neighbors
##
## 1500 samples
##    2 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 1350, 1350, 1350, 1350, 1350, 1350, ...
## Resampling results across tuning parameters:
##
##   k    Accuracy   Kappa
##   1  0.6240000  0.2446251
##   21 0.7393333  0.4745290
##   41 0.7306667  0.4570024
##   61 0.7340000  0.4636743
```



```
##      81  0.7333333  0.4632875
##     101  0.7313333  0.4593480
##     121  0.7326667  0.4624249
##     141  0.7333333  0.4640787
##     161  0.7366667  0.4708178
##     181  0.7313333  0.4602309
##     201  0.7326667  0.4626618
##     221  0.7293333  0.4559741
##     241  0.7306667  0.4585960
##     261  0.7353333  0.4676751
##     281  0.7286667  0.4537842
##     301  0.7253333  0.4463516
##     321  0.7173333  0.4294524
##     341  0.7113333  0.4168003
##     361  0.7080000  0.4099303
##     381  0.7140000  0.4213569
##     401  0.7073333  0.4073761
##     421  0.7100000  0.4126434
##     441  0.7066667  0.4054984
##     461  0.6966667  0.3844183
##     481  0.6860000  0.3612515
##
```

```
## Accuracy was used to select the optimal model using the largest value.
```

```
## The final value used for the model was k = 21.
```

Validation croisée répétée

```
> ctrl3 <- trainControl(method="repeatedcv",repeats=5,number=10)
> e3 <- train(Y~.,data=dapp,method="knn",trControl=ctrl3,tuneGrid=K_cand)
> e3
## k-Nearest Neighbors
##
## 1500 samples
##    2 predictor
##    2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 5 times)
## Summary of sample sizes: 1350, 1350, 1350, 1350, 1350, 1350, ...
## Resampling results across tuning parameters:
##
##    k    Accuracy    Kappa
##    1  0.6232000  0.2438066
##   21  0.7354667  0.4665640
##   41  0.7314667  0.4585144
##   61  0.7317333  0.4592608
##   81  0.7302667  0.4568784
##  101  0.7310667  0.4589567
```

```
## 121 0.7320000 0.4609326
## 141 0.7322667 0.4616077
## 161 0.7336000 0.4643374
## 181 0.7340000 0.4649895
## 201 0.7332000 0.4632905
## 221 0.7325333 0.4620114
## 241 0.7316000 0.4600484
## 261 0.7305333 0.4578098
## 281 0.7286667 0.4536040
## 301 0.7238667 0.4434101
## 321 0.7189333 0.4330787
## 341 0.7136000 0.4215865
## 361 0.7122667 0.4183400
## 381 0.7098667 0.4131761
## 401 0.7090667 0.4112403
## 421 0.7058667 0.4043164
## 441 0.7001333 0.3920207
## 461 0.6952000 0.3811374
## 481 0.6872000 0.3636126
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 21.
```

Critère AUC

```
> donnees1 <- donnees
> names(donnees1)[3] <- c("Class")
> levels(donnees1$Class) <- c("G0", "G1")
> ctrl11 <- trainControl(method="LGOVCV", number=1, index=list(1:1500),
+                        classProbs=TRUE, summary=twoClassSummary)
> e4 <- train(Class~., data=donnees1, method="knn", trControl=ctrl11,
+            metric="ROC", tuneGrid=K_cand)
> e4
## k-Nearest Neighbors
##
## 2000 samples
##    2 predictor
##    2 classes: 'G0', 'G1'
##
## No pre-processing
## Resampling: Repeated Train/Test Splits Estimated (1 reps, 75%)
## Summary of sample sizes: 1500
## Resampling results across tuning parameters:
##
```

##	k	ROC	Sens	Spec
##	1	0.6190866	0.5983264	0.6398467
##	21	0.7171484	0.6903766	0.7432950
##	41	0.7229757	0.6861925	0.7547893
##	61	0.7200500	0.6945607	0.7394636
##	81	0.7255567	0.6945607	0.7432950
##	101	0.7319450	0.6903766	0.7356322
##	121	0.7382452	0.6945607	0.7356322
##	141	0.7353757	0.7029289	0.7318008
##	161	0.7308549	0.7029289	0.7318008
##	181	0.7351272	0.7029289	0.7318008
##	201	0.7340050	0.7029289	0.7356322
##	221	0.7324099	0.7071130	0.7279693
##	241	0.7349028	0.7071130	0.7279693
##	261	0.7365780	0.7071130	0.7356322
##	281	0.7349749	0.6987448	0.7279693
##	301	0.7356963	0.7029289	0.7241379
##	321	0.7341493	0.6861925	0.7318008
##	341	0.7343898	0.6527197	0.7356322
##	361	0.7306385	0.6527197	0.7356322
##	381	0.7301816	0.6359833	0.7394636
##	401	0.7270957	0.6276151	0.7356322
##	421	0.7255487	0.6317992	0.7356322

```
## 441 0.7258933 0.6192469 0.7471264
## 461 0.7220619 0.6150628 0.7471264
## 481 0.7236330 0.6108787 0.7432950
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was k = 121.
```

Motivations

Quelques exemples




Cadre statistique pour l'apprentissage supervisé




Estimation du risque

Annexe 1 : le package tidymodels

Annexe 2 : le package caret

Bibliographie

-  Besse, P. (2018).
Science des données - Apprentissage Statistique.
INSA - Toulouse.
http://www.math.univ-toulouse.fr/~besse/pub/Appren_stat.pdf.
-  Bousquet, O., Boucheron, S., and Lugosi, G. (2003).
Introduction to Statistical Learning Theory, chapter **Advanced Lectures on Machine Learning**.
Springer.
-  Cléménçon, S., Lugosi, G., and Vayatis, N. (2008).
Ranking and empirical minimization of u -statistics.
The Annals of Statistics, 36(2) :844–874.

-  Hastie, T., Tibshirani, R., and Friedman, J. (2009).
The Elements of Statistical Learning : Data Mining, Inference, and Prediction.
Springer, second edition.
-  James, G., Witten, D., Hastie, T., and Tibshirani, R. (2015).
The Elements of Statistical Learning : Data Mining, Inference, and Prediction.
Springer.
-  Vapnik, V. (2000).
The Nature of Statistical Learning Theory.
Springer, second edition.



Wikistat (2020a).

Apprentissage machine — introduction.

<http://wikistat.fr/pdf/st-m-Intro-ApprentStat.pdf>.



Wikistat (2020b).

Qualité de prévision et risque.

<http://wikistat.fr/pdf/st-m-app-risque.pdf>.

Deuxième partie II

Algorithmes linéaires

Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variantes de ridge/lasso

Discrimination binaire

Bibliographie

- **Rappel** : une fonction de prévision $f : \mathbb{R}^d \rightarrow \mathbb{R}$.

- **Rappel** : une fonction de prévision $f : \mathbb{R}^d \rightarrow \mathbb{R}$.

Fonction de prévision linéaire

Une fonction de prévision est dite **linéaire** si elle se met sous la forme

$$f(x) = f_{\beta}(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

- **Rappel** : une fonction de prévision $f : \mathbb{R}^d \rightarrow \mathbb{R}$.

Fonction de prévision linéaire

Une fonction de prévision est dite **linéaire** si elle se met sous la forme

$$f(x) = f_{\beta}(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

Remarque

- Possibilité d'inclure des **effets non linéaires** :

$$f_{\beta}(x) = \beta_0 + \beta_{11} x_1 + \beta_{12} x_1^2 + \beta_{21} x_2 + \beta_{22} x_2^2 + \beta_{12} x_1 x_2 + \beta_{31} x_3 + \beta_{32} \exp(x_3) \dots$$

- Variables **qualitatives** codées en indicatrices :

$$f_{\beta}(x) = \beta_0 + \beta_1 \mathbf{1}_{x_1=A} + \beta_2 \mathbf{1}_{x_1=B} + \beta_3 \mathbf{1}_{x_1=C} + \dots$$

muni d'une contrainte identifiante, par exemple $\beta_1 = 0$.

- Y à valeurs dans \mathbb{R} .
- On utilise souvent le terme **modèle linéaire** :

$$y_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_d x_{id} + \varepsilon_i$$

où les ε_j sont i.i.d tels que $E[\varepsilon_j] = 0$ et $V[\varepsilon_j] = \sigma^2$.

- Y à valeurs dans \mathbb{R} .
- On utilise souvent le terme **modèle linéaire** :

$$y_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_d x_{id} + \varepsilon_i$$

où les ε_i sont i.i.d tels que $E[\varepsilon_i] = 0$ et $V[\varepsilon_i] = \sigma^2$.

- **Fonction de prévision** :

$$m_\beta(x) = E[Y|X = x] = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

Classification binaire

- Y à valeurs dans $\{0, 1\}$.
- La classification s'effectue à partir de la probabilité

$$p(x) = P(Y = 1|X = x).$$

Classification binaire

- Y à valeurs dans $\{0, 1\}$.
- La classification s'effectue à partir de la probabilité

$$p(x) = P(Y = 1|X = x).$$

- **Frontière entre les deux classes :**

$$\{x : p(x) = 1 - p(x)\} = \left\{ x : \log \frac{p(x)}{1 - p(x)} = 0 \right\}.$$

Classification binaire

- Y à valeurs dans $\{0, 1\}$.
- La classification s'effectue à partir de la probabilité

$$p(x) = P(Y = 1|X = x).$$

- **Frontière entre les deux classes** :

$$\{x : p(x) = 1 - p(x)\} = \left\{ x : \log \frac{p(x)}{1 - p(x)} = 0 \right\}.$$

- La frontière est **linéaire** si

$$\log \frac{p(x)}{1 - p(x)} = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

Classification binaire

- Y à valeurs dans $\{0, 1\}$.
- La classification s'effectue à partir de la probabilité

$$p(x) = P(Y = 1|X = x).$$

- **Frontière entre les deux classes** :

$$\{x : p(x) = 1 - p(x)\} = \left\{ x : \log \frac{p(x)}{1 - p(x)} = 0 \right\}.$$

- La frontière est **linéaire** si

$$\log \frac{p(x)}{1 - p(x)} = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

- \implies **Modèle logistique**.

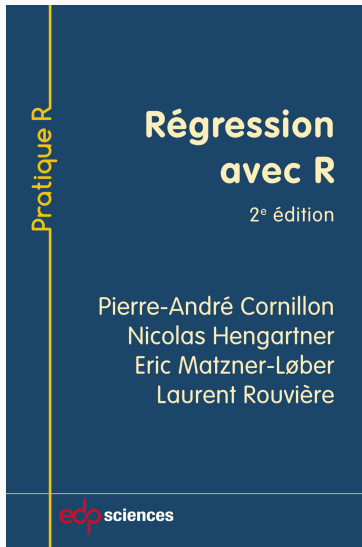
1. Comment **calculer** (ou plutôt **estimer**) les β_j ?

1. Comment **calculer** (ou plutôt **estimer**) les β_j ?
 - MCO-vraisemblance
 - Approches régularisées \implies ridge-lasso...
 - Machines à support vecteur (SVM).

1. Comment **calculer** (ou plutôt **estimer**) les β_j ?
 - MCO-vraisemblance
 - Approches régularisées \implies ridge-lasso...
 - Machines à support vecteur (SVM).
2. Comment **choisir** la combinaison linéaire?

1. Comment **calculer** (ou plutôt **estimer**) les β_j ?
 - MCO-vraisemblance
 - Approches régularisées \implies ridge-lasso...
 - Machines à support vecteur (SVM).
2. Comment **choisir** la combinaison linéaire?
 - Sélection de variables
 - Régression sur composantes \implies PCR-PLS...
 - Transformation de variables \implies résidus partiels, modèle additifs...

[Cornillon et al., 2019] : <https://regression-avec-r.github.io>



Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variante de ridge/lasso

Discrimination binaire

Bibliographie

Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variantes de ridge/lasso

Discrimination binaire

Bibliographie

Minimiser les erreurs

- Les **données** : $(x_1, y_1), \dots, (x_n, y_n)$ à valeurs dans $\mathbb{R}^d \times \mathbb{R}$.
- Le **modèle**

$$y_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_d x_{id} + \varepsilon_i$$

- Les **données** : $(x_1, y_1), \dots, (x_n, y_n)$ à valeurs dans $\mathbb{R}^d \times \mathbb{R}$.
- Le **modèle**

$$y_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_d x_{id} + \varepsilon_i$$

- ε_i représente **l'écart (ou l'erreur)** entre la prévision du modèle β et la valeur observée.

Minimiser les erreurs

- Les **données** : $(x_1, y_1), \dots, (x_n, y_n)$ à valeurs dans $\mathbb{R}^d \times \mathbb{R}$.
- Le **modèle**

$$y_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_d x_{id} + \varepsilon_i$$

- ε_i représente **l'écart (ou l'erreur)** entre la prévision du modèle β et la valeur observée.

Idée

Choisir β de manière à **minimiser ces erreurs**.

Estimateurs des moindres carrés

Définition

On appelle **critère des moindres carrés ordinaires** ou **somme des carrés résiduelles** la fonction de β :

$$\text{SCR}(\beta) = \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_{i1} + \dots + \beta_d x_{id}))^2 = \|\mathbb{Y} - \mathbb{X}\beta\|^2$$

avec

$$\mathbb{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \quad \text{et} \quad \mathbb{X} = \begin{pmatrix} 1 & x_{11} & \dots & x_{1d} \\ \vdots & \vdots & & \vdots \\ 1 & x_{n1} & \dots & x_{nd} \end{pmatrix}.$$

Estimateurs des moindres carrés

Définition

On appelle **critère des moindres carrés ordinaires** ou **somme des carrés résiduelles** la fonction de β :

$$\text{SCR}(\beta) = \sum_{i=1}^n (y_i - (\beta_0 + \beta_1 x_{i1} + \dots + \beta_d x_{id}))^2 = \|\mathbb{Y} - \mathbb{X}\beta\|^2$$

avec

$$\mathbb{Y} = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} \quad \text{et} \quad \mathbb{X} = \begin{pmatrix} 1 & x_{11} & \dots & x_{1d} \\ \vdots & \vdots & & \vdots \\ 1 & x_{n1} & \dots & x_{nd} \end{pmatrix}.$$

Propriété

Si \mathbb{X} est de plein rang alors l'**estimateur des MCO** $\hat{\beta} = (\mathbb{X}^t \mathbb{X})^{-1} \mathbb{X}^t \mathbb{Y}$ minimise $\text{SCR}(\beta)$.

Exemple

- **Données Hitters**, 263 individus, 20 variables

```
> Hitters %>% select(c(1:5,19)) %>% head()
##   AtBat Hits HmRun Runs RBI Salary
## 1   315   81     7   24  38  475.0
## 2   479  130    18   66  72  480.0
## 3   496  141    20   65  78  500.0
## 4   321   87    10   39  42   91.5
## 5   594  169     4   74  51  750.0
## 6   185   37     1   23   8   70.0
```

- **Problème** : Expliquer/prédire le salaire (**Salary**) par les autres variables.

- Calcul des estimateurs MCO avec lm :

```
> mod <- lm(Salary~.,data=Hitters)
> coef(mod)[1:5]
## (Intercept)      AtBat      Hits      HmRun      Runs
## 163.103588    -1.979873    7.500768    4.330883    -2.376210
```

- Calcul des estimateurs MCO avec `lm` :

```
> mod <- lm(Salary~.,data=Hitters)
> coef(mod)[1:5]
## (Intercept)      AtBat      Hits      HmRun      Runs
## 163.103588    -1.979873    7.500768    4.330883   -2.376210
```

- Prévision du salaire de nouveaux individus

```
> xnew %>% select(1:5)
##   AtBat Hits HmRun Runs RBI
## 1   585  139   31   93  94
```

avec `predict` :

```
> predict(mod,newdata=xnew)
##           1
## 1129.376
```

- En **supposant** de plus que les erreurs ε_j suivent une **loi Gaussienne**, on obtient la loi des estimateurs

$$\frac{\hat{\beta}_j - \beta_j}{\hat{\sigma}_{\hat{\beta}_j}} \sim \mathcal{T}_{n-(d+1)}.$$

Modèle gaussien

- En **supposant** de plus que les erreurs ε_j suivent une **loi Gaussienne**, on obtient la loi des estimateurs

$$\frac{\hat{\beta}_j - \beta_j}{\hat{\sigma}_{\hat{\beta}_j}} \sim \mathcal{T}_{n-(d+1)}.$$

- On en déduit des **procédures de test** :

```
> broom::tidy(mod) %>% head()
## # A tibble: 6 x 5
##   term          estimate std.error statistic p.value
##   <chr>         <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)  163.       90.8        1.80  0.0736
## 2 AtBat        -1.98       0.634       -3.12  0.00201
## 3 Hits         7.50       2.38        3.15  0.00181
## 4 HmRun        4.33       6.20        0.698 0.486
## 5 Runs        -2.38       2.98       -0.797 0.426
## 6 RBI         -1.04       2.60       -0.402 0.688
```

- Ainsi que des intervalles de confiance pour les paramètres :

```
> confint(mod) %>% head()
##              2.5 %          97.5 %
## (Intercept) -15.709647 341.9168228
## AtBat       -3.228667  -0.7310792
## Hits        2.817562  12.1839734
## HmRun       -7.884569  16.5463352
## Runs       -8.247625   3.4952055
## RBI        -6.168102   4.0781779
```

- Ainsi que des intervalles de confiance pour les paramètres :

```
> confint(mod) %>% head()
##              2.5 %          97.5 %
## (Intercept) -15.709647 341.9168228
## AtBat       -3.228667  -0.7310792
## Hits        2.817562  12.1839734
## HmRun       -7.884569  16.5463352
## Runs       -8.247625   3.4952055
## RBI        -6.168102   4.0781779
```

- ou pour les prévisions :

```
> predict(mod, newdata=xnew, interval="confidence")
##      fit      lwr      upr
## 1 1129.376 889.2244 1369.528
```


Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variantes de ridge/lasso

Discrimination binaire

Bibliographie

Résidus et variance

Les **résidus** mesurent l'**ajustement** du modèle aux données. Ils sont définis par

$$\hat{\varepsilon} = Y - \hat{Y} = (I - P_X)Y = P_{X^\perp}Y = P_{X^\perp}\varepsilon$$

et vérifient

$$E[\hat{\varepsilon}] = 0 \quad V[\hat{\varepsilon}] = P_{X^\perp}\sigma^2.$$

Résidus et variance

Les **résidus** mesurent l'**ajustement** du modèle aux données. Ils sont définis par

$$\hat{\varepsilon} = Y - \hat{Y} = (I - P_X)Y = P_{X^\perp}Y = P_{X^\perp}\varepsilon$$

et vérifient

$$E[\hat{\varepsilon}] = 0 \quad V[\hat{\varepsilon}] = P_{X^\perp}\sigma^2.$$

- ε_i (**non observés**) homoscédastiques (même variance) et non corrélés
- $\hat{\varepsilon}_i$ (**observés**) hétéroscédastiques et corrélés.

Résidus et variance

Les **résidus** mesurent l'**ajustement** du modèle aux données. Ils sont définis par

$$\hat{\varepsilon} = Y - \hat{Y} = (I - P_X)Y = P_{X^\perp}Y = P_{X^\perp}\varepsilon$$

et vérifient

$$E[\hat{\varepsilon}] = 0 \quad V[\hat{\varepsilon}] = P_{X^\perp}\sigma^2.$$

- ε_i (**non observés**) homoscédastiques (même variance) et non corrélés
- $\hat{\varepsilon}_i$ (**observés**) hétéroscédastiques et corrélés.

Estimation de la variance

$$\hat{\sigma}^2 = \frac{1}{n - d + 1} \|\hat{\varepsilon}\|^2, \quad E[\hat{\sigma}^2] = \sigma^2.$$

Définitions

- résidus $\hat{\varepsilon}_i = y_i - \hat{y}_i$.
- résidus **normalisés** $r_i = \hat{\varepsilon}_i / (\sigma \sqrt{1 - h_{ii}})$
- résidus **standardisés** (**rstandard**) $t_i = \hat{\varepsilon}_i / (\hat{\sigma} \sqrt{1 - h_{ii}})$
- résidus **studentisés** par Validation Croisée (**rstudent**)

$$t_i^* = \frac{\hat{\varepsilon}_i}{\hat{\sigma}_{(i)} \sqrt{1 - h_{ii}}},$$

Définitions

- résidus $\hat{\varepsilon}_i = y_i - \hat{y}_i$.
- résidus **normalisés** $r_i = \hat{\varepsilon}_i / (\sigma \sqrt{1 - h_{ii}})$
- résidus **standardisés** (**rstandard**) $t_i = \hat{\varepsilon}_i / (\hat{\sigma} \sqrt{1 - h_{ii}})$
- résidus **studentisés** par Validation Croisée (**rstudent**)

$$t_i^* = \frac{\hat{\varepsilon}_i}{\hat{\sigma}_{(i)} \sqrt{1 - h_{ii}}},$$

Propriété

Sous l'hypothèse de normalité des résidus,

$$t_i^* \sim \mathcal{T}(n - d).$$

Définition

Une **donnée aberrante** est un point (x_i, y_i) pour lequel la valeur associée à t_i^* est élevée (comparée au seuil donné par la loi du Student) :

$$|t_i^*| > t_{n-p-1}(1 - \alpha/2).$$

Définition

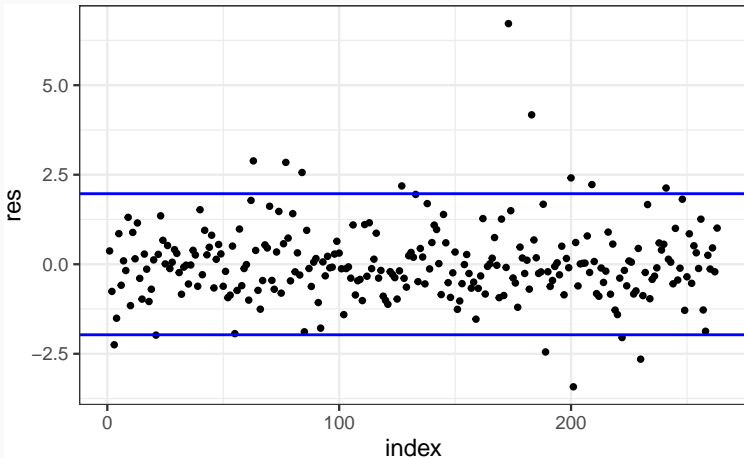
Une **donnée aberrante** est un point (x_i, y_i) pour lequel la valeur associée à t_i^* est élevée (comparée au seuil donné par la loi du Student) :

$$|t_i^*| > t_{n-p-1}(1 - \alpha/2).$$

Diagnostic

1. Visualiser les résidus sur un graphe.
2. Identifier les données avec des résidus élevés.
3. Les **éliminer** de façon **séquentielle**.


```
> res <- rstudent(mod)
> tbl <- tibble(index=1:nrow(Hitters),res=res)
> seuil <- qt(0.975,nrow(Hitters)-(ncol(Hitters)-1))
> ggplot(tbl)+aes(x=index,y=res)+geom_point()+
+   geom_hline(yintercept = c(-seuil,seuil),color="blue")
```



Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variante de ridge/lasso

Discrimination binaire

Bibliographie

- Variable à expliquer **binaire** $\implies y_i \in \{0, 1\}$.
- Nombreuses applications : malade/pas malade, bon/mauvais payeur...

- Variable à expliquer **binaire** $\implies y_i \in \{0, 1\}$.
- Nombreuses applications : malade/pas malade, bon/mauvais payeur...
- **Quantité d'intérêt** : $p(x) = P(Y = 1|X = x)$.

- Variable à expliquer **binaire** $\implies y_i \in \{0, 1\}$.
- Nombreuses applications : malade/pas malade, bon/mauvais payeur...
- **Quantité d'intérêt** : $p(x) = P(Y = 1|X = x)$.

Approche linéaire

$p(x) = p_\beta(x)$ avec

$$\log \frac{p_\beta(x)}{1 - p_\beta(x)} = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

\implies **modèle logistique.**

Estimation

- Comme précédemment les β sont **inconnus**.

Estimation

- Comme précédemment les β sont **inconnus**.
- Le critère des MCO est remplacé par la **log-vraisemblance** (à maximiser) :

$$\mathcal{L}(y_1, \dots, y_n; \beta) = \sum_{i=1}^n [y_i x_i^t \beta - \log(1 + \exp(x_i^t \beta))].$$

Estimation

- Comme précédemment les β sont **inconnus**.
- Le critère des MCO est remplacé par la **log-vraisemblance** (à maximiser) :

$$\mathcal{L}(y_1, \dots, y_n; \beta) = \sum_{i=1}^n [y_i x_i^t \beta - \log(1 + \exp(x_i^t \beta))].$$

- Pas de solution explicite mais de **(bons) algorithmes qui convergent vers le max.**

Estimation

- Comme précédemment les β sont **inconnus**.
- Le critère des MCO est remplacé par la **log-vraisemblance** (à maximiser) :

$$\mathcal{L}(y_1, \dots, y_n; \beta) = \sum_{i=1}^n [y_i x_i^t \beta - \log(1 + \exp(x_i^t \beta))].$$

- Pas de solution explicite mais de **(bons) algorithmes qui convergent vers le max**.

Propriétés

Pour n assez grand, la loi des estimateurs peut être approchée par une **loi gaussienne** :

$$\mathcal{L}(\hat{\beta}) \approx \mathcal{N}(\beta, \Sigma_{\hat{\beta}}).$$

⇒ procédures de test, intervalles de confiance...

Exemple

- On considère les données **SAheart** :

```
> head(SAheart)
##   sbp tobacco  ldl adiposity famhist typea obesity alcohol age chd
## 1 160   12.00 5.73   23.11 Present   49   25.30   97.20 52  1
## 2 144    0.01 4.41   28.61  Absent   55   28.87    2.06 63  1
## 3 118    0.08 3.48   32.28 Present   52   29.14    3.81 46  0
## 4 170    7.50 6.41   38.03 Present   51   31.99   24.26 58  1
## 5 134   13.60 3.50   27.78 Present   60   25.99   57.34 49  1
## 6 132    6.20 6.47   36.21 Present   62   30.77   14.14 45  0
```

- Problème** : expliquer/prédire la variable binaire **chd** par les autres variables.

- On obtient les estimateurs avec `glm`

```
> logit <- glm(chd~.,data=SAheart,family="binomial")
> broom::tidy(logit)
## # A tibble: 10 x 5
##   term                estimate std.error statistic    p.value
##   <chr>                <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)        -6.15     1.31     -4.70  0.00000258
## 2 sbp                 0.00650   0.00573    1.14   0.256
## 3 tobacco            0.0794    0.0266    2.98   0.00285
## 4 ldl                 0.174     0.0597    2.92   0.00355
## 5 adiposity          0.0186    0.0293    0.635  0.526
## 6 famhistPresent     0.925     0.228     4.06   0.0000490
## 7 typea              0.0396    0.0123    3.21   0.00131
## 8 obesity            -0.0629   0.0442   -1.42   0.155
## 9 alcohol            0.000122  0.00448    0.0271 0.978
## 10 age               0.0452    0.0121    3.73   0.000193
```

Prévision

- Nouvel individu $x \in \mathbb{R}^d$.
- **Question** : que prédire ?

Prévision

- Nouvel individu $x \in \mathbb{R}^d$.
- Question : que prédire ?

2 possibilités

1. La probabilité $p(x) = P(Y = 1|X = x)$.
2. La classe de x (0 ou 1).

- Nouvel individu $x \in \mathbb{R}^d$.
- Question : que prédire ?

2 possibilités

1. La probabilité $p(x) = P(Y = 1|X = x)$.
2. La classe de x (0 ou 1).

⇒ la classe se déduisant souvent de la probabilité, il est souvent préférable de prédire cette dernière.

Prévision

- Nouvel individu $x \in \mathbb{R}^d$.
- **Question** : que prédire ?

2 possibilités

1. La **probabilité** $p(x) = P(Y = 1|X = x)$.
2. La **classe** de x (0 ou 1).

⇒ la **classe se déduisant souvent de la probabilité**, il est souvent préférable de prédire cette dernière.

Critère de prévision

Ils dépendent de la quantité prédite, par exemple

- **courbe ROC** pour des probabilités.
- **erreur de classification** pour des classes.

- Les prévisions de la probabilité de l'évènement {**chd=1**} pour de nouveaux individus

```
> xnew
##   sbp tobacco  ldl adiposity famhist typea obesity alcohol age
## 1 146         0 6.62    25.69 Absent    60   28.07    8.23  63
```

- s'obtiennent avec **predict** :

```
> predict(logit,newdata=xnew,type="response")
##           1
## 0.4719671
```


Conclusion

Remarque

La qualité de ces modèles (et donc des prévisions) reposent sur deux postulats :

1. le **modèle est bon** : Y s'explique bien par une combinaison linéaire des X ;
2. les **estimateurs sont bons** : ils possèdent de bonnes propriétés statistiques.

Conclusion

Remarque

La qualité de ces modèles (et donc des prévisions) reposent sur deux postulats :

1. le **modèle est bon** : Y s'explique bien par une combinaison linéaire des X ;
2. les **estimateurs sont bons** : ils possèdent de bonnes propriétés statistiques.

- La qualité du modèle est toujours **difficile à vérifier** \implies ajouter d'autres effets dans la combinaison linéaire (quadratique, interactions...).
- On en sait plus sur la **performance des estimateurs** :

Conclusion

Remarque

La qualité de ces modèles (et donc des prévisions) reposent sur deux postulats :

1. le **modèle est bon** : Y s'explique bien par une combinaison linéaire des X ;
2. les **estimateurs sont bons** : ils possèdent de bonnes propriétés statistiques.

- La qualité du modèle est toujours **difficile à vérifier** \implies ajouter d'autres effets dans la combinaison linéaire (quadratique, interactions...).
- On en sait plus sur la **performance des estimateurs** :
 1. **Trop de variables** \implies \nearrow de la variance (sur-ajustement).
 2. **Colinéarités** \implies \nearrow de la variance (sur-ajustement).

Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variante de ridge/lasso

Discrimination binaire

Bibliographie

- Une approche naturelle pour répondre aux 2 problèmes évoqués précédemment est de sélectionner des variables explicatives parmi $\{X_1, \dots, X_d\}$.

Idée

Supprimer les variables

- qui n'expliquent pas Y .
- dont l'effet est déjà expliqué par d'autres variables

⇒ ce n'est pas parce qu'une variable n'est pas sélectionnée qu'elle n'est pas liée à Y !

Best subset selection

- d variables explicatives $\implies 2^d$ modèles concurrents.
- **Idée** : **construire** les 2^d modèles et les **comparer**.

Best subset selection

- d variables explicatives $\implies 2^d$ modèles concurrents.
- **Idée** : construire les 2^d modèles et les comparer.

Algorithme BSS

Entrée : un critère de choix de modèle (AIC, BIC...).

Pour $j = 0, \dots, d$:

1. Construire les $\binom{d}{j}$ modèles linéaires à j variables ;
2. Choisir parmi ces modèles celui qui a la plus petite SCR. On note \mathcal{M}_j le modèle sélectionné.

Retourner : le meilleur modèle parmi $\mathcal{M}_0, \mathcal{M}_1, \dots, \mathcal{M}_d$ au sens du critère de choix de modèle.

Exemples de critères (voir [Cornillon et al., 2019])

- **AIC** : Akaike Information Criterion

$$-2\mathcal{L}_n(\hat{\beta}) + 2d.$$

- **BIC** : Bayesian Information Criterion

$$-2\mathcal{L}_n(\hat{\beta}) + \log(n)d.$$

- **R^2 ajusté** :

$$R_a^2 = 1 - \frac{n-1}{n-d+1}(1-R^2) \quad \text{où} \quad R^2 = \frac{SSR}{SST} = \frac{\|\hat{Y} - \bar{Y}\mathbf{1}\|^2}{\|Y - \bar{Y}\mathbf{1}\|^2}.$$

- **C_p de Mallow** :

$$C_p = \frac{1}{n} \left(\sum_{i=1}^n (Y_i - \hat{Y}_i)^2 + 2d\hat{\sigma}^2 \right).$$

- Ces critères sont constitués de deux parties :
 1. une qui mesure la **qualité d'ajustement** du modèle ;
 2. une autre qui mesure sa **complexité**.

- Ces critères sont constitués de deux parties :
 1. une qui mesure la **qualité d'ajustement** du modèle ;
 2. une autre qui mesure sa **complexité**.

Exemple AIC

- $-2\mathcal{L}_n(\hat{\beta})$ mesure l'ajustement ;
- $2p$ mesure la complexité.

- Ces critères sont constitués de deux parties :
 1. une qui mesure la **qualité d'ajustement** du modèle ;
 2. une autre qui mesure sa **complexité**.

Exemple AIC

- $-2\mathcal{L}_n(\hat{\beta})$ mesure l'ajustement ;
- $2p$ mesure la complexité.

⇒ l'idée est de choisir un modèle de **complexité minimale** qui **ajuste bien** les données.

- On peut utiliser les packages `leaps` et `bestglm`.
- On propose de présenter `bestglm` qui fait appel à `leaps` pour la régression et fonctionne également pour le modèle logistique.

- On peut utiliser les packages `leaps` et `bestglm`.
- On propose de présenter `bestglm` qui fait appel à `leaps` pour la régression et fonctionne également pour le modèle logistique.

```
> Hitters1 <- Hitters[,c(1:18,20,19)]
> sel.var <- bestglm(Hitters1)
> sel.var$Subsets %>% select(c(1:5,22)) %>% head()
##      (Intercept) AtBat Hits HmRun Runs      BIC
## 0             TRUE FALSE FALSE FALSE FALSE 3213.768
## 1             TRUE FALSE FALSE FALSE FALSE 3117.350
## 2             TRUE FALSE  TRUE FALSE FALSE 3079.270
## 3             TRUE FALSE  TRUE FALSE FALSE 3072.569
## 4             TRUE FALSE  TRUE FALSE FALSE 3066.387
## 5             TRUE  TRUE  TRUE FALSE FALSE 3064.125
```

- On obtient le **modèle sélectionné** avec :

```
> sel.var$BestModel %>% broom::tidy()
## # A tibble: 7 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>    <dbl>    <dbl>    <dbl>
## 1 (Intercept)    91.5     65.0      1.41 1.60e- 1
## 2 AtBat         -1.87     0.527    -3.54 4.70e- 4
## 3 Hits           7.60     1.66     4.57 7.46e- 6
## 4 Walks          3.70     1.21     3.06 2.49e- 3
## 5 CRBI           0.643    0.0644    9.98 5.05e-20
## 6 DivisionW    -123.     39.8     -3.09 2.24e- 3
## 7 PutOuts       0.264    0.0748    3.53 4.84e- 4
```

- On obtient le **modèle sélectionné** avec :

```
> sel.var$BestModel %>% broom::tidy()
## # A tibble: 7 x 5
##   term          estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>    <dbl>   <dbl>
## 1 (Intercept)    91.5      65.0      1.41 1.60e- 1
## 2 AtBat         -1.87     0.527    -3.54 4.70e- 4
## 3 Hits          7.60     1.66     4.57 7.46e- 6
## 4 Walks         3.70     1.21     3.06 2.49e- 3
## 5 CRBI          0.643    0.0644    9.98 5.05e-20
## 6 DivisionW    -123.     39.8     -3.09 2.24e- 3
## 7 PutOuts       0.264    0.0748    3.53 4.84e- 4
```

Remarque

- L'approche **exhaustive** peut se révéler coûteuse en temps de calcul lorsque $d > 50$.
- On utilise généralement des méthodes **pas à pas** dans ce cas.

Pas à pas ascendant

Algorithme forward

Entrée : un critère de choix de modèle (AIC, BIC...)

1. Construire \mathcal{M}_0 le modèle linéaire qui contient uniquement la constante ;
2. Pour $j = 0, \dots, d - 1$:
 - 2.1 Construire les $d - j$ modèles linéaires en ajoutant une variable, parmi les variables non utilisées, à \mathcal{M}_j ;
 - 2.2 Choisir, parmi ces $d - j$ modèles, celui qui minimise la SCR $\rightarrow \mathcal{M}_{j+1}$.

Retourner : le meilleur modèle parmi $\mathcal{M}_0, \mathcal{M}_1, \dots, \mathcal{M}_d$ au sens du critère de choix de modèle.

Le coin R

Utiliser `method=forward` dans `bestglm`.

Pas à pas descendant

Algorithme backward

Entrée : un critère de choix de modèle (AIC, BIC...)

1. Construire \mathcal{M}_d le modèle linéaire complet (avec toutes les variables explicatives);
2. Pour $j = d, \dots, 1$:
 - 2.1 Construire les j modèles linéaires en supprimant une variable, parmi les variables non utilisées, à \mathcal{M}_j ;
 - 2.2 Choisir, parmi ces j modèles, celui qui minimise la SCR $\rightarrow \mathcal{M}_{j-1}$.

Retourner : le meilleur modèle parmi $\mathcal{M}_0, \mathcal{M}_1, \dots, \mathcal{M}_d$ au sens du critère de choix de modèle.

Le coin R

Utiliser `method=backward` dans `bestglm`.

Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

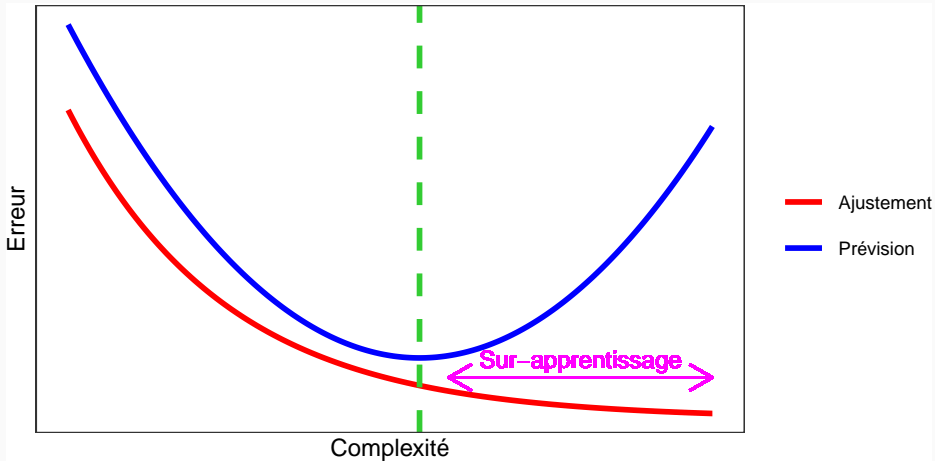
Régression ridge

Régression Lasso

Variante de ridge/lasso

Discrimination binaire

Bibliographie



Complexité linéaire

Le **nombre de variables** est une mesure de la complexité des algorithmes linéaires.

- On génère des données $(x_i, y_i), i = 1, \dots, 500$ selon le modèle

$$y_i = 1x_{i1} + 0x_{i2} + \dots + 0x_{iq} + \varepsilon_i$$

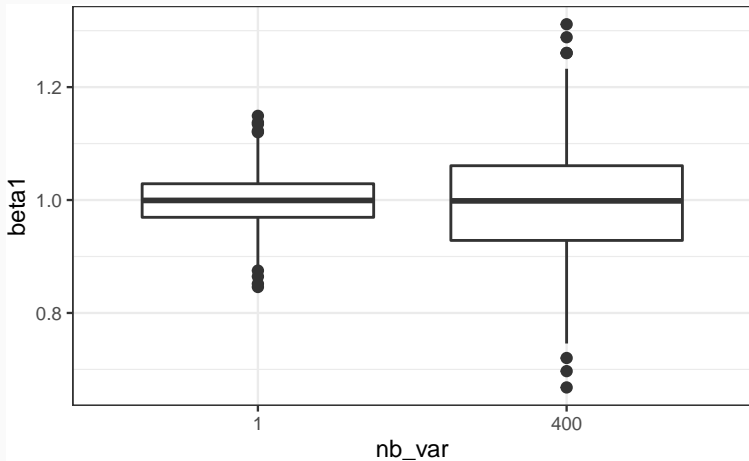
où $x_1, \dots, x_q, \varepsilon$ sont i.i.d. de loi $\mathcal{N}(0, 1)$.

- On génère des données $(x_i, y_i), i = 1, \dots, 500$ selon le modèle

$$y_i = 1x_{i1} + 0x_{i2} + \dots + 0x_{iq} + \varepsilon_i$$

où $x_1, \dots, x_q, \varepsilon$ sont i.i.d. de loi $\mathcal{N}(0, 1)$.

- Seule X_1 est **explicative**, les $q - 1$ autres variables peuvent être vues comme du **bruit**.
- On calcule l'**estimateur de MCO de β_1** sur 1000 répétitions. On trace les boxplot de ces estimateurs pour $q = 0$ et $q = 400$.



Conclusion

Plus de **variance** (donc **moins de précision**) lorsque le nombre de variables inutiles augmente.

- Lorsque le nombre de variables d est grand, les estimateurs des moindres carrés du modèle linéaire

$$Y = \beta_1 X_1 + \dots + \beta_d X_d + \varepsilon$$

possèdent généralement une grande variance.

- Lorsque le nombre de variables d est grand, les estimateurs des moindres carrés du modèle linéaire

$$Y = \beta_1 X_1 + \dots + \beta_d X_d + \varepsilon$$

possèdent généralement une grande variance.

Idée des méthodes pénalisés

- Contraindre la valeur des estimateurs des moindres carrés de manière à réduire la variance (quitte à augmenter un peu le biais).

- Lorsque le nombre de variables d est grand, les estimateurs des moindres carrés du modèle linéaire

$$Y = \beta_1 X_1 + \dots + \beta_d X_d + \varepsilon$$

possèdent généralement une grande variance.

Idée des méthodes pénalisés

- Contraindre la valeur des estimateurs des moindres carrés de manière à réduire la variance (quitte à augmenter un peu le biais).
- Comment ? En imposant une contrainte sur la valeur des estimateurs des moindres carrés :

$$\hat{\beta}^{pen} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^n \left(y_i - \sum_{j=1}^d x_{ij} \beta_j \right)^2$$

sous la contrainte $\|\beta\|_? \leq t$.

- Quelle **norme** utiliser pour la contrainte ?

- Quelle **norme** utiliser pour la contrainte ?
- **Existence/unicité** des estimateurs ? **Solutions explicites** du problème d'optimisation ?

- Quelle **norme** utiliser pour la contrainte ?
- **Existence/unicité** des estimateurs ? **Solutions explicites** du problème d'optimisation ?
- Comment **choisir t** ?
 - t petit \implies estimateurs **contraints** (proche de 0) ;
 - t grand \implies estimateurs des **moindres carrés** (non pénalisés).

- Cas similaire déjà vu pour LDA.
- **Modèle standard LDA** : $\mathcal{L}(X|Y = k) = \mathcal{N}(\mu_k, \Sigma)$.
- μ_k et Σ sont généralement estimés pour les **moyennes et matrice de covariance empiriques**.

- Cas similaire déjà vu pour LDA.
- **Modèle standard LDA** : $\mathcal{L}(X|Y = k) = \mathcal{N}(\mu_k, \Sigma)$.
- μ_k et Σ sont généralement estimés pour les **moyennes et matrice de covariance empiriques**.

LDA régularisée

On **régularise** la matrice de covariance en **augmentant les valeurs de la diagonale**

$$(1 - \gamma)\hat{\Sigma} + \gamma\hat{\sigma}^2 I_p.$$

Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variantes de ridge/lasso

Discrimination binaire

Bibliographie

- La **régression ridge** consiste à minimiser le critère des moindres carrés pénalisé par la norme 2 des coefficients.

Définition

1. Les **estimateurs ridge** $\hat{\beta}^R$ s'obtiennent en minimisant

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^d x_{ij} \beta_j \right)^2 \quad \text{sous la contrainte} \quad \sum_{j=1}^d \beta_j^2 \leq t \quad (1)$$

- La **régression ridge** consiste à minimiser le critère des moindres carrés pénalisé par la norme 2 des coefficients.

Définition

1. Les **estimateurs ridge** $\hat{\beta}^R$ s'obtiennent en minimisant

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^d x_{ij} \beta_j \right)^2 \quad \text{sous la contrainte} \quad \sum_{j=1}^d \beta_j^2 \leq t \quad (1)$$

2. ou de façon **équivalente**

$$\hat{\beta}^R = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^d x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^d \beta_j^2 \right\}. \quad (2)$$

Quelques remarques

- Les définitions (1) et (2) sont **équivalentes** dans le sens où pour tout t il existe un unique λ tels que les solutions aux deux problèmes d'optimisation **coïncident**.

Quelques remarques

- Les définitions (1) et (2) sont **équivalentes** dans le sens où pour tout t il existe un unique λ tels que les solutions aux deux problèmes d'optimisation **coïncident**.
- La **constante** β_0 n'entre généralement **pas** dans la **pénalité**.

Quelques remarques

- Les définitions (1) et (2) sont **équivalentes** dans le sens où pour tout t il existe un unique λ tels que les solutions aux deux problèmes d'optimisation **coïncident**.
- La **constante** β_0 n'entre généralement **pas** dans la **pénalité**.
- L'estimateur **dépend** bien entendu du paramètre t (ou λ) :
$$\hat{\beta}^R = \hat{\beta}^R(t) = \hat{\beta}^R(\lambda).$$

Quelques remarques

- Les définitions (1) et (2) sont **équivalentes** dans le sens où pour tout t il existe un unique λ tels que les solutions aux deux problèmes d'optimisation **coïncident**.
- La **constante** β_0 n'entre généralement **pas** dans la **pénalité**.
- L'estimateur **dépend** bien entendu du paramètre t (ou λ) :
$$\hat{\beta}^R = \hat{\beta}^R(t) = \hat{\beta}^R(\lambda).$$
- Les variables explicatives sont le plus souvent **réduites** pour **éviter les problèmes d'échelle** dans la pénalité.

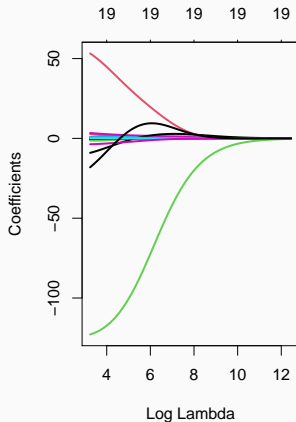
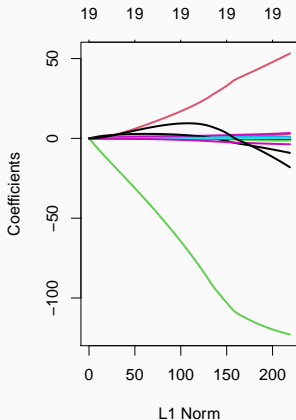
Exemple avec les données Hitters

- Il existe **plusieurs fonctions et packages** qui permettent de faire de la régression pénalisée sur **R**. Nous présentons ici **glmnet**.
- **glmnet** n'accepte pas d'objet **formule**. Il faut spécifier la **matrice** des X et le **vecteur** des Y :

```
> Hitters.X <- model.matrix(Salary~.,data=Hitters)[,-1]
```

Ridge avec glmnet

```
> library(glmnet)
> reg.ridge <- glmnet(Hitters.X,Hitters$Salary,alpha=0)
> par(mfrow=c(1,2))
> plot(reg.ridge,lwd=2)
> plot(reg.ridge,lwd=2,xvar="lambda")
```



Propriétés des estimateurs ridge

Propriétés

1. Lorsque les variables explicatives sont **centrée-réduites**, l'estimateur Ridge solution de (2) s'écrit

$$\hat{\beta}^R = \hat{\beta}^R(\lambda) = (\mathbb{X}^t\mathbb{X} + \lambda\mathbb{I})^{-1}\mathbb{X}^t\mathbb{Y}.$$

2. On déduit

$$\text{biais}(\hat{\beta}^R) = -\lambda(\mathbb{X}^t\mathbb{X} + \lambda\mathbb{I})^{-1}\beta$$

et

$$V(\hat{\beta}^R) = \sigma^2(\mathbb{X}^t\mathbb{X} + \lambda\mathbb{I})^{-1}\mathbb{X}^t\mathbb{X}(\mathbb{X}^t\mathbb{X} + \lambda\mathbb{I})^{-1}.$$

Propriétés des estimateurs ridge

Propriétés

1. Lorsque les variables explicatives sont **centrée-réduites**, l'estimateur Ridge solution de (2) s'écrit

$$\hat{\beta}^R = \hat{\beta}^R(\lambda) = (\mathbb{X}^t\mathbb{X} + \lambda\mathbb{I})^{-1}\mathbb{X}^t\mathbb{Y}.$$

2. On déduit

$$\text{biais}(\hat{\beta}^R) = -\lambda(\mathbb{X}^t\mathbb{X} + \lambda\mathbb{I})^{-1}\beta$$

et

$$V(\hat{\beta}^R) = \sigma^2(\mathbb{X}^t\mathbb{X} + \lambda\mathbb{I})^{-1}\mathbb{X}^t\mathbb{X}(\mathbb{X}^t\mathbb{X} + \lambda\mathbb{I})^{-1}.$$

Commentaires

- Si $\lambda = 0$, on retrouve le biais et la variance de l'estimateur des **MCO**.
- $\lambda \nearrow \implies$ biais \nearrow et variance \searrow et réciproquement lorsque $\lambda \searrow$.

- Il est **crucial** : si $\lambda \approx 0$ alors $\hat{\beta}^R \approx \hat{\beta}^{MCO}$, si λ "grand" alors $\hat{\beta}^R \approx 0$.

- Il est **crucial** : si $\lambda \approx 0$ alors $\hat{\beta}^R \approx \hat{\beta}^{MCO}$, si λ "grand" alors $\hat{\beta}^R \approx 0$.
- Le choix de λ se fait le plus souvent de façon "classique" :
 1. **Estimation d'un critère** de choix de modèle pour toutes les valeurs de λ ;

- Il est **crucial** : si $\lambda \approx 0$ alors $\hat{\beta}^R \approx \hat{\beta}^{MCO}$, si λ "grand" alors $\hat{\beta}^R \approx 0$.
- Le choix de λ se fait le plus souvent de façon "classique" :
 1. **Estimation d'un critère** de choix de modèle pour toutes les valeurs de λ ;
 2. Choix du λ qui **minimise** le critère estimé.

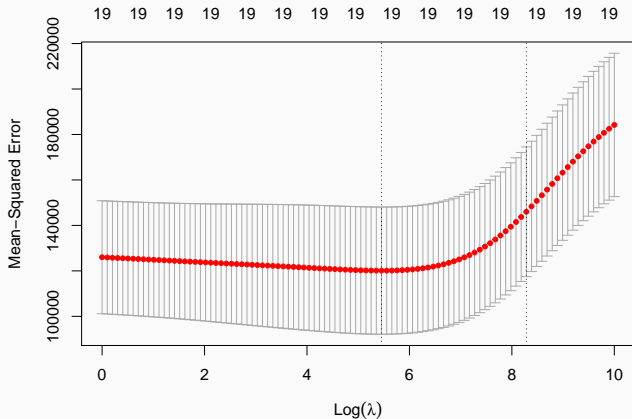
Choix de λ

- Il est **crucial** : si $\lambda \approx 0$ alors $\hat{\beta}^R \approx \hat{\beta}^{MCO}$, si λ "grand" alors $\hat{\beta}^R \approx 0$.
- Le choix de λ se fait le plus souvent de façon "classique" :
 1. **Estimation d'un critère** de choix de modèle pour toutes les valeurs de λ ;
 2. Choix du λ qui **minimise** le critère estimé.
- **Exemple** : la fonction `cv.glmnet` choisit la valeur de λ qui minimise l'**erreur quadratique moyenne**

$$E[(Y - m_{\hat{\beta}^R(\lambda)}(X))^2]$$

estimée par **validation croisée**.

```
> set.seed(321)
> reg.cvridge <- cv.glmnet(Hitters.X,Hitters$Salary,alpha=0,
+                          lambda=exp(seq(0,10,length=100)))
> bestlam <- reg.cvridge$lambda.min
> bestlam
## [1] 233.8186
> plot(reg.cvridge)
```



Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variante de ridge/lasso

Discrimination binaire

Bibliographie

- La **régression lasso** consiste à minimiser le critère des moindres carrés pénalisé par la norme 1 des coefficients.

Définition [**Tibshirani, 1996**]

1. Les **estimateurs lasso** $\hat{\beta}^L$ s'obtiennent en minimisant

$$\sum_{i=1}^n \left(Y_i - \beta_0 - \sum_{j=1}^d X_{ij} \beta_j \right)^2 \quad \text{sous la contrainte} \quad \sum_{j=1}^d |\beta_j| \leq t \quad (3)$$

2. ou de façon **équivalente**

$$\hat{\beta}^L = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^n \left(Y_i - \beta_0 - \sum_{j=1}^d X_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^d |\beta_j| \right\}. \quad (4)$$

Comparaison Ridge-Lasso

- Dans le cas où la matrice \mathbb{X} est **orthonormée**, on a une **écriture explicite** pour les estimateurs ridge et lasso.

Comparaison Ridge-Lasso

- Dans le cas où la matrice \mathbb{X} est **orthonormée**, on a une **écriture explicite** pour les estimateurs ridge et lasso.

Propriété

Si la matrice de design \mathbb{X} est orthonormée, alors

$$\hat{\beta}_j^R = \frac{\hat{\beta}_j}{1 + \lambda} \quad \text{et} \quad \hat{\beta}_j^L = \text{signe}(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+$$

où $\hat{\beta}_j$ est l'estimateur MCO de β_j .

Comparaison Ridge-Lasso

- Dans le cas où la matrice \mathbb{X} est **orthonormée**, on a une **écriture explicite** pour les estimateurs ridge et lasso.

Propriété

Si la matrice de design \mathbb{X} est orthonormée, alors

$$\hat{\beta}_j^R = \frac{\hat{\beta}_j}{1 + \lambda} \quad \text{et} \quad \hat{\beta}_j^L = \text{signe}(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+$$

où $\hat{\beta}_j$ est l'estimateur MCO de β_j .

Commentaires

- **Ridge** "diminue" l'estimateur MCO de façon **proportionnelle** ;

Comparaison Ridge-Lasso

- Dans le cas où la matrice \mathbb{X} est **orthonormée**, on a une **écriture explicite** pour les estimateurs ridge et lasso.

Propriété

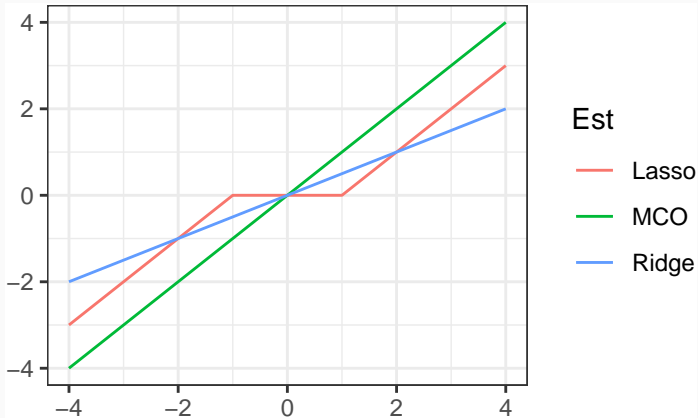
Si la matrice de design \mathbb{X} est orthonormée, alors

$$\hat{\beta}_j^R = \frac{\hat{\beta}_j}{1 + \lambda} \quad \text{et} \quad \hat{\beta}_j^L = \text{signe}(\hat{\beta}_j)(|\hat{\beta}_j| - \lambda)_+$$

où $\hat{\beta}_j$ est l'estimateur MCO de β_j .

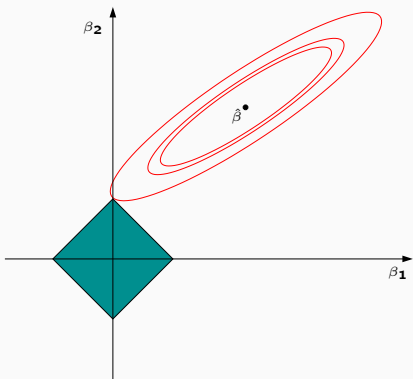
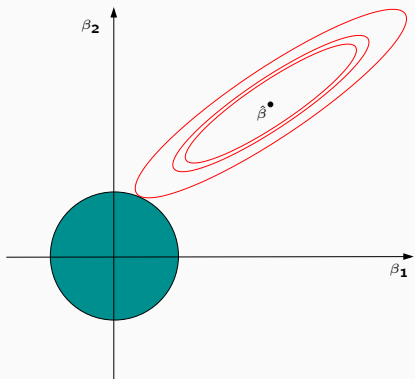
Commentaires

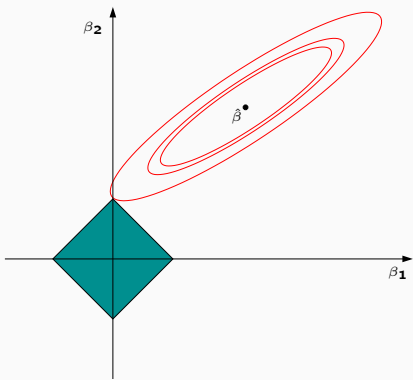
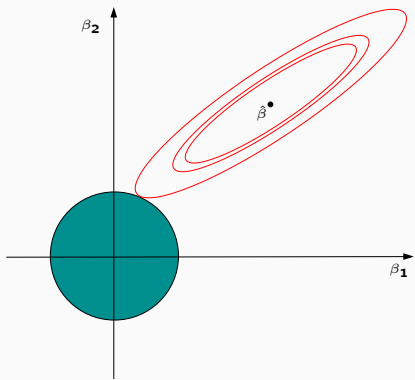
- **Ridge** "diminue" l'estimateur MCO de façon **proportionnelle** ;
- **Lasso** **translate et tronque** l'estimateur MCO (lorsque ce dernier est petit).



Conclusion

Le lasso va avoir tendance à "mettre" des coefficients à 0 et donc à faire de la sélection de variables.





Remarque

Ces approches reviennent (d'une certaine façon) à projeter l'estimateur des MCO sur les boules unités associées à

1. la norme 2 pour la régression ridge ;
2. la norme 1 pour le lasso.

Quelques remarques

- Comme pour la régression ridge :
 - on préfère souvent **réduire la matrice de design** avant d'effectuer la régression lasso ;

Quelques remarques

- Comme pour la régression ridge :
 - on préfère souvent **réduire la matrice de design** avant d'effectuer la régression lasso ;
 - Le choix de λ est **crucial** (il est le plus souvent sélectionné en minimisant un critère empirique).

Quelques remarques

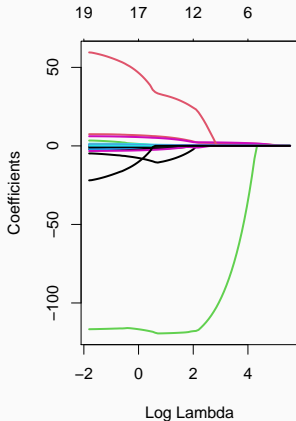
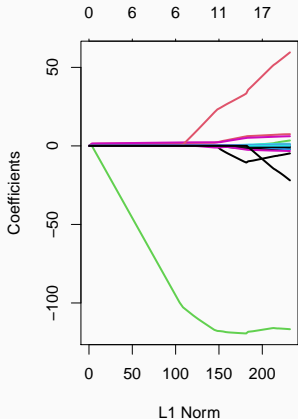
- Comme pour la régression ridge :
 - on préfère souvent **réduire la matrice de design** avant d'effectuer la régression lasso ;
 - Le choix de λ est **crucial** (il est le plus souvent sélectionné en minimisant un critère empirique).
 - $\lambda \nearrow \implies$ biais \nearrow et variance \searrow et réciproquement lorsque $\lambda \searrow$.

Quelques remarques

- Comme pour la régression ridge :
 - on préfère souvent **réduire la matrice de design** avant d'effectuer la régression lasso ;
 - Le choix de λ est **crucial** (il est le plus souvent sélectionné en minimisant un critère empirique).
 - $\lambda \nearrow \implies$ biais \nearrow et variance \searrow et réciproquement lorsque $\lambda \searrow$.
- **MAIS**, contrairement à ridge : $\lambda \nearrow \implies$ **le nombre de coefficients nuls augmente** ([Bühlmann and van de Geer, 2011]).

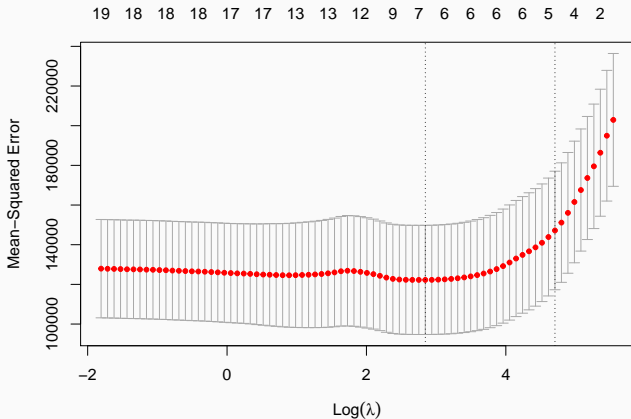
Le coin R

```
> reg.lasso <- glmnet(Hitters.X,Hitters$Salary,alpha=1)
> par(mfrow=c(1,2))
> plot(reg.lasso,lwd=2)
> plot(reg.lasso,lwd=2,xvar="lambda")
```



Sélection de λ

```
> set.seed(321)
> reg.cvlasso <- cv.glmnet(Hitters.X,Hitters$Salary,alpha=1)
> bestlam <- reg.cvlasso$lambda.min
> bestlam
## [1] 17.19108
> plot(reg.cvlasso)
```



- Il existe plusieurs façons de résoudre le problème numérique d'optimisation lasso (ou ridge).
- Un des plus utilisés est l'algorithme de descente de coordonnées [Hastie et al., 2015].

- Il existe plusieurs façons de résoudre le problème numérique d'optimisation lasso (ou ridge).
- Un des plus utilisés est l'algorithme de descente de coordonnées [Hastie et al., 2015].
- On considère le problème lasso

$$\hat{\beta}^L = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^n \left(Y_i - \beta_0 - \sum_{j=1}^d X_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^d |\beta_j| \right\}$$

avec les variables explicatives centrées-réduites (pour simplifier).

1. **Initialisation** : $\hat{\beta}_0 = \bar{y}$, $\hat{\beta}_j = \dots, j = 1, \dots, d$.
2. Répéter jusqu'à convergence :
Pour $j = 1, \dots, d$:
 - 2.1 Calculer les **résidus partiels** $r_i^{(j)} = y_i - \sum_{k \neq j} x_{ik} \hat{\beta}_k$;
 - 2.2 Faire la **régression simple** des y_i contre $r_i^{(j)} \implies \tilde{\beta}_j$;
 - 2.3 **Mettre à jour** $\hat{\beta}_j = \text{signe}(\tilde{\beta}_j)(|\tilde{\beta}_j| - \lambda)_+$
3. **Retourner** : $\hat{\beta}_j, j = 1, \dots, d$.

Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variantes de ridge/lasso

Discrimination binaire

Bibliographie

Différentes pénalités

- Les approches **ridge** et **lasso** diffèrent uniquement au niveau de la **pénalité** ajoutée au critère des moindres carrés.
- **Norme 2** pour **ridge** et **norme 1** pour le **lasso**.

Différentes pénalités

- Les approches **ridge** et **lasso** diffèrent uniquement au niveau de la **pénalité** ajoutée au critère des moindres carrés.
- **Norme 2** pour **ridge** et **norme 1** pour le **lasso**.
- Il existe tout un tas d'**autres stratégies** de pénalisations.
- Nous en présentons quelques unes dans cette partie.
- On pourra consulter [[Hastie et al., 2015](#)] pour plus de détails.

- [Zou and Hastie, 2005] ont proposé de combiner les approches ridge et lasso en proposant une pénalité (appelée elastic net) de la forme

$$\lambda \sum_{j=1}^d ((1 - \alpha)\beta_j^2 + \alpha|\beta_j|)$$

où $\alpha \in [0, 1]$.

- [Zou and Hastie, 2005] ont proposé de combiner les approches ridge et lasso en proposant une pénalité (appelée elastic net) de la forme

$$\lambda \sum_{j=1}^d ((1 - \alpha)\beta_j^2 + \alpha|\beta_j|)$$

où $\alpha \in [0, 1]$.

- Le paramètre α définit le compromis ridge/lasso :
 - $\alpha = 1 \implies$ Lasso ;
 - $\alpha = 0 \implies$ Ridge ;

- [Zou and Hastie, 2005] ont proposé de combiner les approches ridge et lasso en proposant une pénalité (appelée elastic net) de la forme

$$\lambda \sum_{j=1}^d ((1 - \alpha)\beta_j^2 + \alpha|\beta_j|)$$

où $\alpha \in [0, 1]$.

- Le paramètre α définit le compromis ridge/lasso :
 - $\alpha = 1 \implies$ Lasso ;
 - $\alpha = 0 \implies$ Ridge ;
 - Ce paramètre correspond (évidemment) à l'argument **alpha** de la fonction `glmnet`.

- [Zou and Hastie, 2005] ont proposé de combiner les approches ridge et lasso en proposant une pénalité (appelée elastic net) de la forme

$$\lambda \sum_{j=1}^d ((1 - \alpha)\beta_j^2 + \alpha|\beta_j|)$$

où $\alpha \in [0, 1]$.

- Le paramètre α définit le compromis ridge/lasso :
 - $\alpha = 1 \implies$ Lasso ;
 - $\alpha = 0 \implies$ Ridge ;
 - Ce paramètre correspond (évidemment) à l'argument **alpha** de la fonction `glmnet`.
- **Avantage** : on a plus de flexibilité car la pénalité elastic net propose une gamme de modèles beaucoup plus large que lasso et ridge ;

- [Zou and Hastie, 2005] ont proposé de combiner les approches ridge et lasso en proposant une pénalité (appelée elastic net) de la forme

$$\lambda \sum_{j=1}^d ((1 - \alpha)\beta_j^2 + \alpha|\beta_j|)$$

où $\alpha \in [0, 1]$.

- Le paramètre α définit le compromis ridge/lasso :
 - $\alpha = 1 \implies$ Lasso ;
 - $\alpha = 0 \implies$ Ridge ;
 - Ce paramètre correspond (évidemment) à l'argument **alpha** de la fonction `glmnet`.
- **Avantage** : on a plus de flexibilité car la pénalité elastic net propose une gamme de modèles beaucoup plus large que lasso et ridge ;
- **Inconvénient** : en plus du λ il faut aussi sélectionner le α !

- Dans certaines applications, les variables explicatives appartiennent à des groupes de variables prédéfinis.

Group Lasso

- Dans certaines applications, les variables explicatives appartiennent à des groupes de variables prédéfinis.
- Nécessité de "shrinker" ou sélectionner les variables par groupe.

Group Lasso

- Dans certaines applications, les variables explicatives appartiennent à des groupes de variables prédéfinis.
- Nécessité de "shrinker" ou sélectionner les variables par groupe.

Exemple : variables qualitatives

- 2 variables explicatives qualitatives X_1 et X_2 et une variable explicative continue X_3 .
- Le modèle s'écrit

$$Y = \beta_0 + \beta_1 \mathbf{1}_{X_1=A} + \beta_2 \mathbf{1}_{X_1=B} + \beta_3 \mathbf{1}_{X_1=C} \\ + \beta_4 \mathbf{1}_{X_2=D} + \beta_5 \mathbf{1}_{X_2=E} + \beta_6 \mathbf{1}_{X_2=F} + \beta_7 \mathbf{1}_{X_2=G} + \beta_8 X_3 + \varepsilon$$

muni des contraintes $\beta_1 = \beta_4 = 0$.

Group Lasso

- Dans certaines applications, les variables **explicatives** appartiennent à des **groupes de variables** prédéfinis.
- Nécessité de "**shrinker**" ou **sélectionner** les variables **par groupe**.

Exemple : variables qualitatives

- 2 variables explicatives qualitatives X_1 et X_2 et une variable explicative continue X_3 .
- Le **modèle** s'écrit

$$Y = \beta_0 + \beta_1 1_{X_1=A} + \beta_2 1_{X_1=B} + \beta_3 1_{X_1=C} \\ + \beta_4 1_{X_2=D} + \beta_5 1_{X_2=E} + \beta_6 1_{X_2=F} + \beta_7 1_{X_2=G} + \beta_8 X_3 + \varepsilon$$

muni des contraintes $\beta_1 = \beta_4 = 0$.

- 3 groupes : $X_1 = (1_{X_1=B}, 1_{X_1=C})$, $X_2 = (1_{X_2=E}, 1_{X_2=F}, 1_{X_2=G})$ et $X_3 = X_3$.

Définition

En présence de d variables réparties en L groupes X_1, \dots, X_L de cardinal d_1, \dots, d_L . On note $\beta_\ell, \ell = 1, \dots, L$ le vecteur des coefficients associé au groupe X_ℓ . Les **estimateurs group-lasso** s'obtiennent en minimisant le critère

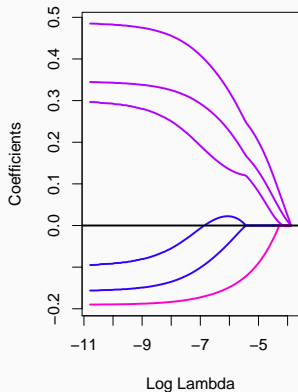
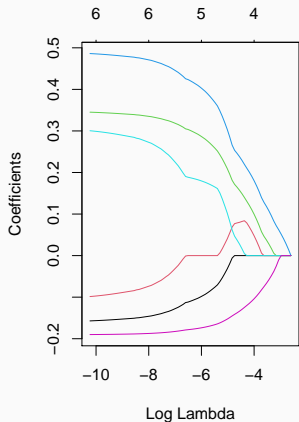
$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{\ell=1}^L X_{i\ell} \beta_\ell \right)^2 + \lambda \sum_{\ell=1}^L \sqrt{d_\ell} \|\beta_\ell\|_2$$

Remarque

Puisque $\|\beta_\ell\|_2 = 0$ ssi $\beta_{\ell 1} = \dots = \beta_{\ell d_\ell} = 0$, cette procédure encourage la **mise à zéro** des coefficients d'un **même groupe**.

- La fonction `gglasso` du package `gglasso` permet de faire du **groupe lasso** sur R.

```
> summary(donnees)
##           X1                X2                X3                Y
## Length:200      Length:200      Min.   :0.009496  Min.   : -3.23315
## Class :character  Class :character  1st Qu.:0.237935  1st Qu.: -0.50404
## Mode  :character  Mode  :character  Median :0.485563  Median :  0.16759
##                                     Mean  :0.483286  Mean   :  0.09792
##                                     3rd Qu.:0.734949  3rd Qu.:  0.66918
##                                     Max.   :0.998741  Max.   :  3.04377
> D <- model.matrix(Y~.,data=donnees)[,-1]
> model <- glmnet(D,Y,alpha=1)
> groupe <- c(1,1,2,2,2,3)
> library(gglasso)
> model1 <- gglasso(D,Y,group=groupe)
> plot(model1)
```



Remarque

Les coefficients **s'annulent par groupe** lorsque λ augmente (graphe de droite).

Sparse group lasso

- La **norme 2** de la pénalité group-lasso implique que, généralement, tous les coefficients d'un groupe sont **tous nuls** ou **tous non nuls**.
- Dans certains cas, il peut être intéressant de mettre de la **sparsité** dans les groupes aussi. Comment ?

Sparse group lasso

- La **norme 2** de la pénalité group-lasso implique que, généralement, tous les coefficients d'un groupe sont **tous nuls** ou **tous non nuls**.
- Dans certains cas, il peut être intéressant de mettre de la **sparsité** dans les groupes aussi. Comment ?
- En ajoutant **la norme 1** dans la pénalité.

Sparse group lasso

- La **norme 2** de la pénalité group-lasso implique que, généralement, tous les coefficients d'un groupe sont **tous nuls** ou **tous non nuls**.
- Dans certains cas, il peut être intéressant de mettre de la **sparsité** dans les groupes aussi. Comment ?
- En ajoutant **la norme 1** dans la pénalité.

Pénalité sparse group lasso

$$\lambda \sum_{\ell=1}^L [(1 - \alpha) \|\beta_{\ell}\|_2 + \alpha \|\beta_{\ell}\|_1].$$

- Sur **R** : package **SGL**.

Fused lasso

- Utile pour prendre en compte la **spatialité des données**.
- **Idée** : deux coefficients successifs doivent être proches.

Pénalité fused lasso

$$\lambda_1 \sum_{j=1}^d |\beta_j|$$

Fused lasso

- Utile pour prendre en compte la **spatialité des données**.
- **Idée** : deux coefficients successifs doivent être proches.

Pénalité fused lasso

$$\lambda_1 \sum_{j=1}^d |\beta_j| + \lambda_2 \sum_{j=2}^d |\beta_{j+1} - \beta_j|$$

qui peut se re-paramétriser en

$$\lambda \sum_{j=2}^d |\beta_{j+1} - \beta_j|.$$

- Sur **R** : package **genlasso**.

Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variantes de ridge/lasso

Discrimination binaire

Bibliographie

- Les méthodes **ridge et lasso** ont été présentées dans un cadre de régression linéaire.
- Ces techniques d'adaptent directement à la **régression logistique** $\mathcal{Y} = \{-1, 1\}$.

Discrimination binaire

- Les méthodes **ridge et lasso** ont été présentées dans un cadre de régression linéaire.
- Ces techniques d'adaptent directement à la **régression logistique** $\mathcal{Y} = \{-1, 1\}$.
- Les **pénalités** sont **identiques**.
- **Seul changement** : le critère moindre carré est remplacé par la déviance \implies ce qui revient à **minimiser l'opposé de la vraisemblance plus la pénalité**.

Définition

On note $\tilde{y}_i = (y_i + 1)/2$.

- On appelle **estimateur ridge** en régression logistique l'estimateur

$$\hat{\beta}^R = \operatorname{argmin}_{\beta} \left\{ - \sum_{i=1}^n (\tilde{y}_i x_i^t \beta - \log(1 + \exp(x_i^t \beta))) + \lambda \sum_{j=1}^d \beta_j^2 \right\}.$$

- On appelle **estimateur lasso** en régression logistique l'estimateur

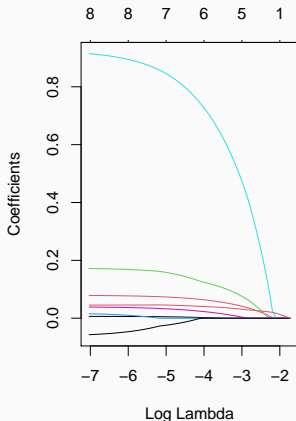
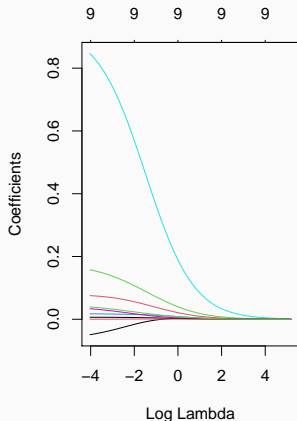
$$\hat{\beta}^L = \operatorname{argmin}_{\beta} \left\{ - \sum_{i=1}^n (\tilde{y}_i x_i^t \beta - \log(1 + \exp(x_i^t \beta))) + \lambda \sum_{j=1}^d |\beta_j| \right\}.$$

- Pour faire du ridge ou lasso en logistique, il suffit d'ajouter l'argument `family=binomial` dans `glmnet`.
- Tout reste identique pour le reste (tracé du chemin des coefficients, choix du λ ...).
- Exemple : données SAheart

```
> head(SAheart)
##   sbp tobacco   ldl adiposity famhist typea obesity alcohol age chd
## 1 160   12.00 5.73   23.11 Present   49   25.30   97.20 52  1
## 2 144    0.01 4.41   28.61  Absent   55   28.87    2.06 63  1
## 3 118    0.08 3.48   32.28 Present   52   29.14    3.81 46  0
## 4 170    7.50 6.41   38.03 Present   51   31.99   24.26 58  1
## 5 134   13.60 3.50   27.78 Present   60   25.99   57.34 49  1
## 6 132    6.20 6.47   36.21 Present   62   30.77   14.14 45  0
```

- On obtient les chemins de régularisation ridge et lasso avec les commandes suivantes :

```
> SAheart.X <- model.matrix(chd~.,data=SAheart)
> log.ridge <- glmnet(SAheart.X,SAheart$chd,family="binomial",alpha=0)
> log.lasso <- glmnet(SAheart.X,SAheart$chd,family="binomial",alpha=1)
> plot(log.ridge,xvar="lambda")
```



Le modèle de régression linéaire

Estimateurs des moindres carrés

Résidus

Le modèle logistique

Sélection de variables

Régularisation

Régression ridge

Régression Lasso

Variante de ridge/lasso

Discrimination binaire

Bibliographie



Aronszajn, N. (1950).

Theory of reproducing kernels.

Transactions of the American Mathematical Society, 68 :337–404.



Bühlmann, P. and van de Geer, S. (2011).

Statistics for high-dimensional data.

Springer.



Cornillon, P., Hengartner, N., Matzner-Løber, E., and Rouvière, L. (2019).

Régression avec R.

EDP Sciences.



Fromont, M. (2015).

Apprentissage statistique.

Université Rennes 2, diapos de cours.



Hastie, T., Tibshirani, R., and Friedman, J. (2009).

The Elements of Statistical Learning : Data Mining, Inference, and Prediction.

Springer, second edition.






Hastie, T., Tibshirani, R., and Wainwright, M. (2015).

Statistical Learning with Sparsity : The Lasso and Generalizations.

CRC Press.

https:

[//web.stanford.edu/~hastie/StatLearnSparsity_files/SLS.pdf](https://web.stanford.edu/~hastie/StatLearnSparsity_files/SLS.pdf).

-  Karatzoglou, A., Smola, A., Hornik, K., and Zeileis, A. (2004).
kernlab – an s4 package for kernel methods in r.
Journal of Statistical Software, 11(9).
-  Tibshirani, R. (1996).
Regression shrinkage and selection via the lasso.
Journal of the Royal Statistical Society, Series B, 58 :267–288.
-  Zou, H. and Hastie, T. (2005).
Regularization and variable selection via the elastic net.
Journal of the Royal Statistical Society, Series B, 67 :301–320.

Troisième partie III

Algorithmes non linéaires

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- Algorithmes **linéaires** :

$$f(x) = f_{\beta}(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

- **Problème** : tous les problèmes ne sont pas linéaires.

- Algorithmes **linéaires** :

$$f(x) = f_{\beta}(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

- **Problème** : tous les problèmes ne sont pas linéaires.
- Possible d'**ajouter de la non linéarité** dans les algorithmes linéaires : effets quadratiques, interaction...

- Algorithmes **linéaires** :

$$f(x) = f_{\beta}(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

- **Problème** : tous les problèmes ne sont pas linéaires.
- Possible d'**ajouter de la non linéarité** dans les algorithmes linéaires : effets quadratiques, interaction...
- **Difficile pour l'utilisateur** de trouver quels effets ajouter ! Surtout lorsque d est grand.

- Algorithmes **linéaires** :

$$f(x) = f_{\beta}(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_d x_d.$$

- **Problème** : tous les problèmes ne sont pas linéaires.
- Possible d'**ajouter de la non linéarité** dans les algorithmes linéaires : effets quadratiques, interaction...
- **Difficile pour l'utilisateur** de trouver quels effets ajouter ! Surtout lorsque d est grand.

Dans cette partie

Présentation de quelques algorithmes **non linéaires** :

- Méthodes par **arbres**.
- **Réseaux de neurones**.

Arbres binaires

Choix des coupures

- Cas de la régression

- Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

- Algorithme

- Choix des paramètres

- Erreur OOB et importance des variables

Bibliographie

- Les arbres sont des algorithmes de prédiction qui fonctionnent en régression et en discrimination.
- Il existe différentes variantes permettant de construire des prédicteurs par arbres.
- Nous nous focalisons dans cette partie sur la méthode CART [Breiman et al., 1984] qui est la plus utilisée.

Arbres binaires

Choix des coupures

- Cas de la régression

- Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

- Algorithme

- Choix des paramètres

- Erreur OOB et importance des variables

Bibliographie

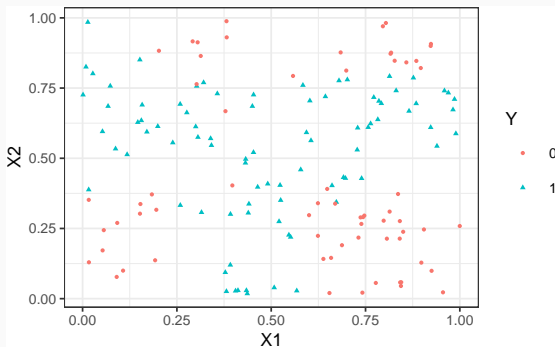
- On cherche à expliquer une variable Y par d variables explicatives X_1, \dots, X_d .

- On cherche à expliquer une variable Y par d variables explicatives X_1, \dots, X_d .
- Y peut admettre un nombre quelconque de modalités et les variables X_1, \dots, X_d peuvent être qualitatives et/ou quantitatives.

- On cherche à expliquer une variable Y par d variables explicatives X_1, \dots, X_d .
- Y peut admettre un nombre quelconque de modalités et les variables X_1, \dots, X_d peuvent être qualitatives et/ou quantitatives.
- Néanmoins, pour simplifier on se place dans un premier temps en discrimination binaire : Y admet 2 modalités (-1 ou 1). On suppose de plus que l'on a simplement 2 variables explicatives quantitatives.

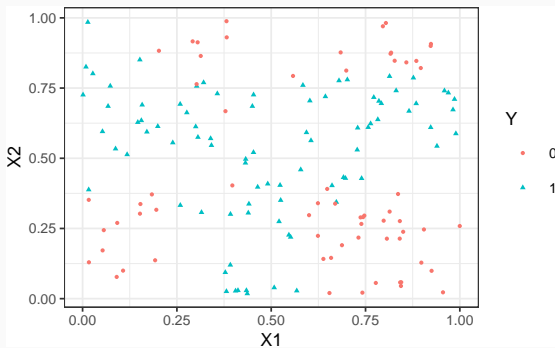
Représentation des données

- On dispose de n observations $(x_1, y_1), \dots, (x_n, y_n)$ où $x_i \in \mathbb{R}^2$ et $y_i \in \{0, 1\}$.



Représentation des données

- On dispose de n observations $(x_1, y_1), \dots, (x_n, y_n)$ où $x_i \in \mathbb{R}^2$ et $y_i \in \{0, 1\}$.

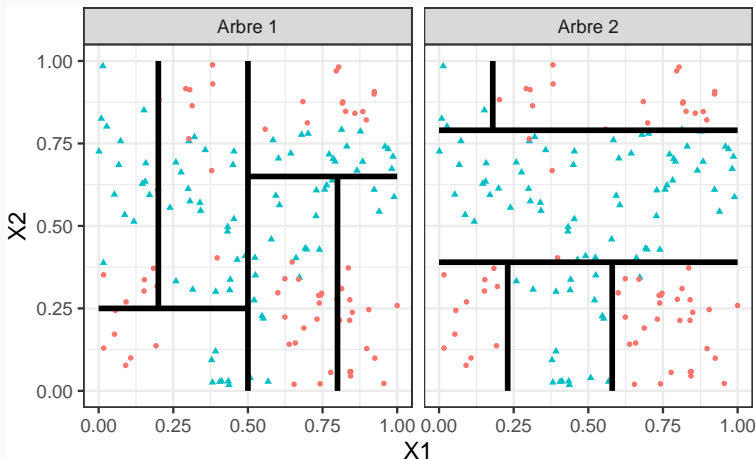


Approche par arbres

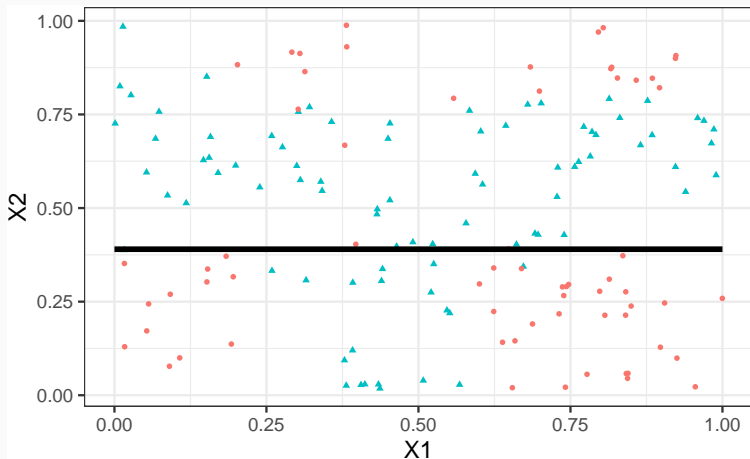
Trouver une **partition** des observations qui **sépare** "au mieux" les points rouges des points bleus.

Arbres binaires

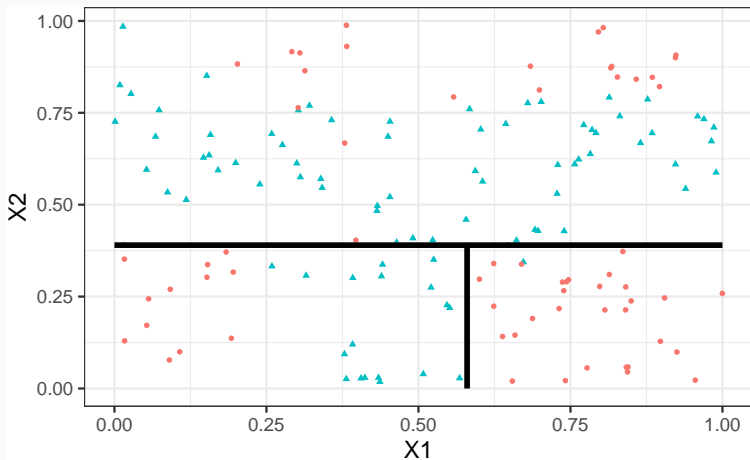
- La **méthode CART** propose de construire une partition basée sur des divisions **successives parallèles aux axes**.
- 2 exemples de partition :



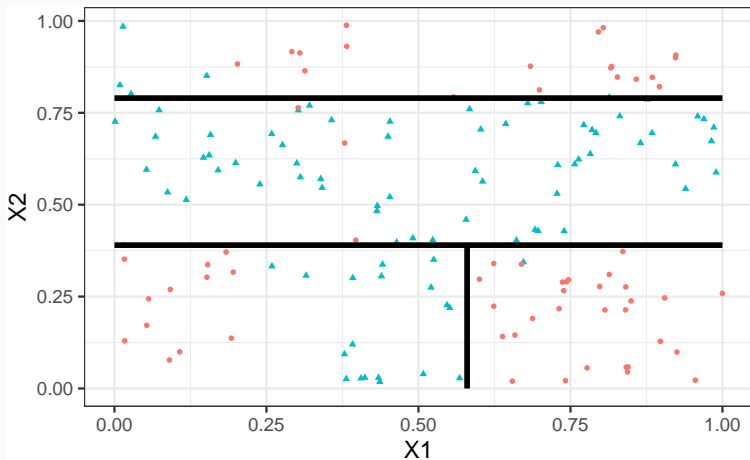
- A chaque étape, la méthode cherche une **nouvelle division** : une **variable** et un **seuil** de coupure.



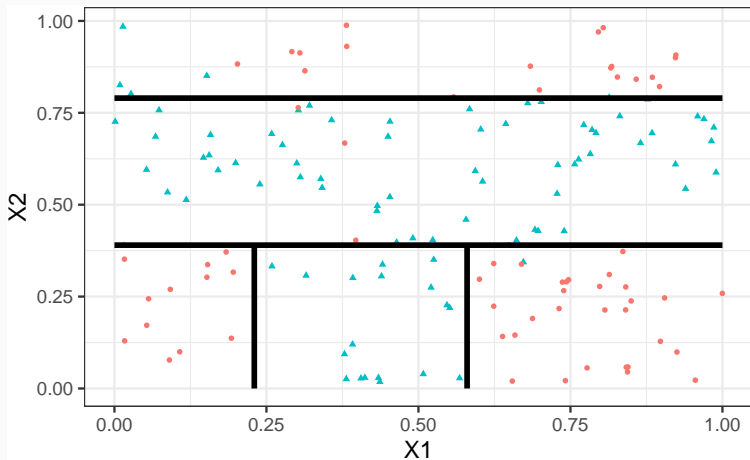
- A chaque étape, la méthode cherche une **nouvelle division** : une **variable** et un **seuil** de coupure.



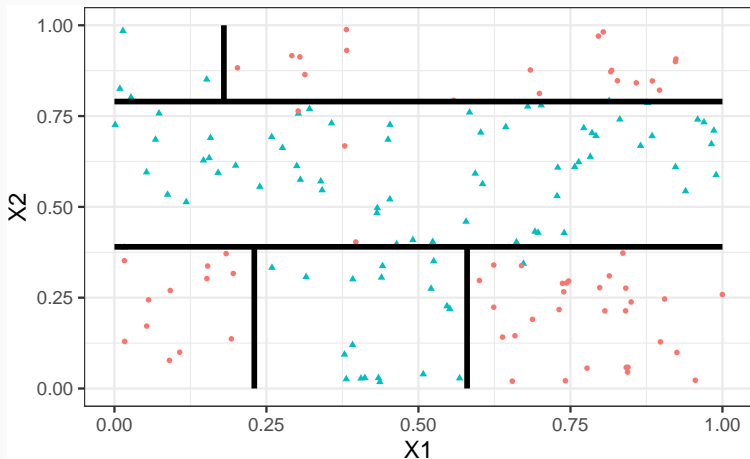
- A chaque étape, la méthode cherche une **nouvelle division** : une **variable** et un **seuil** de coupure.



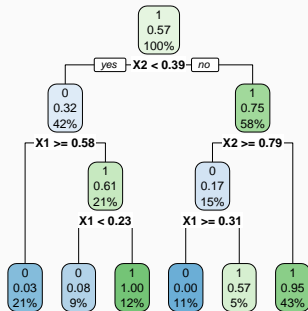
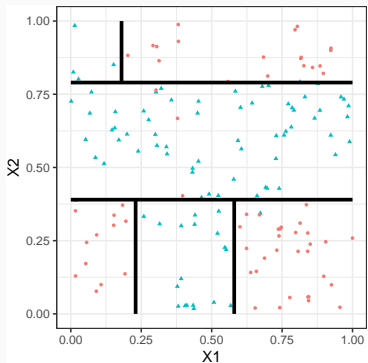
- A chaque étape, la méthode cherche une **nouvelle division** : une **variable** et un **seuil** de coupure.



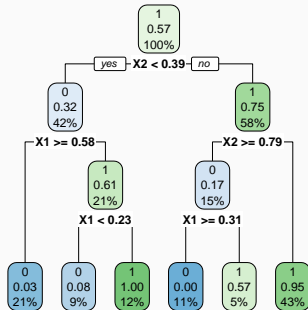
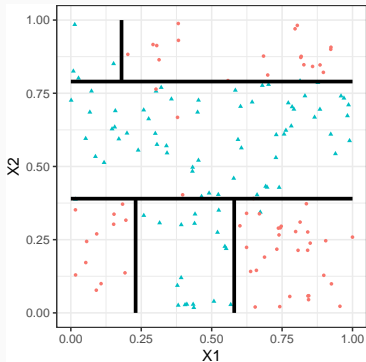
- A chaque étape, la méthode cherche une **nouvelle division** : une **variable** et un **seuil** de coupure.



Représentation de l'arbre



Représentation de l'arbre



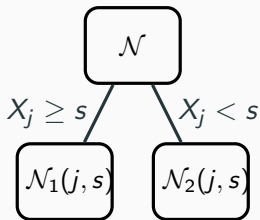
Remarque

Visuel de **droite** plus pertinent :

- Plus d'information.
- Généralisation à plus de deux dimensions.

Vocabulaire

- Chaque coupure divise une partie de \mathbb{R}^d en deux parties appelées **nœuds**.
- Le premier nœud, qui contient toutes les observations, est le **nœud racine**.
- Une coupure divise un nœud en deux **nœuds fils** :



- Les nœuds qui ne sont pas découpés (en bas de l'arbre) sont les **nœuds terminaux** ou **feuilles** de l'arbre.

Arbre et algorithme de prévision

- L'arbre construit, les **prévisions** se déduisent à partir de **moyennes faites dans les feuilles**.
- On note $\mathcal{N}(x)$ la feuille de l'arbre qui contient $x \in \mathbb{R}^d$, les prévisions s'obtiennent selon :
 1. **Régression** \implies moyenne des y_i de la feuille

$$m_n(x) = \frac{1}{|\mathcal{N}(x)|} \sum_{i: x_i \in \mathcal{N}(x)} y_i$$

Arbre et algorithme de prévision

- L'arbre construit, les **prévisions** se déduisent à partir de **moyennes faites dans les feuilles**.
- On note $\mathcal{N}(x)$ la feuille de l'arbre qui contient $x \in \mathbb{R}^d$, les prévisions s'obtiennent selon :

1. **Régression** \implies moyenne des y_i de la feuille

$$m_n(x) = \frac{1}{|\mathcal{N}(x)|} \sum_{i: x_i \in \mathcal{N}(x)} y_i$$

2. **Classification (classe)** \implies vote à la majorité :

$$g_n(x) = \operatorname{argmax}_k \sum_{i: x_i \in \mathcal{N}(x)} 1_{y_i=k}$$

3. **Classification (proba)** \implies proportion d'obs. du groupe k :

$$S_{k,n}(x) = \frac{1}{|\mathcal{N}(x)|} \sum_{i: x_i \in \mathcal{N}(x)} 1_{y_i=k}$$

1. Comment **découper** un nœud ?

⇒ si on dispose d'un algorithme pour découper un nœud, il suffira de le répéter.

1. Comment **découper** un nœud ?

⇒ si on dispose d'un algorithme pour découper un nœud, il suffira de le répéter.

2. Comment choisir la **profondeur de l'arbre** ?

- Profondeur **maximale** ? (on découpe jusqu'à ne plus pouvoir)

1. Comment **découper** un nœud ?

⇒ si on dispose d'un algorithme pour découper un nœud, il suffira de le répéter.

2. Comment choisir la **profondeur de l'arbre** ?

- Profondeur **maximale** ? (on découpe jusqu'à ne plus pouvoir)
sur-ajustement ?
- Critère d'arrêt ?

1. Comment **découper** un nœud ?

⇒ si on dispose d'un algorithme pour découper un nœud, il suffira de le répéter.

2. Comment choisir la **profondeur de l'arbre** ?

- Profondeur **maximale** ? (on découpe jusqu'à ne plus pouvoir **sur-ajustement** ?
- Critère d'arrêt ?
- Élagage ? (on construit un arbre profond et on enlève des branches "inutiles" ...).

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- Une coupure = un couple $(j, s) \in \{1, \dots, d\} \times \mathbb{R}$.
- Idée : définir un critère mesure la performance d'une coupure et choisir celle qui optimise le critère.

- Une coupure = un couple $(j, s) \in \{1, \dots, d\} \times \mathbb{R}$.
- Idée : définir un critère mesure la performance d'une coupure et choisir celle qui optimise le critère.
- Coupure performante \implies les deux nœuds fils sont homogènes vis-à-vis de Y .

- Une coupure = un couple $(j, s) \in \{1, \dots, d\} \times \mathbb{R}$.
- Idée : définir un critère mesure la performance d'une coupure et choisir celle qui optimise le critère.
- Coupure performante \implies les deux nœuds fils sont homogènes vis-à-vis de Y .

Fonction d'impureté

- Objectif : mesurer l'homogénéité d'un nœud.

- Une coupure = un couple $(j, s) \in \{1, \dots, d\} \times \mathbb{R}$.
- Idée : définir un critère mesure la performance d'une coupure et choisir celle qui optimise le critère.
- Coupure performante \implies les deux nœuds fils sont homogènes vis-à-vis de Y .

Fonction d'impureté

- Objectif : mesurer l'homogénéité d'un nœud.
- Intérêt : choisir la coupure qui maximise la pureté des nœuds fils.

- L'impureté \mathcal{I} d'un nœud doit être :
 1. faible lorsque un nœud est homogène : les valeurs de Y dans le nœud sont proches.
 2. élevée lorsque un nœud est hétérogène : les valeurs de Y dans le nœud sont dispersées.

Critère de découpe

- L'**impureté** \mathcal{I} d'un nœud doit être :
 1. **faible** lorsque un nœud est homogène : les valeurs de Y dans le nœud sont **proches**.
 2. **élevée** lorsque un nœud est hétérogène : les valeurs de Y dans le nœud sont **dispersées**.

L'idée

Une fois \mathcal{I} définie, on choisira le couple (j, s) qui **maximise le gain d'impureté** :

$$\Delta(j, s) = p(\mathcal{N})\mathcal{I}(\mathcal{N}) - (p(\mathcal{N}_1(j, s))\mathcal{I}(\mathcal{N}_1(j, s)) + p(\mathcal{N}_2(j, s))\mathcal{I}(\mathcal{N}_2(j, s)))$$

où $p(\mathcal{N})$ représente la proportion d'observations dans le nœud \mathcal{N} .

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- Une mesure naturelle de l'**impureté** d'un nœud \mathcal{N} en **régression** est la **variance** du nœud :

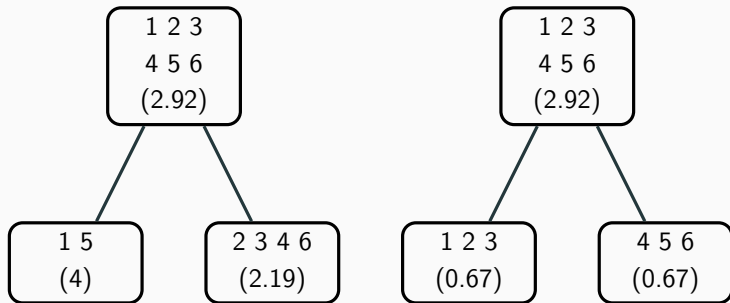
$$\mathcal{I}(\mathcal{N}) = \frac{1}{|\mathcal{N}|} \sum_{i: x_i \in \mathcal{N}} (y_i - \bar{y}_{\mathcal{N}})^2,$$

où $\bar{y}_{\mathcal{N}}$ désigne la moyenne des Y_i dans \mathcal{N} .

- Une mesure naturelle de l'**impureté** d'un nœud \mathcal{N} en **régression** est la **variance** du nœud :

$$\mathcal{I}(\mathcal{N}) = \frac{1}{|\mathcal{N}|} \sum_{i:x_i \in \mathcal{N}} (y_i - \bar{y}_{\mathcal{N}})^2,$$

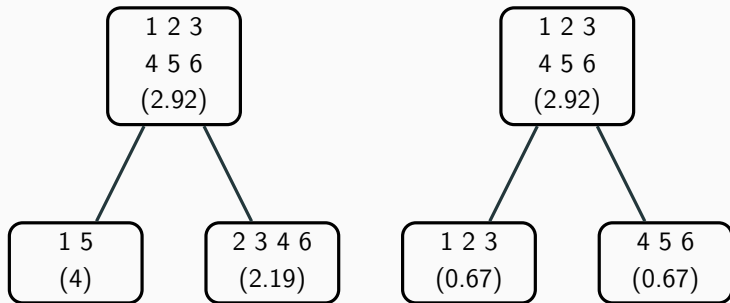
où $\bar{y}_{\mathcal{N}}$ désigne la moyenne des Y_i dans \mathcal{N} .



- Une mesure naturelle de l'**impureté** d'un nœud \mathcal{N} en **régression** est la **variance** du nœud :

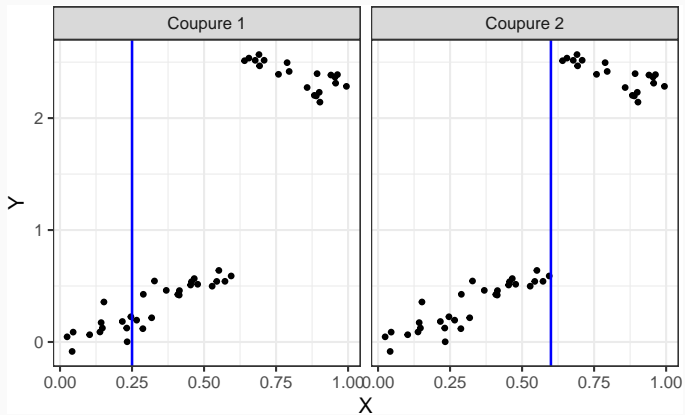
$$\mathcal{I}(\mathcal{N}) = \frac{1}{|\mathcal{N}|} \sum_{i: x_i \in \mathcal{N}} (y_i - \bar{y}_{\mathcal{N}})^2,$$

où $\bar{y}_{\mathcal{N}}$ désigne la moyenne des Y_i dans \mathcal{N} .

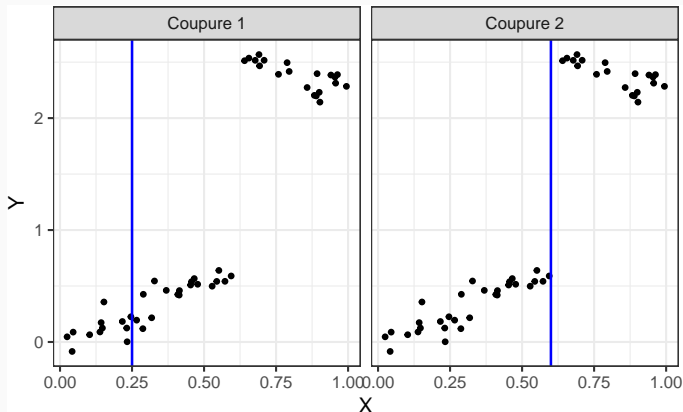


⇒ coupure de **droite** plus performante.

Exemple



Exemple



	$\mathcal{I}(\mathcal{N})$	$\mathcal{I}(\mathcal{N}_1)$	$\mathcal{I}(\mathcal{N}_2)$	Δ
Gauche	1.05	0.01	0.94	0.34
Droite	1.05	0.04	0.01	1.02

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- Les $Y_i, i = 1, \dots, n$ sont à valeurs dans $\{1, \dots, K\}$.

- Les $Y_i, i = 1, \dots, n$ sont à valeurs dans $\{1, \dots, K\}$.
- On cherche une fonction \mathcal{I} telle que $\mathcal{I}(\mathcal{N})$ soit
 - **petite** si un **label majoritaire** se distingue clairement dans \mathcal{N} ;
 - **grande** sinon.

- Les $Y_i, i = 1, \dots, n$ sont à valeurs dans $\{1, \dots, K\}$.
- On cherche une fonction \mathcal{I} telle que $\mathcal{I}(\mathcal{N})$ soit
 - **petite** si un **label majoritaire** se distingue clairement dans \mathcal{N} ;
 - **grande** sinon.

Impureté

L'**impureté** d'un nœud \mathcal{N} en classification se mesure selon

$$\mathcal{I}(\mathcal{N}) = \sum_{j=1}^K f(p_j(\mathcal{N}))$$

où

- $p_j(\mathcal{N})$ représente la proportion d'observations de la classe j dans le nœud \mathcal{N} .
- f est une fonction (concave) $[0, 1] \rightarrow \mathbb{R}^+$ telle que $f(0) = f(1) = 0$.

Exemples de fonctions f

- Si \mathcal{N} est pur, on veut $\mathcal{I}(\mathcal{N}) = 0$

Exemples de fonctions f

- Si \mathcal{N} est pur, on veut $\mathcal{I}(\mathcal{N}) = 0 \implies$ c'est pourquoi f doit vérifier $f(0) = f(1) = 0$.

Exemples de fonctions f

- Si \mathcal{N} est pur, on veut $\mathcal{I}(\mathcal{N}) = 0 \implies$ c'est pourquoi f doit vérifier $f(0) = f(1) = 0$.
- Les 2 mesures d'impureté les plus classiques sont :
 1. **Gini** : $f(p) = p(1 - p)$;
 2. **Information** : $f(p) = -p \log(p)$.

Exemples de fonctions f

- Si \mathcal{N} est pur, on veut $\mathcal{I}(\mathcal{N}) = 0 \implies$ c'est pourquoi f doit vérifier $f(0) = f(1) = 0$.
- Les 2 mesures d'impureté les plus classiques sont :
 1. **Gini** : $f(p) = p(1 - p)$;
 2. **Information** : $f(p) = -p \log(p)$.

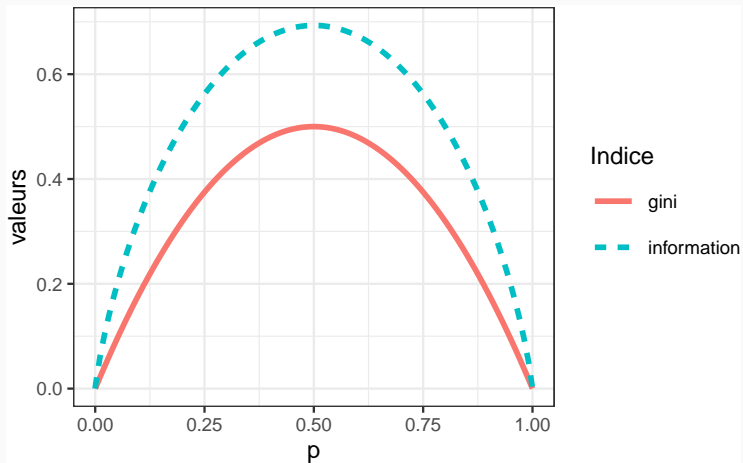
Cas binaire

Dans ce cas on a

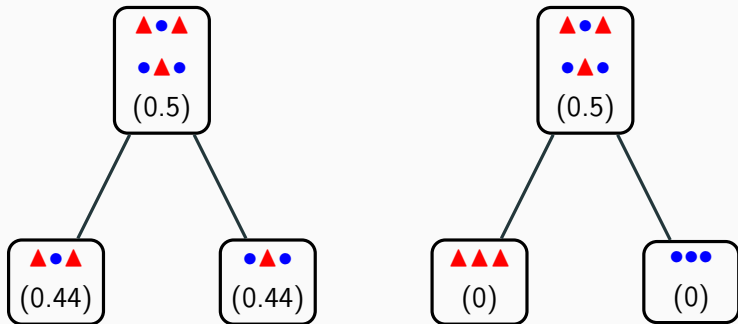
1. $\mathcal{I}(\mathcal{N}) = 2p(1 - p)$ pour **Gini**
2. $\mathcal{I}(\mathcal{N}) = -p \log p - (1 - p) \log(1 - p)$ pour **Information**

où p désigne la proportion de 1 (ou 0) dans \mathcal{N} .

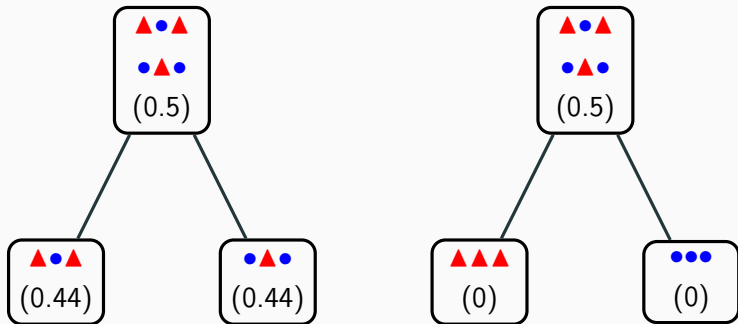
Impureté dans le cas binaire



Example 1

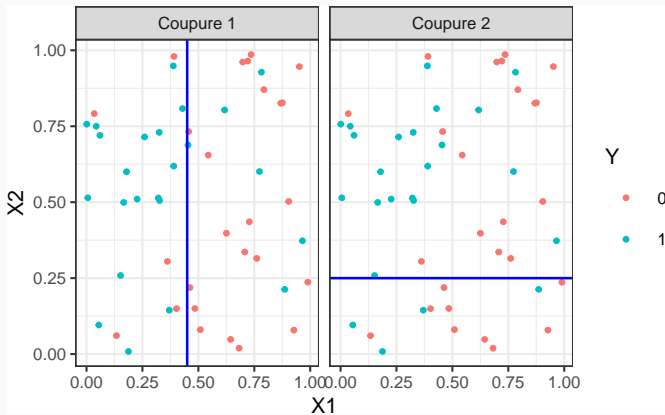


Exemple 1

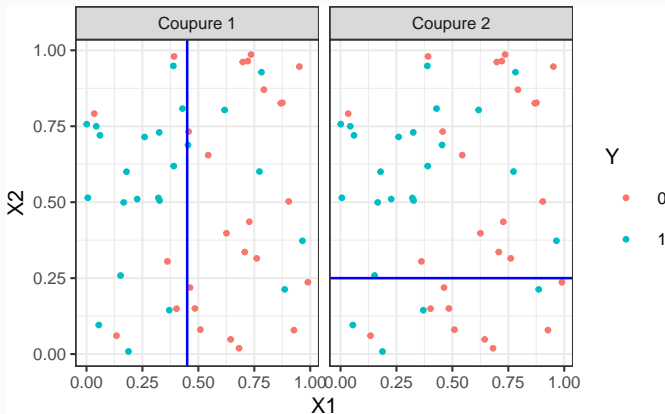


⇒ coupure de **droite** plus performante.

Exemple 2



Exemple 2



	$\mathcal{I}(\mathcal{N})$	$\mathcal{I}(\mathcal{N}_1)$	$\mathcal{I}(\mathcal{N}_2)$	Δ
Gauche	0.50	0.34	0.35	0.16
Droite	0.50	0.43	0.50	0.02

- Arbres binaires

- Choix des coupures

 - Cas de la régression

 - Cas de la classification supervisée

- Elagage

- Importance des variables

- Bagging et forêts aléatoires

 - Bagging

 - Forêts aléatoires

 - Algorithme

 - Choix des paramètres

 - Erreur OOB et importance des variables

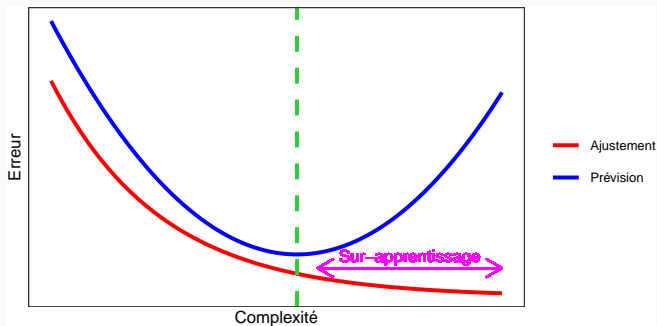
- Bibliographie

Pourquoi élaguer ?

- Les coupures permettent de **séparer les données selon Y**
⇒ plus on coupe mieux on ajuste !

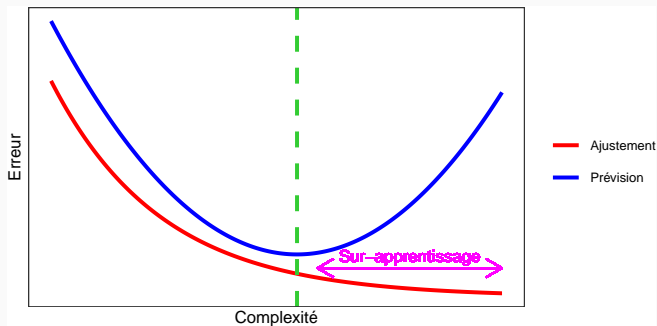
Pourquoi élaguer ?

- Les coupures permettent de **séparer les données selon Y**
⇒ plus on coupe mieux on ajuste !
- Risque de **sur-ajustement** si on coupe trop !



Pourquoi élaguer ?

- Les coupures permettent de **séparer les données selon Y**
⇒ plus on coupe mieux on ajuste !
- Risque de **sur-ajustement** si on coupe trop !



Complexité d'un arbre

Représentée par son **nombre de coupures** ou sa **profondeur**.

Comment faire ?

- Tester tous les arbres ?

Comment faire ?

- Tester tous les arbres ?
⇒ possible uniquement sur de petits échantillons !
- Critère d'arrêt : ne plus découper si une certaine condition est vérifiée.

Comment faire ?

- **Tester tous les arbres ?**
⇒ possible uniquement sur de petits échantillons !
- **Critère d'arrêt** : ne plus découper si une certaine condition est vérifiée.
⇒ possible mais... une coupure peut ne pas être pertinente alors que des **coupures plus basses** le seront !

Comment faire ?

- Tester tous les arbres ?
⇒ possible uniquement sur de petits échantillons !
- Critère d'arrêt : ne plus découper si une certaine condition est vérifiée.
⇒ possible mais... une coupure peut ne pas être pertinente alors que des coupures plus basses le seront !

Élaguer

1. Considérer un arbre (trop) profond ⇒ qui sur-ajuste ;
2. Supprimer les branches peu utiles.

- Tester **tous les sous-arbres** d'un arbre très profond se révèlent souvent **trop coûteux** en temps de calcul.

Élagage CART

- Tester **tous les sous-arbres** d'un arbre très profond se révèlent souvent **trop coûteux** en temps de calcul.
- [Breiman et al., 1984] propose une stratégie d'élagage qui permet de se ramener à **une suite d'arbres emboîtés**

$$\mathcal{T}_{max} = \mathcal{T}_0 \supset \mathcal{T}_1 \supset \dots \supset \mathcal{T}_K.$$

de **taille raisonnable** (plus petite que n).

Élagage CART

- Tester **tous les sous-arbres** d'un arbre très profond se révèlent souvent **trop coûteux** en temps de calcul.
- [Breiman et al., 1984] propose une stratégie d'élagage qui permet de se ramener à **une suite d'arbres emboîtés**

$$\mathcal{T}_{max} = \mathcal{T}_0 \supset \mathcal{T}_1 \supset \dots \supset \mathcal{T}_K.$$

de **taille raisonnable** (plus petite que n).

- Il est ensuite possible de **choisir un arbre dans cette suite** par des méthodes traditionnelles :
 1. choix d'un risque ;
 2. optimisation de ce risque (par validation croisée par exemple).

Construction de la suite de sous arbres

- Soit T un arbre à $|T|$ nœuds terminaux $\mathcal{N}_1, \dots, \mathcal{N}_{|T|}$.
- Soit $R(\mathcal{N})$ un risque (d'ajustement) dans le nœud \mathcal{N} :
 - **Régression** :

$$R_m(T) = \frac{1}{N_m} \sum_{i: x_i \in \mathcal{N}_m} (y_i - \bar{y}_{\mathcal{N}_m})^2$$

- **Classification** :

$$R_m(T) = \frac{1}{N_m} \sum_{i: x_i \in \mathcal{N}_m} 1_{y_i \neq y_{\mathcal{N}_m}}$$

Construction de la suite de sous arbres

- Soit T un arbre à $|T|$ nœuds terminaux $\mathcal{N}_1, \dots, \mathcal{N}_{|T|}$.
- Soit $R(\mathcal{N})$ un risque (d'ajustement) dans le nœud \mathcal{N} :
 - **Régression** :

$$R_m(T) = \frac{1}{N_m} \sum_{i: x_i \in \mathcal{N}_m} (y_i - \bar{y}_{\mathcal{N}_m})^2$$

- **Classification** :

$$R_m(T) = \frac{1}{N_m} \sum_{i: x_i \in \mathcal{N}_m} 1_{y_i \neq y_{\mathcal{N}_m}}$$

Définition

Soit $\alpha \geq 0$, le critère **coût/complexité** est défini par :

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m R_m(T) + \alpha |T|.$$

Idée

- $C_\alpha(T)$ est un critère qui prend en compte l'adéquation d'un arbre et sa complexité.
- L'idée est de chercher un arbre T_α qui minimise $C_\alpha(T)$ pour une valeur de α bien choisie.

Idée

- $C_\alpha(T)$ est un critère qui prend en compte l'adéquation d'un arbre et sa complexité.
- L'idée est de chercher un arbre T_α qui minimise $C_\alpha(T)$ pour une valeur de α bien choisie.

Remarque

- $\alpha = 0 \implies T_\alpha = T_0 = T_{\max}$.
- $\alpha = +\infty \implies T_\alpha = T_{+\infty} = T_{\text{root}}$ arbre sans coupure.

Idée

- $C_\alpha(T)$ est un critère qui prend en compte l'adéquation d'un arbre et sa complexité.
- L'idée est de chercher un arbre T_α qui minimise $C_\alpha(T)$ pour une valeur de α bien choisie.

Remarque

- $\alpha = 0 \implies T_\alpha = T_0 = T_{\max}$.
- $\alpha = +\infty \implies T_\alpha = T_{+\infty} = T_{\text{root}}$ arbre sans coupure.

Question (a priori difficile)

Comment calculer T_α qui minimise $C_\alpha(T)$?

Lemme 1

Si T_1 et T_2 sont deux sous-arbres de T_{\max} avec $R_\alpha(T_1) = R_\alpha(T_2)$. Alors $T_1 \subset T_2$ ou $T_2 \subset T_1$

\implies garantit une unique solution de **taille minimale**.

Deux lemmes

Lemme 1

Si T_1 et T_2 sont deux sous-arbres de T_{\max} avec $R_\alpha(T_1) = R_\alpha(T_2)$. Alors $T_1 \subset T_2$ ou $T_2 \subset T_1$

\implies garantit une unique solution de **taille minimale**.

Lemme 2

Si $\alpha > \alpha'$ alors $T_\alpha = T_{\alpha'}$ ou $T_\alpha \subset T_{\alpha'}$.

\implies garantit une **stabilité des solutions** lorsque α parcourt \mathbb{R}^+ \implies elles vont être **emboîtées** les unes dans les autres.

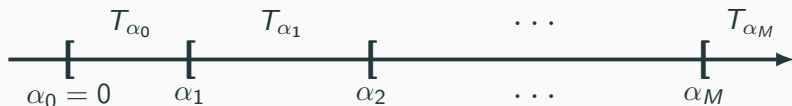
Théorème [Breiman et al., 1984]

Il existe une suite finie $\alpha_0 = 0 < \alpha_1 < \dots < \alpha_M$ avec $M \leq |T_{\max}|$ et une suite associée d'arbres emboîtés $(T_{\alpha_m})_m$

$$T_{\max} = T_{\alpha_0} \supset T_{\alpha_1} \supset \dots \supset T_{\alpha_M} = T_{\text{root}}$$

telle que $\forall \alpha \in [\alpha_m, \alpha_{m+1}[$

$$T_m \in \underset{T \subseteq T_{\max}}{\operatorname{argmin}} C_\alpha(T).$$



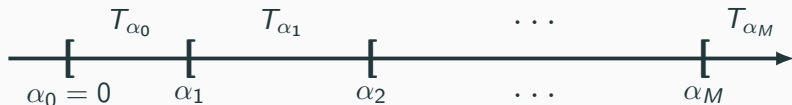
Théorème [Breiman et al., 1984]

Il existe une suite finie $\alpha_0 = 0 < \alpha_1 < \dots < \alpha_M$ avec $M \leq |T_{\max}|$ et une suite associée d'arbres emboîtés $(T_{\alpha_m})_m$

$$T_{\max} = T_{\alpha_0} \supset T_{\alpha_1} \supset \dots \supset T_{\alpha_M} = T_{\text{root}}$$

telle que $\forall \alpha \in [\alpha_m, \alpha_{m+1}[$

$$T_m \in \underset{T \subseteq T_{\max}}{\operatorname{argmin}} C_\alpha(T).$$



Commentaires

- Nombre de minimiseurs de $C_\alpha(T)$ est "petit".
- Ils s'obtiennent en **élaguant** : en supprimant des branches.

Exemple

- On visualise la suite de sous-arbres avec la fonction `printcp` ou dans l'objet `rpart` :

```
> library(rpart)
> set.seed(123)
> arbre <- rpart(Y~.,data=don.2D.arbre,cp=0.0001,minsplit=2)
> arbre$cptable
```

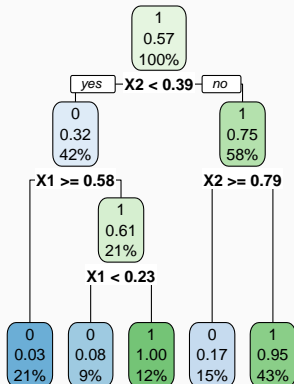
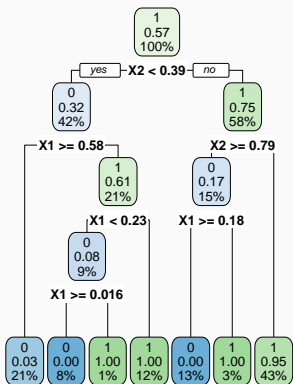
##	CP	nsplit	rel error	xerror	xstd
## 1	0.353846154	0	1.00000000	1.00000000	0.09336996
## 2	0.230769231	1	0.64615385	0.7076923	0.08688336
## 3	0.138461538	2	0.41538462	0.5076923	0.07805324
## 4	0.061538462	4	0.13846154	0.2153846	0.05481185
## 5	0.015384615	5	0.07692308	0.1846154	0.05111769
## 6	0.007692308	6	0.06153846	0.2461538	0.05816388
## 7	0.000100000	14	0.00000000	0.2153846	0.05481185

- Suite de 7 arbres emboîtés.
- CP : complexity parameter, il mesure la complexité de l'arbre : CP $\searrow \implies$ complexité \nearrow .
- nsplit : nombre de coupures de l'arbre.
- rel.error : erreur (normalisée) calculée sur les données d'apprentissage \implies erreur d'ajustement.
- xerror : erreur (normalisée) calculée par validation croisée 10 blocs \implies erreur de prévision (voir diapos suivantes).
- xstd : écart-type associé à l'erreur de validation croisée.

Visualisation

- On peut les visualiser en combinant `prune` (extraction) et `rpart.plot` (tracé) :

```
> arbre1 <- prune(arbre, cp=0.01)
> arbre2 <- prune(arbre, cp=0.1)
> library(rpart.plot)
> rpart.plot(arbre1); rpart.plot(arbre2)
```



Choix de l'arbre final

- Choisir un arbre dans la suite revient à choisir une valeur de α .

Choix de l'arbre final

- Choisir un arbre dans la suite revient à **choisir une valeur de α** .
- Ce choix s'effectue généralement de façon classique :
 1. Choix d'un **risque**.
 2. **Estimation** du risque par **ré-échantillonnage** (CV par exemple) pour tous les α_m .
 3. **Sélection** du α_m qui **minimise** le risque estimé.

Choix de l'arbre final

- Choisir un arbre dans la suite revient à **choisir une valeur de α** .
- Ce choix s'effectue généralement de façon classique :
 1. Choix d'un **risque**.
 2. **Estimation** du risque par **ré-échantillonnage** (CV par exemple) pour tous les α_m .
 3. **Sélection** du α_m qui **minimise** le risque estimé.

Remarque

La fonction `rpart` effectue par défaut une validation croisée 10 blocs en prenant :

- le **risque quadratique** en régression.
- l'**erreur de classification** en classification.

1. Calculer

$$\beta_0 = 0, \quad \beta_1 = \sqrt{\alpha_1 \alpha_2}, \quad \dots \quad \beta_{M-1} = \sqrt{\alpha_{M-1} \alpha_M}, \quad \beta_M = +\infty.$$

2. Pour $k = 1, \dots, K$

2.1 Construire l'arbre maximal sur l'ensemble des données privé du k^e bloc, c'est-à-dire $\mathcal{B}^{-k} = \{(x_i, y_i) : i \in \{1, \dots, n\} \setminus B_k\}$.

2.2 Appliquer l'algorithme d'élagage à cet arbre maximal, puis extraire les arbres qui correspondent aux valeurs $\beta_m, m = 0, \dots, M \implies T_{\beta_m}(\cdot, \mathcal{B}^{-k})$.

2.3 Calculer les valeurs prédites par chaque arbre sur le bloc k : $T_{\beta_m}(x_i, \mathcal{B}^{-k}), i \in B_k$.

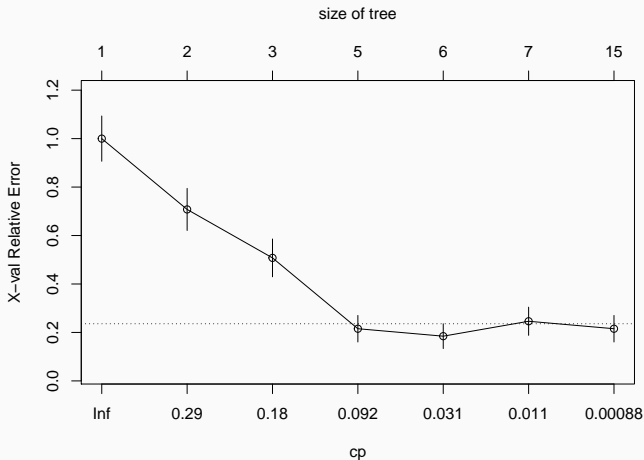
3. En déduire les erreurs pour chaque β_m :

$$\widehat{\mathcal{R}}(\beta_m) = \frac{1}{n} \sum_{k=1}^K \sum_{i \in B_k} \ell(y_i, T_{\beta_m}(x_i, \mathcal{B}^{-k})).$$

Retourner : une valeur α_m telle que $\widehat{\mathcal{R}}(\beta_m)$ est minimum.

- Les erreurs de validation croisée se trouvent dans la colonne `xerror` de l'élément `cptable`.
- On peut les visualiser avec `plotcp` :

```
> plotcp(arbre)
```



- Il reste à choisir l'arbre qui minimise l'erreur de prévision :

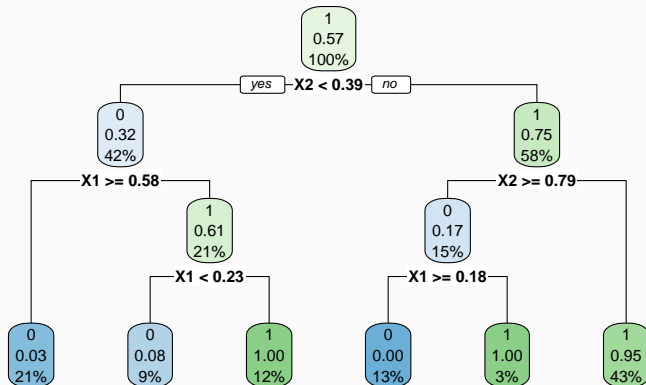
```
> cp_opt <- as_tibble(arbre$cptable) %>% arrange(xerror) %>%  
+ slice(1) %>% select(CP) %>% as.numeric()  
> cp_opt  
## [1] 0.01538462
```

- Il reste à choisir l'arbre qui **minimise l'erreur de prévision** :

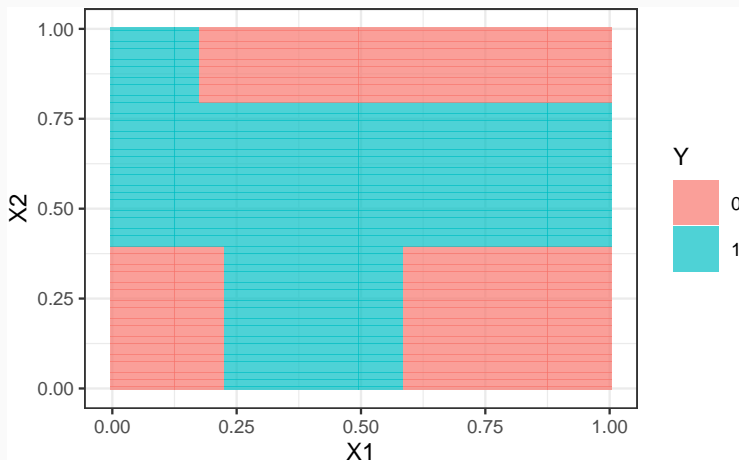
```
> cp_opt <- as_tibble(arbre$cptable) %>% arrange(xerror) %>%  
+ slice(1) %>% select(CP) %>% as.numeric()  
> cp_opt  
## [1] 0.01538462
```

- et à le visualiser :

```
> arbre_final <- prune(arbre, cp=cp_opt)  
> rpart.plot(arbre_final)
```



- 2 variables explicatives \implies on peut visualiser l'arbre final
- en coloriant le carré $[0, 1]^2$ en fonction des valeurs prédites.



- Nouvel individu :

```
> xnew <- tibble(X1=0.4,X2=0.5)
```

- Nouvel individu :

```
> xnew <- tibble(X1=0.4,X2=0.5)
```

- Prévision de la classe :

```
> predict(arbre_final,newdata=xnew,type="class")  
## 1  
## 1  
## Levels: 0 1
```

- Nouvel individu :

```
> xnew <- tibble(X1=0.4,X2=0.5)
```

- Prévision de la classe :

```
> predict(arbre_final,newdata=xnew,type="class")  
## 1  
## 1  
## Levels: 0 1
```

- Prévision des probabilités :

```
> predict(arbre_final,newdata=xnew,type="prob")  
##           0           1  
## 1 0.046875 0.953125
```

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- La **visualisation de l'arbre** peut donner une idée sur l'**importance des variables** dans l'algorithme.
- **Pas suffisant !** Il se peut en effet que des variables possèdent une grande importance sans pour autant apparaître explicitement dans l'arbre !

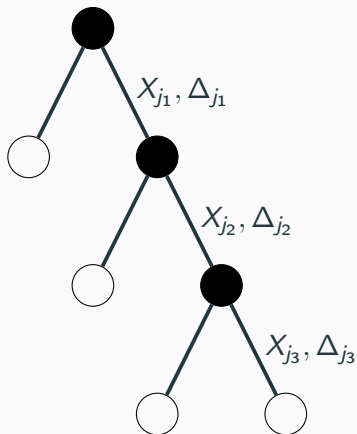
- La **visualisation de l'arbre** peut donner une idée sur l'**importance des variables** dans l'algorithme.
- **Pas suffisant !** Il se peut en effet que des variables possèdent une grande importance sans pour autant apparaître explicitement dans l'arbre !
 - Difficile de **quantifier l'importance** juste en regardant l'arbre !
 - Il se peut en effet que des variables possèdent une grande importance **sans pour autant apparaître en haut** de l'arbre !

- La **visualisation de l'arbre** peut donner une idée sur l'**importance des variables** dans l'algorithme.
- **Pas suffisant !** Il se peut en effet que des variables possèdent une grande importance sans pour autant apparaître explicitement dans l'arbre !
 - Difficile de **quantifier l'importance** juste en regardant l'arbre !
 - Il se peut en effet que des variables possèdent une grande importance **sans pour autant apparaître en haut** de l'arbre !

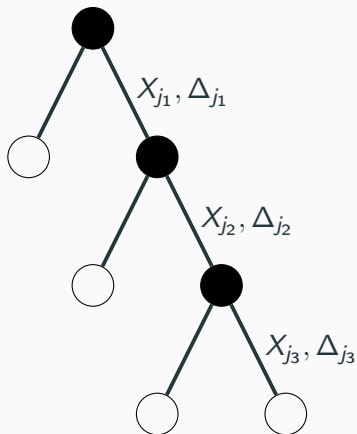
Mesure d'importance d'un arbre

Basée sur le **gain d'impureté** des nœuds internes.

- Nœuds internes $\implies N_t, t = 1, \dots, J - 1;$
- Variables de coupure $\implies X_{j_t};$
- Gain d'impureté $\implies i_{j_t}^2.$

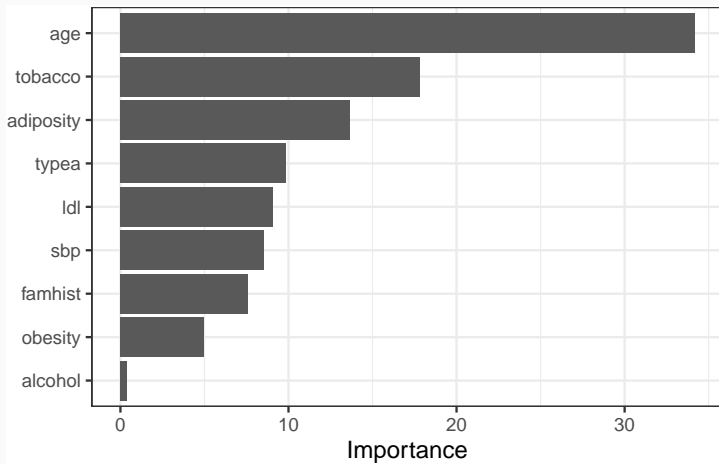


- Nœuds internes $\implies N_t, t = 1, \dots, J - 1;$
- Variables de coupure $\implies X_{j_t};$
- Gain d'impureté $\implies i_{j_t}^2.$



Mesure d'impureté de la variable ℓ

$$\mathcal{I}_\ell(T) = \sum_{t=1}^{|T|-1} \Delta_t 1_{j_t=\ell}.$$



1. Avantages :

- Méthode « simple » relativement facile à mettre en œuvre.
- Fonctionne en régression et en classification.
- Résultats interprétables (à condition que l'arbre ne soit pas trop profond).

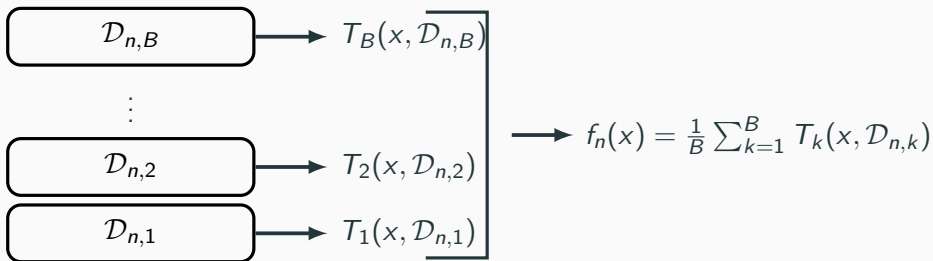
1. Avantages :

- Méthode « simple » relativement facile à mettre en œuvre.
- Fonctionne en régression et en classification.
- Résultats interprétables (à condition que l'arbre ne soit pas trop profond).

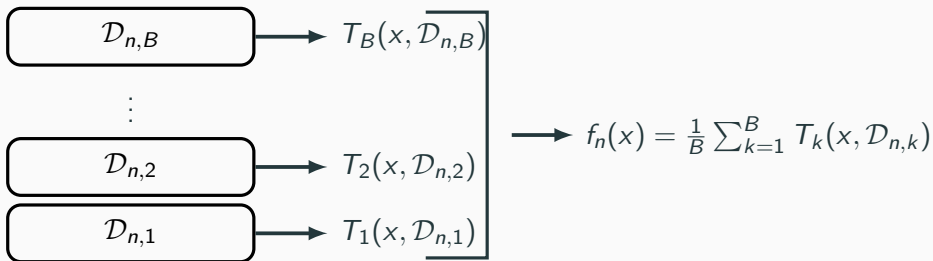
2. Inconvénients :

- Performances prédictives limitées.
- méthode connue pour être instable, sensible à de légères perturbations de l'échantillon.
 - ⇒ Cet inconvénient sera un avantage pour des agrégations bootstrap
 - ⇒ forêts aléatoires.

- **Idée** : construire un grand nombre d'algorithmes "simples" et les agréger pour obtenir une seule prévision. Par exemple



- **Idée** : construire un grand nombre d'**algorithmes "simples"** et les agréger pour obtenir une seule prévision. Par exemple



Questions

1. Comment choisir les **échantillons** $\mathcal{D}_{n,b}$?
2. Comment choisir les **algorithmes** ?
3. ...

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- Idem que précédemment, on cherche à **expliquer** une variable Y par d variables explicatives X_1, \dots, X_d .

- Idem que précédemment, on cherche à **expliquer** une variable Y par d variables explicatives X_1, \dots, X_d .
- Pour simplifier on se place en **régression** : Y est à valeurs dans \mathbb{R} mais tout ce qui va être fait s'étant directement à la **classification binaire ou multiclassés**.

- Idem que précédemment, on cherche à **expliquer** une variable Y par d variables explicatives X_1, \dots, X_d .
- Pour simplifier on se place en **régression** : Y est à valeurs dans \mathbb{R} mais tout ce qui va être fait s'étant directement à la **classification binaire ou multiclassés**.
- **Notations** :
 - (X, Y) un couple aléatoire à valeurs dans $\mathbb{R}^d \times \mathbb{R}$.
 - $\mathcal{D}_n = (X_1, Y_1), \dots, (X_n, Y_n)$ un n -échantillon i.i.d. de même loi que (X, Y) .

- Un algorithme de la forme :

$$f_n(x) = \frac{1}{B} \sum_{b=1}^B T_b(x)$$

- **Hypothèse** : les T_1, \dots, T_b sont **identiquement distribuées**.

- Un algorithme de la forme :

$$f_n(x) = \frac{1}{B} \sum_{b=1}^B T_b(x)$$

- **Hypothèse** : les T_1, \dots, T_b sont **identiquement distribuées**.

Propriété

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)]$$

où $\rho(x) = \text{corr}(T_1(x), T_2(x))$.

- Un algorithme de la forme :

$$f_n(x) = \frac{1}{B} \sum_{b=1}^B T_b(x)$$

- **Hypothèse** : les T_1, \dots, T_b sont **identiquement distribuées**.

Propriété

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)]$$

où $\rho(x) = \text{corr}(T_1(x), T_2(x))$.

Conséquence

- **Biais** non modifié.
- **Variance** \searrow si $B \nearrow$ et $\rho(x) \searrow$.

- Ajuster le même algorithme sur les mêmes données n'est d'aucun intérêt.

- Ajuster le même algorithme sur les mêmes données n'est d'aucun intérêt.
- Ajuster le même algorithme sur des sous-échantillons disjoints est d'un intérêt limité.

- Ajuster le même algorithme sur les mêmes données n'est d'aucun intérêt.
- Ajuster le même algorithme sur des sous-échantillons disjoints est d'un intérêt limité.
- Utiliser un grand nombre d'algorithmes différents est compliqué...

- Ajuster le même algorithme sur les mêmes données n'est d'aucun intérêt.
- Ajuster le même algorithme sur des sous-échantillons disjoints est d'un intérêt limité.
- Utiliser un grand nombre d'algorithmes différents est compliqué...

Idée

Ajuster le même algorithme sur des échantillons bootstraps.

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- Le **bagging** désigne un ensemble de méthodes introduit par Léo Breiman [[Breiman, 1996](#)].
- **Bagging** : vient de la contraction de **Bootstrap Aggregating**.
- **Idée** : plutôt que de construire un seul estimateur, en construire un grand nombre (sur des échantillons **bootstrap**) et les **agréger**.

Idée : échantillons bootstrap

- Echantillon **initial** :

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

Idée : échantillons bootstrap

- Echantillon **initial** :

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

- Echantillons **bootstrap** : tirage de taille n avec remise

3	4	6	10	3	9	10	7	7	1	T_1
2	8	6	2	10	10	2	9	5	6	T_2
2	9	4	4	7	7	2	3	6	7	T_3
6	1	3	3	9	3	8	10	10	1	T_4
3	7	10	3	2	8	6	9	10	2	T_5
	\vdots								\vdots	
7	10	3	4	9	10	10	8	6	1	T_B

Idée : échantillons bootstrap

- Echantillon **initial** :

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

- Echantillons **bootstrap** : tirage de taille n avec remise

3	4	6	10	3	9	10	7	7	1	T_1
2	8	6	2	10	10	2	9	5	6	T_2
2	9	4	4	7	7	2	3	6	7	T_3
6	1	3	3	9	3	8	10	10	1	T_4
3	7	10	3	2	8	6	9	10	2	T_5
	⋮								⋮	
7	10	3	4	9	10	10	8	6	1	T_B

- A la fin, on **agrège** :

$$f_n(x) = \frac{1}{B} \sum_{b=1}^B T_b(x)$$

Algorithme bagging

Entrées :

- B un entier positif ;
- T un algorithme de prévision.

Pour b entre 1 et B :

1. Faire un tirage aléatoire avec remise de taille n dans $\{1, \dots, n\}$. On note θ_b l'ensemble des indices sélectionnés et $\mathcal{D}_{n,b}^* = \{(x_i, y_i), i \in \theta_b\}$ l'échantillon bootstrap associé.
2. Entraîner l'algorithme T sur $\mathcal{D}_{n,b}^* \implies T(\cdot, \theta_b, \mathcal{D}_n)$.

Retourner : $f_n(x) = \frac{1}{B} \sum_{b=1}^B T(x, \theta_b, \mathcal{D}_n)$.

Un algorithme pas forcément aléatoire

- L'**aléa bootstrap** implique que l'algorithme "change" lorsqu'on l'exécute plusieurs fois mais...

Un algorithme pas forcément aléatoire

- L'**aléa bootstrap** implique que l'algorithme "change" lorsqu'on l'exécute plusieurs fois mais...

$$\lim_{B \rightarrow +\infty} \frac{1}{B} \sum_{b=1}^B T(x, \theta_b, \mathcal{D}_n) = E_{\theta}[T(x, \theta, \mathcal{D}_n)] = \bar{f}_n(x, \mathcal{D}_n)$$

Un algorithme pas forcément aléatoire

- L'**aléa bootstrap** implique que l'algorithme "change" lorsqu'on l'exécute plusieurs fois mais...

$$\lim_{B \rightarrow +\infty} \frac{1}{B} \sum_{b=1}^B T(x, \theta_b, \mathcal{D}_n) = E_{\theta}[T(x, \theta, \mathcal{D}_n)] = \bar{f}_n(x, \mathcal{D}_n)$$

Conséquence

- L'algorithme se **stabilise** (converge) lorsque $B \nearrow$.

Un algorithme pas forcément aléatoire

- L'**aléa bootstrap** implique que l'algorithme "change" lorsqu'on l'exécute plusieurs fois mais...

$$\lim_{B \rightarrow +\infty} \frac{1}{B} \sum_{b=1}^B T(x, \theta_b, \mathcal{D}_n) = E_{\theta}[T(x, \theta, \mathcal{D}_n)] = \bar{f}_n(x, \mathcal{D}_n)$$

Conséquence

- L'algorithme se **stabilise** (converge) lorsque $B \nearrow$.
- Recommandation : choisir B le **plus grand possible**.

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

Conclusion

- Bagging ne modifie pas le biais.

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

Conclusion

- Bagging ne modifie pas le biais.
- B grand $\implies V[f_n(x)] \approx \rho(x)V[T_1(x)]$

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

Conclusion

- Bagging ne modifie pas le biais.
- B grand $\implies V[f_n(x)] \approx \rho(x)V[T_1(x)] \implies$ la variance diminue d'autant plus que la corrélation entre les prédicteurs diminue.

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

Conclusion

- Bagging ne modifie pas le biais.
- B grand $\implies V[f_n(x)] \approx \rho(x)V[T_1(x)] \implies$ la variance diminue d'autant plus que la corrélation entre les prédicteurs diminue.
- Il est donc nécessaire d'agréger des estimateurs sensibles à de légères perturbations de l'échantillon.

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

Conclusion

- Bagging ne modifie pas le biais.
- B grand $\implies V[f_n(x)] \approx \rho(x)V[T_1(x)] \implies$ la variance diminue d'autant plus que la corrélation entre les prédicteurs diminue.
- Il est donc nécessaire d'agréger des estimateurs sensibles à de légères perturbations de l'échantillon.
- Les arbres sont connus pour posséder de telles propriétés.

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

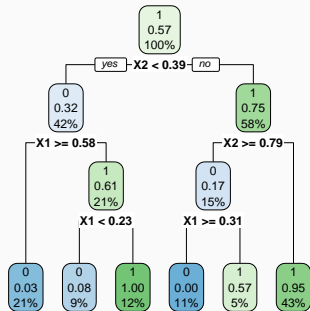
Algorithme

Choix des paramètres

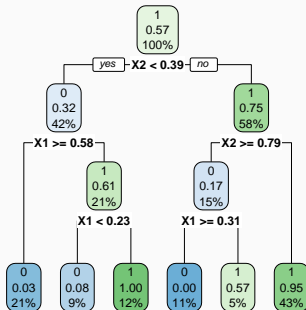
Erreur OOB et importance des variables

Bibliographie

Rappels sur les arbres



Rappels sur les arbres



Complexité

Profondeur

- **petite** : biais ↗, variance ↘
- **grande** : biais ↘, variance ↗ (sur-apprentissage).

- Comme son nom l'indique, une forêt aléatoire est définie à partir d'un ensemble d'arbres.

- Comme son nom l'indique, une **forêt aléatoire** est définie à partir d'un ensemble d'arbres.

Définition

Soit $T_k(x)$, $k = 1, \dots, B$ des prédicteurs par arbre ($T_k : \mathbb{R}^d \rightarrow \mathbb{R}$). Le prédicteur des **forêts aléatoires** est obtenu par agrégation de cette collection d'arbres :

$$f_n(x) = \frac{1}{B} \sum_{k=1}^B T_k(x).$$

- Forêts aléatoires = collection d'arbres.

Forêts aléatoires

- Forêts aléatoires = collection d'arbres.
- Les forêts aléatoires les plus utilisées sont (de loin) celles proposées par Léo Breiman (au début des années 2000).

Forêts aléatoires

- Forêts aléatoires = collection d'arbres.
- Les forêts aléatoires les plus utilisées sont (de loin) celles proposées par Léo Breiman (au début des années 2000).
- Elles consistent à agréger des arbres construits sur des échantillons bootstrap.

Forêts aléatoires

- Forêts aléatoires = collection d'arbres.
- Les forêts aléatoires les plus utilisées sont (de loin) celles proposées par Léo Breiman (au début des années 2000).
- Elles consistent à agréger des arbres construits sur des échantillons bootstrap.
- On pourra trouver de la doc à l'url
<http://www.stat.berkeley.edu/~breiman/RandomForests/>
et consulter la thèse de Robin Genuer [Genuer, 2010].

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

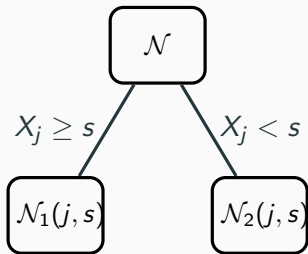
Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

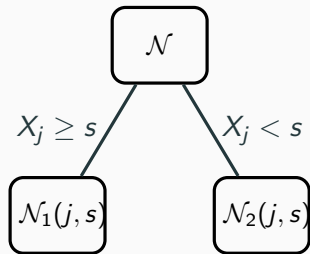
Coupures "aléatoires"



Arbres pour forêt

- Breiman propose de sélectionner la "meilleure" variable dans un ensemble composé **uniquement de m try variables choisies aléatoirement parmi les d variables initiales.**

Coupures "aléatoires"



Arbres pour forêt

- Breiman propose de sélectionner la "meilleure" variable dans un ensemble composé **uniquement de m try variables choisies aléatoirement parmi les d variables initiales.**
- **Objectif** : **diminuer la corrélation** entre les arbres que l'on agrège.

Algorithme forêts aléatoires

Entrées :

- B un entier positif ;
- **mtry** un entier entre 1 et d ;
- **min.node.size** un entier plus petit que n .

Pour b entre 1 et B :

1. Faire un tirage aléatoire avec remise de taille n dans $\{1, \dots, n\}$. On note \mathcal{I}_b l'ensemble des indices sélectionnés et $\mathcal{D}_{n,b}^* = \{(x_i, y_i), i \in \mathcal{I}_b\}$ l'échantillon bootstrap associé.
2. Construire un arbre CART à partir de $\mathcal{D}_{n,b}^*$ en découpant chaque nœud de la façon suivante :
 - 2.1 Choisir **mtry** variables au hasard parmi les d variables explicatives ;
 - 2.2 Sélectionner la meilleure coupure $X_j \leq s$ en ne considérant que les **mtry** variables sélectionnées ;
 - 2.3 Ne pas découper un nœud s'il contient moins de **min.node.size** observations.
3. On note $T(\cdot, \theta_b, \mathcal{D}_n)$ l'arbre obtenu.

Retourner : $f_n(x) = \frac{1}{B} \sum_{b=1}^B T(x, \theta_b, \mathcal{D}_n)$.

Type de prévision

La prévision dépend de la **nature de Y** et de ce que l'on souhaite **estimer**

- **Régression** : $T(x, \theta_b, \mathcal{D}_n) \in \mathbb{R}$ et

$$m_n(x) = \frac{1}{B} \sum_{b=1}^B T(x, \theta_b, \mathcal{D}_n).$$

Type de prévision

La prévision dépend de la **nature de Y** et de ce que l'on souhaite **estimer**

- **Régression** : $T(x, \theta_b, \mathcal{D}_n) \in \mathbb{R}$ et

$$m_n(x) = \frac{1}{B} \sum_{b=1}^B T(x, \theta_b, \mathcal{D}_n).$$

- **Classification** (classe) : $T(x, \theta_b, \mathcal{D}_n) \in \{1, \dots, K\}$ et

$$g_n(x) \in \operatorname{argmax}_{k \in \{1, \dots, K\}} \sum_{b=1}^B 1_{T(x, \theta_b, \mathcal{D}_n)=k}, \quad k = 1, \dots, K.$$

Type de prévision

La prévision dépend de la **nature de Y** et de ce que l'on souhaite **estimer**

- **Régression** : $T(x, \theta_b, \mathcal{D}_n) \in \mathbb{R}$ et

$$m_n(x) = \frac{1}{B} \sum_{b=1}^B T(x, \theta_b, \mathcal{D}_n).$$

- **Classification** (classe) : $T(x, \theta_b, \mathcal{D}_n) \in \{1, \dots, K\}$ et

$$g_n(x) \in \operatorname{argmax}_{k \in \{1, \dots, K\}} \sum_{b=1}^B 1_{T(x, \theta_b, \mathcal{D}_n)=k}, \quad k = 1, \dots, K.$$

- **Classification** (proba) : $T_k(x, \theta_b, \mathcal{D}_n) \in [0, 1]$ et

$$S_{n,k}(x) = \frac{1}{B} \sum_{b=1}^B T_k(x, \theta_b, \mathcal{D}_n), \quad k = 1, \dots, K.$$

- Notamment 2 packages avec à peu près la même syntaxe.
- **randomforest** : le plus ancien et probablement encore le plus utilisé.
- **ranger** [Wright and Ziegler, 2017] : plus efficace au niveau **temps de calcul** (codé en C++).

```
> library(ranger)
> set.seed(12345)
> foret <- ranger(type~.,data=spam)
> foret
## ranger(type ~ ., data = spam)
## Type: Classification
## Number of trees: 500
## Sample size: 4601
## Number of independent variables: 57
## Mtry: 7
## Target node size: 1
## Variable importance mode: none
## Splitrule: gini
## OOB prediction error: 4.59 %
```

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- B réglé \implies le plus grand possible.

En pratique on pourra s'assurer que le **courbe d'erreur** en fonction du nombre d'arbres est **stabilisée**.

- B réglé \implies le plus grand possible.

En pratique on pourra s'assurer que la **courbe d'erreur** en fonction du nombre d'arbres est **stabilisée**.

- Pour les autres paramètres on étudie à nouveau :

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

Conséquence

- Le biais n'étant pas amélioré par "l'agrégation bagging", il est recommandé d'agréger des estimateurs qui possèdent un **biais faible** (**contrairement au boosting**).

- B réglé \implies le plus grand possible.

En pratique on pourra s'assurer que la **courbe d'erreur** en fonction du nombre d'arbres est **stabilisée**.

- Pour les autres paramètres on étudie à nouveau :

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

Conséquence

- Le biais n'étant pas amélioré par "l'agrégation bagging", il est recommandé d'agréger des estimateurs qui possèdent un **biais faible** (**contrairement au boosting**).
- Arbres "**profonds**", **peu d'observations dans les nœuds terminaux**.

- B réglé \implies le plus grand possible.

En pratique on pourra s'assurer que la **courbe d'erreur** en fonction du nombre d'arbres est **stabilisée**.

- Pour les autres paramètres on étudie à nouveau :

$$E[f_n(x)] = E[T_1(x)] \quad \text{et} \quad V[f_n(x)] = \rho(x)V[T_1(x)] + \frac{1 - \rho(x)}{B}V[T_1(x)].$$

Conséquence

- Le biais n'étant pas amélioré par "l'agrégation bagging", il est recommandé d'agréger des estimateurs qui possèdent un **biais faible** (**contrairement au boosting**).
- Arbres "**profonds**", **peu d'observations dans les nœuds terminaux**.
- Par défaut dans **randomForest**, ***min.node.size*** = 5 en régression et 1 en classification.

Choix de m_{try}

- Il est en relation avec la corrélation entre les arbres $\rho(x)$.

Choix de m try

- Il est en relation avec la corrélation entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le compromis biais/variance de la forêt.

Choix de *mtry*

- Il est en relation avec la corrélation entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le compromis biais/variance de la forêt.
- *mtry* ↘

Choix de *mtry*

- Il est en relation avec la corrélation entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le compromis biais/variance de la forêt.
- *mtry* ↘
 1. tendance à se rapprocher d'un choix "aléatoire" des variables de découpe des arbres

Choix de *mtry*

- Il est en **relation avec la corrélation** entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le **compromis biais/variance** de la forêt.
- *mtry* ↘
 1. tendance à se rapprocher d'un **choix "aléatoire"** des variables de découpe des arbres \implies les arbres sont de plus en plus différents

Choix de $mtry$

- Il est en relation avec la corrélation entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le compromis biais/variance de la forêt.
- $mtry \searrow$
 1. tendance à se rapprocher d'un choix "aléatoire" des variables de découpe des arbres \implies les arbres sont de plus en plus différents $\implies \rho(x) \searrow \implies$ la variance de la forêt diminue.

Choix de $mtry$

- Il est en relation avec la corrélation entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le compromis biais/variance de la forêt.
- $mtry$ ↘
 1. tendance à se rapprocher d'un choix "aléatoire" des variables de découpe des arbres \implies les arbres sont de plus en plus différents \implies $\rho(x)$ ↘ \implies la variance de la forêt diminue.
 2. mais... le biais des arbres ↗

Choix de $mtry$

- Il est en **relation avec la corrélation** entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le **compromis biais/variance** de la forêt.
- $mtry \searrow$
 1. tendance à se rapprocher d'un **choix "aléatoire"** des variables de découpe des arbres \implies les arbres sont de plus en plus différents $\implies \rho(x) \searrow \implies$ **la variance de la forêt diminue**.
 2. mais... le biais des arbres $\nearrow \implies$ le **biais de la forêt** \nearrow .

Choix de $mtry$

- Il est en relation avec la corrélation entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le compromis biais/variance de la forêt.
- $mtry \searrow$
 1. tendance à se rapprocher d'un choix "aléatoire" des variables de découpe des arbres \implies les arbres sont de plus en plus différents $\implies \rho(x) \searrow \implies$ la variance de la forêt diminue.
 2. mais... le biais des arbres $\nearrow \implies$ le biais de la forêt \nearrow .
- Inversement lorsque $mtry \nearrow$ (risque de sur-ajustement).

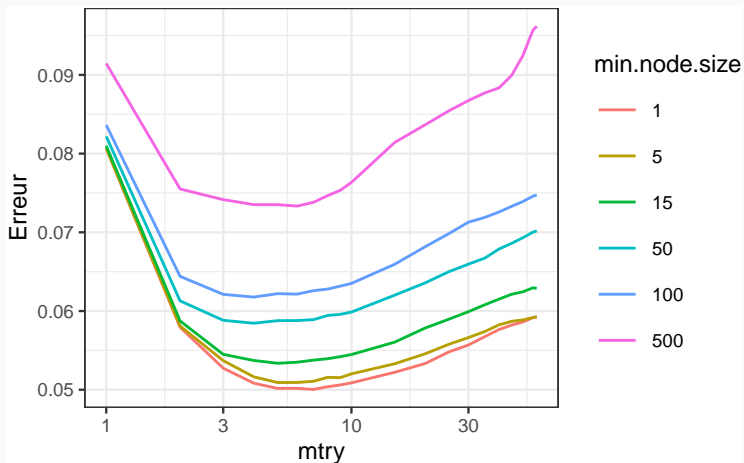
Choix de $mtry$

- Il est en relation avec la corrélation entre les arbres $\rho(x)$.
- Ce paramètre a une influence sur le compromis biais/variance de la forêt.
- $mtry \searrow$
 1. tendance à se rapprocher d'un choix "aléatoire" des variables de découpe des arbres \implies les arbres sont de plus en plus différents $\implies \rho(x) \searrow \implies$ la variance de la forêt diminue.
 2. mais... le biais des arbres $\nearrow \implies$ le biais de la forêt \nearrow .
- Inversement lorsque $mtry \nearrow$ (risque de sur-ajustement).

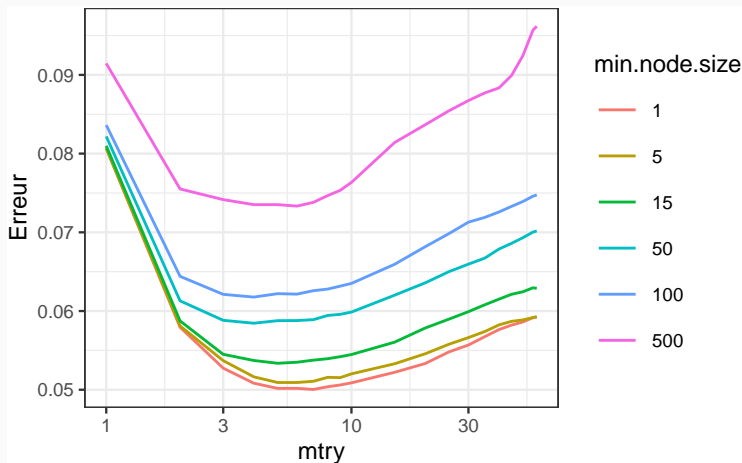
Conclusion

- Il est recommandé de comparer les performances de la forêt pour plusieurs valeurs de $mtry$.
- Par défaut $mtry = d/3$ en régression et \sqrt{d} en classification.

- Visualisation d'erreur en fonction de **min.node.size** et **mtry**



- Visualisation d'erreur en fonction de **min.node.size** et **mtry**



Commentaires

min.node.size petit et **mtry** à calibrer.

- On peut bien entendu **calibrer ces paramètres** avec les approches traditionnelles mais...

- On peut bien entendu **calibrer ces paramètres** avec les approches traditionnelles mais...
- les valeurs par défaut sont souvent performantes !

- On peut bien entendu **calibrer ces paramètres** avec les approches traditionnelles mais...
- les valeurs par défaut sont souvent performantes !
- On pourra quand même faire quelques essais, notamment pour **mtry**.

Un exemple avec tidymodels

1. Initialisation du workflow :

```
> tune_spec <- rand_forest(mtry = tune(),min_n= tune()) %>%  
+   set_engine("ranger") %>%  
+   set_mode("classification")  
> rf_wf <- workflow() %>% add_model(tune_spec) %>% add_formula(type ~ .)
```

2. Ré-échantillonnage et grille de paramètres :

```
> blocs <- vfold_cv(spam, v = 10,repeats = 5)  
> rf_grid <- expand_grid(mtry=c(seq(1,55,by=5),57),  
+                       min_n=c(1,5,15,50,100,500))
```

3. Calcul des erreurs :

```
> rf_res <- rf_wf %>% tune_grid(resamples = blocs, grid = rf_grid)
```

4. Visualisation des résultats (AUC et accuracy) :

```
> rf_res %>% show_best("roc_auc") %>% select(-8)
```

```
## # A tibble: 5 x 7
```

##	mtry	min_n	.metric	.estimator	mean	n	std_err
##	<dbl>	<dbl>	<chr>	<chr>	<dbl>	<int>	<dbl>
## 1	4	1	roc_auc	binary	0.988	50	0.000614
## 2	5	1	roc_auc	binary	0.988	50	0.000623
## 3	6	1	roc_auc	binary	0.988	50	0.000617
## 4	5	5	roc_auc	binary	0.988	50	0.000621
## 5	7	1	roc_auc	binary	0.988	50	0.000645

```
> rf_res %>% show_best("accuracy") %>% select(-8)
```

```
## # A tibble: 5 x 7
```

##	mtry	min_n	.metric	.estimator	mean	n	std_err
##	<dbl>	<dbl>	<chr>	<chr>	<dbl>	<int>	<dbl>
## 1	4	1	accuracy	binary	0.954	50	0.00159
## 2	6	1	accuracy	binary	0.954	50	0.00141
## 3	7	1	accuracy	binary	0.954	50	0.00149
## 4	5	1	accuracy	binary	0.954	50	0.00153
## 5	8	1	accuracy	binary	0.953	50	0.00146

Remarque

On retrouve bien `min.node.size` petit et `mtry` proche de la valeur par défaut (7).

Remarque

On retrouve bien `min.node.size` petit et `mtry` proche de la valeur par défaut (7).

5. Ajustement de l'algorithme final :

```
> foret_finale <- rf_wf %>%  
+   finalize_workflow(list(mtry=7,min_n=1)) %>%  
+   fit(data=spam)
```

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging

Forêts aléatoires

Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

- Comme pour tous les algorithmes de prévision on peut évaluer la performance des forêts aléatoires en estimant un risque par ré-échantillonnage.

- Comme pour tous les algorithmes de prévision on peut évaluer la performance des forêts aléatoires en estimant un risque par ré-échantillonnage.
- Les tirages bootstraps permettent de définir une alternative, souvent moins couteuse en temps de calcul, au ré-échantillonnage.

- Comme pour tous les algorithmes de prévision on peut évaluer la performance des forêts aléatoires en estimant un risque par ré-échantillonnage.
- Les tirages bootstraps permettent de définir une alternative, souvent moins couteuse en temps de calcul, au ré-échantillonnage.
- Idée/astuce : utiliser les observations non sélectionnées dans les échantillons bootstraps pour estimer le risque.

OOB illustration

3	4	6	10	3	9	10	7	7	1	T_1
2	8	6	2	10	10	2	9	5	6	T_2
2	9	4	4	7	7	2	3	6	7	T_3
6	1	3	3	9	3	8	10	10	1	T_4
3	7	10	3	2	8	6	9	10	2	T_5
7	10	3	4	9	10	10	8	6	1	T_6

OOB illustration

3	4	6	10	3	9	10	7	7	1	T_1
2	8	6	2	10	10	2	9	5	6	T_2
2	9	4	4	7	7	2	3	6	7	T_3
6	1	3	3	9	3	8	10	10	1	T_4
3	7	10	3	2	8	6	9	10	2	T_5
7	10	3	4	9	10	10	8	6	1	T_6

- Les échantillons 2, 3 et 5 ne contiennent pas la première observation, donc

$$\hat{y}_1 = \frac{1}{3}(T_2(x_1) + T_3(x_1) + T_5(x_1)).$$

- On fait de même pour toutes les observations $\implies \hat{y}_2, \dots, \hat{y}_n$.

OOB illustration

3	4	6	10	3	9	10	7	7	1	T_1
2	8	6	2	10	10	2	9	5	6	T_2
2	9	4	4	7	7	2	3	6	7	T_3
6	1	3	3	9	3	8	10	10	1	T_4
3	7	10	3	2	8	6	9	10	2	T_5
7	10	3	4	9	10	10	8	6	1	T_6

- Les échantillons 2, 3 et 5 ne contiennent pas la première observation, donc

$$\hat{y}_1 = \frac{1}{3}(T_2(x_1) + T_3(x_1) + T_5(x_1)).$$

- On fait de même pour toutes les observations $\implies \hat{y}_2, \dots, \hat{y}_n$.
- On calcule l'erreur selon

$$\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 \quad \text{ou} \quad \frac{1}{n} \sum_{i=1}^n 1_{\hat{y}_i \neq y_i}.$$

OOB définition

- Pour $i = 1, \dots, n$ on note

$$\text{OOB}(i) = \{b \leq B : i \notin \mathcal{I}_b\}$$

l'ensemble des tirages bootstrap qui **ne contiennent pas** i et

$$f_{n,\text{OOB}(i)}(x_i) = \frac{1}{|\text{OOB}(i)|} \sum_{b \in \text{OOB}(i)} T(x_i, \theta_b, \mathcal{D}_n)$$

la prévision de la forêt en ne considérant **que les arbres pour lesquels** i **n'est pas dans le tirage bootstrap.**

OOB définition

- Pour $i = 1, \dots, n$ on note

$$\text{OOB}(i) = \{b \leq B : i \notin \mathcal{I}_b\}$$

l'ensemble des tirages bootstrap qui **ne contiennent pas** i et

$$f_{n,\text{OOB}(i)}(x_i) = \frac{1}{|\text{OOB}(i)|} \sum_{b \in \text{OOB}(i)} T(x_i, \theta_b, \mathcal{D}_n)$$

la prévision de la forêt en ne considérant **que les arbres pour lesquels i n'est pas dans le tirage bootstrap**.

- L'**erreur OOB** s'obtient en confrontant ces prévisions aux valeurs observées, par exemple

$$\frac{1}{n} \sum_{i=1}^n (y_i - f_{n,\text{OOB}(i)}(x_i))^2 \quad \text{ou} \quad \frac{1}{n} \sum_{i=1}^n \mathbf{1}_{f_{n,\text{OOB}(i)}(x_i) \neq y_i}$$

OOB définition

- Pour $i = 1, \dots, n$ on note

$$\text{OOB}(i) = \{b \leq B : i \notin \mathcal{I}_b\}$$

l'ensemble des tirages bootstrap qui **ne contiennent pas** i et

$$f_{n,\text{OOB}(i)}(x_i) = \frac{1}{|\text{OOB}(i)|} \sum_{b \in \text{OOB}(i)} T(x_i, \theta_b, \mathcal{D}_n)$$

la prévision de la forêt en ne considérant **que les arbres pour lesquels i n'est pas dans le tirage bootstrap**.

- L'**erreur OOB** s'obtient en confrontant ces prévisions aux valeurs observées, par exemple

$$\frac{1}{n} \sum_{i=1}^n (y_i - f_{n,\text{OOB}(i)}(x_i))^2 \quad \text{ou} \quad \frac{1}{n} \sum_{i=1}^n \mathbf{1}_{f_{n,\text{OOB}(i)}(x_i) \neq y_i}$$

⇒ erreur renvoyée par défaut dans **ranger** et **randomforest**.

Importance des variables

Deux mesures sont généralement utilisées.

- **Score d'impureté** : simplement la moyenne des importances de X_j dans chaque arbre de la forêt :

$$\mathcal{I}_j^{\text{imp}} = \frac{1}{B} \sum_{b=1}^B \mathcal{I}_j(T_b),$$

voir chapitre sur les arbres pour la définition de $\mathcal{I}_j(T_b)$.

Importance des variables

Deux mesures sont généralement utilisées.

- **Score d'impureté** : simplement la moyenne des importances de X_j dans chaque arbre de la forêt :

$$\mathcal{I}_j^{\text{imp}} = \frac{1}{B} \sum_{b=1}^B \mathcal{I}_j(T_b),$$

voir chapitre sur les arbres pour la définition de $\mathcal{I}_j(T_b)$.

- **Importance par permutation** : comparer les erreurs de chaque arbre sur l'échantillon
 1. OOB de l'arbre ;
 2. OOB en permutant les valeurs de la variables j .

⇒ **Idée** : Si X_j est importante ces erreurs doivent être très différentes.

Importance par permutation

- On présente ce score en régression mais rien ne change pour la classification.
- On note

$$\text{Err}(\text{OOB}_b) = \frac{1}{|\text{OOB}_b|} \sum_{i \in \text{OOB}_b} (y_i - T(x_i, \theta_b, \mathcal{D}_n))^2,$$

avec

$$\text{OOB}_b = \{i \leq n : i \notin \mathcal{I}_b\}.$$

⇒ Erreur de l'arbre b calculée sur les données OOB.

Importance par permutation

- On présente ce score en régression mais rien ne change pour la classification.
- On note

$$\text{Err}(\text{OOB}_b) = \frac{1}{|\text{OOB}_b|} \sum_{i \in \text{OOB}_b} (y_i - T(x_i, \theta_b, \mathcal{D}_n))^2,$$

avec

$$\text{OOB}_b = \{i \leq n : i \notin \mathcal{I}_b\}.$$

\implies Erreur de l'arbre b calculée sur les données OOB.

- On recalcule cette erreur mais sur OOB_b où on permute les valeurs de la j^{e} colonne.

$$\begin{bmatrix} x_{11} & \dots & x_{1j} & \dots & x_{1d} \\ x_{21} & \dots & x_{2j} & \dots & x_{2d} \\ x_{51} & \dots & x_{3j} & \dots & x_{3d} \\ x_{41} & \dots & x_{4j} & \dots & x_{4d} \\ x_{51} & \dots & x_{5j} & \dots & x_{5d} \end{bmatrix} \implies \begin{bmatrix} x_{11} & \dots & x_{3j} & \dots & x_{1d} \\ x_{21} & \dots & x_{5j} & \dots & x_{2d} \\ x_{51} & \dots & x_{1j} & \dots & x_{3d} \\ x_{41} & \dots & x_{2j} & \dots & x_{4d} \\ x_{51} & \dots & x_{4j} & \dots & x_{5d} \end{bmatrix}$$

- On note \tilde{x}_i^j les individus de l'échantillon OOB_b permuté et on calcule

$$\text{Err}(\text{OOB}_b^j) = \frac{1}{|\text{OOB}_b|} \sum_{i \in \text{OOB}_b} (y_i - T(\tilde{x}_i^j, \theta_b, \mathcal{D}_n))^2.$$

$$\begin{bmatrix} x_{11} & \dots & x_{1j} & \dots & x_{1d} \\ x_{21} & \dots & x_{2j} & \dots & x_{2d} \\ x_{51} & \dots & x_{3j} & \dots & x_{3d} \\ x_{41} & \dots & x_{4j} & \dots & x_{4d} \\ x_{51} & \dots & x_{5j} & \dots & x_{5d} \end{bmatrix} \Rightarrow \begin{bmatrix} x_{11} & \dots & x_{3j} & \dots & x_{1d} \\ x_{21} & \dots & x_{5j} & \dots & x_{2d} \\ x_{51} & \dots & x_{1j} & \dots & x_{3d} \\ x_{41} & \dots & x_{2j} & \dots & x_{4d} \\ x_{51} & \dots & x_{4j} & \dots & x_{5d} \end{bmatrix}$$

- On note \tilde{x}_i^j les individus de l'échantillon OOB_b permuté et on calcule

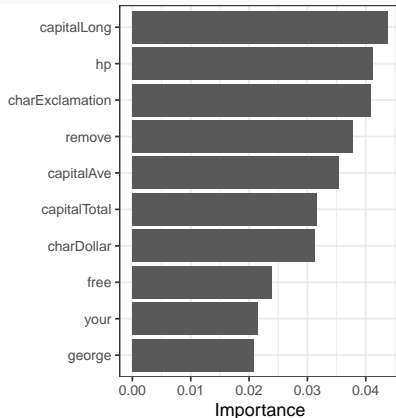
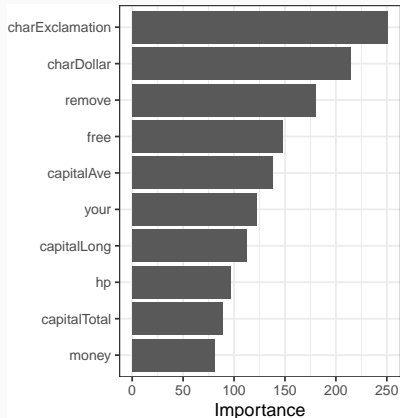
$$\text{Err}(\text{OOB}_b^j) = \frac{1}{|\text{OOB}_b|} \sum_{i \in \text{OOB}_b} (y_i - T(\tilde{x}_i^j, \theta_b, \mathcal{D}_n))^2.$$

Importance par permutation

$$\mathcal{I}_j^{\text{perm}} = \frac{1}{B} \sum_{b=1}^B (\text{Err}(\text{OOB}_b^j) - \text{Err}(\text{OOB}_b)).$$

- On peut **calculer et visualiser** facilement ces importances avec **ranger** :

```
> set.seed(1234)
> foret.imp <- ranger(type~.,data=spam,importance="impurity")
> foret.perm <- ranger(type~.,data=spam,importance="permutation")
> vip(foret.imp);vip(foret.perm)
```



Beaucoup d'avantages

- Bonnes performances prédictives \implies souvent parmi les algorithmes de tête dans les compétitions [[Fernández-Delgado et al., 2014](#)].
- Facile à calibrer.

Conclusion

Beaucoup d'avantages

- Bonnes performances prédictives \implies souvent parmi les algorithmes de tête dans les compétitions [[Fernández-Delgado et al., 2014](#)].
- Facile à calibrer.

Assez peu d'inconvénients

Coté boîte noire (mais guère plus que les autres méthodes...)

Arbres binaires

Choix des coupures

Cas de la régression

Cas de la classification supervisée

Elagage

Importance des variables

Bagging et forêts aléatoires

Bagging




Forêts aléatoires




Algorithme

Choix des paramètres

Erreur OOB et importance des variables

Bibliographie

-  Breiman, L. (1996).
Bagging predictors.
Machine Learning, 26(2) :123–140.
-  Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984).
Classification and regression trees.
Wadsworth & Brooks.
-  Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. (2014).
Do we need hundreds of classifiers to solve real world classification problems ?
Journal of Machine Learning Research, 15 :3133–3181.

-  Freund, Y. and Schapire, R. (1996).
Experiments with a new boosting algorithm.
In Proceedings of the Thirteenth International Conference on Machine Learning.
-  Friedman, J. H. (2001).
Greedy function approximation : A gradient boosting machine.
Annals of Statistics, 29 :1189–1232.
-  Friedman, J. H. (2002).
Stochastic gradient boosting.
Computational Statistics & Data Analysis, 28 :367–378.



Genuer, R. (2010).

Forêts aléatoires : aspects théoriques, sélection de variables et applications.

PhD thesis, Université Paris XI.



Hastie, T., Tibshirani, R., and Friedman, J. (2009).

The Elements of Statistical Learning : Data Mining, Inference, and Prediction.




Springer, second edition.



McCulloch, W. and Pitts, W. (1943).

A logical calculus of ideas immanent in nervous activity.

Bulletin of Mathematical Biophysics, 5 :115–133.

-  Rosenblatt, F. (1958).
The perceptron : a probabilistic model for information storage and organization in the brain.
Psychological Review, 65 :386–408.
-  Rumelhart, D. E., Hinton, G. E., and R. J. Williams, R. J. (1986).
Learning representations by back-propagating errors.
Nature, pages 533–536.
-  Wright, M. and Ziegler, A. (2017).
ranger : A fast implementation of random forests for high dimensional data in c++ and r.
Journal of Statistical Software, 17(1).

Discussion/comparaison des algorithmes

	Linéaire	SVM	Réseau	Arbre	Forêt	Boosting
Performance	■	■	■	▼	▲	▲
Calibration	▼	▼	▼	▲	▲	▲
Coût calc.	■	▼	▼	▲	▲	▲
Interprétation	▲	▼	▼	■	▼	▼

Commentaires

- Résultats pour **données tabulaires**.
- Différent pour **données structurées** (image, texte..)
 ⇒ performance ↗ réseaux pré-entraînés ⇒ **apprentissage profond/deep learning**.