Apprentissage supervisé - Machine learning

L. Rouvière
laurent.rouviere@univ-rennes2.fr

Janvier 2022

Présentation

- Objectifs : comprendre les aspects théoriques et pratiques de l'apprentissage supervisé ainsi que de quelques algorithmes 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 : energie, finance, marketing, sport.

Programme

- Matériel :
  - slides : https://lrouviere.github.io/machine_learning/
  - Tutoriel : https://lrouviere.github.io/TUTO_ML/ (chapitres 1, 3 et 5).

- 3 parties :
  1. Machine Learning : cadre, objectif, risque...
  2. Segmentation : arbres CART
  3. Agrégation : forêts aléatoires

Objectifs/questions

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

Première partie I

Apprentissage : contexte et formalisation

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...

$\Longrightarrow$ 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 $\Longrightarrow$ Data Mining $\Longrightarrow$ Apprentissage statistique $\Longrightarrow$ Big Data $\Longrightarrow$ Intelligence artificielle...

Approche statistique

Objectif $\Longrightarrow$ expliquer

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

Approche statistique

Objectif $\Longrightarrow$ 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...

Approche machine learning

Objectif $\Longrightarrow$ prédire

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

Approche machine learning

Objectif $\Longrightarrow$ 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 $\Longrightarrow$ le modèle doit être interprétable !
  - complexité moins importante en machine learning $\Longrightarrow$ on veut "juste bien prédire".

Statistique vs apprentissage

- Les objectifs diffèrent :
  - recherche de complexité minimale en statistique $\Longrightarrow$ le modèle doit être interprétable !
  - complexité moins importante en machine learning $\Longrightarrow$ on veut "juste bien prédire".
- Approches néanmoins complémentaires :
  - bien expliquer $\Longrightarrow$ bien prédire ;
  - "récentes" évolutions d'aide à l'interprétation des algorithmes ML $\Longrightarrow$ 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 $\Longrightarrow$ le modèle doit être interprétable !
  - complexité moins importante en machine learning $\Longrightarrow$ on veut "juste bien prédire".
- Approches néanmoins complémentaires :
  - bien expliquer $\Longrightarrow$ bien prédire ;
  - "récentes" évolutions d'aide à l'interprétation des algorithmes ML $\Longrightarrow$ 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 ;
- Systèmes de recommendation : 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 ;
- Systèmes de recommendation : identifier les produits susceptibles d'intéresser des consommateurs.

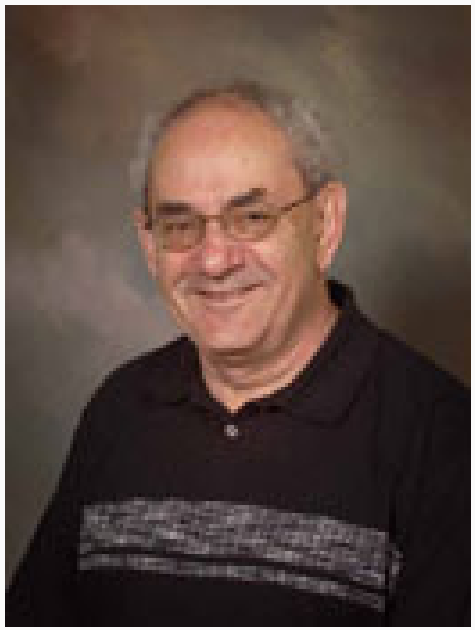
Nombreuses applications

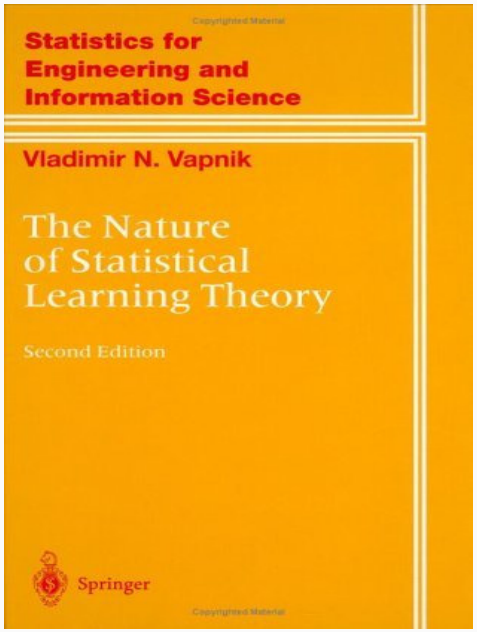
finance, économie, marketing, biologie, médecine...

Théorie de l'apprentissage statistique

Approche mathématique

- Ouvrage fondateur : [Vapnik, 2000]

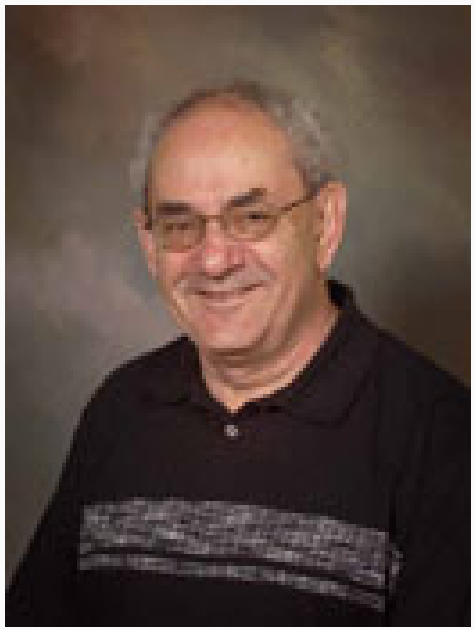


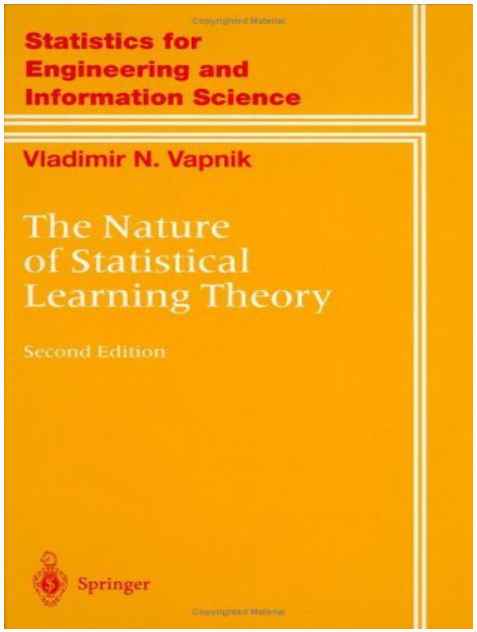


Théorie de l'apprentissage statistique

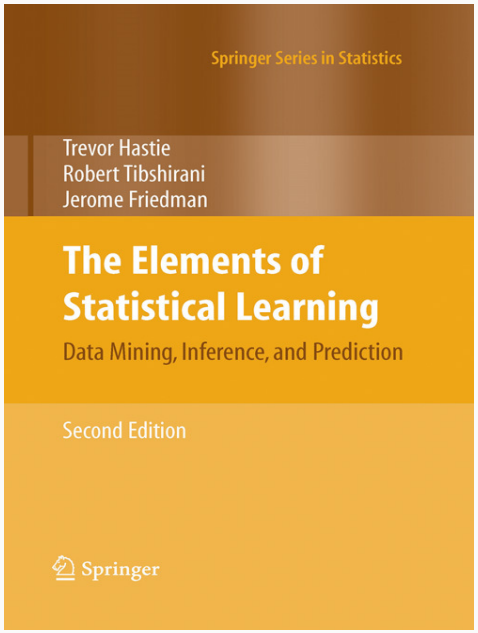
Approche mathématique

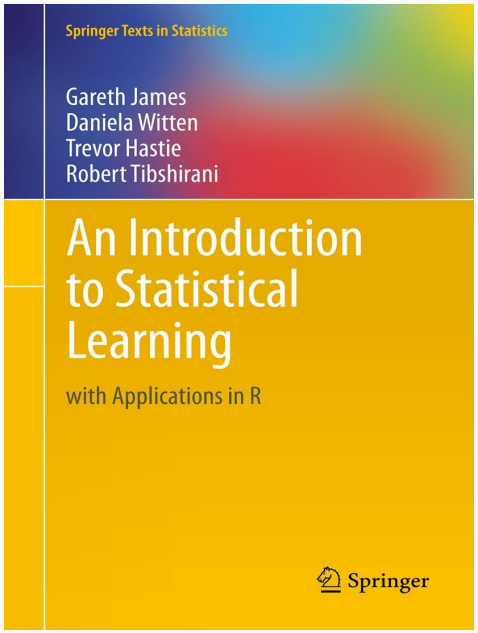
- Ouvrage fondateur : [Vapnik, 2000]
- voir aussi [Bousquet et al., 2003].





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/

Wikistat

- 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]
  - ...

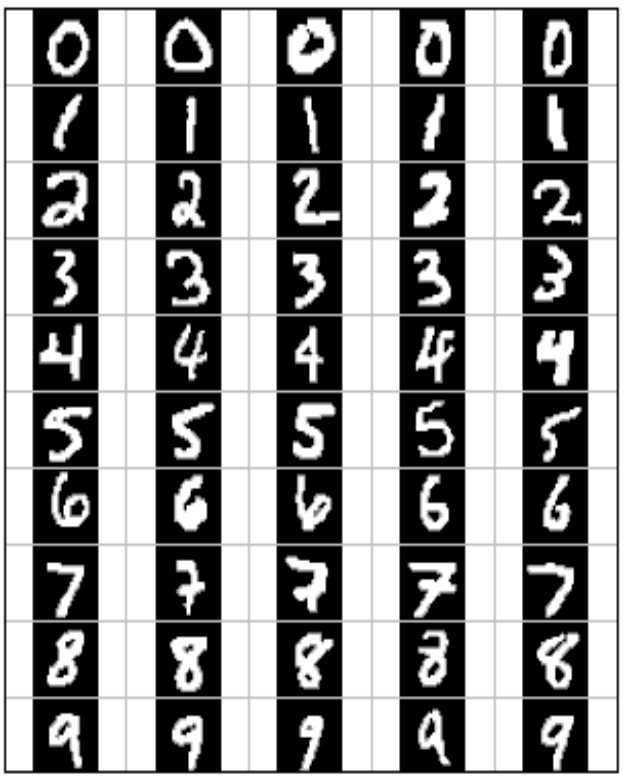
Wikistat

- 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.

Reconnaissance de l'écriture

Apprentissage statistique

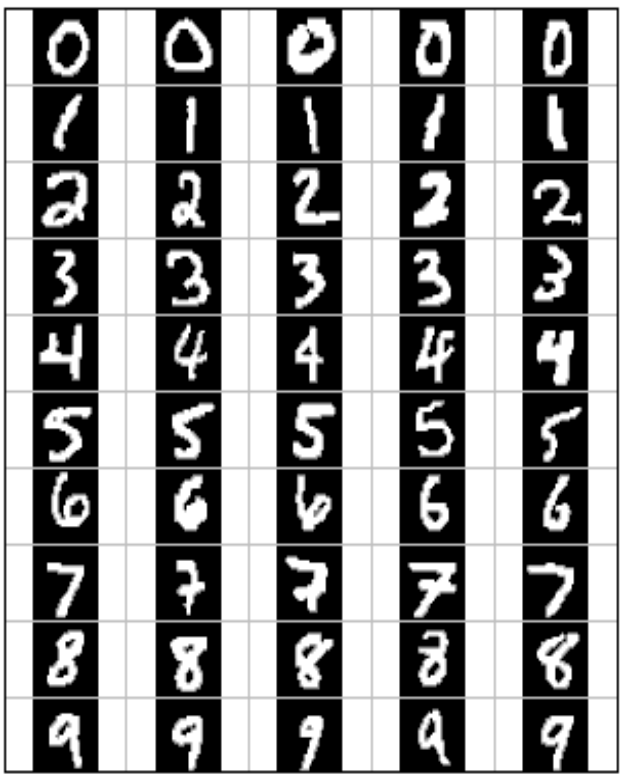
Comprendre et apprendre un comportement à partir d'exemples.

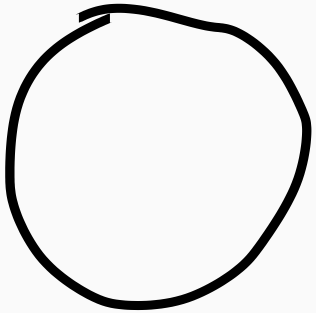


Reconnaissance de l'écriture

Apprentissage statistique

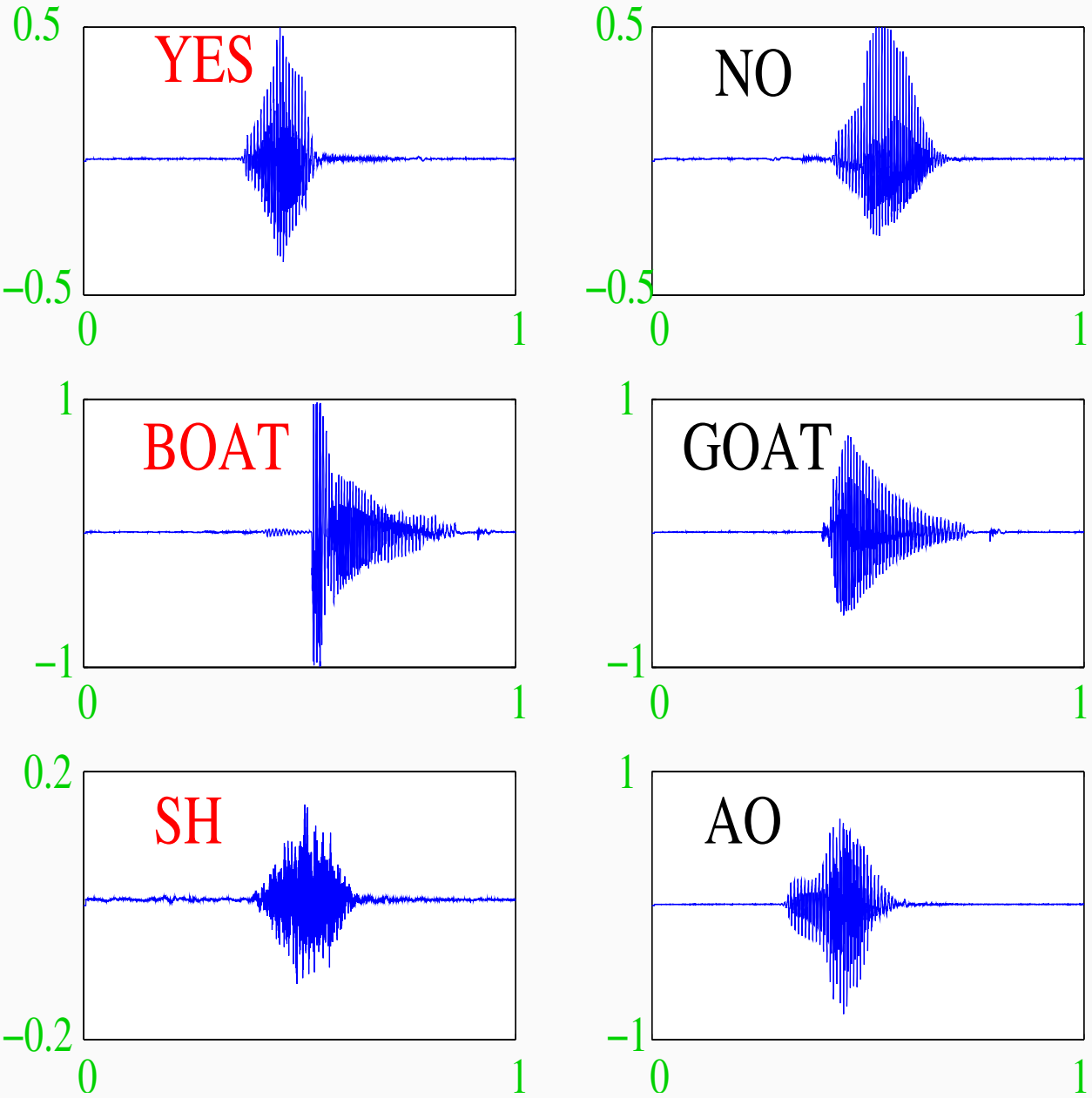
Comprendre et apprendre un comportement à partir d'exemples.



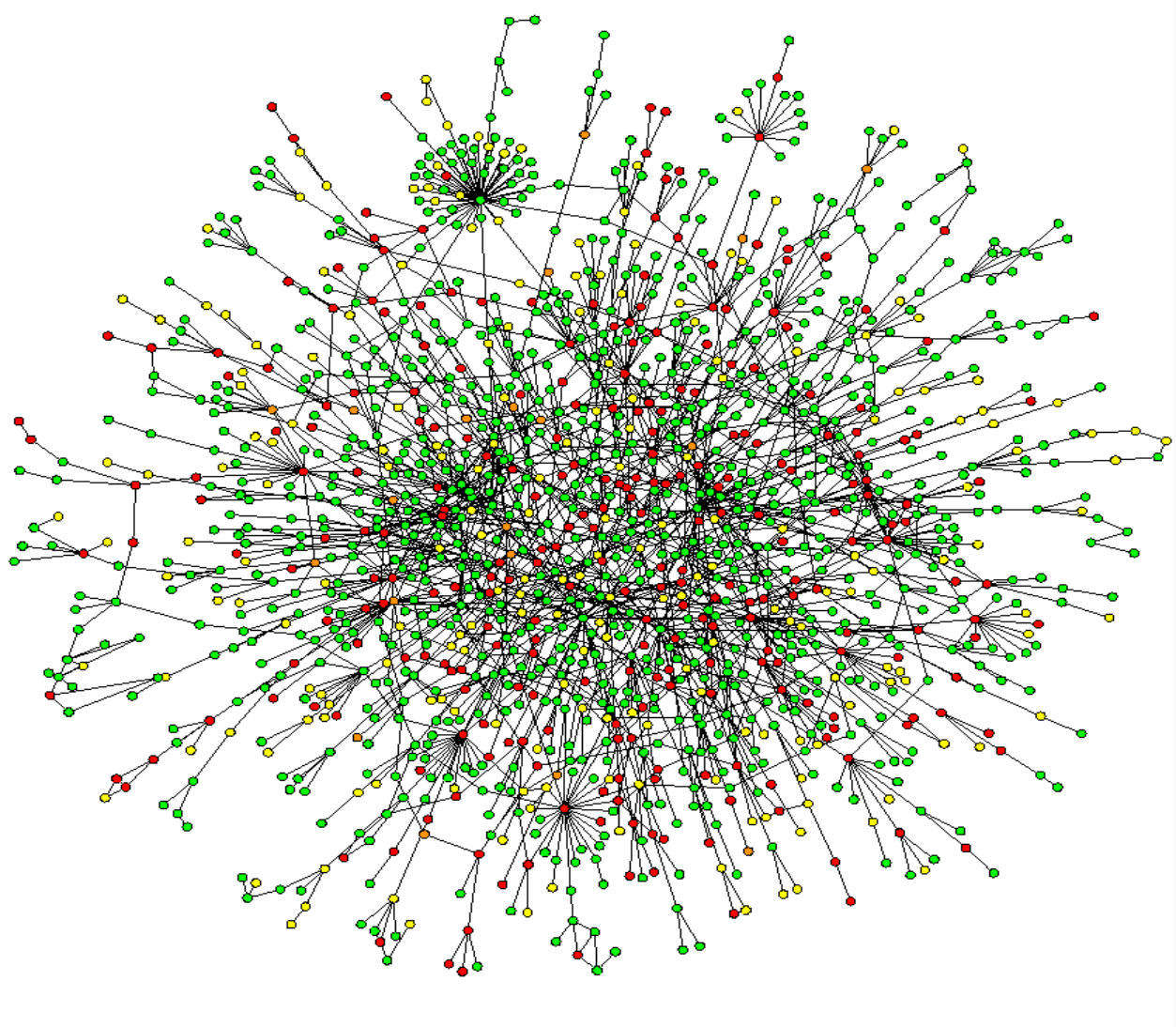


Qu'est-ce qui est écrit ? 0, 1, 2... ?

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 ?

Régression vs classification

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

Objectifs

1. Expliquer le(s) méchanisme(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 : $d_n = (x_1, y_1), \ldots, (x_n, y_n)$ où $x_i \in \mathcal{X}$ représente l'entrée et $y_i \in \mathcal{Y}$ la sortie.

Objectifs

1. Expliquer le(s) méchanisme(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 (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 $d_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ 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 $d_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ 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 $f : \mathcal{X} \to \mathcal{Y}$ telle que

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

Un début de formalisation mathématique

- Etant données des observations $d_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ 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 $f : \mathcal{X} \to \mathcal{Y}$ telle que

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

- 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 $d_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ 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 $f : \mathcal{X} \to \mathcal{Y}$ telle que

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

- 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 $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ telle que

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

Vision statistique

- On suppose que les données $d_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ sont des réalisations d'un n-échantillon $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ de loi inconnue.
- Les X_i sont des variables aléatoires à valeurs dans $\mathcal{X}$, les Y_i dans $\mathcal{Y}$.
- Le plus souvent on supposera que les couples $(X_i, Y_i), i = 1, \ldots, n$ sont i.i.d de loi (inconnue) P.

Vision statistique

- On suppose que les données $d_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ sont des réalisations d'un n-échantillon $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ de loi inconnue.
- Les X_i sont des variables aléatoires à valeurs dans $\mathcal{X}$, les Y_i dans $\mathcal{Y}$.
- Le plus souvent on supposera que les couples $(X_i, Y_i), i = 1, \ldots, n$ sont i.i.d de loi (inconnue) P.

Performance d'une fonction de prévision

- Etant donné une fonction de perte $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$, la performance d'une fonction de prévision $f : \mathcal{X} \to \mathcal{Y}$ est mesurée par

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

où (X, Y) est indépendant des (X_i, Y_i) et de même loi P.

Vision statistique

- On suppose que les données $d_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ sont des réalisations d'un n-échantillon $\mathcal{D}_n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ de loi inconnue.
- Les X_i sont des variables aléatoires à valeurs dans $\mathcal{X}$, les Y_i dans $\mathcal{Y}$.
- Le plus souvent on supposera que les couples $(X_i, Y_i), i = 1, \ldots, n$ sont i.i.d de loi (inconnue) P.

Performance d'une fonction de prévision

- Etant donné une fonction de perte $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$, la performance d'une fonction de prévision $f : \mathcal{X} \to \mathcal{Y}$ est mesurée par

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

où (X, Y) est indépendant des (X_i, Y_i) et de même loi P.

- $\mathcal{R}(f)$ est appelé risque ou erreur de généralisation de f.

Fonction de prévision optimale

- $\mathcal{R}(f) \Longrightarrow$ "Erreur moyenne" de f par rapport à la loi des données.
- Idée : trouver f qui a la plus petite erreur.

Fonction de prévision optimale

- $\mathcal{R}(f) \Longrightarrow$ "Erreur moyenne" de f par rapport à la loi des données.
- Idée : trouver f qui a la plus petite erreur.

Aspect théorique

- Pour une fonction de perte $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$ donnée, le problème théorique consiste à trouver

$$f^\star \in \operatorname*{argmin}_f \mathcal{R}(f) \Longleftrightarrow \mathcal{R}(f^\star) \leq \mathcal{R}(f) \ \forall f$$

- Une telle fonction $f^\star$ (si elle existe) est appelée fonction de prévision optimale pour la perte ℓ.

Aspect pratique

- La fonction de prévision optimale $f^\star$ dépend le plus souvent de la loi P des (X, Y) qui est en pratique inconnue.

Aspect pratique

- La fonction de prévision optimale $f^\star$ dépend le plus souvent de la loi P des (X, Y) qui est en pratique inconnue.
- Le job du statisticien est de trouver un estimateur $f_n = f_n(., \mathcal{D}_n)$ tel que $\mathcal{R}(f_n) \approx \mathcal{R}(f^\star)$.

Aspect pratique

- La fonction de prévision optimale $f^\star$ dépend le plus souvent de la loi P des (X, Y) qui est en pratique inconnue.
- Le job du statisticien est de trouver un estimateur $f_n = f_n(., \mathcal{D}_n)$ tel que $\mathcal{R}(f_n) \approx \mathcal{R}(f^\star)$.

Définition

Un algorithme de prévision est représenté par une suite $(f_n)_n$ d'applications (mesurables) telles que pour $n \geq 1$, $f_n : \mathcal{X} \times (\mathcal{X} \times \mathcal{Y})^n \to \mathcal{Y}$.

Propriétés statistiques d'un algorithme

- 1 un algorithme : 1 estimateur $f_n(.) = f_n(., \mathcal{D}_n)$ de $f^\star$.

Propriétés statistiques d'un algorithme

- 1 un algorithme : 1 estimateur $f_n(.) = f_n(., \mathcal{D}_n)$ de $f^\star$.

Propriétés statistiques

- Biais : $\mathrm{E}[f_n(x)] - f^\star(x) \Longrightarrow$ prévisions "en moyenne" ;
- Variance : $\mathrm{V}[f_n(x)] \Longrightarrow$ stabilité des prévisions ;
- Consistance : $\lim_{n\to\infty} \mathcal{R}(f_n) = \mathcal{R}(f^\star) \Longrightarrow$ précision quand n augmente ;
- ...

- Algorithme simple qui permet de répondre à des problème de régression et de classification.

- Algorithme simple qui permet de répondre à des problème de régression et de classification.
- Approche non paramétrique basée sur des moyennes locales.

- Algorithme simple qui permet de répondre à des problème de régression et de classification.
- Approche non paramétrique basée sur des moyennes locales.

Idée

Estimer la fonction inconnue au point x par une moyenne des y_i tels que x_i est proche de x.

kppv en régression

- $(x_1, y_1), \ldots, (x_n, y_n)$ avec $x_i \in \mathbb{R}^d$ et $y_i \in \mathbb{R}$.

Définition

Soit $k \leq n$. L'estimateur des k plus proches voisins de $m^\star(x)$ est défini par

$$m_{n,k}(x) = \frac{1}{k} \sum_{i \in \text{kppv}(x)} y_i$$

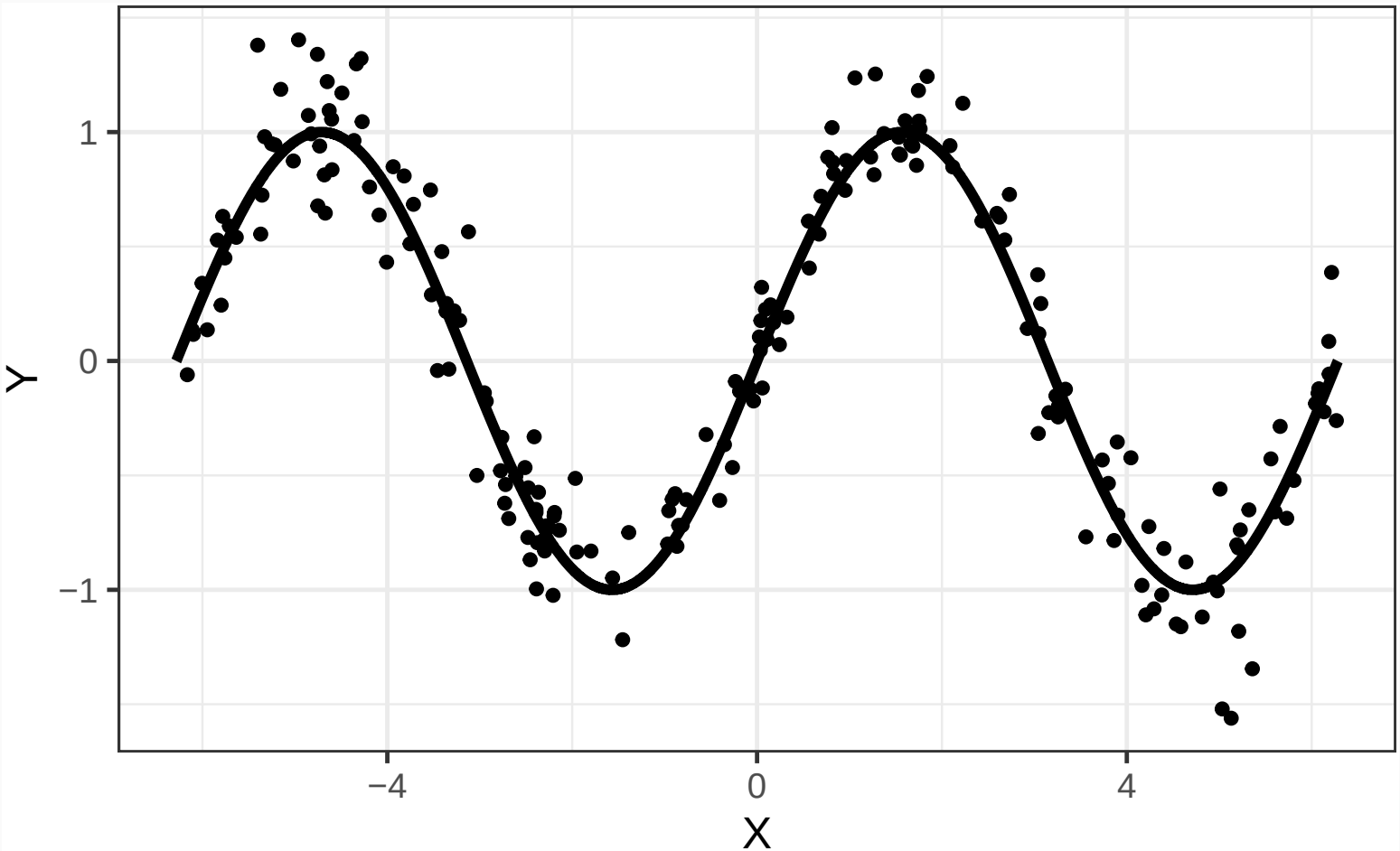
avec

$$\text{kppv}(x) = \{i \leq n : \|x - x_i\| \leq \|x - x_{(k)}\|\}$$

et $\|x - x_{(k)}\|$ la k^{e} plus petite valeur parmi $\{\|x - x_1\|, \ldots, \|x - x_n\|\}$.

Exemple

- On veut estimer la fonction sinus à partir du nuage de points

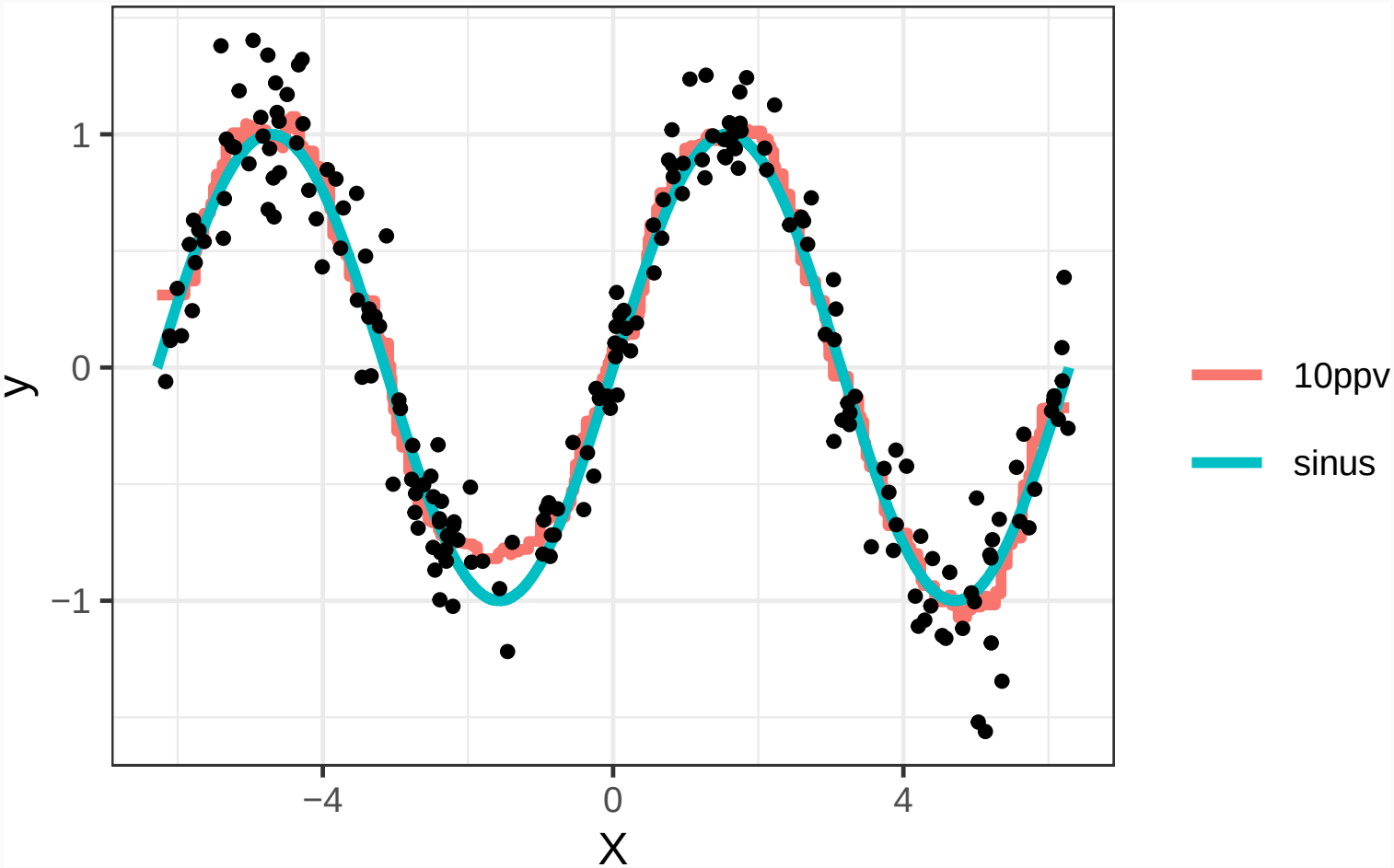


- La fonction kknn du package kknn permet d'entrainer l'algorithme des kppv

```
> library(kknn)
> knn10 <- kknn(Y~.,train=nuage_sinus,test=grille.sinus,
+               k=10,kernel="rectangular")
```

- La fonction kknn du package kknn permet d'entrainer l'algorithme des kppv

```
> library(kknn)
> knn10 <- kknn(Y~.,train=nuage_sinus,test=grille.sinus,
+               k=10,kernel="rectangular")
```



kppv en classification binaire

- $(x_1, y_1), \ldots, (x_n, y_n)$ avec $x_i \in \mathbb{R}^d$ et $y_i \in \{0, 1\}$.

Définition

Soit $k \leq n$. L'algorithme des k plus proches voisins est défini par

$$g_{n,k}(x) = \begin{cases} 1 & \text{si } \sum_{i \in \text{kppv}(x)} 1_{y_i=1} \geq \sum_{i \in \text{kppv}(x)} 1_{y_i=0} \\ 0 & \text{sinon.} \end{cases}$$

pour la prévision des groupes

kppv en classification binaire

- $(x_1, y_1), \ldots, (x_n, y_n)$ avec $x_i \in \mathbb{R}^d$ et $y_i \in \{0, 1\}$.

Définition

Soit $k \leq n$. L'algorithme des k plus proches voisins est défini par

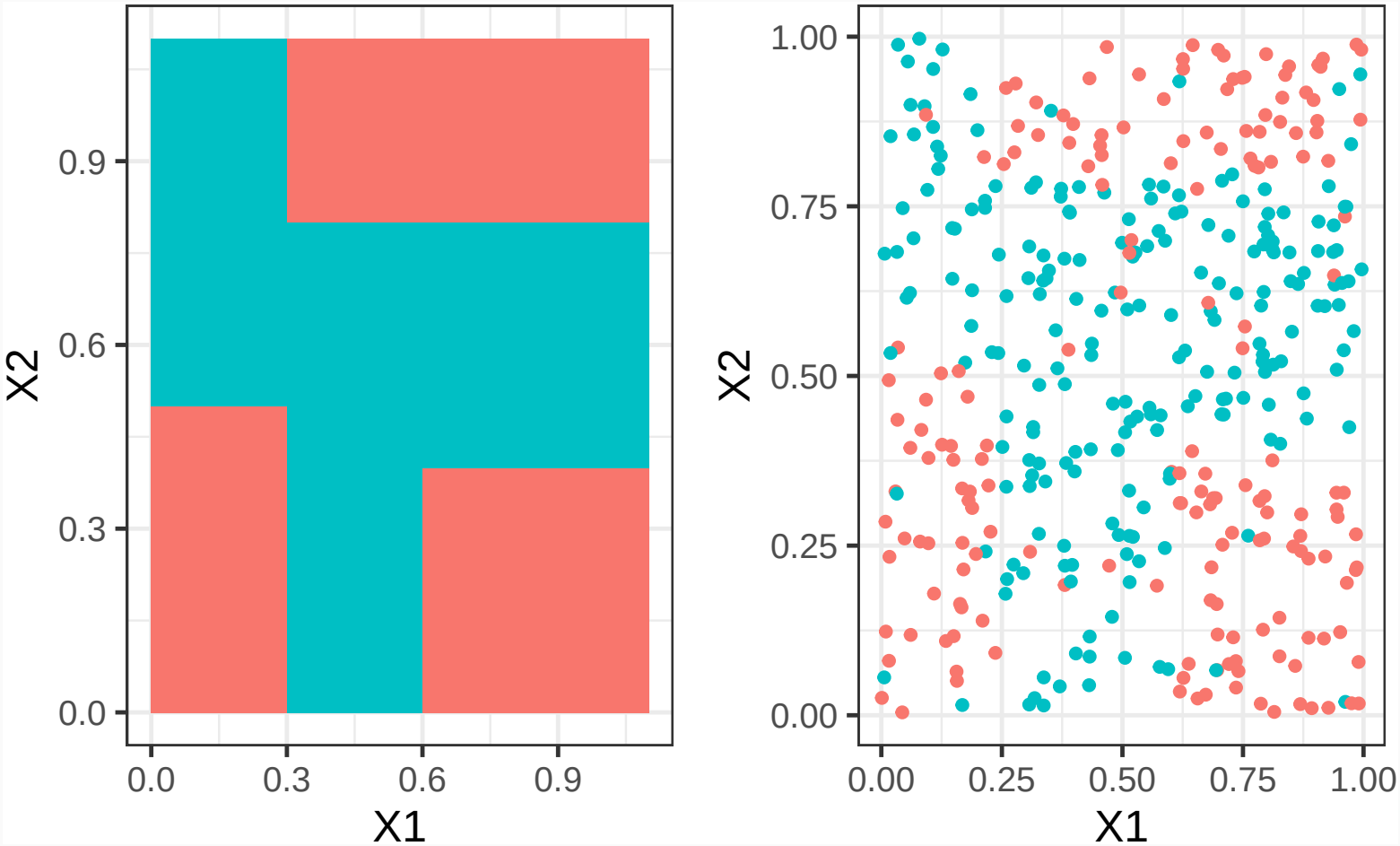
$$g_{n,k}(x) = \begin{cases} 1 & \text{si } \sum_{i \in \text{kppv}(x)} 1_{y_i=1} \geq \sum_{i \in \text{kppv}(x)} 1_{y_i=0} \\ 0 & \text{sinon.} \end{cases}$$

pour la prévision des groupes et par

$$S_{n,k}(x) = \frac{1}{|\text{kppv}(x)|} \sum_{i \in \text{kppv}(x)} 1_{y_i=1}$$

pour la prévision de la probabilité $\mathrm{P}(Y = 1 | X = x)$.

Exemple

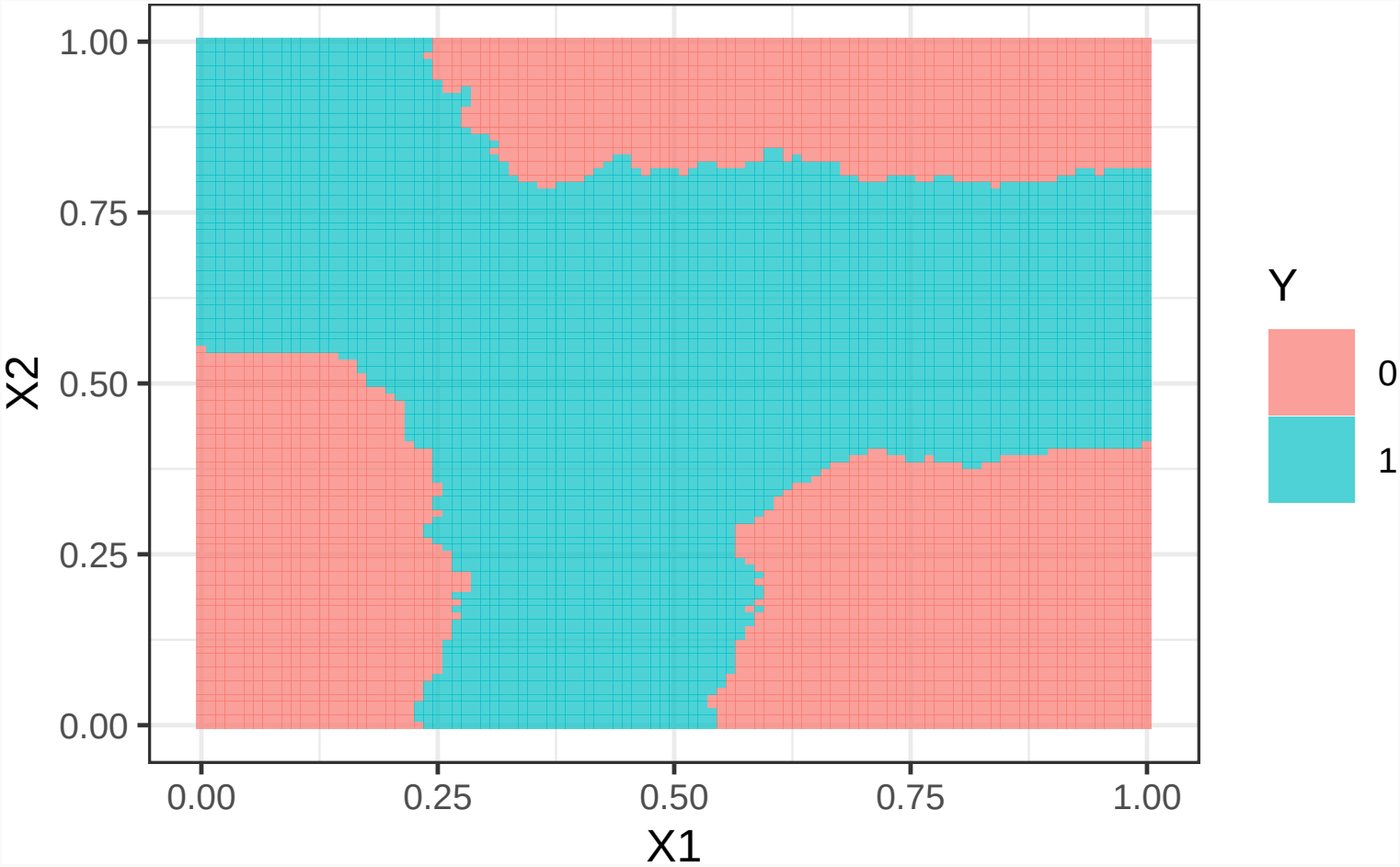

0.9
0.6
0.3
0.0
X2
0.0
0.3
0.6
0.9
X1
1.00
0.75
0.50
0.25
0.00
X2
0.00
0.25
0.50
0.75
1.00
X1

- La fonction kknn du package kknn permet d'entrainer l'algorithme des kppv

```
> prev <- kknn(Y~.,train=ex.classif2D,test=px,k=25,
+              kernel="rectangular")$fitted.values
```

- La fonction kknn du package kknn permet d'entrainer l'algorithme des kppv

```
> prev <- kknn(Y~.,train=ex.classif2D,test=px,k=25,
+              kernel="rectangular")$fitted.values
```



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.

- Une fonction de perte $\ell : \mathcal{Y} \times \widetilde{\mathcal{Y}} \to \mathbb{R}^+$ dépend de l'espace des observations $\mathcal{Y}$ et de celui des prévisions $\widetilde{\mathcal{Y}}$.

- Une fonction de perte $\ell : \mathcal{Y} \times \widetilde{\mathcal{Y}} \to \mathbb{R}^+$ dépend de l'espace des observations $\mathcal{Y}$ et de celui des prévisions $\widetilde{\mathcal{Y}}$.
- On distingue 3 catégories de fonction de perte en fonction de ces espaces :
  1. Prévisions numériques : problème de régression où on cherche à prédire la valeur de Y : $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$;

- Une fonction de perte $\ell : \mathcal{Y} \times \widetilde{\mathcal{Y}} \to \mathbb{R}^+$ dépend de l'espace des observations $\mathcal{Y}$ et de celui des prévisions $\widetilde{\mathcal{Y}}$.
- On distingue 3 catégories de fonction de perte en fonction de ces espaces :
  1. Prévisions numériques : problème de régression où on cherche à prédire la valeur de Y : $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$;
  2. Prévision de groupes : problème de classification où on veut prédire un label : $\ell : \{1, \ldots, K\} \times \{1, \ldots, K\} \to \mathbb{R}^+$;

- Une fonction de perte $\ell : \mathcal{Y} \times \widetilde{\mathcal{Y}} \to \mathbb{R}^+$ dépend de l'espace des observations $\mathcal{Y}$ et de celui des prévisions $\widetilde{\mathcal{Y}}$.
- On distingue 3 catégories de fonction de perte en fonction de ces espaces :
  1. Prévisions numériques : problème de régression où on cherche à prédire la valeur de Y : $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$;
  2. Prévision de groupes : problème de classification où on veut prédire un label : $\ell : \{1, \ldots, K\} \times \{1, \ldots, K\} \to \mathbb{R}^+$;
  3. Prévision de probabilités : problème de classification où on veut prédire les probabilités $\mathrm{P}(Y = k|X = x)$: $\ell : \{1, \ldots, K\} \times \mathbb{R}^K \to \mathbb{R}^+$.

Régression

- $\mathcal{Y} = \mathbb{R}$, une prévision = un réel $\Longrightarrow$ $m : \mathcal{X} \to \mathbb{R}$;

Régression

- $\mathcal{Y} = \mathbb{R}$, une prévision = un réel $\Longrightarrow$ $m : \mathcal{X} \to \mathbb{R}$;
- Une perte = une distance entre deux nombres, par exemple la perte quadratique :

$$\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$$
$$(y, y') \mapsto (y - y')^2$$

- Le risque (risque quadratique) est alors donné par

$$\mathcal{R}(m) = \mathsf{E}[(Y - m(X)]^2$$

Régression

- $\mathcal{Y} = \mathbb{R}$, une prévision = un réel $\Longrightarrow$ $m : \mathcal{X} \to \mathbb{R}$;
- Une perte = une distance entre deux nombres, par exemple la perte quadratique :

$$\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$$
$$(y, y') \mapsto (y - y')^2$$

- Le risque (risque quadratique) est alors donné par

$$\mathcal{R}(m) = \mathsf{E}[(Y - m(X)]^2$$

- et la fonction optimale (inconnue), appelée fonction de régression, par

$$m^\star(x) = \mathsf{E}[Y|X = x].$$

Classification

- $\mathcal{Y} = \{1, \ldots, K\}$, une prévision = un groupe $\Longrightarrow g : \mathcal{X} \to \{1, \ldots, K\}$;
- Une perte = 1 coût pour une mauvaise prévision, par exemple la perte indicatrice

$$\ell : \{1, \ldots, K\} \times \{1, \ldots, K\} \to \mathbb{R}^+$$
$$(y, y') \mapsto 1_{y \neq y'}$$

- Le risque (erreur de classification) est alors donné par

$$\mathcal{R}(g) = \mathsf{E}[1_{g(X) \neq Y}] = \mathsf{P}(g(X) \neq Y).$$

- et la fonction optimale (inconnue), appelée règle de Bayes, par

$$g^\star(x) = \operatorname*{argmax}_k \mathsf{P}(Y = k | X = x).$$

Classification binaire

- $\mathcal{Y} = \{-1, 1\}$, une prévision = un groupe $\Longrightarrow g : \mathcal{X} \to \{-1, 1\}$.
- Ce cadre permet une analyse plus fine des différents types d'erreur.

Classification binaire

- $\mathcal{Y} = \{-1, 1\}$, une prévision = un groupe $\Longrightarrow g : \mathcal{X} \to \{-1, 1\}$.
- Ce cadre permet une analyse plus fine des différents types d'erreur.
- En effet, seules 4 situations peuvent se produire

	$g(x) = -1$	$g(x) = 1$
$y = -1$	VN	FP
$y = 1$	FN	VP

- On peut les quantifier en terme de probabilités.

Pour aller plus vite

Erreurs binaires

- Spécificité $\Longrightarrow$ bien prédire les négatifs :

$$\mathrm{sp}(g) = \mathrm{P}(g(X) = -1 | Y = -1),$$

- Sensibilité $\Longrightarrow$ bien prédire les positifs :

$$\mathrm{se}(g) = \mathrm{P}(g(X) = 1 | Y = 1),$$

- Taux de faux négatifs $\Longrightarrow$ prédire négatif à tort :

$$\mathrm{fn}(g) = \mathrm{P}(g(X) = -1 | Y = 1),$$

- Taux de faux positifs $\Longrightarrow$ prédire positif à tort :

$$\mathrm{fp}(g) = \mathrm{P}(g(X) = 1 | Y = -1).$$

Erreurs binaires

- Spécificité $\Longrightarrow$ bien prédire les négatifs :

$$\mathrm{sp}(g) = \mathrm{P}(g(X) = -1 | Y = -1),$$

- Sensibilité $\Longrightarrow$ bien prédire les positifs :

$$\mathrm{se}(g) = \mathrm{P}(g(X) = 1 | Y = 1),$$

- Taux de faux négatifs $\Longrightarrow$ prédire négatif à tort :

$$\mathrm{fn}(g) = \mathrm{P}(g(X) = -1 | Y = 1),$$

- Taux de faux positifs $\Longrightarrow$ prédire positif à tort :

$$\mathrm{fp}(g) = \mathrm{P}(g(X) = 1 | Y = -1).$$

Critères binaires

De nombreux critères s'obtiennent en combinant ces probabilités :

$$\mathrm{EC}(g) = \mathrm{P}(g(X) \neq Y) = \mathrm{fp}(g)\mathrm{P}(Y = -1) + \mathrm{fn}(g)\mathrm{P}(Y = 1).$$

Quelques critères binaires

- Balanced Accuracy :

$$\frac{1}{2}\mathsf{P}(g(X)=-1|Y=-1)+\frac{1}{2}\mathsf{P}(g(X)=1|Y=1)=\frac{1}{2}(\mathsf{se}(g)+\mathsf{sp}(g)).$$

Quelques critères binaires

- Balanced Accuracy :

$$\frac{1}{2}\mathrm{P}(g(X)=-1|Y=-1)+\frac{1}{2}\mathrm{P}(g(X)=1|Y=1)=\frac{1}{2}(\mathrm{se}(g)+\mathrm{sp}(g)).$$

- F_1-score :

$$2\,\frac{\text{Precision}\ \times \text{Recall}}{\text{Precision}\ + \text{Recall}},$$

avec

$$\text{Precision}\ \mathrm{P}(Y=1|g(X)=1) \quad \text{et} \quad \text{Recall}=\mathrm{P}(g(X)=1|Y=1).$$

- Kappa de Cohen...

Quelques critères binaires

- Balanced Accuracy :

$$\frac{1}{2}\mathrm{P}(g(X)=-1|Y=-1)+\frac{1}{2}\mathrm{P}(g(X)=1|Y=1)=\frac{1}{2}(\mathrm{se}(g)+\mathrm{sp}(g)).$$

- F_1-score :

$$2\,\frac{\text{Precision}\ \times \text{Recall}}{\text{Precision}\ + \text{Recall}},$$

avec

$$\text{Precision}\ \mathrm{P}(Y=1|g(X)=1) \quad \text{et} \quad \text{Recall}=\mathrm{P}(g(X)=1|Y=1).$$

- Kappa de Cohen...

Remarque

Mieux adapté que l'erreur de classification au cas de données déséquilibrées.

Classification (pour des probabilités)

- $\mathcal{Y} = \{1, \ldots, K\}$, une prévision = $K - 1$ probabilités
  $p_k(x) = \mathrm{P}(Y = k | X = x), k = 1, \ldots, K - 1.$

Classification (pour des probabilités)

- $\mathcal{Y} = \{1, \ldots, K\}$, une prévision = $K - 1$ probabilités
  $p_k(x) = \mathrm{P}(Y = k | X = x), k = 1, \ldots, K - 1.$
- Les fonctions de perte sont généralement définies comme généralisation de pertes spécifiques au problème de classification binaire.

Classification (pour des probabilités)

- $\mathcal{Y} = \{1, \ldots, K\}$, une prévision = $K - 1$ probabilités
  $p_k(x) = \mathrm{P}(Y = k | X = x), k = 1, \ldots, K - 1.$
- Les fonctions de perte sont généralement définies comme généralisation de pertes spécifiques au problème de classification binaire.
- Classification binaire avec $\mathcal{Y} = \{-1, 1\}$ et $S : \mathcal{X} \to \mathbb{R}$
  ($S(x) = \mathrm{P}(Y = 1 | X = x)$ ou une transformation bijective de cette probabilité) $\Longrightarrow$ fonction de score.

Fonction de score

- Objectif d'un score : ordonner

Fonction de score

- Objectif d'un score : ordonner

- avant (d'éventuellement) classer en fixant un seuil $s \in \mathbb{R}$:

$$g_s(x) = \begin{cases} 1 & \text{si } S(x) > s \\ -1 & \text{sinon.} \end{cases}$$

Fonction de score

- Objectif d'un score : ordonner

- avant (d'éventuellement) classer en fixant un seuil $s \in \mathbb{R}$:

$$g_s(x) = \begin{cases} 1 & \text{si } S(x) > s \\ -1 & \text{sinon.} \end{cases}$$

- Pour un seuil s donné, on a les erreurs (FP et FN)

$$\alpha(s) = \mathrm{P}(S(X) > s | Y = -1) = 1 - \mathrm{sp}(s)$$

et

$$\beta(s) = \mathrm{P}(S(X) \leq s | Y = 1) = 1 - \mathrm{se}(s).$$

Courbe ROC

- Idée : s'affranchir du choix de s en visualisant les erreurs $\alpha(s)$ et $\beta(s)$ sur un graphe 2D pour toutes les valeurs de s.

Courbe ROC

- Idée : s'affranchir du choix de s en visualisant les erreurs $\alpha(s)$ et $\beta(s)$ sur un graphe 2D pour toutes les valeurs de s.

Définition

La courbe ROC de S est la courbe paramétrée par les valeurs de seuil s dont les abscisses et ordonnées sont définies par

$$\begin{cases} x(s) = \mathrm{P}(S(X) > s | Y = -1) = \alpha(s) \\ y(s) = \mathrm{P}(S(X) > s | Y = 1) = 1 - \beta(s). \end{cases}$$

Courbe ROC

- Idée : s'affranchir du choix de s en visualisant les erreurs $\alpha(s)$ et $\beta(s)$ sur un graphe 2D pour toutes les valeurs de s.

Définition

La courbe ROC de S est la courbe paramétrée par les valeurs de seuil s dont les abscisses et ordonnées sont définies par

$$\begin{cases} x(s) = \mathrm{P}(S(X) > s | Y = -1) = \alpha(s) \\ y(s) = \mathrm{P}(S(X) > s | Y = 1) = 1 - \beta(s). \end{cases}$$

Visualisation

- Abscisses : les faux positifs ou la spécificité ;
- Ordonnées : les faux négatifs ou la sensibilité.

Analyse de la courbe ROC

- Une proba est entre 0 et 1 $\Longrightarrow$ ROC vit dans le carré $[0,1]^2$.

Analyse de la courbe ROC

- Une proba est entre 0 et 1 $\Longrightarrow$ ROC vit dans le carré $[0,1]^2$.
- $x(-\infty) = y(-\infty) = 1$ et $x(+\infty) = y(+\infty) = 1 \Longrightarrow$ ROC part du point $(1,1)$ pour arriver en $(0,0)$.

Analyse de la courbe ROC

- Une proba est entre 0 et 1 $\Longrightarrow$ ROC vit dans le carré $[0,1]^2$.
- $x(-\infty) = y(-\infty) = 1$ et $x(+\infty) = y(+\infty) = 1 \Longrightarrow$ ROC part du point $(1,1)$ pour arriver en $(0,0)$.
- ROC parfaite : il existe $s^\star$ tel que $\alpha(s^\star) = \beta(s^\star) = 0 \Longrightarrow$ ROC est définie par l'union des segments

$$[(1,1);(0,1)] \quad \text{et} \quad [(0,1);(0,0)].$$

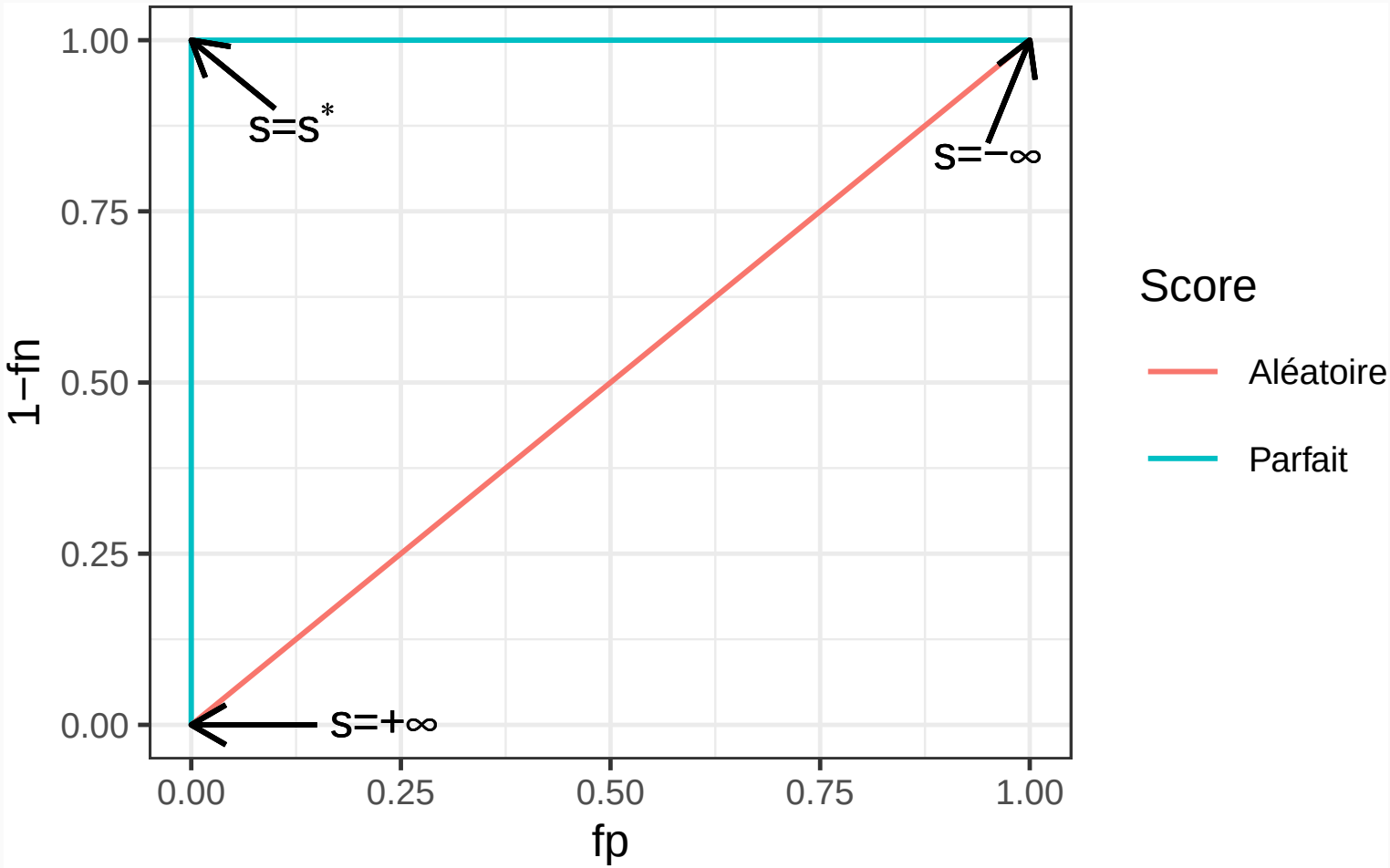
Analyse de la courbe ROC

- Une proba est entre 0 et 1 $\Longrightarrow$ ROC vit dans le carré $[0,1]^2$.
- $x(-\infty) = y(-\infty) = 1$ et $x(+\infty) = y(+\infty) = 1 \Longrightarrow$ ROC part du point $(1,1)$ pour arriver en $(0,0)$.
- ROC parfaite : il existe $s^\star$ tel que $\alpha(s^\star) = \beta(s^\star) = 0 \Longrightarrow$ ROC est définie par l'union des segments

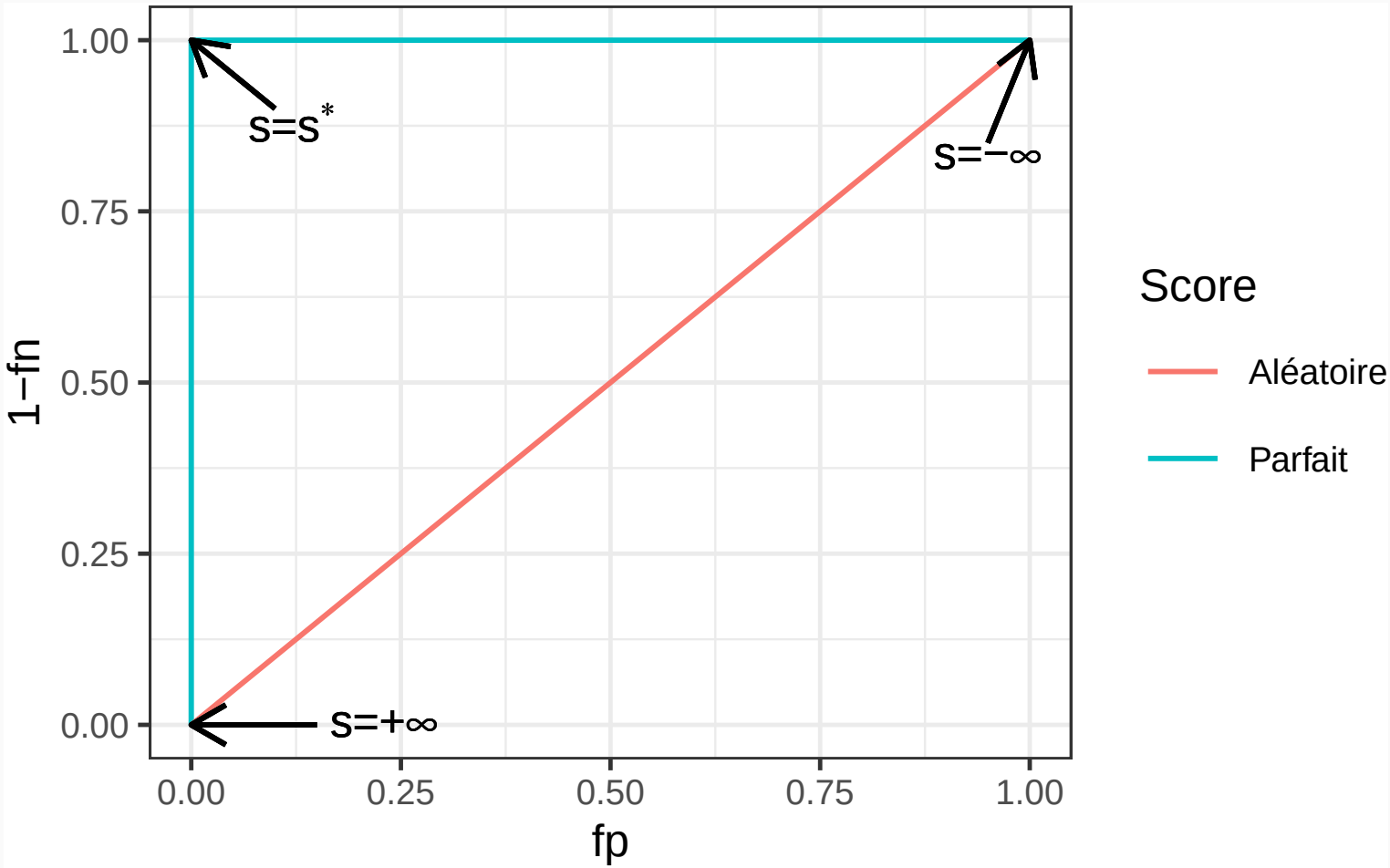
$$[(1,1);(0,1)] \quad \text{et} \quad [(0,1);(0,0)].$$

- Mauvaise ROC : $S(X)$ et Y sont indépendantes $\Longrightarrow x(s) = y(s)$ pour tout $s \in \mathbb{R}$ et ROC correspond à la première bissectrice.

Visualisation

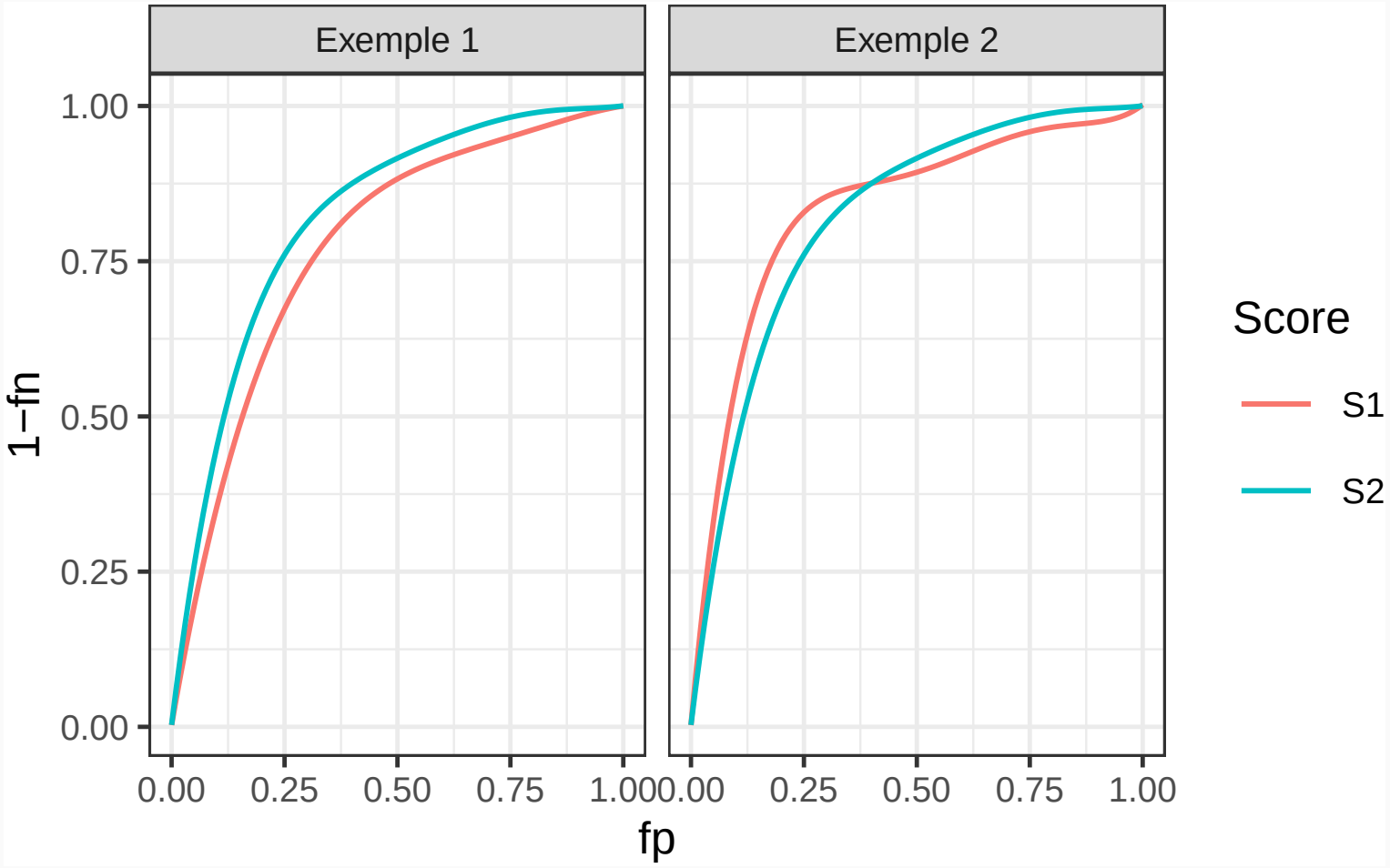


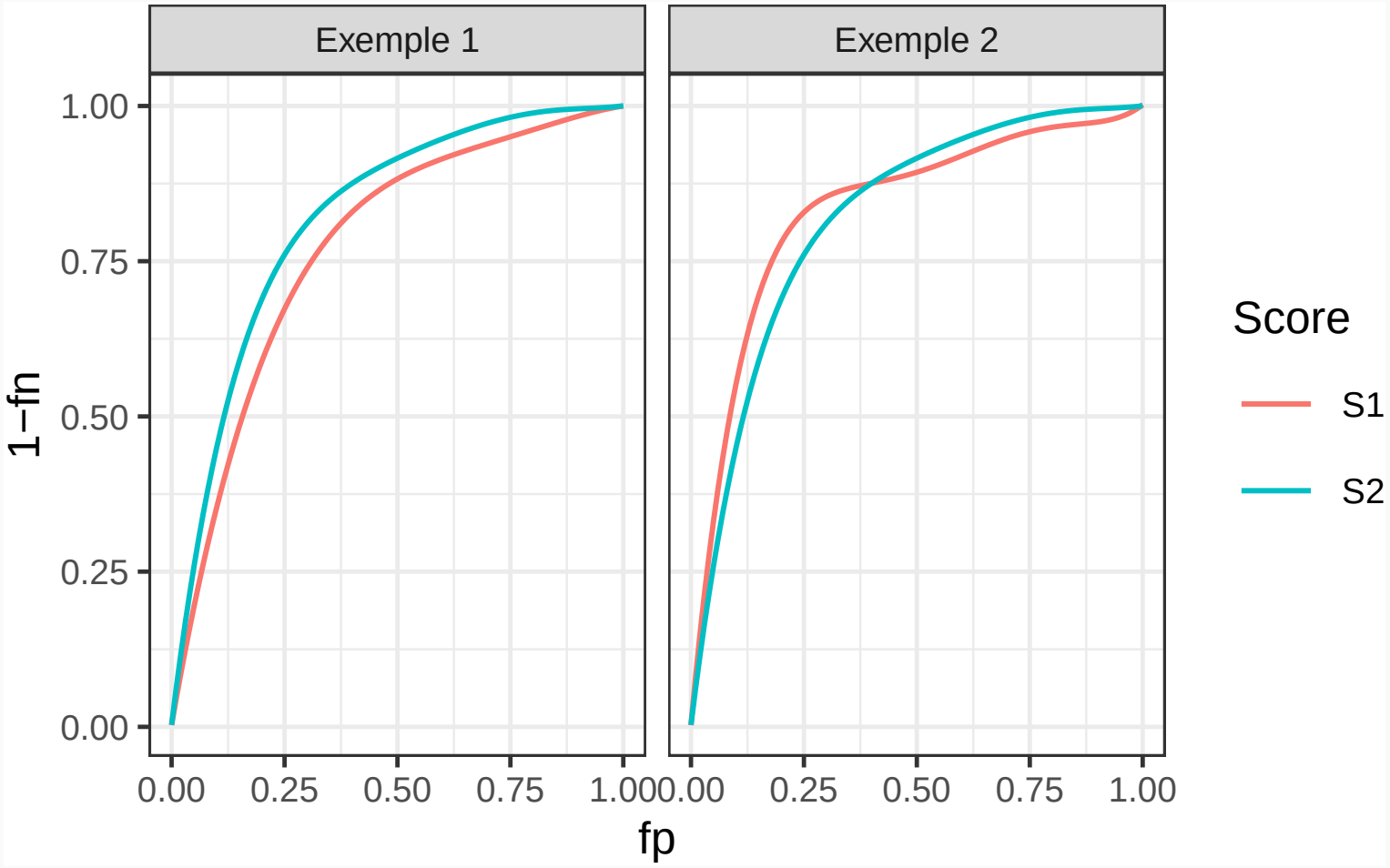
Visualisation



Interprétation

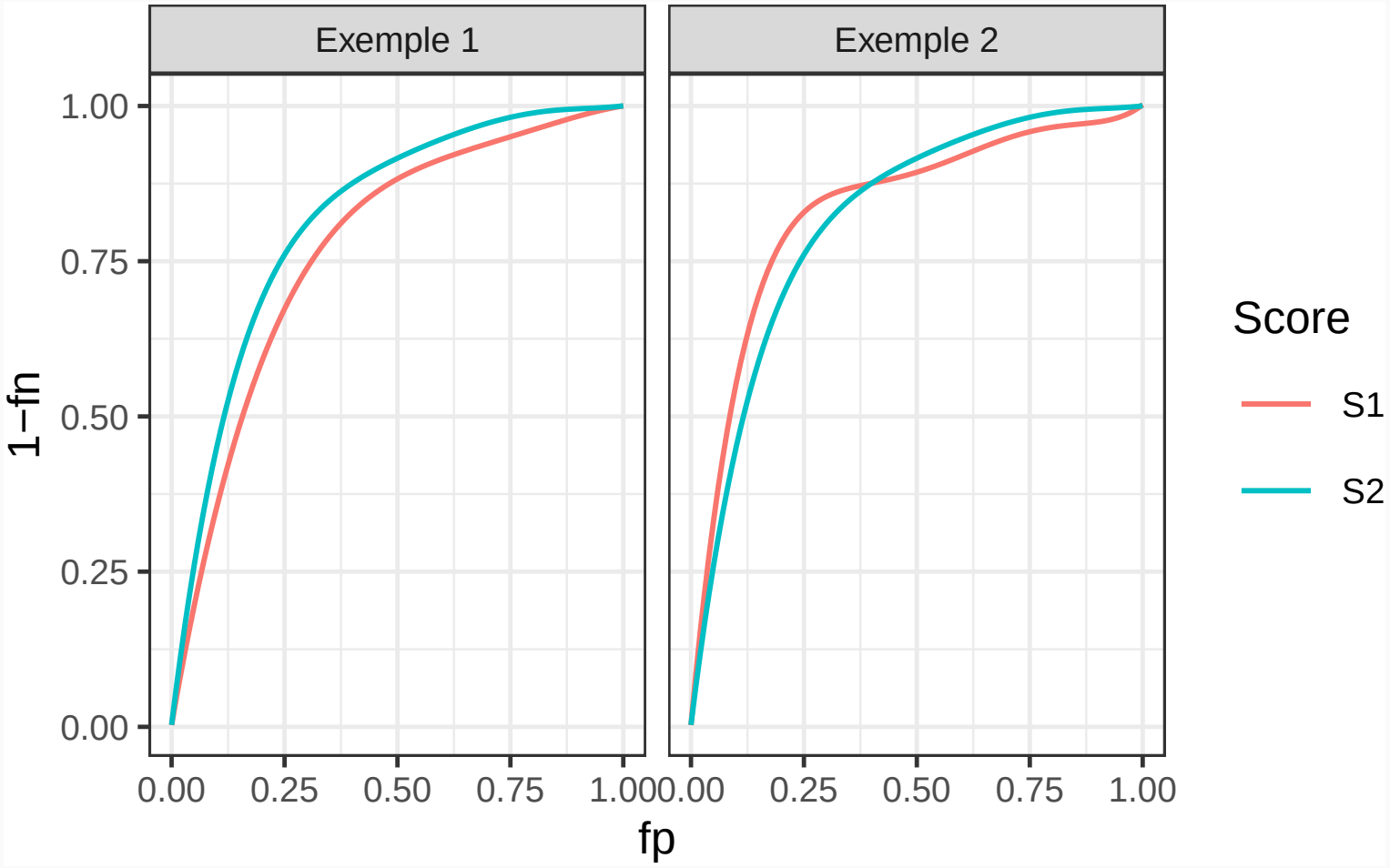
On évalue la performance d'un score par sa capacité à se rapprocher le plus vite possible de la droite $y = 1$.


Exemple 1
Exemple 2
1.00
0.75
0.50
0.25
0.00
1−fn
0.00
0.25
0.50
0.75
1.00
fp
Score
S1
S2



Comparaison

- Exemple 1 : S2 meilleur que S1.
- Exemple 2 : il y a débat...



Comparaison

- Exemple 1 : S2 meilleur que S1.
- Exemple 2 : il y a débat...
- Idée : utiliser l'aire sous la courbe.

AUC

Définition

On appelle AUC l'aire sous la courbe ROC de S.

Propriété

- $0.5 \leq \text{AUC}(S) \leq 1$.
- Plus l'AUC est grand, meilleur est le score.

Interprétation de l'AUC

Propriété

Soit (X_1, Y_1) et (X_2, Y_2) indépendants et de même loi que (X, Y), on a

$$\begin{aligned}\mathrm{AUC}(S) =& \mathrm{P}(S(X_1) > S(X_2) | Y_1 = 1, Y_2 = -1) \\ &+ \frac{1}{2}\mathrm{P}(S(X_1) = S(X_2) | Y_1 = 1, Y_2 = -1).\end{aligned}$$

En particulier si $S(X)$ est continue alors

$$\mathrm{AUC}(S) = \mathrm{P}(S(X_1) \geq S(X_2) | Y_1 = 1, Y_2 = -1).$$

Interprétation

- L'AUC correspond à la probabilité que le score ordonne correctement deux observations prélevées aléatoirement dans les groupes -1 et 1.

Interprétation de l'AUC

Propriété

Soit (X_1, Y_1) et (X_2, Y_2) indépendants et de même loi que (X, Y), on a

$$\begin{aligned}\text{AUC}(S) =& \text{P}(S(X_1) > S(X_2) | Y_1 = 1, Y_2 = -1) \\ &+ \frac{1}{2}\text{P}(S(X_1) = S(X_2) | Y_1 = 1, Y_2 = -1).\end{aligned}$$

En particulier si $S(X)$ est continue alors

$$\text{AUC}(S) = \text{P}(S(X_1) \geq S(X_2) | Y_1 = 1, Y_2 = -1).$$

Interprétation

- L'AUC correspond à la probabilité que le score ordonne correctement deux observations prélevées aléatoirement dans les groupes -1 et 1.
- $\text{AUC}(S) = 0.9 \Longrightarrow$ dans 90% des cas, le score d'un individu positif sera plus grand que le score d'un individu négatif.

Perte AUC et score optimal

- Remarquons que

$$\mathrm{AUC}(S) = \mathrm{E}[1_{S(X_1)>S(X_2)} | Y_1 = 1, Y_2 = -1].$$

Perte AUC et score optimal

- Remarquons que

$$\mathrm{AUC}(S) = \mathrm{E}[1_{S(X_1)>S(X_2)} | Y_1 = 1, Y_2 = -1].$$

- L'AUC peut donc s'écrire comme l'espérance d'une fonction de perte particulière

$$\ell((y_1, y_2), (S(x_1), S(x_2))) = 1_{S(x_1)>S(x_2)} \quad \text{avec} \quad y_1 = 1 \text{ et } y_2 = -1.$$

Perte AUC et score optimal

- Remarquons que

$$\mathrm{AUC}(S) = \mathrm{E}[1_{S(X_1)>S(X_2)} | Y_1 = 1, Y_2 = -1].$$

- L'AUC peut donc s'écrire comme l'espérance d'une fonction de perte particulière

$$\ell((y_1, y_2), (S(x_1), S(x_2))) = 1_{S(x_1)>S(x_2)} \quad \text{avec} \quad y_1 = 1 \text{ et } y_2 = -1.$$

Proposition

Le score optimal par rapport à l'AUC est

$$S^\star(x) = \mathrm{P}(Y = 1 | X = x).$$

En effet pour tout score $S : \mathcal{X} \to \mathbb{R}$ on a

$$\mathrm{AUC}(S^\star) \geq \mathrm{AUC}(S).$$

Résumé

	Perte $\ell(y, f(x))$	Risque $\mathcal{R}(f)$	Champion $f^\star$
Régression	$(y - m(x))^2$	$\mathrm{E}[Y - m(X)]^2$	$\mathrm{E}[Y \mid X = x]$
Classif. binaire	$1_{y \neq g(x)}$	$\mathrm{P}(Y \neq g(X))$	Bayes
Scoring	$1_{S(x_1) > S(x_2)}$	$\mathrm{AUC}(S)$	$\mathrm{P}(Y = 1 \mid X = x)$

Le package yardstick

- Nous verrons dans la section suivante que ces critères se calculent (ou plutôt s'estiment) en confrontant les valeurs observées y_i aux valeurs prédites d'un algorithme. Par exemple

```
> head(tbl)
## # A tibble: 6 x 3
##   obs   proba class
##   <fct> <dbl> <fct>
## 1 0     0.117 0
## 2 0     0.288 0
## 3 1     0.994 1
## 4 0     0.528 1
## 5 0     0.577 1
## 6 1     0.997 1
```

Le package yardstick

- Nous verrons dans la section suivante que ces critères se calculent (ou plutôt s'estiment) en confrontant les valeurs observées y_i aux valeurs prédites d'un algorithme. Par exemple

```
> head(tbl)
## # A tibble: 6 x 3
##   obs   proba class
##   <fct> <dbl> <fct>
## 1 0     0.117 0
## 2 0     0.288 0
## 3 1     0.994 1
## 4 0     0.528 1
## 5 0     0.577 1
## 6 1     0.997 1
```

- Le package yardstick contient un ensemble de fonctions qui permettent de calculer les critères :

https://yardstick.tidymodels.org/articles/metric-types.html

Exemples

- Erreur de classification (ou plutôt accuracy) avec accuracy :

```
> library(yardstick)
> tbl %>% accuracy(truth=obs,estimate=class)
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.834
```

Exemples

- Erreur de classification (ou plutôt accuracy) avec accuracy :

```
> library(yardstick)
> tbl %>% accuracy(truth=obs,estimate=class)
## # A tibble: 1 x 3
##   .metric  .estimator .estimate
##   <chr>    <chr>          <dbl>
## 1 accuracy binary         0.834
```

- AUC avec roc_auc

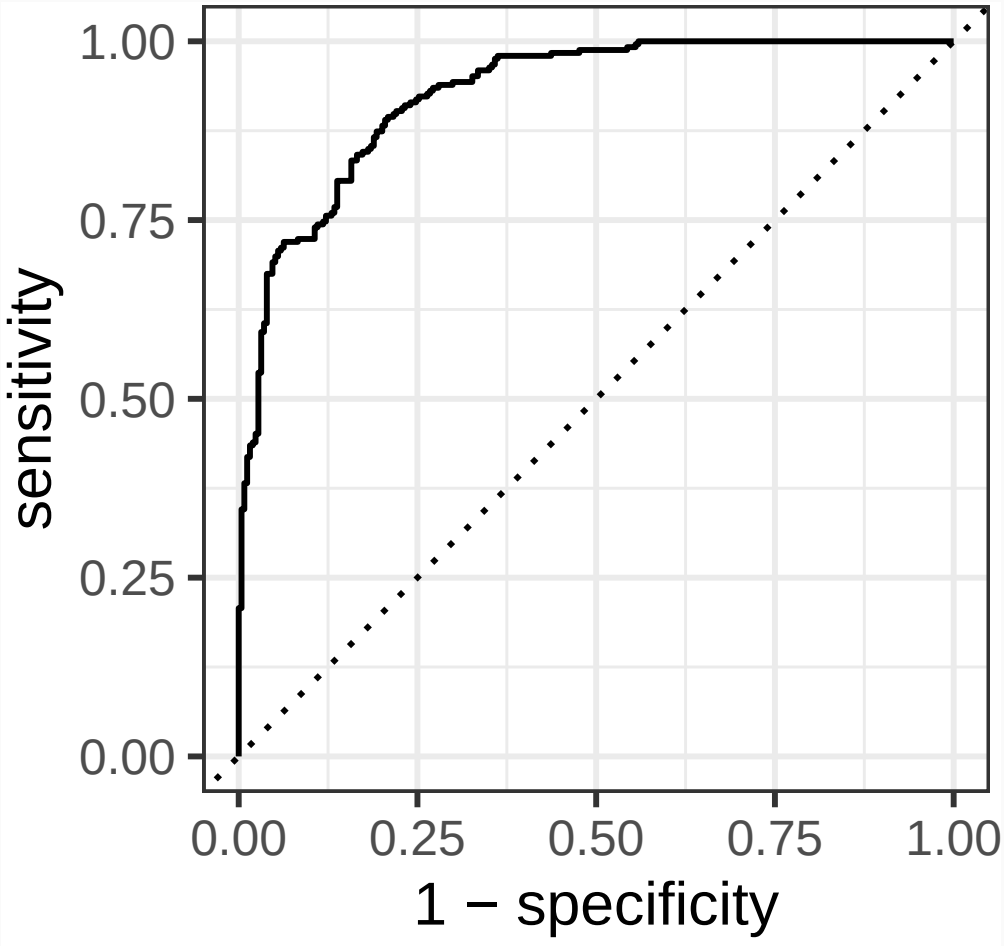
```
> tbl %>% roc_auc(truth=obs,estimate=proba,event_level="second")
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.926
```

- On peut aussi définir plusieurs critères :

```
> multi_metric <- metric_set(accuracy,bal_accuracy,f_meas,kap)
> tbl %>% multi_metric(truth=obs,estimate=class,event_level="second")
## # A tibble: 4 x 3
##   .metric      .estimator .estimate
##   <chr>        <chr>          <dbl>
## 1 accuracy     binary         0.834
## 2 bal_accuracy binary         0.834
## 3 f_meas       binary         0.832
## 4 kap          binary         0.668
```

- et tracer des courbes ROC avec roc_curve et autoplot

```
> tbl %>% roc_curve(truth=obs,estimate=proba,event_level="second") %>%
+   autoplot()
```



- La plupart des modèles statistiques renvoient des estimateurs qui dépendent de paramètres λ à calibrer.

- La plupart des modèles statistiques renvoient des estimateurs qui dépendent de paramètres λ à calibrer.

Exemples

- nombres de variables dans un modèle linéaire ou logistique.
- paramètre de pénalités pour les régressions pénalisées.
- profondeur des arbres.
- nombre de plus proches voisins.
- nombre d'itérations en boosting.
- ...

- La plupart des modèles statistiques renvoient des estimateurs qui dépendent de paramètres λ à calibrer.

Exemples

- nombres de variables dans un modèle linéaire ou logistique.
- paramètre de pénalités pour les régressions pénalisées.
- profondeur des arbres.
- nombre de plus proches voisins.
- nombre d'itérations en boosting.
- ...

Remarque importante

Le choix de ces paramètres est le plus souvent crucial pour la performance de l'estimateur sélectionné.

- Le paramètre λ à sélectionner représente la complexité du modèle :

- Le paramètre λ à sélectionner représente la complexité du modèle :

Complexité $\Longrightarrow$ compromis biais/variance

- λ petit $\Longrightarrow$ modèle peu flexible $\Longrightarrow$ mauvaise adéquation sur les données $\Longrightarrow$ biais $\nearrow$, variance $\searrow$.

- Le paramètre λ à sélectionner représente la complexité du modèle :

Complexité $\Longrightarrow$ compromis biais/variance

- λ petit $\Longrightarrow$ modèle peu flexible $\Longrightarrow$ mauvaise adéquation sur les données $\Longrightarrow$ biais $\nearrow$, variance $\searrow$.
- λ grand $\Longrightarrow$ modèle trop flexible $\Longrightarrow$ sur-ajustement $\Longrightarrow$ biais $\searrow$, variance $\nearrow$.

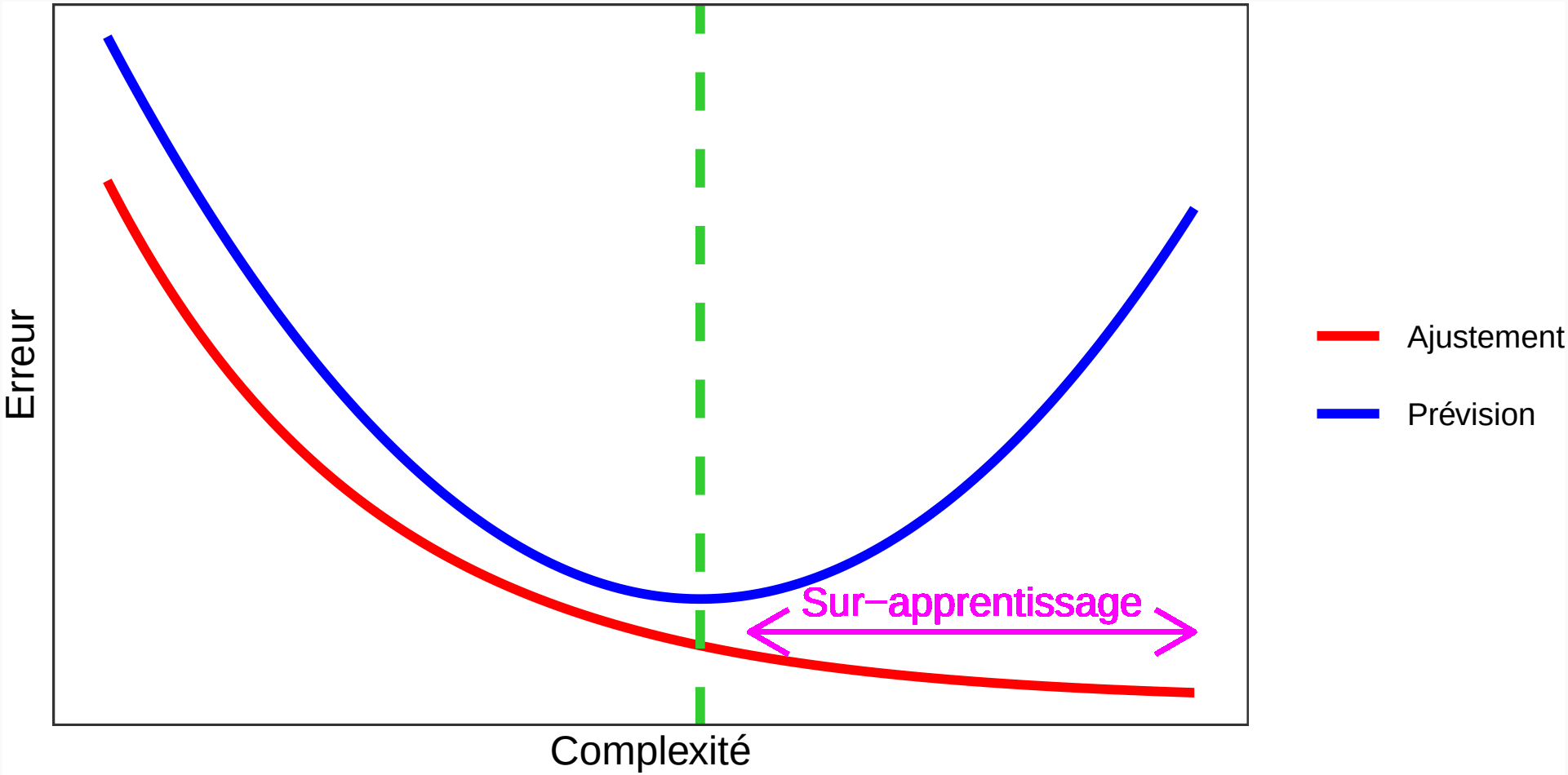
- Le paramètre λ à sélectionner représente la complexité du modèle :

Complexité $\Longrightarrow$ compromis biais/variance

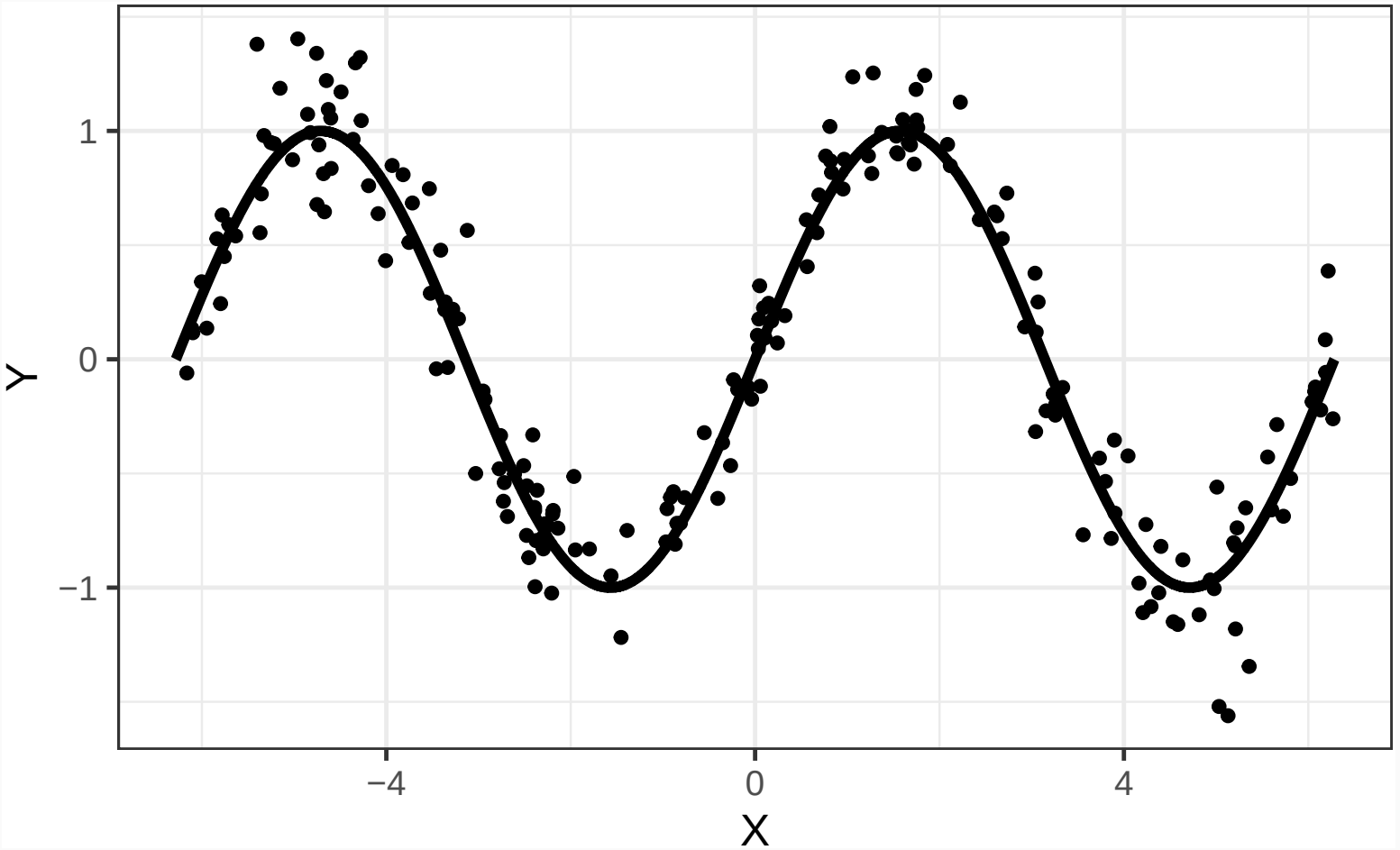
- λ petit $\Longrightarrow$ modèle peu flexible $\Longrightarrow$ mauvaise adéquation sur les données $\Longrightarrow$ biais $\nearrow$, variance $\searrow$.
- λ grand $\Longrightarrow$ modèle trop flexible $\Longrightarrow$ sur-ajustement $\Longrightarrow$ biais $\searrow$, variance $\nearrow$.

Overfitting

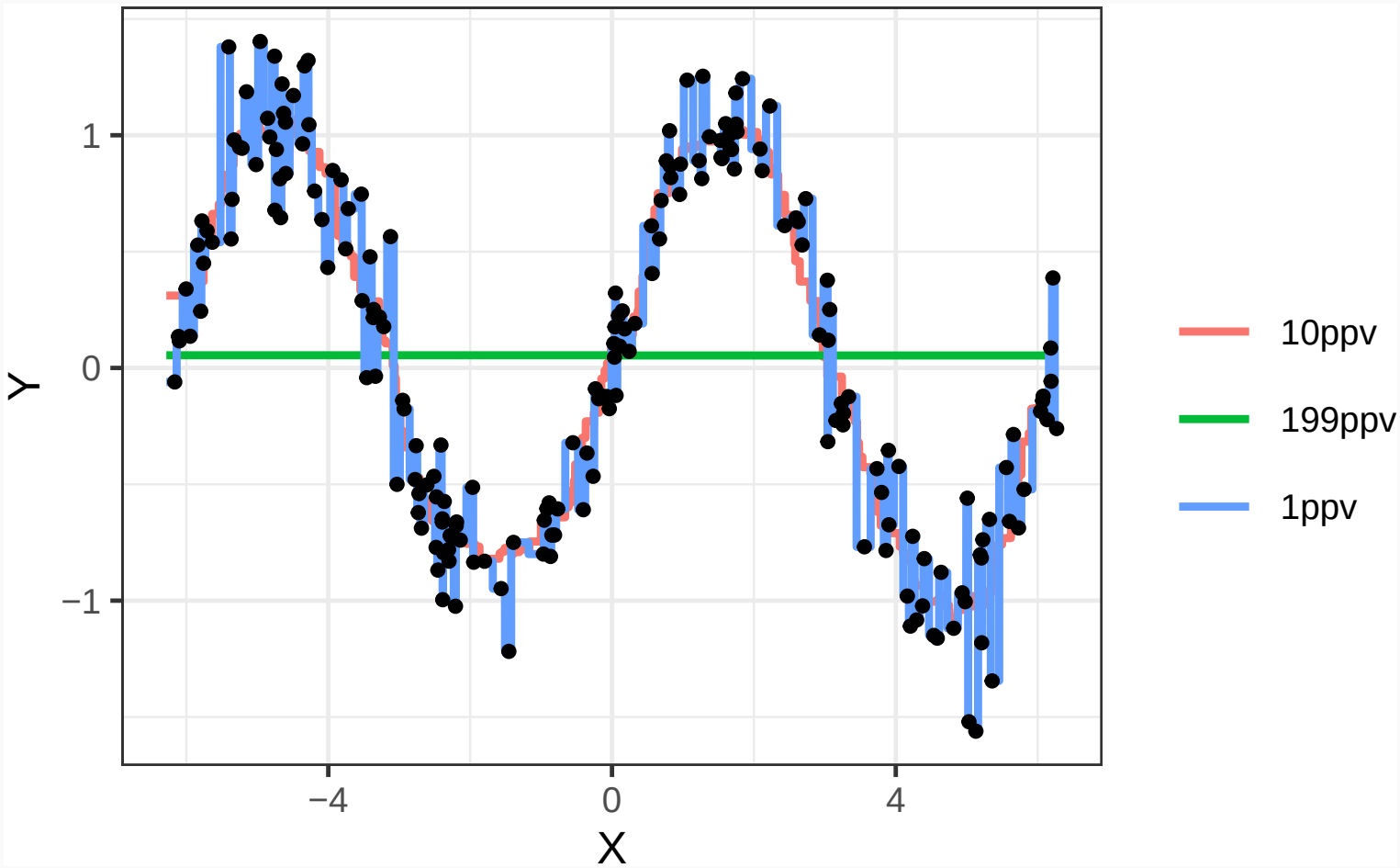
Sur-ajuster signifie que le modèle va (trop) bien ajuster les données d'apprentissage, il aura du mal à s'adapter à de nouveaux individus.


Erreur
Complexité
Sur−apprentissage
Ajustement
Prévision

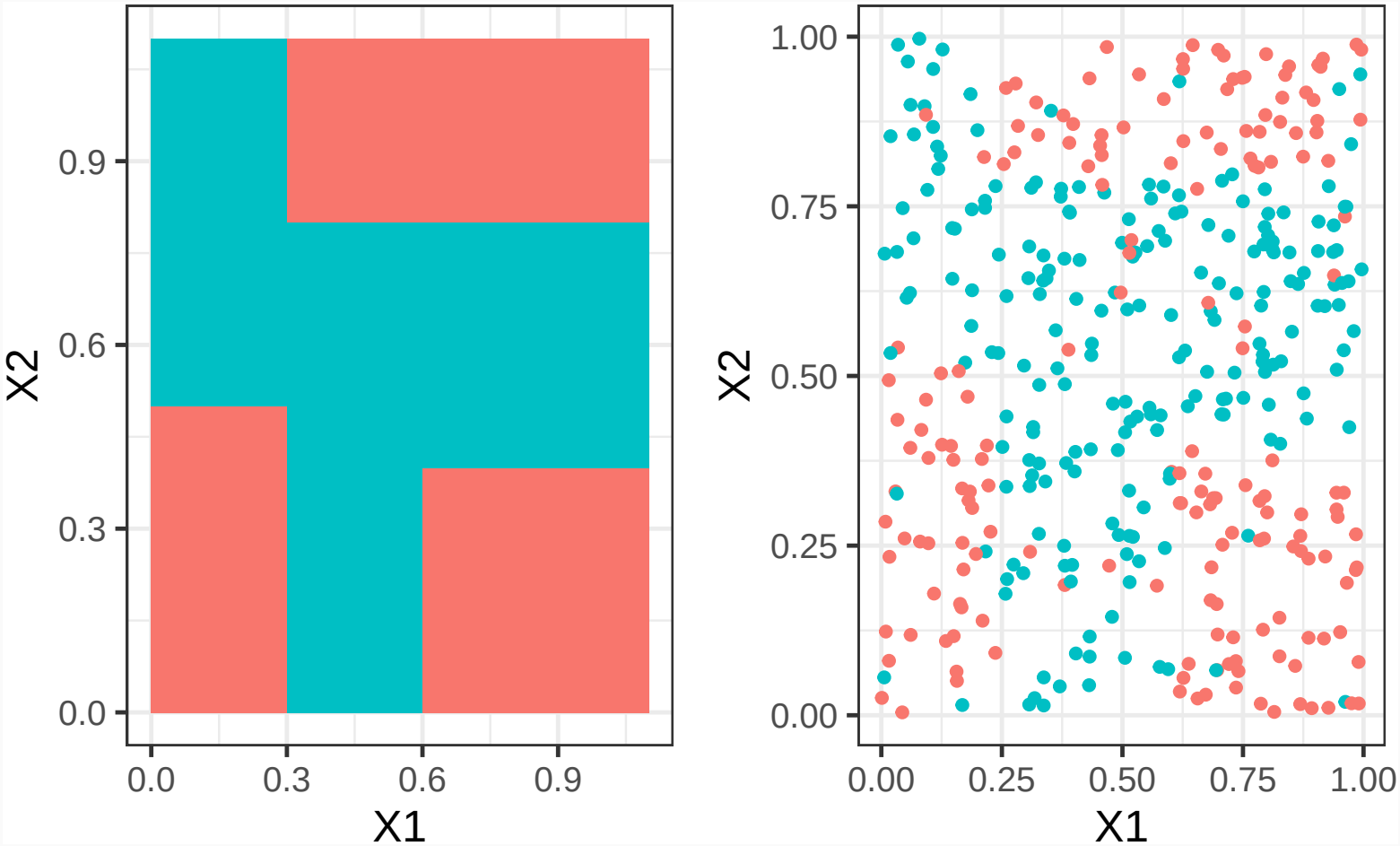
Overfitting en régression



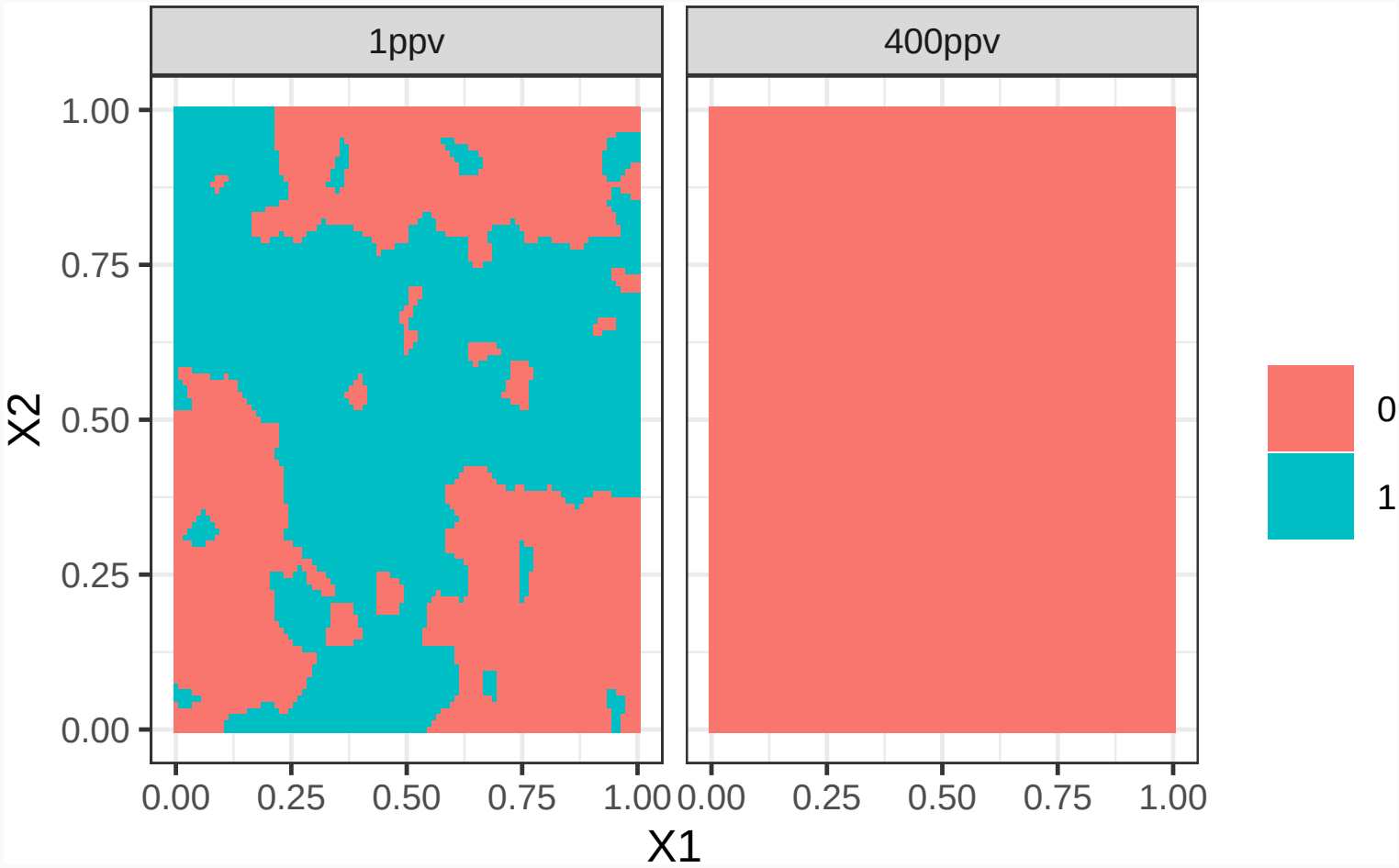
Overfitting en régression



Overfitting en classification supervisée



Overfitting en classification supervisée



Application shiny

https://lrouviere.shinyapps.io/overfitting_app/

- 1 algorithme $f_n(x, \mathcal{D}_n)$ peut être vu comme 1 estimateur de la fonction de prévision optimale $f^\star(x)$.
- Comme tout estimateur, il possède des propriétés comme

- 1 algorithme $f_n(x, \mathcal{D}_n)$ peut être vu comme 1 estimateur de la fonction de prévision optimale $f^\star(x)$.
- Comme tout estimateur, il possède des propriétés comme
  1. la variance $\mathrm{V}[f_n(x, \mathcal{D}_n)]$

- 1 algorithme $f_n(x, \mathcal{D}_n)$ peut être vu comme 1 estimateur de la fonction de prévision optimale $f^\star(x)$.
- Comme tout estimateur, il possède des propriétés comme
  1. la variance $\mathrm{V}[f_n(x, \mathcal{D}_n)] \Longrightarrow$ mesure la dispersion des prévisions au point x par rapport à la loi des données $\mathcal{D}_n$.

- 1 algorithme $f_n(x, \mathcal{D}_n)$ peut être vu comme 1 estimateur de la fonction de prévision optimale $f^\star(x)$.
- Comme tout estimateur, il possède des propriétés comme
  1. la variance $\mathrm{V}[f_n(x, \mathcal{D}_n)] \Longrightarrow$ mesure la dispersion des prévisions au point x par rapport à la loi des données $\mathcal{D}_n$.
  2. le biais $\mathrm{E}[f_n(x, \mathcal{D}_n)] - f^\star(x)$

- 1 algorithme $f_n(x, \mathcal{D}_n)$ peut être vu comme 1 estimateur de la fonction de prévision optimale $f^\star(x)$.
- Comme tout estimateur, il possède des propriétés comme
  1. la variance $\mathrm{V}[f_n(x, \mathcal{D}_n)] \Longrightarrow$ mesure la dispersion des prévisions au point x par rapport à la loi des données $\mathcal{D}_n$.
  2. le biais $\mathrm{E}[f_n(x, \mathcal{D}_n)] - f^\star(x) \Longrightarrow$ mesure l'écart entre la moyenne de ces prévisions et la fonction optimale.

Remarque

La quête de le complexité optimale d'un algorithme se retrouve dans la recherche du meilleur compromis biais/variance.

- Complexité (trop) faible

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer $\Longrightarrow \nearrow$ biais

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer $\Longrightarrow \nearrow$ biais $\Longrightarrow$ sous-apprentissage.

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer $\Longrightarrow \nearrow$ biais $\Longrightarrow$ sous-apprentissage.
- Complexité (trop) grande

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer $\Longrightarrow \nearrow$ biais $\Longrightarrow$ sous-apprentissage.

- Complexité (trop) grande $\Longrightarrow \nearrow$ sensibilité aux données d'apprentissage

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer $\Longrightarrow \nearrow$ biais $\Longrightarrow$ sous-apprentissage.

- Complexité (trop) grande $\Longrightarrow \nearrow$ sensibilité aux données d'apprentissage $\Longrightarrow \nearrow$ dispersion et donc $\nearrow$ variance mais

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer $\Longrightarrow \nearrow$ biais $\Longrightarrow$ sous-apprentissage.

- Complexité (trop) grande $\Longrightarrow \nearrow$ sensibilité aux données d'apprentissage $\Longrightarrow \nearrow$ dispersion et donc $\nearrow$ variance mais $\Longrightarrow \searrow$ biais

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer $\Longrightarrow \nearrow$ biais $\Longrightarrow$ sous-apprentissage.

- Complexité (trop) grande $\Longrightarrow \nearrow$ sensibilité aux données d'apprentissage $\Longrightarrow \nearrow$ dispersion et donc $\nearrow$ variance mais $\Longrightarrow \searrow$ biais $\Longrightarrow$ sur-apprentissage.

- Complexité (trop) faible $\Longrightarrow \searrow$ sensibilité aux données d'apprentissage $\Longrightarrow \searrow$ dispersion et donc $\searrow$ variance mais $\nearrow$ difficulté à capturer les spécificités de la fonction à estimer $\Longrightarrow \nearrow$ biais $\Longrightarrow$ sous-apprentissage.

- Complexité (trop) grande $\Longrightarrow \nearrow$ sensibilité aux données d'apprentissage $\Longrightarrow \nearrow$ dispersion et donc $\nearrow$ variance mais $\Longrightarrow \searrow$ biais $\Longrightarrow$ sur-apprentissage.

Conclusion

Le sur-apprentissage se traduit généralement par une variance trop élevée due à une trop grande complexité de l'algorithme.

L'exemple des kppv

- On peut retrouver les remarques précédentes avec des arguments mathématiques.
- Exemple des kppv en régression : sous des hypothèses standards en statistique non-paramétrique, on a

$$\mathrm{E}\|m_{n,k} - m^\star\| = \mathrm{E}\int |m_{n,k}(x) - m^\star(x)|\mu \mathrm{d}x \leq \frac{c_1}{k} + c_2\left(\frac{k}{n}\right)^{2/d}.$$

- Décomposition biais/variance.

L'exemple des kppv

- On peut retrouver les remarques précédentes avec des arguments mathématiques.
- Exemple des kppv en régression : sous des hypothèses standards en statistique non-paramétrique, on a

$$\mathrm{E}\|m_{n,k} - m^\star\| = \mathrm{E}\int |m_{n,k}(x) - m^\star(x)|\mu \mathrm{d}x \leq \frac{c_1}{k} + c_2\left(\frac{k}{n}\right)^{2/d}.$$

- Décomposition biais/variance.
- k grand $\Longrightarrow$ biais $\nearrow$ - variance $\searrow$ $\Longrightarrow$ sous-apprentissage.

L'exemple des kppv

- On peut retrouver les remarques précédentes avec des arguments mathématiques.
- Exemple des kppv en régression : sous des hypothèses standards en statistique non-paramétrique, on a

$$\mathrm{E}\|m_{n,k} - m^\star\| = \mathrm{E}\int |m_{n,k}(x) - m^\star(x)|\mu \mathrm{d}x \leq \frac{c_1}{k} + c_2\left(\frac{k}{n}\right)^{2/d}.$$

- Décomposition biais/variance.
- k grand $\Longrightarrow$ biais $\nearrow$ - variance $\searrow$ $\Longrightarrow$ sous-apprentissage.
- k petit $\Longrightarrow$ biais $\searrow$ - variance $\nearrow$ $\Longrightarrow$ sur-apprentissage.

Rappels

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

Objectif

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

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

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

Rappels

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

Objectif

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

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

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

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 $\mathcal{R}(f_n) = \mathrm{E}[\ell(Y, f_n(X))]$.

Risque empirique

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

Méthodes de ré-échantillonnage : validation croisée, bootstrap...

Présentation

- Différentes méthodes pour estimer $\mathcal{R}(f_n)$.

Présentation

- Différentes méthodes pour estimer $\mathcal{R}(f_n)$.
- Presque toujours la même idée : séparer les données en blocs

Présentation

- Différentes méthodes pour estimer $\mathcal{R}(f_n)$.
- Presque toujours la même idée : séparer les données en blocs
  1. entrainer l'algorithme sur certains blocs

Présentation

- Différentes méthodes pour estimer $\mathcal{R}(f_n)$.
- Presque toujours la même idée : séparer les données en blocs
  1. entrainer l'algorithme sur certains blocs
  2. le tester (prédire) sur d'autres

Présentation

- Différentes méthodes pour estimer $\mathcal{R}(f_n)$.
- Presque toujours la même idée : séparer les données en blocs
  1. entrainer l'algorithme sur certains blocs
  2. le tester (prédire) sur d'autres
  3. en déduire l'estimateur du risque

Présentation

- Différentes méthodes pour estimer $\mathcal{R}(f_n)$.
- Presque toujours la même idée : séparer les données en blocs
  1. entrainer l'algorithme sur certains blocs
  2. le tester (prédire) sur d'autres
  3. en déduire l'estimateur du risque
- La différence entre les différentes approches se trouve dans la manière de construire les blocs.

Apprentissage - Validation ou Validation hold out

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

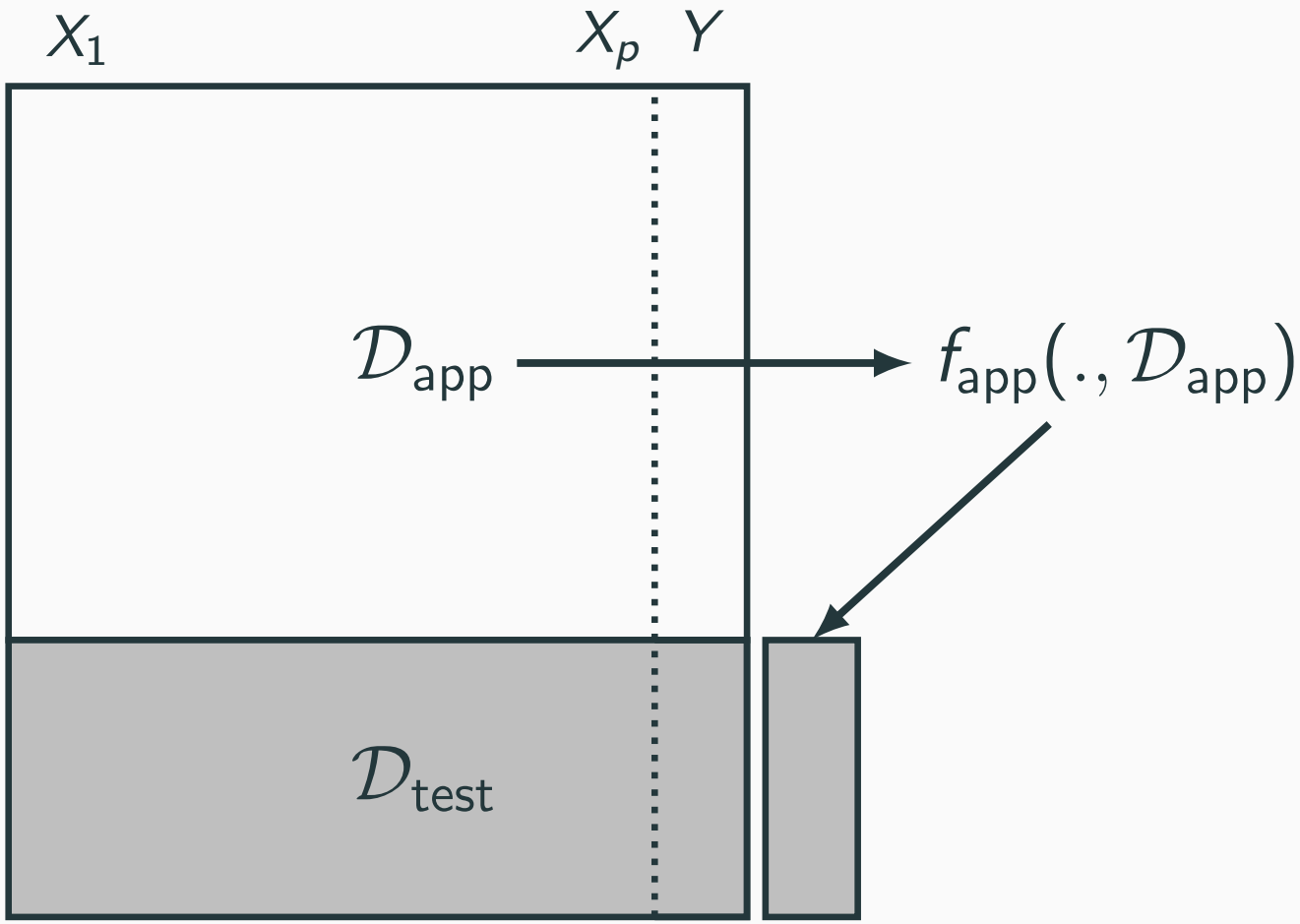
Algorithme

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

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

Retourner :

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



Commentaires

Nécessite d'avoir un nombre suffisant d'observations dans

1. $\mathcal{D}_{\text{app}}$ 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

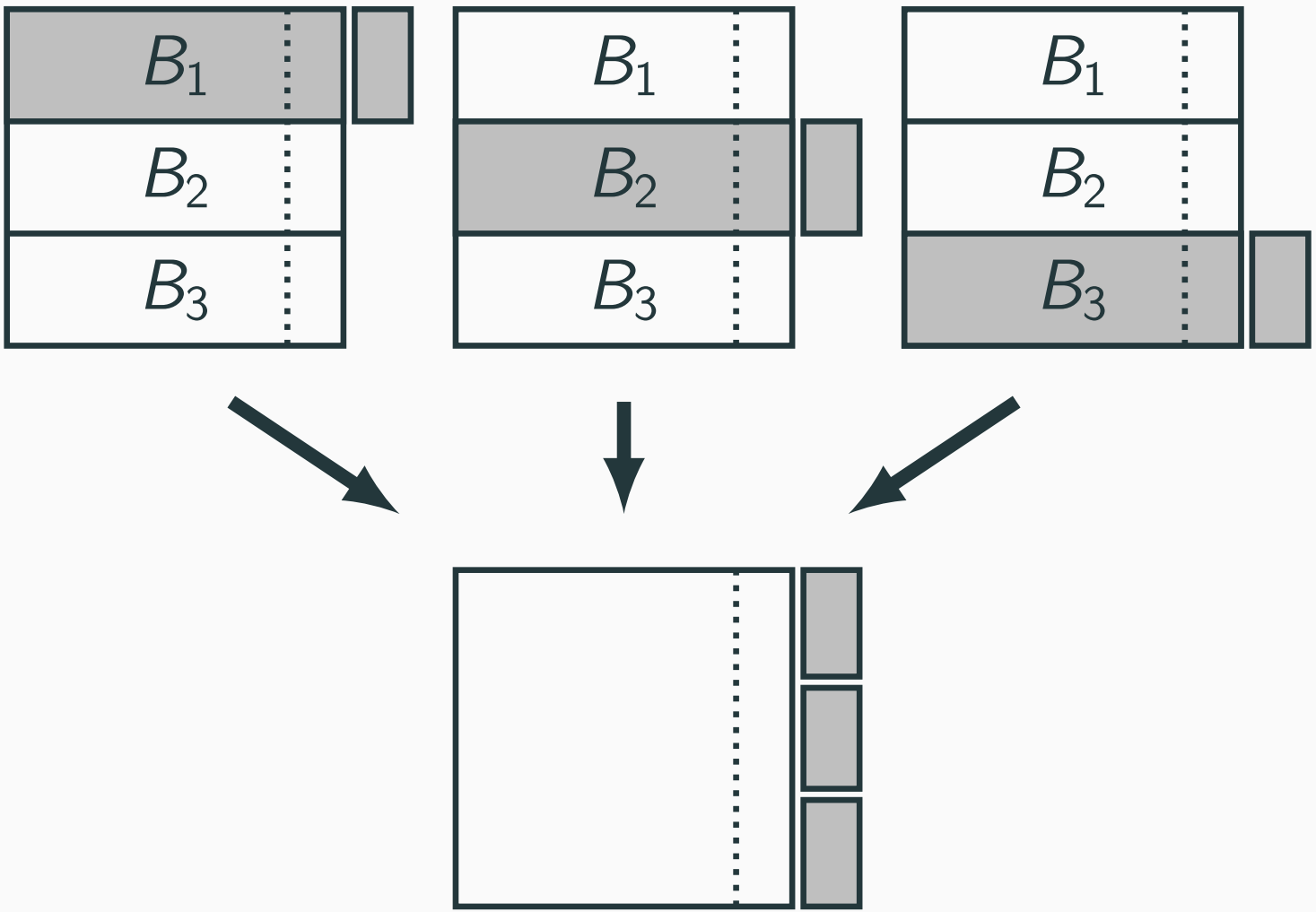
Entrée : $\{B_1, \ldots, B_K\}$ une partition de $\{1, \ldots, n\}$ en K blocs.

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

1. Ajuster l'algorithme de prévision en utilisant l'ensemble des données privé du k^e bloc, c'est-à-dire $\mathcal{B}_k = \{(x_i, y_i) : i \in \{1, \ldots, n\} \backslash B_k\}$. On désigne par $f_k(.) = f_k(., \mathcal{B}_k)$ 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 : $\frac{1}{K} \sum_{k=1}^{K} \widehat{\mathcal{R}}(f_k)$.


B_1
B_2
B_3
B_1
B_2
B_3
B_1
B_2
B_3

Commentaires

- Le choix de K doit être fait par l'utilisateur (souvent $K = 10$).
- Avantage : plus adapté que la technique apprentissage/validation $\Longrightarrow$ 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 $\Longrightarrow$ 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.

$\Longrightarrow$ recommandé uniquement lorsque n est petit.

Autres approches

- 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].

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 $\Longrightarrow \widehat{\mathcal{R}}(f_{n,\theta})$

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

Exemple

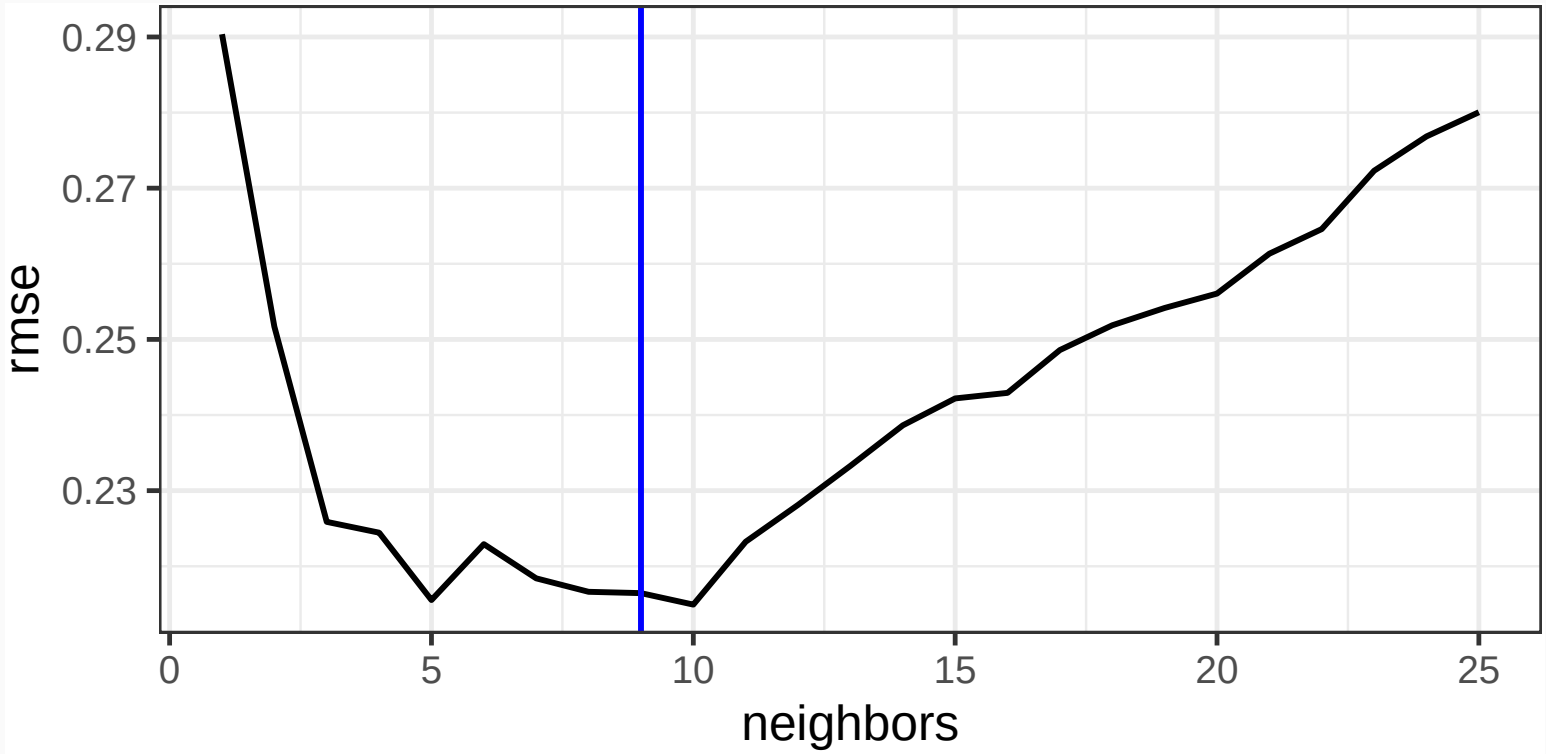
Problème : choisir k pour l'exemple du sinus.

- Grille : $\{1, 2, \ldots, 25\}$;
- Risque : RMSE ;
- Ré-échantillonnage : validation croisée 10 blocs.

Exemple

Problème : choisir k pour l'exemple du sinus.

- Grille : $\{1, 2, \ldots, 25\}$;
- Risque : RMSE ;
- Ré-échantillonnage : validation croisée 10 blocs.



Présentation du package

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

- Le procédé de calibration d'un algorithme 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...)

- Le procédé de calibration d'un algorithme 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.

Les données

- 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
```

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("kknn")
```

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("kknn")
```

- On créé 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)
```

$\Longrightarrow$ consulter https://www.tidymodels.org/find/parsnip/ pour

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 :

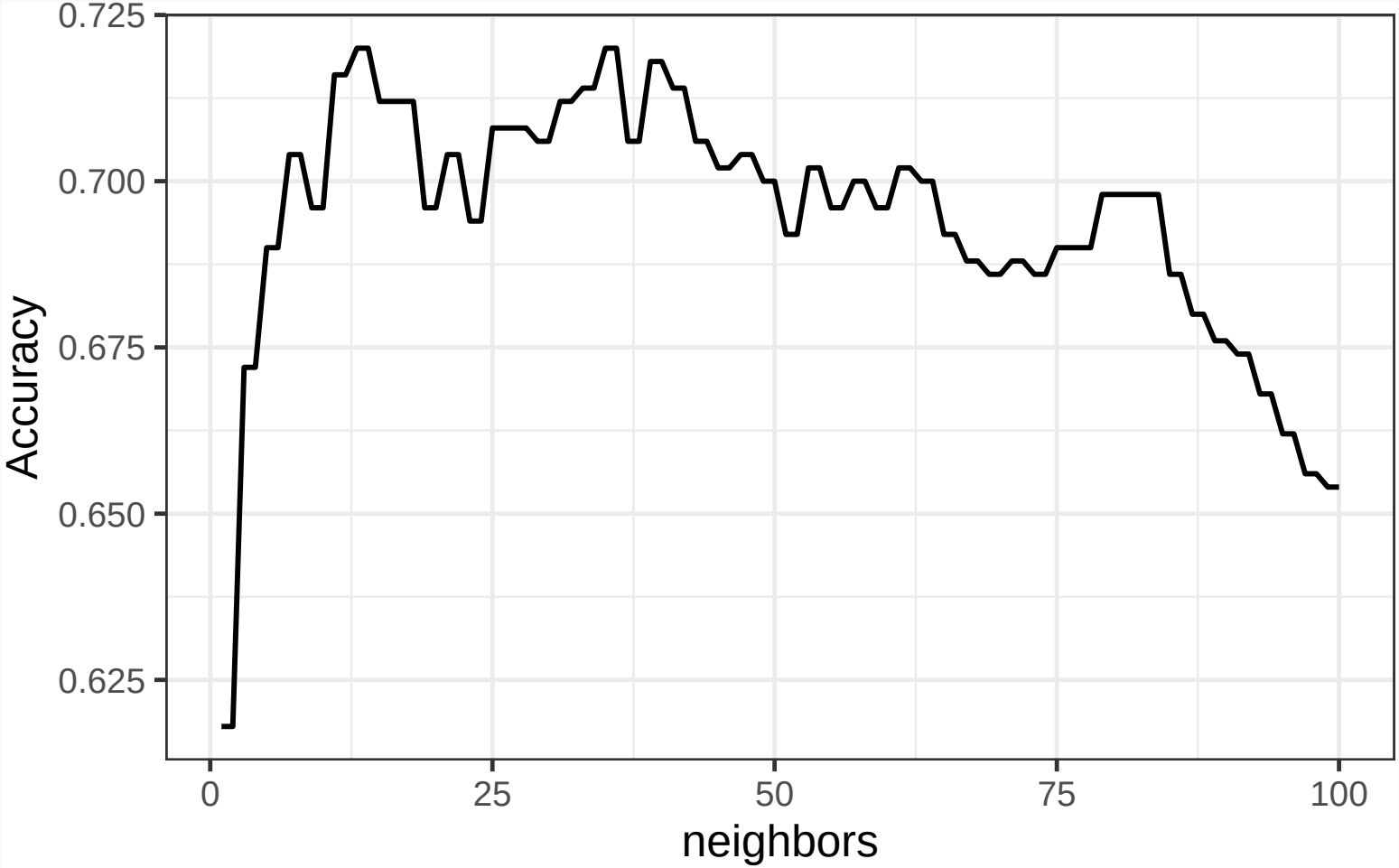
```
> ppv.cv <- ppv_wf %>%
+   tune_grid(
+     resamples = re_ech_cv,
+     grid = grille_k,
+     metrics=metric_set(accuracy))
```

- 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
```

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 entrainant 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 entrainant 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.
- $\Longrightarrow$ facilement personnalisable.
- Aisé de changer le critère, la méthode de ré-échantillonnage...

- Une méthode de ré-échantillonnage renvoie un estimateur $\widehat{\mathcal{R}}(f_n)$ du risque $\mathcal{R}(f_n)$.
- Comme pour tout estimateur, il est important d'étudier ses propriétés pour connaître sa précision.
- Une telle étude aidera l'utilisateur à choisir le meilleur algorithme.

- Une méthode de ré-échantillonnage renvoie un estimateur $\widehat{\mathcal{R}}(f_n)$ du risque $\mathcal{R}(f_n)$.
- Comme pour tout estimateur, il est important d'étudier ses propriétés pour connaître sa précision.
- Une telle étude aidera l'utilisateur à choisir le meilleur algorithme.

Remarque

- Lorsque la méthode utilisée est répétée sur plusieurs blocs, il est "facile" d'estimer la variance de $\widehat{\mathcal{R}}(f_n)$.
- Nous l'illustrons avec la validation croisée.

- Rappel : l'estimateur de validation croisée s'écrit $\widehat{\mathcal{R}}_{\mathsf{CV}}(f_n) = \frac{1}{K} \sum_{k=1}^{K} \widehat{\mathcal{R}}(f_k)$.

- Rappel : l'estimateur de validation croisée s'écrit $\widehat{\mathcal{R}}_{\mathsf{CV}}(f_n) = \frac{1}{K}\sum_{k=1}^{K}\widehat{\mathcal{R}}(f_k)$.
- On a donc

$$\mathsf{V}[\widehat{\mathcal{R}}_{\mathsf{CV}}(f_n)|\mathcal{D}_n] = \mathsf{V}\left[\frac{1}{K}\sum_{k=1}^{K}\widehat{\mathcal{R}}(f_k)\Big|\mathcal{D}_n\right] = \frac{1}{K}\mathsf{V}[\widehat{\mathcal{R}}(f_1)|\mathcal{D}_n].$$

- Rappel : l'estimateur de validation croisée s'écrit $\widehat{\mathcal{R}}_{\mathrm{CV}}(f_n) = \frac{1}{K}\sum_{k=1}^{K}\widehat{\mathcal{R}}(f_k)$.
- On a donc

$$\mathrm{V}[\widehat{\mathcal{R}}_{\mathrm{CV}}(f_n)|\mathcal{D}_n] = \mathrm{V}\left[\frac{1}{K}\sum_{k=1}^{K}\widehat{\mathcal{R}}(f_k)\Big|\mathcal{D}_n\right] = \frac{1}{K}\mathrm{V}[\widehat{\mathcal{R}}(f_1)|\mathcal{D}_n].$$

- La variance $\mathrm{V}[\widehat{\mathcal{R}}(f_1)|\mathcal{D}_n]$ désigne la variance de l'erreur calculée sur un des K blocs (elles sont toutes égales), on peut l'estimer par

$$\widehat{\mathrm{V}}[\widehat{\mathcal{R}}(f_1)|\mathcal{D}_n] = \frac{1}{K-1}\sum_{k=1}^{K}(\widehat{\mathcal{R}}(f_k) - \widehat{\mathcal{R}}_{\mathrm{CV}}(f_n))^2.$$

- Rappel : l'estimateur de validation croisée s'écrit $\widehat{\mathcal{R}}_{\mathrm{CV}}(f_n) = \frac{1}{K}\sum_{k=1}^{K}\widehat{\mathcal{R}}(f_k)$.
- On a donc

$$\mathrm{V}[\widehat{\mathcal{R}}_{\mathrm{CV}}(f_n)|\mathcal{D}_n] = \mathrm{V}\left[\frac{1}{K}\sum_{k=1}^{K}\widehat{\mathcal{R}}(f_k)\Big|\mathcal{D}_n\right] = \frac{1}{K}\mathrm{V}[\widehat{\mathcal{R}}(f_1)|\mathcal{D}_n].$$

- La variance $\mathrm{V}[\widehat{\mathcal{R}}(f_1)|\mathcal{D}_n]$ désigne la variance de l'erreur calculée sur un des K blocs (elles sont toutes égales), on peut l'estimer par

$$\widehat{\mathrm{V}}[\widehat{\mathcal{R}}(f_1)|\mathcal{D}_n] = \frac{1}{K-1}\sum_{k=1}^{K}(\widehat{\mathcal{R}}(f_k) - \widehat{\mathcal{R}}_{\mathrm{CV}}(f_n))^2.$$

- On déduit l'estimateur du risque de validation croisée en posant

$$\frac{1}{K(K-1)}\sum_{k=1}^{K}(\widehat{\mathcal{R}}(f_k) - \widehat{\mathcal{R}}_{\mathrm{CV}}(f_n))^2.$$

- Cette variance, ou plutôt sa racine carrée (son écart-type), est automatiquement calculée par tune_grid :

```
> ppv.cv %>% collect_metrics() %>% select(1:6) %>% head()
## # A tibble: 6 x 6
##   neighbors .metric  .estimator  mean     n std_err
##       <int> <chr>    <chr>      <dbl> <int>   <dbl>
## 1         1 accuracy binary     0.618    10  0.0247
## 2         2 accuracy binary     0.618    10  0.0247
## 3         3 accuracy binary     0.672    10  0.0215
## 4         4 accuracy binary     0.672    10  0.0215
## 5         5 accuracy binary     0.69     10  0.0209
## 6         6 accuracy binary     0.69     10  0.0209
```

- Cette variance, ou plutôt sa racine carrée (son écart-type), est automatiquement calculée par tune_grid :

```
> ppv.cv %>% collect_metrics() %>% select(1:6) %>% head()
## # A tibble: 6 x 6
##   neighbors .metric  .estimator  mean     n std_err
##       <int> <chr>    <chr>      <dbl> <int>   <dbl>
## 1         1 accuracy binary     0.618    10  0.0247
## 2         2 accuracy binary     0.618    10  0.0247
## 3         3 accuracy binary     0.672    10  0.0215
## 4         4 accuracy binary     0.672    10  0.0215
## 5         5 accuracy binary     0.69     10  0.0209
## 6         6 accuracy binary     0.69     10  0.0209
```

- Il est intéressant de la visualiser en même temps que le risque estimé.

- L'utilisateur peut ainsi choisir un algorithme de complexité minimale tel que le risque soit "proche" du risque optimal.

- L'utilisateur peut ainsi choisir un algorithme de complexité minimale tel que le risque soit "proche" du risque optimal.

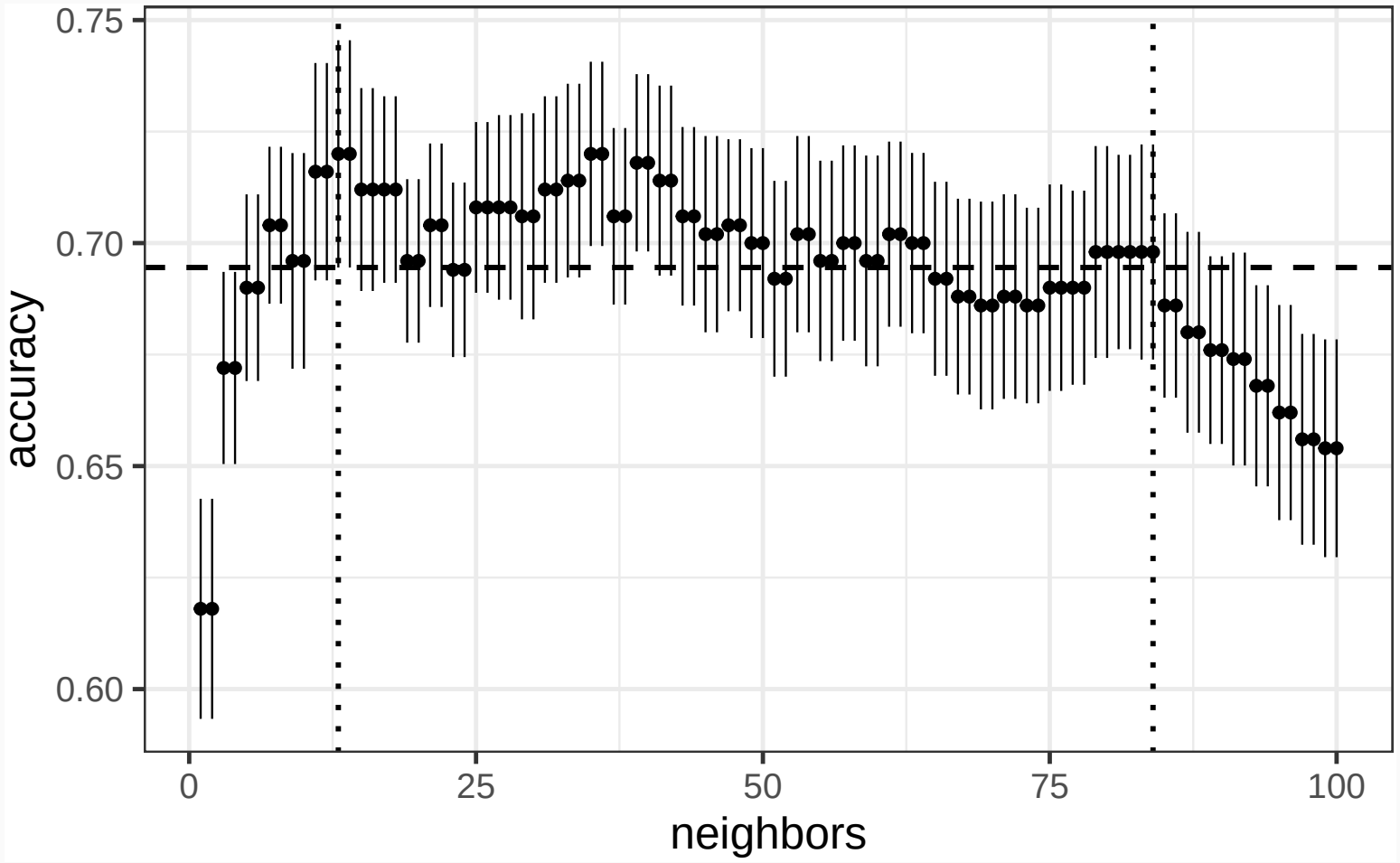
Règle one-standard-error [Breiman et al., 1984].

Choisir l'algorithme de complexité minimale parmi ceux dont le risque ne dépasse pas le meilleur risque à un écart-type près.

- L'utilisateur peut ainsi choisir un algorithme de complexité minimale tel que le risque soit "proche" du risque optimal.

Règle one-standard-error [Breiman et al., 1984].

Choisir l'algorithme de complexité minimale parmi ceux dont le risque ne dépasse pas le meilleur risque à un écart-type près.



- On retrouve la valeur choisie par cette règle avec la fonction select_by_one_std_err :

```
> ppv.cv %>% select_by_one_std_err(desc(neighbors)) %>% select(-7)
## # A tibble: 1 x 8
##   neighbors .metric  .estimator  mean     n std_err .best .bound
##       <int> <chr>    <chr>      <dbl> <int>   <dbl> <dbl>  <dbl>
## 1        84 accuracy binary     0.698    10  0.0241  0.72  0.695
```

- On retrouve la valeur choisie par cette règle avec la fonction select_by_one_std_err :

```
> ppv.cv %>% select_by_one_std_err(desc(neighbors)) %>% select(-7)
## # A tibble: 1 x 8
##   neighbors .metric  .estimator  mean     n std_err .best .bound
##       <int> <chr>    <chr>      <dbl> <int>   <dbl> <dbl>  <dbl>
## 1        84 accuracy binary     0.698    10  0.0241  0.72  0.695
```

- On sélectionne ici plus de voisins, on a donc une complexité plus petite.

- Les méthodes d'estimation du risque sont construites à partir de prévisions sur différents blocs.

- Les méthodes d'estimation du risque sont construites à partir de prévisions sur différents blocs.
- La variance de ces estimateurs peut parfois se révéler élevée $\Longrightarrow \nearrow$ difficulté pour choisir le meilleur algorithme.

- Les méthodes d'estimation du risque sont construites à partir de prévisions sur différents blocs.
- La variance de ces estimateurs peut parfois se révéler élevée $\Longrightarrow \nearrow$ difficulté pour choisir le meilleur algorithme.

Une solution

Répéter les méthodes de ré-échantillonnage sur plusieurs découpages des données.

Algorithme : répétition du ré-échantillonnage

Entrées :

- Un algorithme d'estimation du risque (validation hold out, validation croisée).
- M nombre de répétitions.

Pour m variant de 1 à M :

- Estimer le risque par l'algorithme choisi $\Longrightarrow \widehat{\mathcal{R}}_m(f_n)$.

Retourner : $\frac{1}{M}\sum_{m=1}^{M}\widehat{\mathcal{R}}_m(f_n)$.

Algorithme : répétition du ré-échantillonnage

Entrées :

- Un algorithme d'estimation du risque (validation hold out, validation croisée).
- M nombre de répétitions.

Pour m variant de 1 à M :

- Estimer le risque par l'algorithme choisi $\Longrightarrow \widehat{\mathcal{R}}_m(f_n)$.

Retourner : $\frac{1}{M}\sum_{m=1}^{M}\widehat{\mathcal{R}}_m(f_n)$.

- Avantages : estimation plus précise du risque.

Algorithme : répétition du ré-échantillonnage

Entrées :

- Un algorithme d'estimation du risque (validation hold out, validation croisée).
- M nombre de répétitions.

Pour m variant de 1 à M :

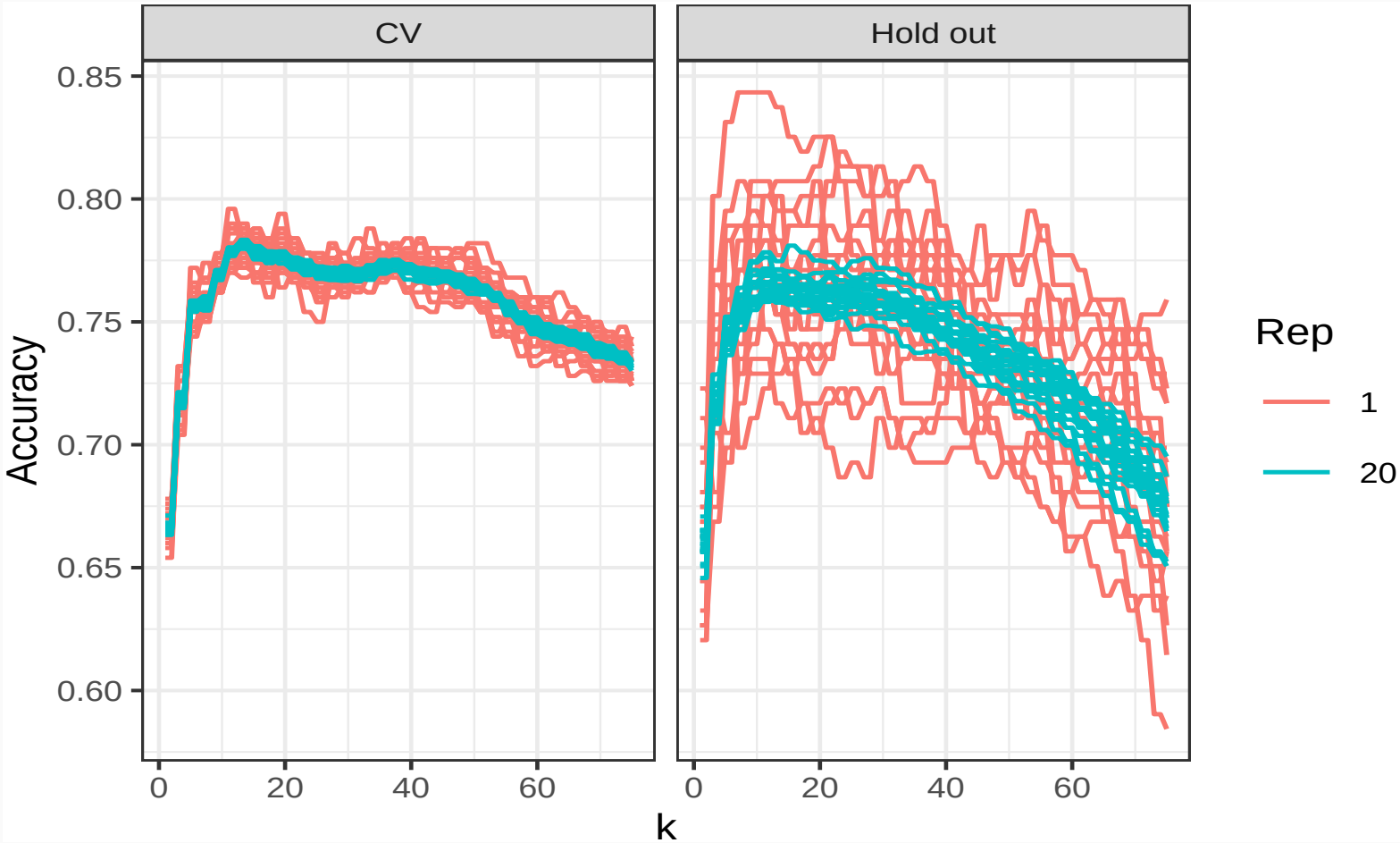
- Estimer le risque par l'algorithme choisi $\Longrightarrow \widehat{\mathcal{R}}_m(f_n)$.

Retourner : $\frac{1}{M}\sum_{m=1}^{M}\widehat{\mathcal{R}}_m(f_n)$.

- Avantages : estimation plus précise du risque.
- Inconvénients : plus couteux en temps de calcul (intéressant de paralléliser).

Exemple

- Courbes de risque par Monte Carlo sans répétition et avec 20 répétitions.



- On observe clairement une diminution de la variabilité avec répétitions.

Le coin R

- Facile à mettre en oeuvre avec tidymodels : il suffit de définir les découpages.
- Validation hold out répétée avec mc_cv :

```
> mv_cv(don.2D.500,prom=2/3,times=20)
```

- Validation croisée répétée avec vfold_cv :

```
> vflod_cv(don.2D.500,v=10,repeats=20)
```

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)
> ctrl2 <- trainControl(method="cv",number=10)
> e2 <- train(Y~.,data=dapp,method="knn",trControl=ctrl2,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="LGOCV",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.
```

Références i


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.


Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984).
***Classification and regression trees*.**
Wadsworth & Brooks.

Références ii


Clémençon, 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.

Références iii

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

Arbres

Présentation

- 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.

Notations

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

Notations

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

- Y peut admettre un nombre quelconque de modalités et les variables $X_1, \ldots, X_d$ peuvent être qualitatives et/ou quantitatives.

Notations

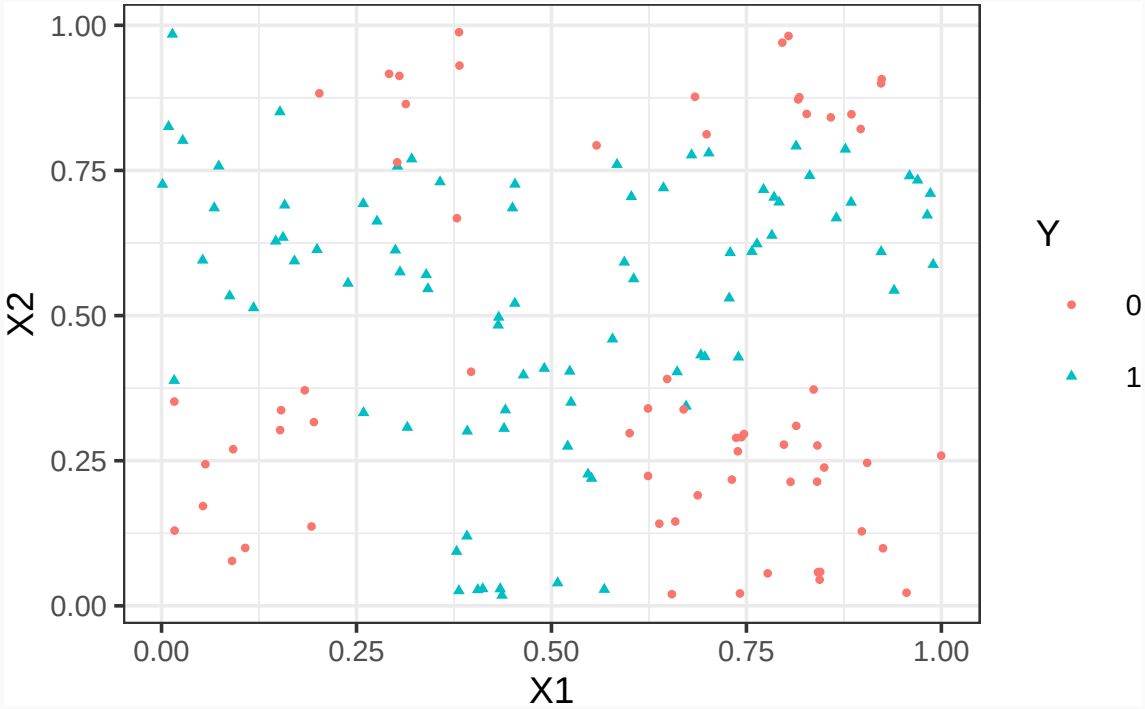
- On cherche à expliquer une variable Y par d variables explicatives $X_1, \ldots, X_d$.

- Y peut admettre un nombre quelconque de modalités et les variables $X_1, \ldots, X_d$ 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.

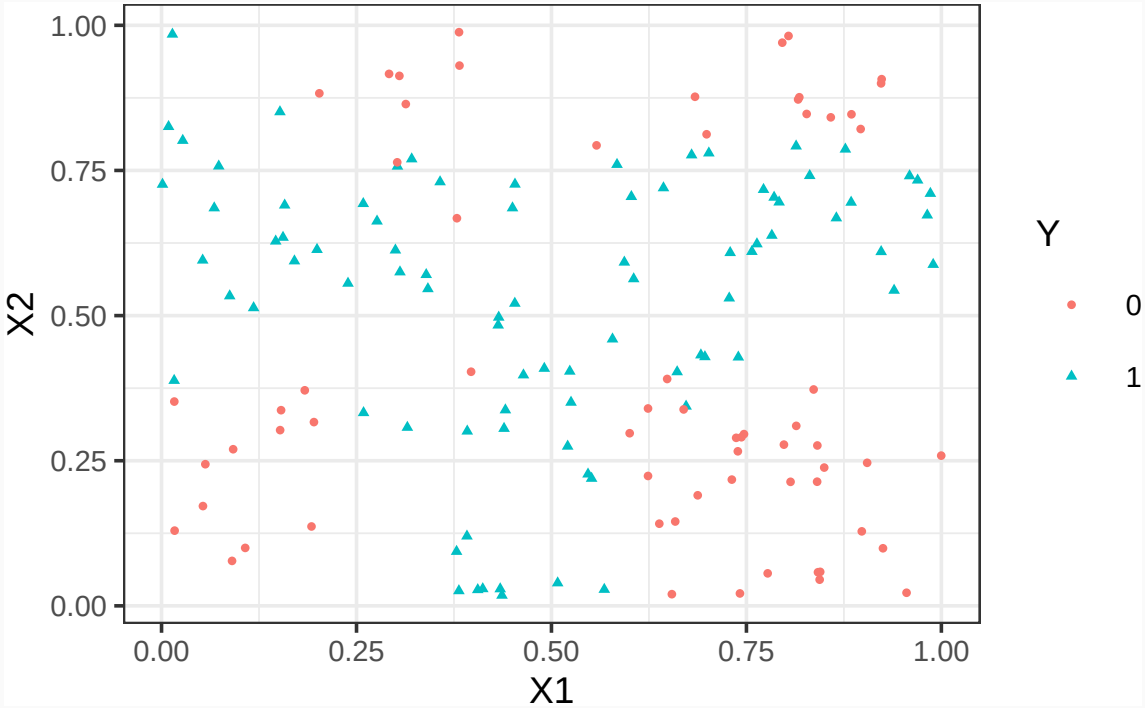
Représentation des données

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



Représentation des données

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

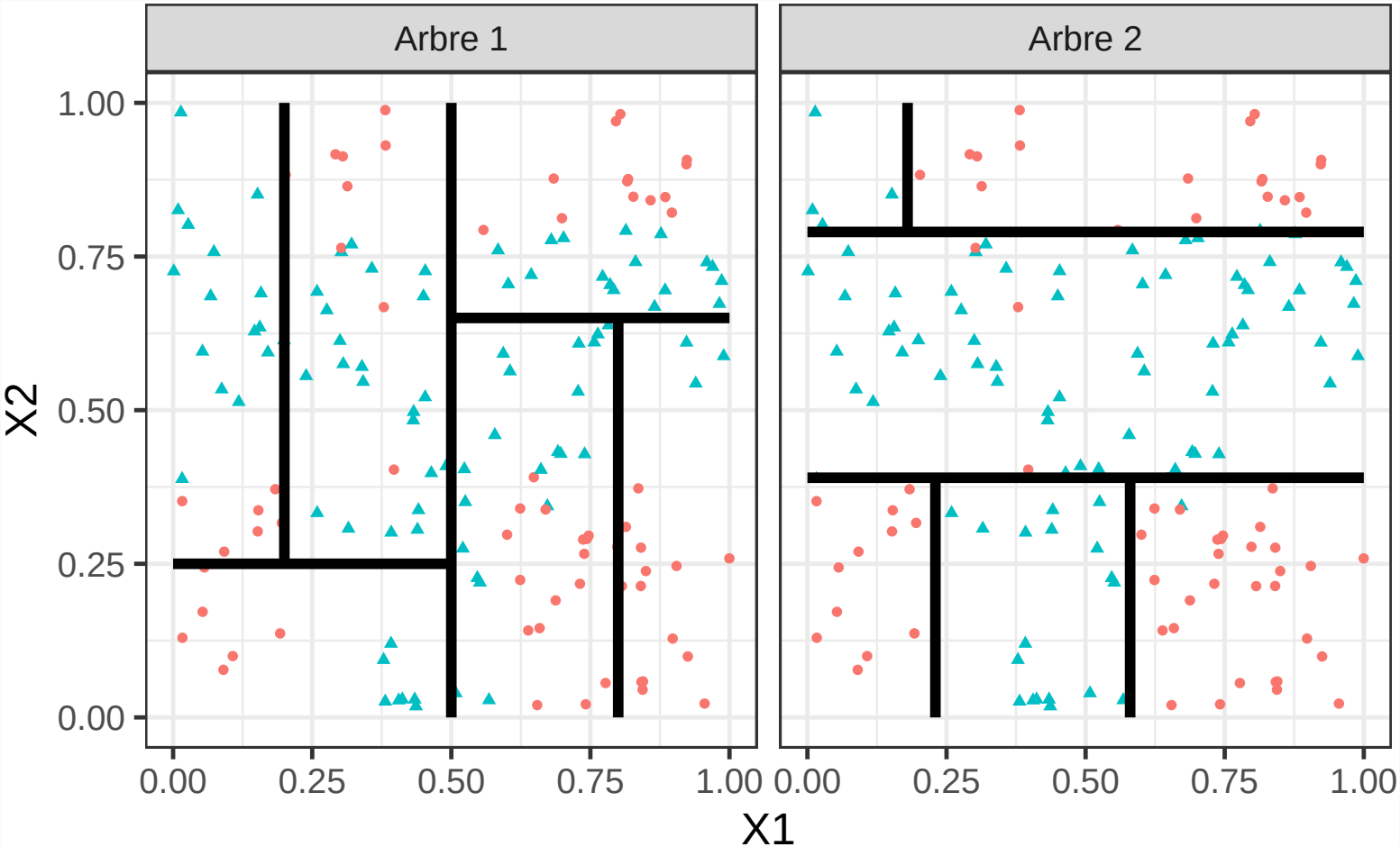


Approche par arbres

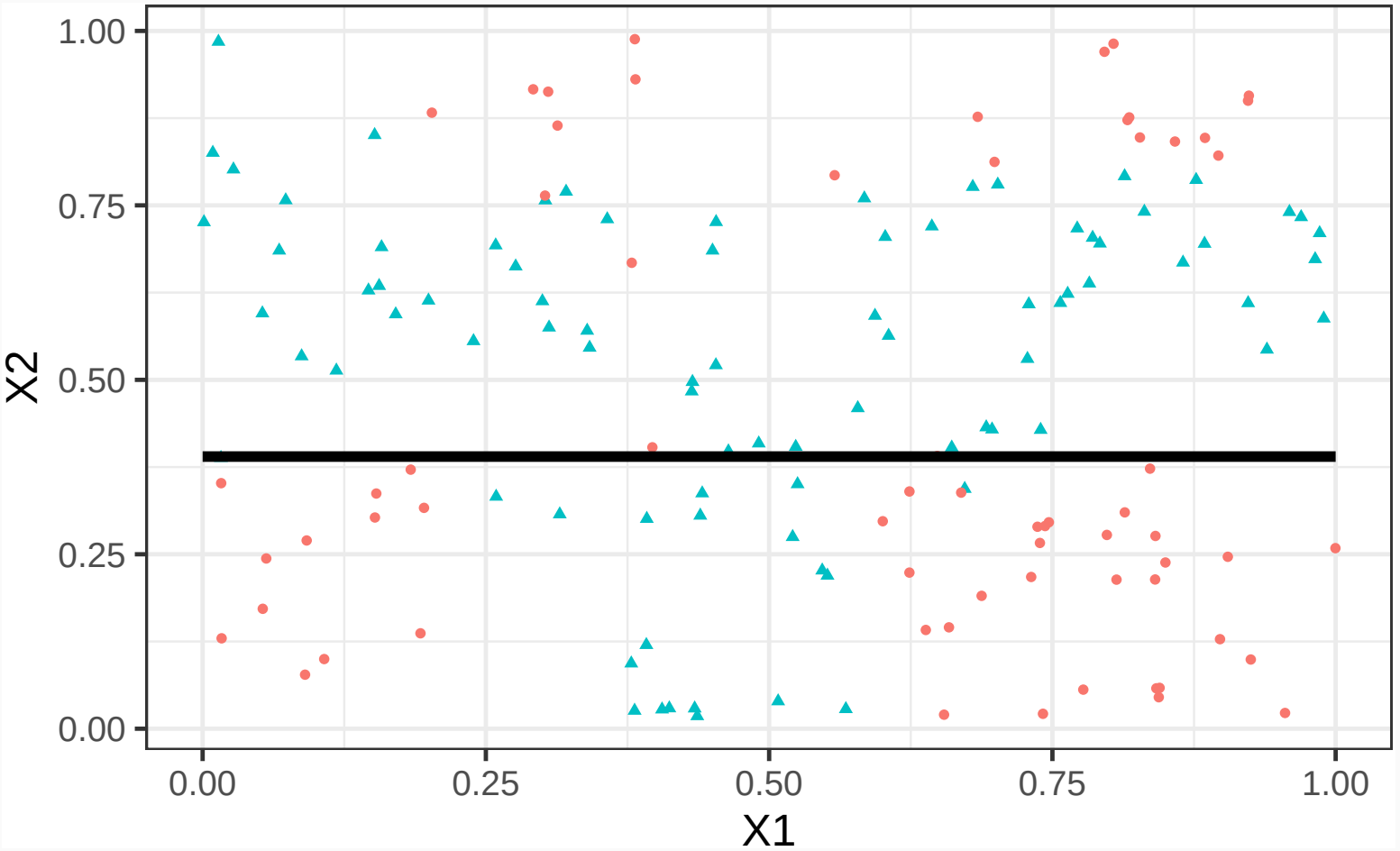
Trouver une partition des observations qui sépare "au mieux" les points rouges des points bleus.

Arbres binaires

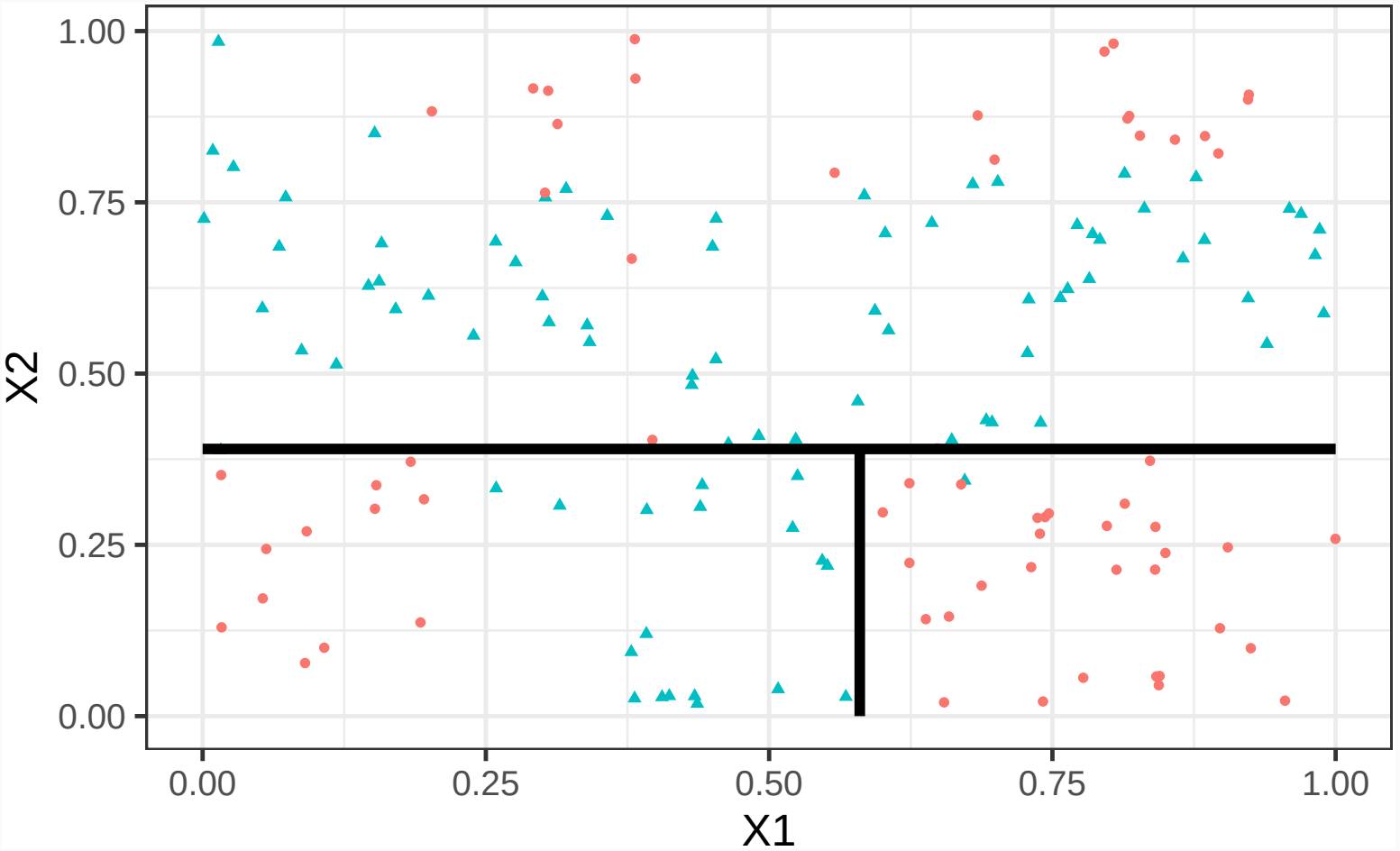
- La méthode CART propose de construire une partition basée sur des divisions successives parallèles aux axes.
- 2 exemples de partition :



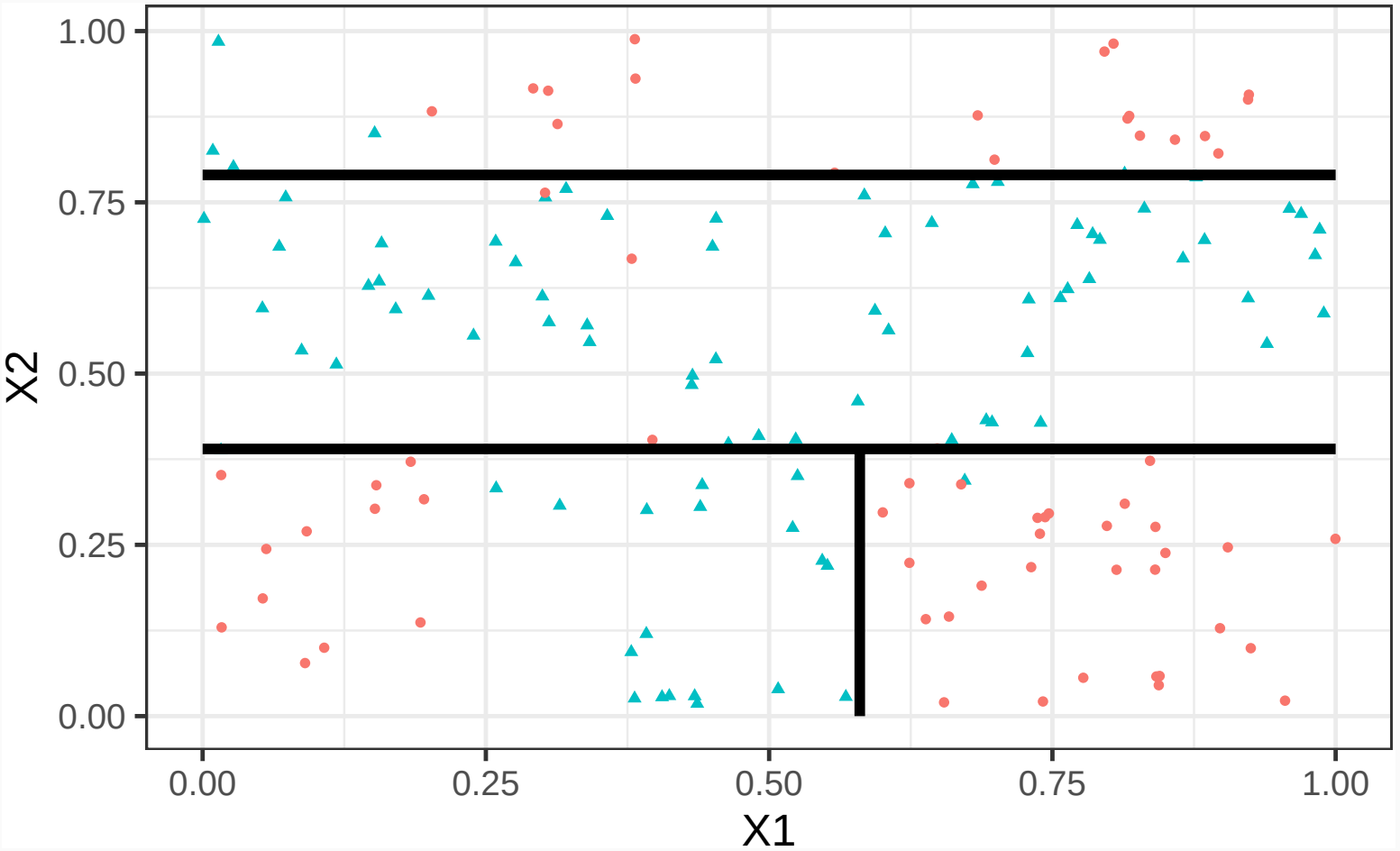
- A chaque étape, la méthode cherche une nouvelle division : une variable et un seuil de coupure.



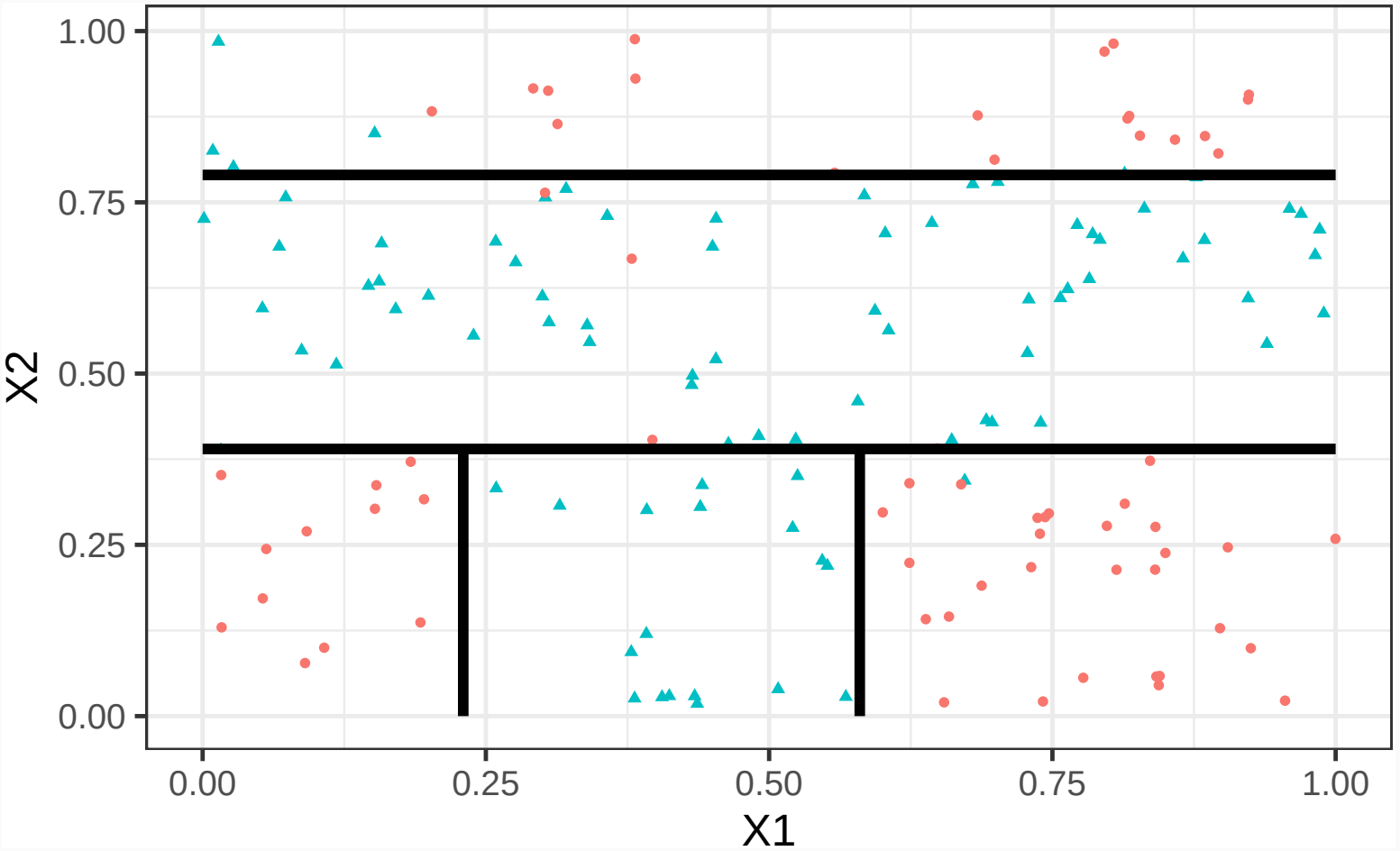
- A chaque étape, la méthode cherche une nouvelle division : une variable et un seuil de coupure.



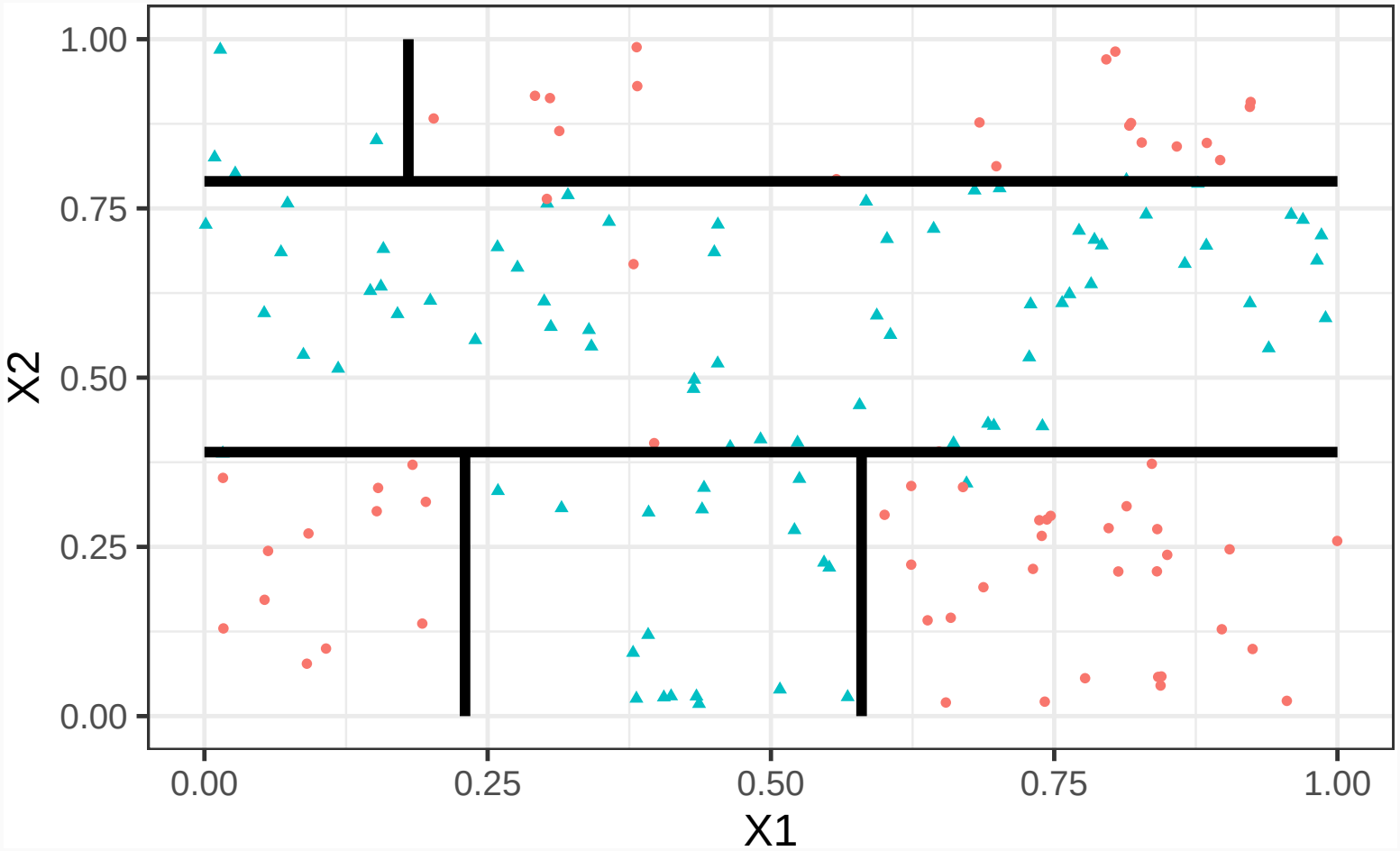
- A chaque étape, la méthode cherche une nouvelle division : une variable et un seuil de coupure.



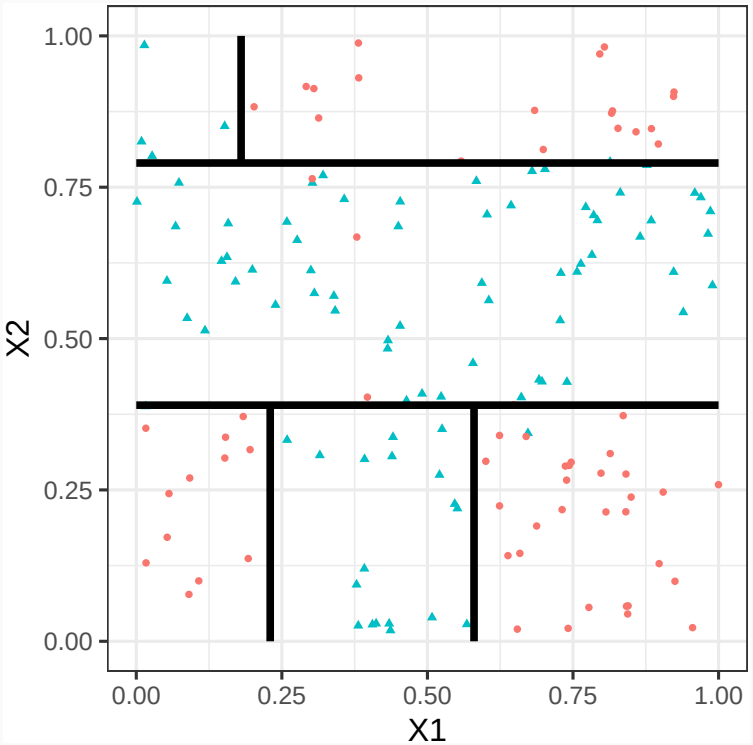
- A chaque étape, la méthode cherche une nouvelle division : une variable et un seuil de coupure.

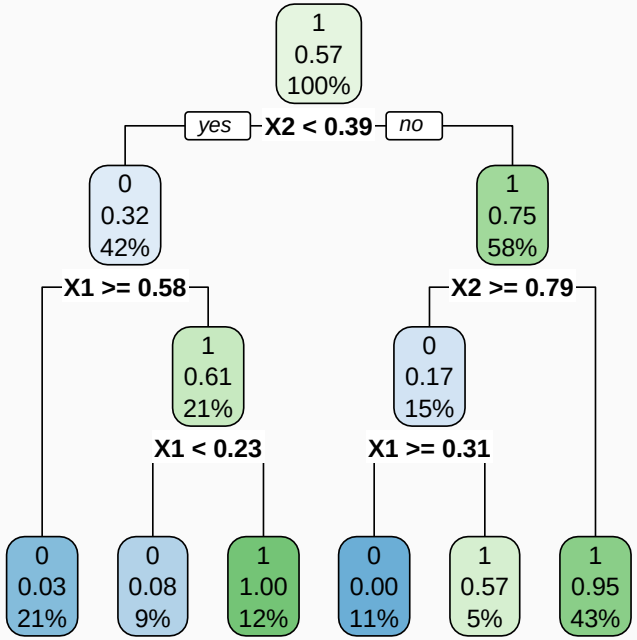


- A chaque étape, la méthode cherche une nouvelle division : une variable et un seuil de coupure.

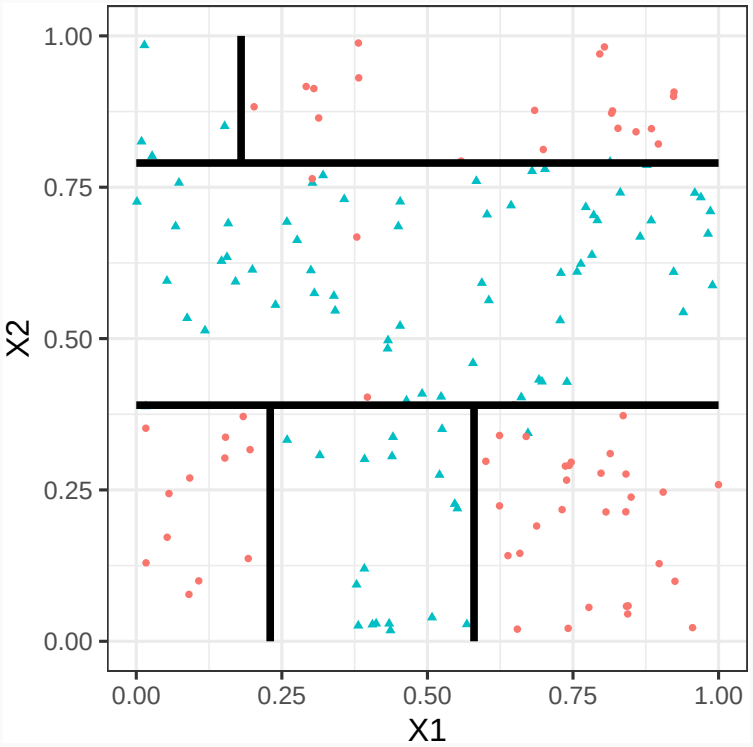


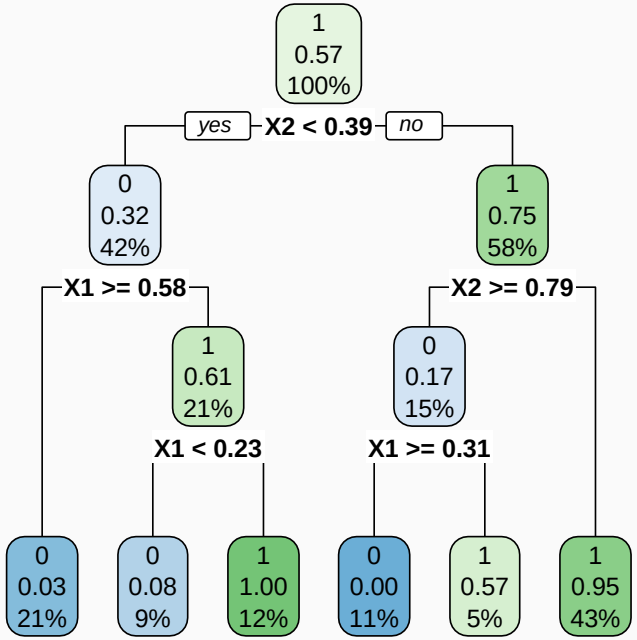
Représentation de l'arbre





Représentation de l'arbre





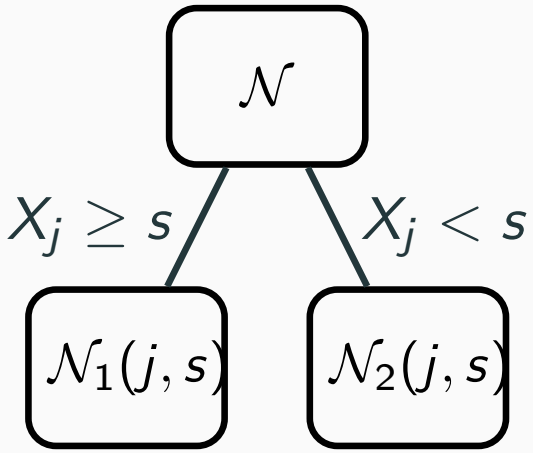
Remarque

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 en noeud 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 $\Longrightarrow$ 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 $\Longrightarrow$ 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) $\Longrightarrow$ 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) $\Longrightarrow$ proportion d'obs. du groupe k :

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

Questions

1. Comment découper un nœud ?

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

Questions

1. Comment découper un nœud ?

   $\Longrightarrow$ 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)

Questions

1. Comment découper un nœud ?

   $\Longrightarrow$ 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 ?

Questions

1. Comment découper un nœud ?

   $\Longrightarrow$ 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"...).

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

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

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

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.

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 $\mathcal{N}$.

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

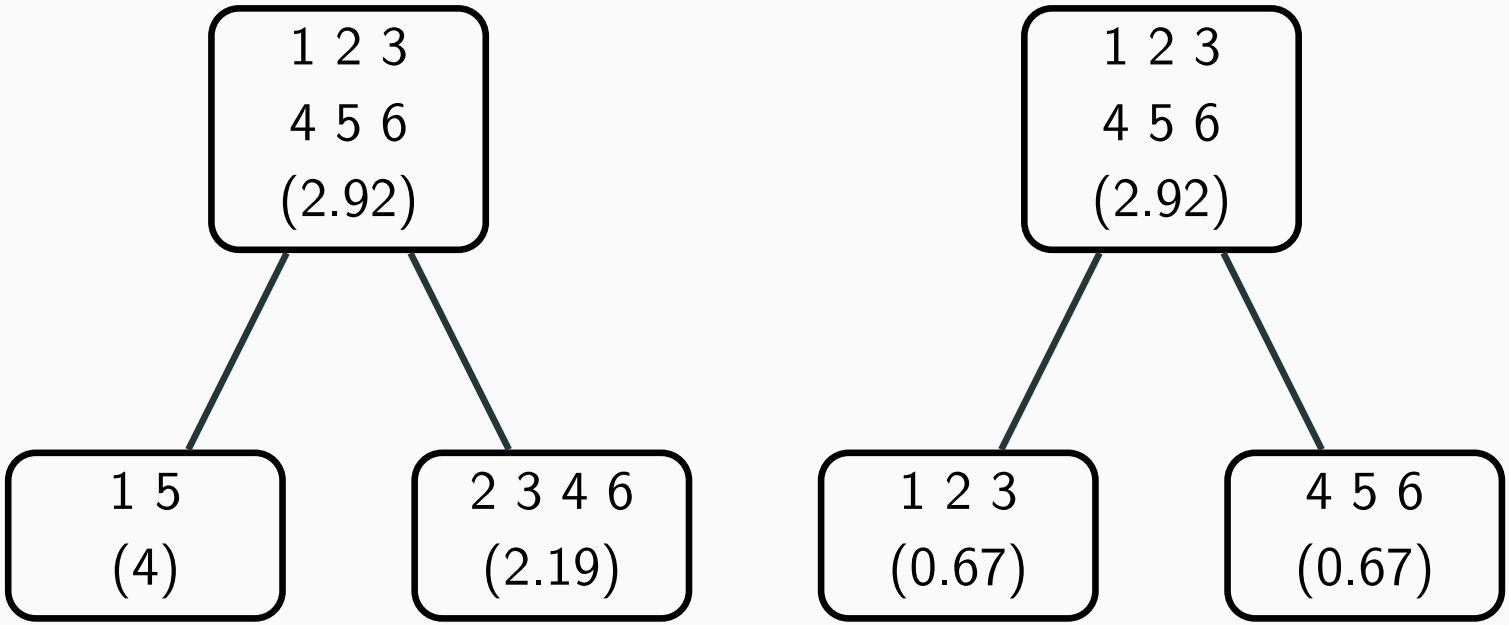
$$\mathcal{I}(\mathcal{N}) = \frac{1}{|\mathcal{N}|} \sum_{i:x_i \in \mathcal{N}} (y_i - \bar{y}_{\mathcal{N}})^2,$$

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 :

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

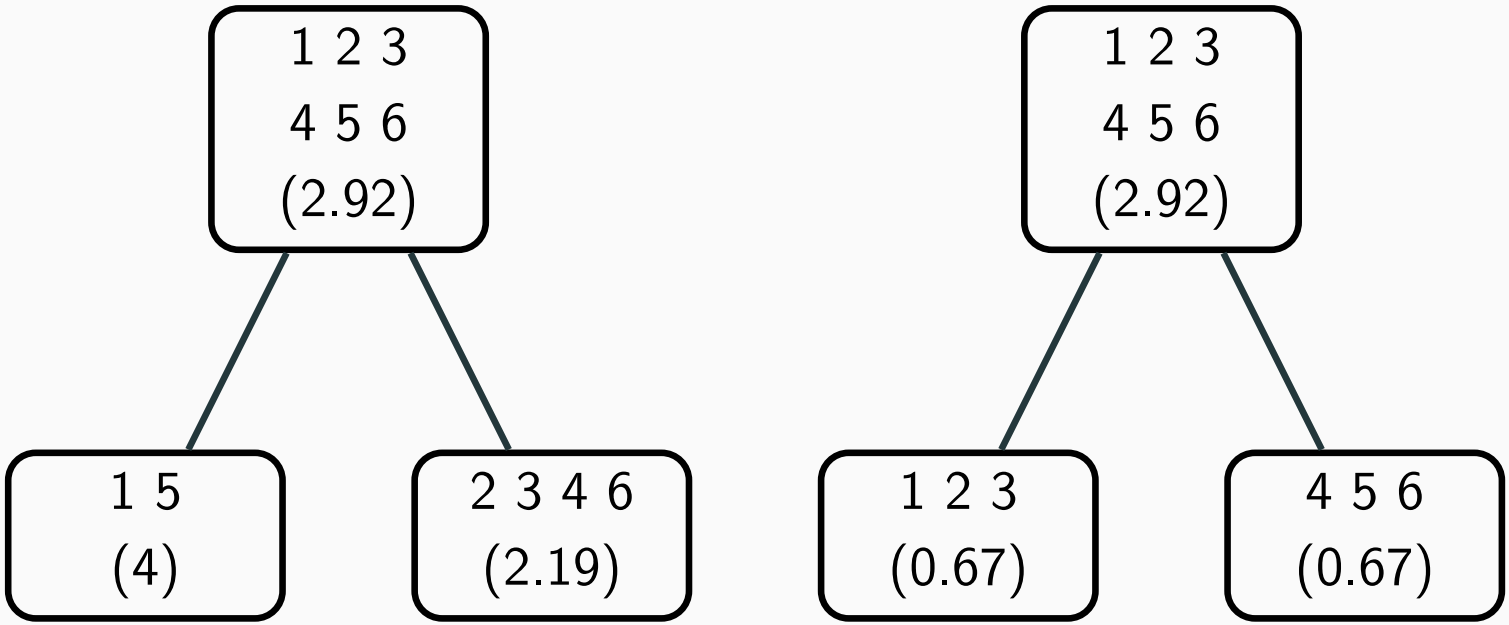
où $\bar{y}_{\mathcal{N}}$ désigne la moyenne des Y_i dans $\mathcal{N}$.



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

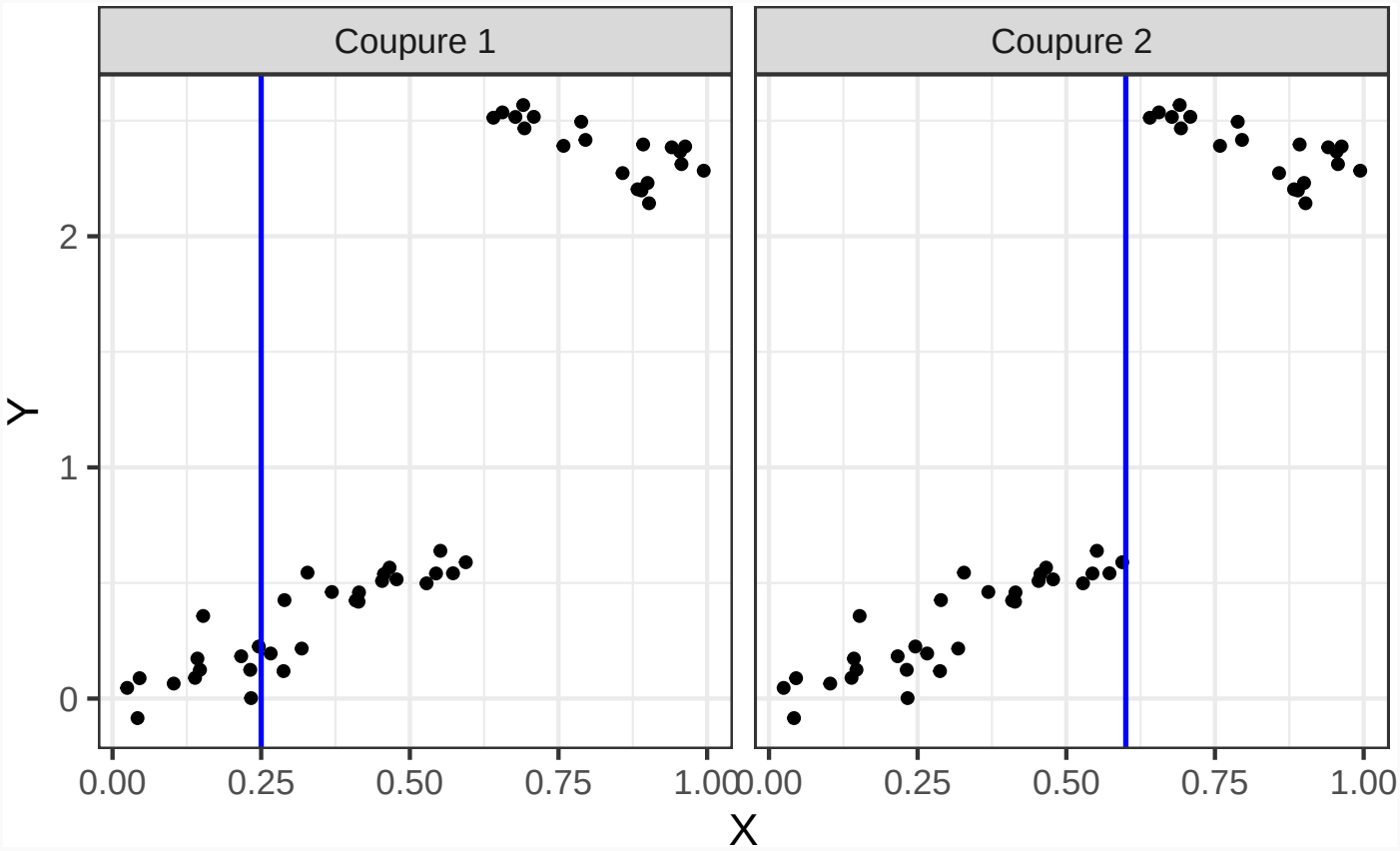
$$\mathcal{I}(\mathcal{N}) = \frac{1}{|\mathcal{N}|} \sum_{i:x_i \in \mathcal{N}} (y_i - \bar{y}_{\mathcal{N}})^2,$$

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

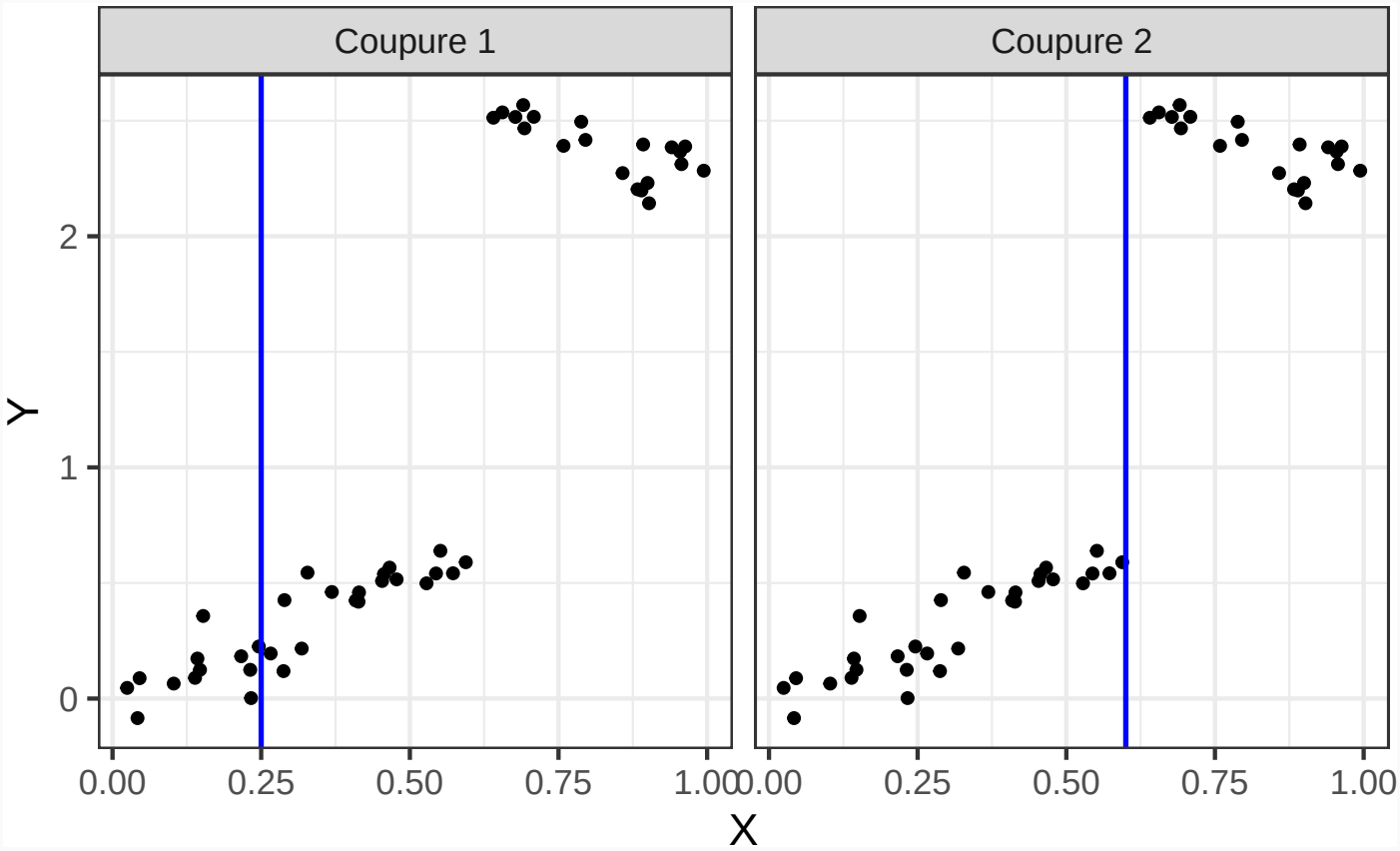


$\Longrightarrow$ coupure de droite plus performante.

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

Pour aller plus vite

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

- Les $Y_i, i = 1, \ldots, n$ sont à valeurs dans $\{1, \ldots, K\}$.
- 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 $Y_i, i = 1, \ldots, n$ sont à valeurs dans $\{1, \ldots, K\}$.
- 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) $[0, 1] \to \mathbb{R}^+$ 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 $\mathcal{I}(\mathcal{N}) = 0 \Longrightarrow$ c'est pourquoi f doit vérifier $f(0) = f(1) = 0$.

Exemples de fonctions f

- Si $\mathcal{N}$ est pur, on veut $\mathcal{I}(\mathcal{N}) = 0 \Longrightarrow$ 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 $\mathcal{I}(\mathcal{N}) = 0 \Longrightarrow$ 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 : $f(p) = -p\log(p)$.

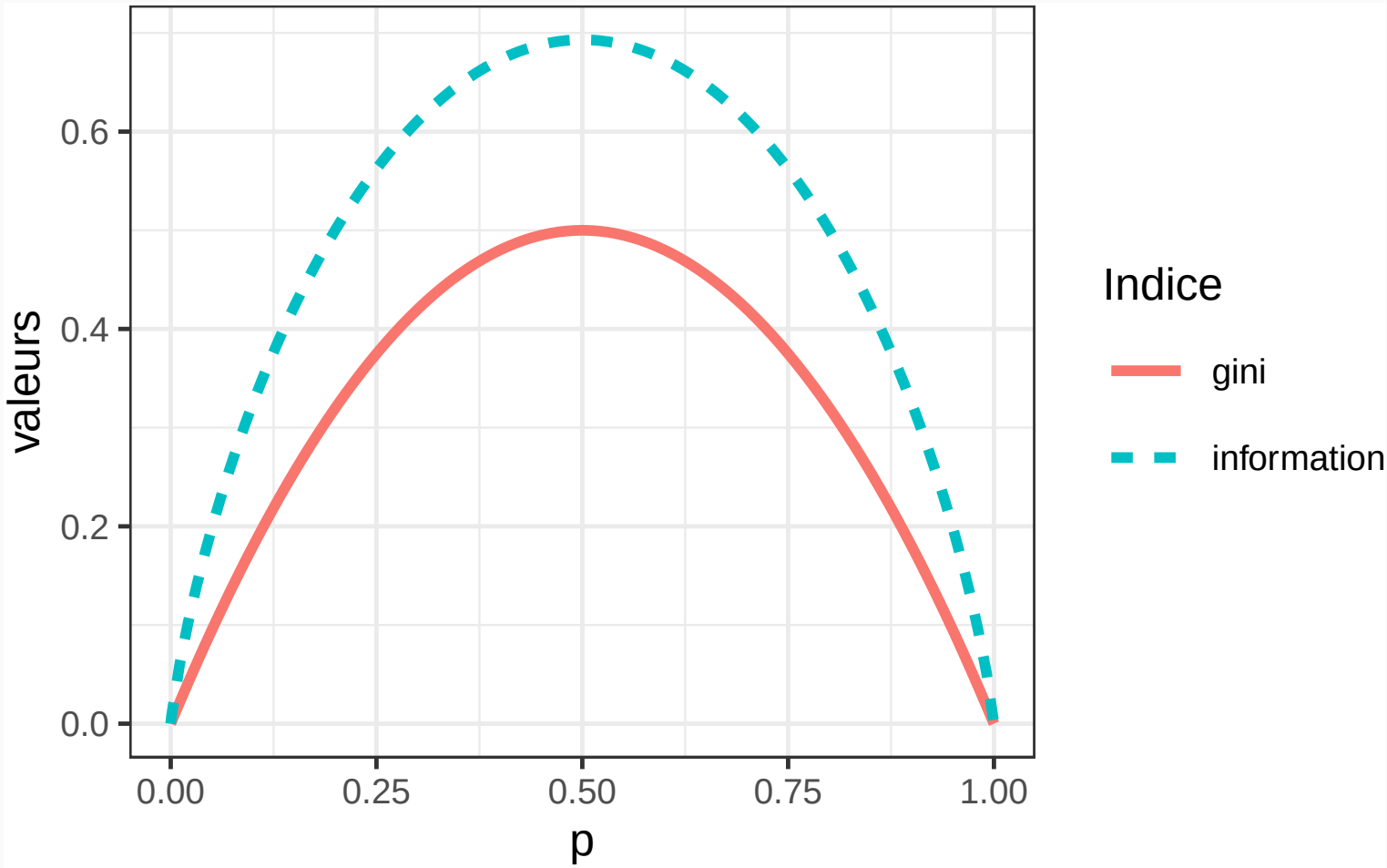
Cas binaire

Dans ce cas on a

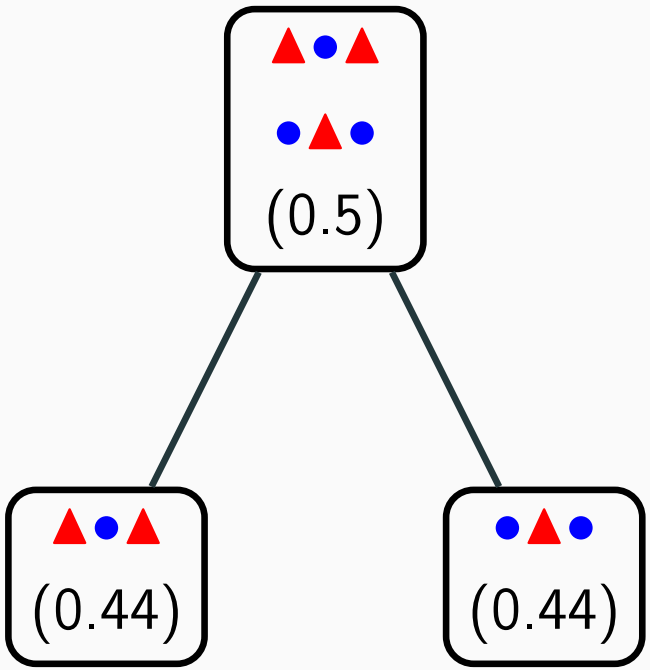
1. $\mathcal{I}(\mathcal{N}) = 2p(1-p)$ pour Gini
2. $\mathcal{I}(\mathcal{N}) = -p\log p - (1-p)\log(1-p)$ pour Information

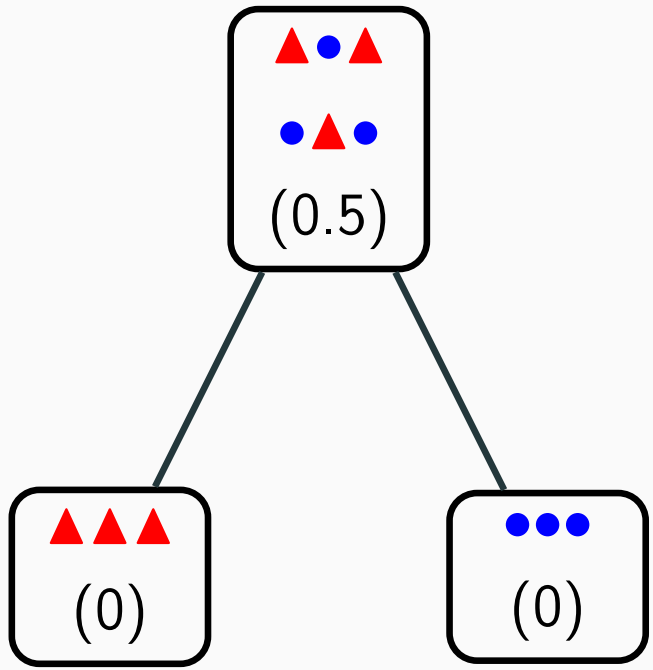
où p désigne la proportion de 1 (ou 0) dans $\mathcal{N}$.

Impureté dans le cas binaire

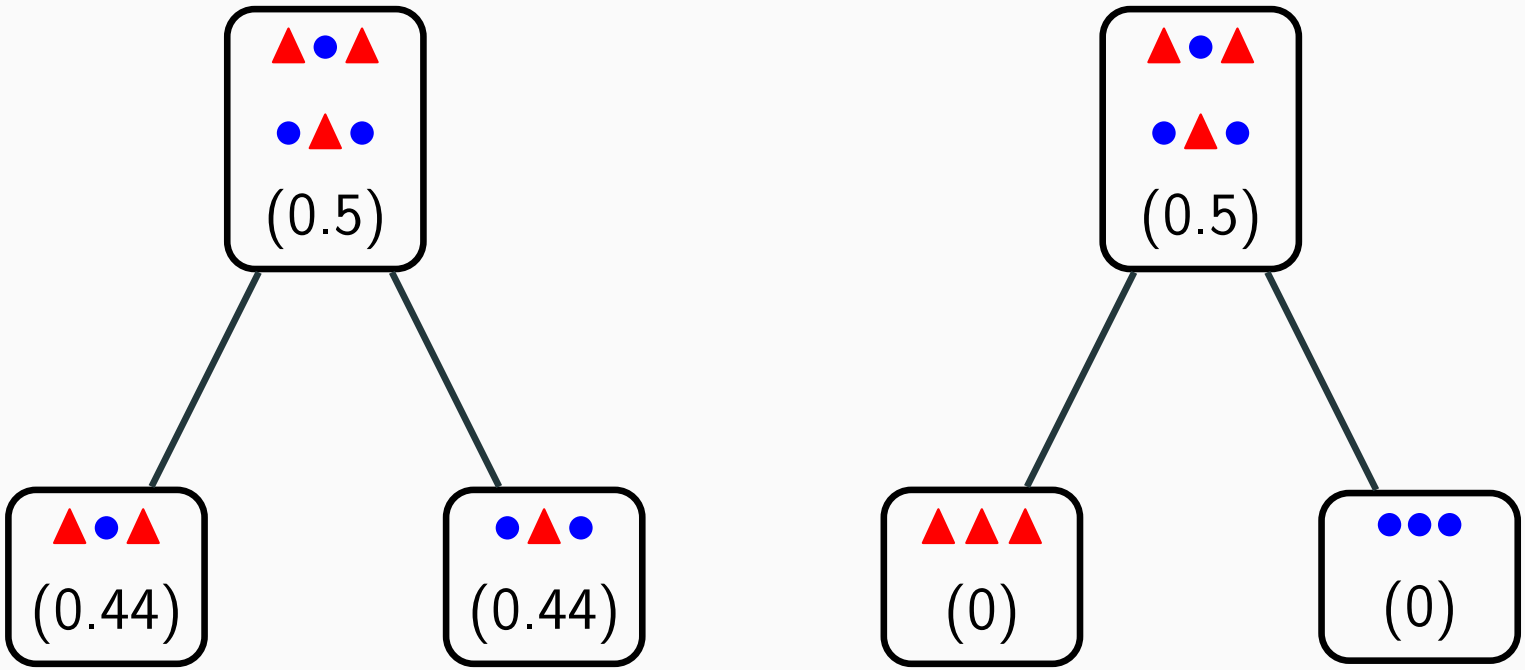


Exemple 1



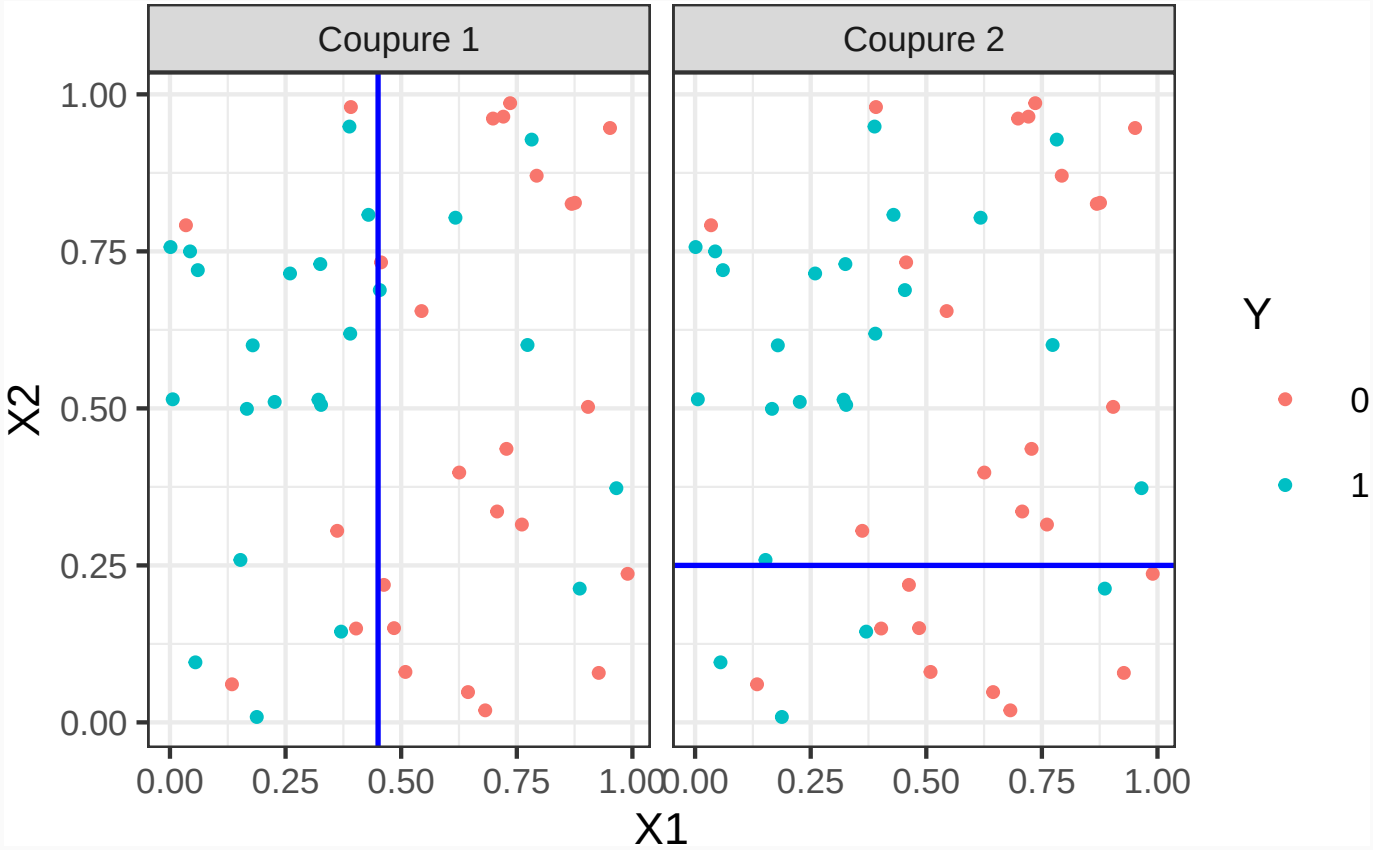


Exemple 1

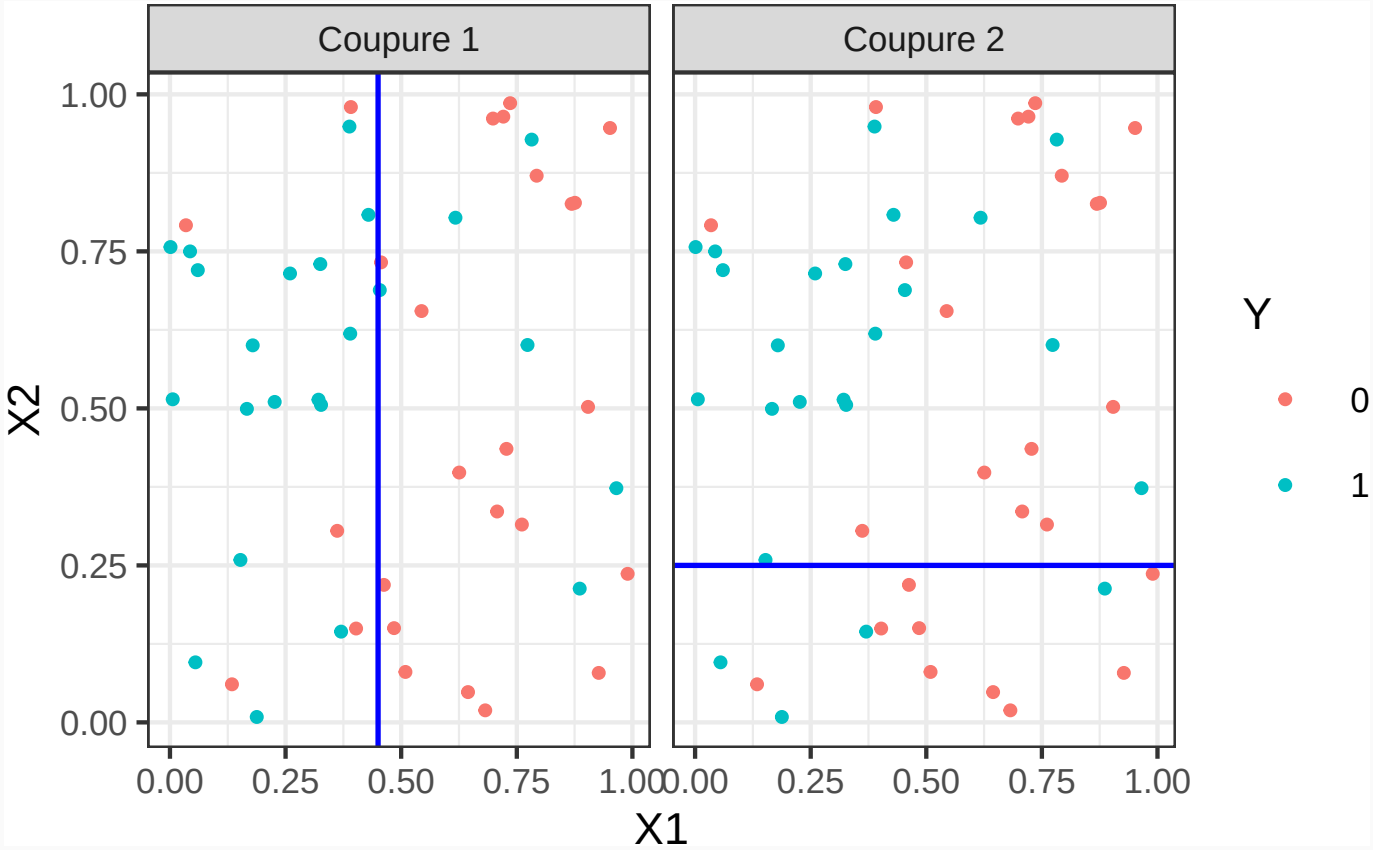


$\Longrightarrow$ coupure de droite plus performante.

Exemple 2


Coupure 1
Coupure 2
1.00
0.75
0.50
0.25
0.00
X2
0.00
0.25
0.50
0.75
1.00
X1
Y
0
1

Exemple 2



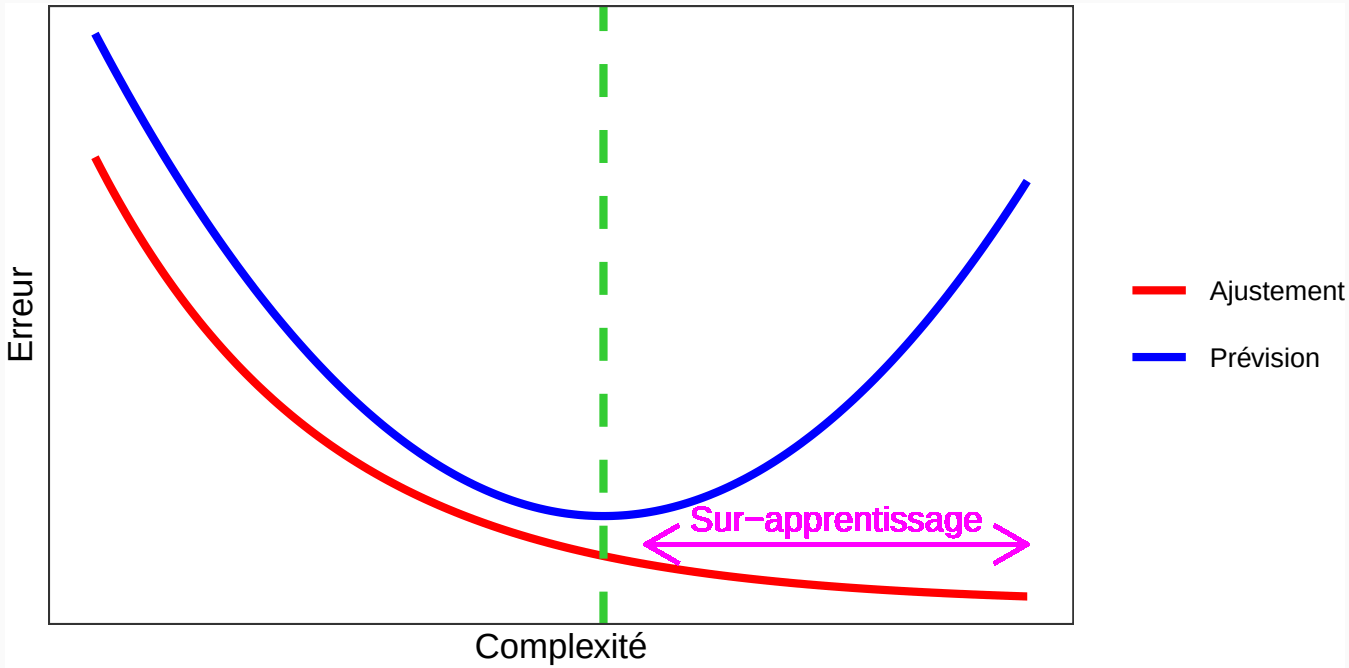
	$\mathcal{I}(\mathcal{N})$	$\mathcal{I}(\mathcal{N}_1)$	$\mathcal{I}(\mathcal{N}_2)$	Δ
Gauche	0.50	0.34	0.35	0.16
Droite	0.50	0.43	0.50	0.02

Pourquoi élaguer ?

- Les coupures permettent de séparer les données selon Y
  $\Longrightarrow$ plus on coupe mieux on ajuste !

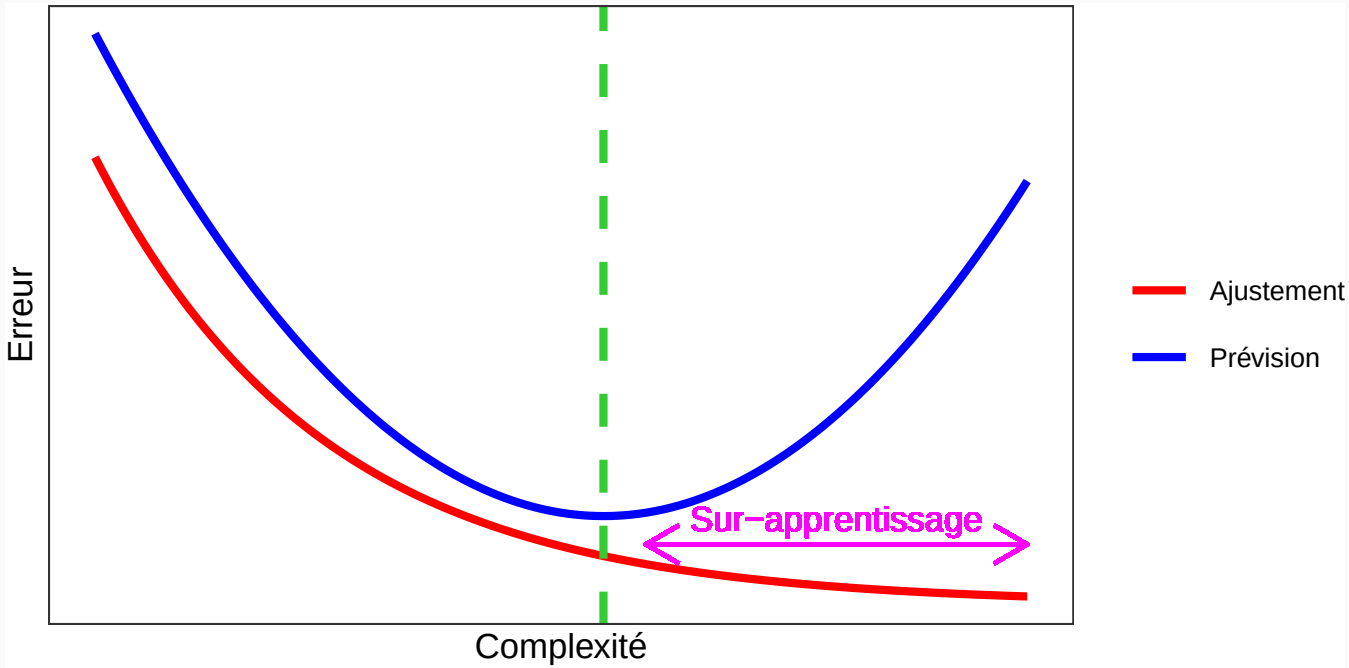
Pourquoi élaguer ?

- Les coupures permettent de séparer les données selon Y
  $\Longrightarrow$ 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
  $\Longrightarrow$ 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 ?

  $\Longrightarrow$ possible uniquement sur de petits échantillons !
- Critère d'arrêt : ne plus découper si une certaine condition est vérifiée.

Comment faire ?

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

Comment faire ?

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

Élaguer

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

Élagage CART

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

Élagage CART

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

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

de taille raisonnable (plus petite que n).

Élagage CART

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

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

  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).

Pour aller plus vite

Construction de la suite de sous arbres

- Soit T un arbre à $|T|$ nœuds terminaux $\mathcal{N}_1, \ldots, \mathcal{N}_{|T|}$.
- 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 $\mathcal{N}_1, \ldots, \mathcal{N}_{|T|}$.
- 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

- $\alpha = 0 \Longrightarrow T_\alpha = T_0 = T_{\text{max}}$.
- $\alpha = +\infty \Longrightarrow T_\alpha = T_{+\infty} = T_{\text{root}}$ 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

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

Question (a priori difficile)

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

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 $T_2 \subset T_1$

$\Longrightarrow$ 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 $T_2 \subset T_1$

$\Longrightarrow$ garantit une unique solution de taille minimale.

Lemme 2

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

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

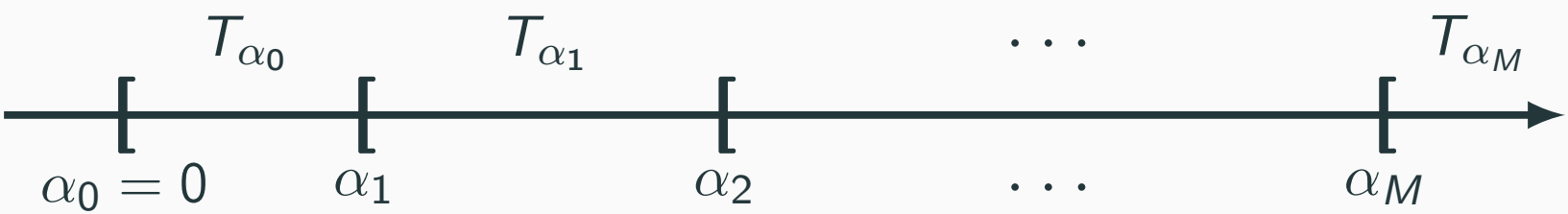
Théorème [Breiman et al., 1984]

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

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

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

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



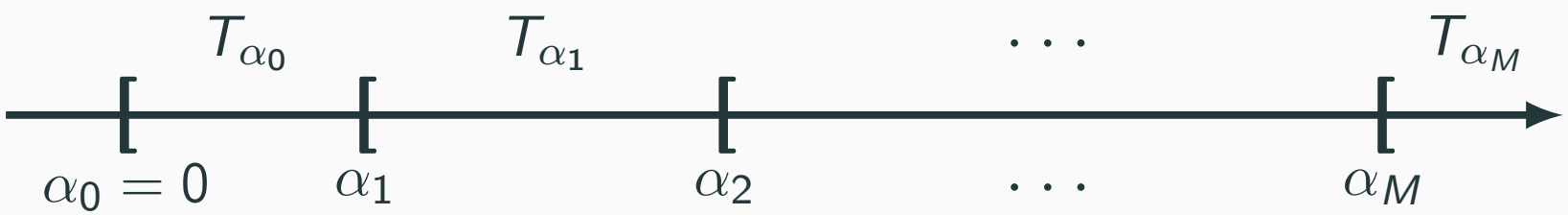
Théorème [Breiman et al., 1984]

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

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

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

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



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 1.0000000 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
```

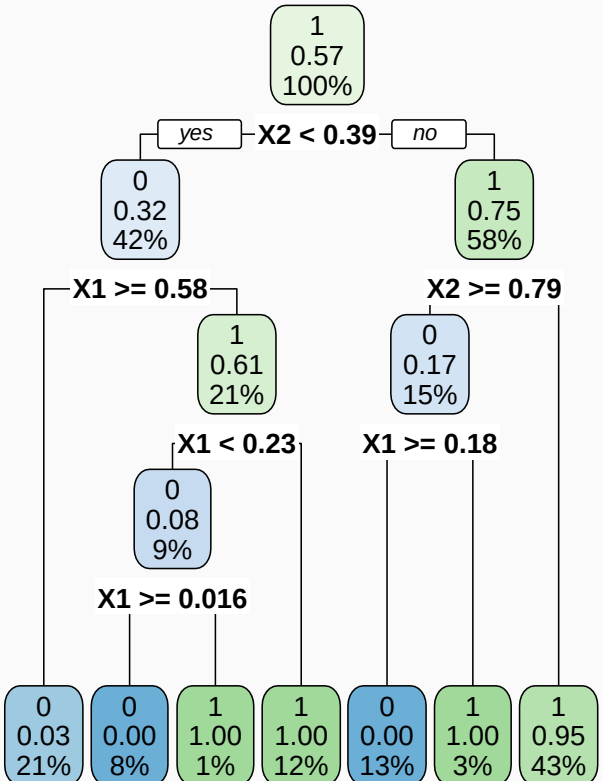
Sorties printcp

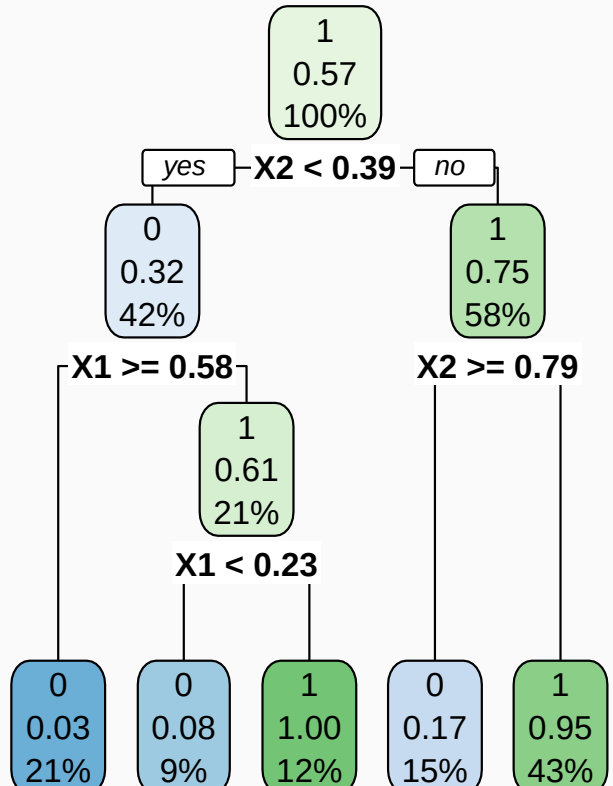
- Suite de 7 arbres emboités.
- CP : complexity parameter, il mesure la complexité de l'arbre : CP $\searrow \Longrightarrow$ complexité $\nearrow$.
- nsplit : nombre de coupures de l'arbre.
- rel.error : erreur (normalisée) calculée sur les données d'apprentissage $\Longrightarrow$ erreur d'ajustement.
- xerror : erreur (normalisée) calculée par validation croisée 10 blocs $\Longrightarrow$ 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.

Validation croisée rpart

1. Calculer

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

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

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

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

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

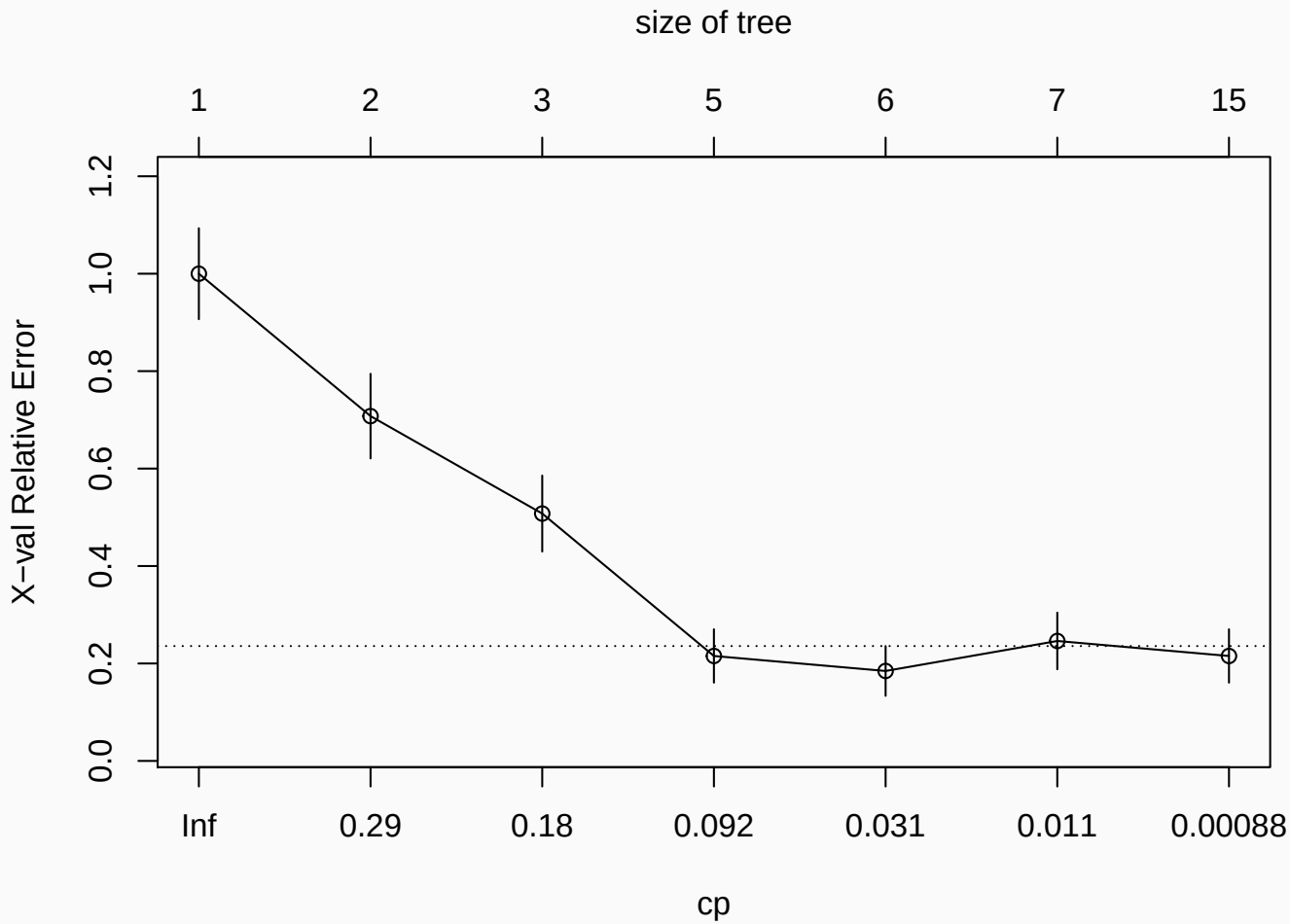
3. En déduire les erreurs pour chaque β_m :

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

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 :

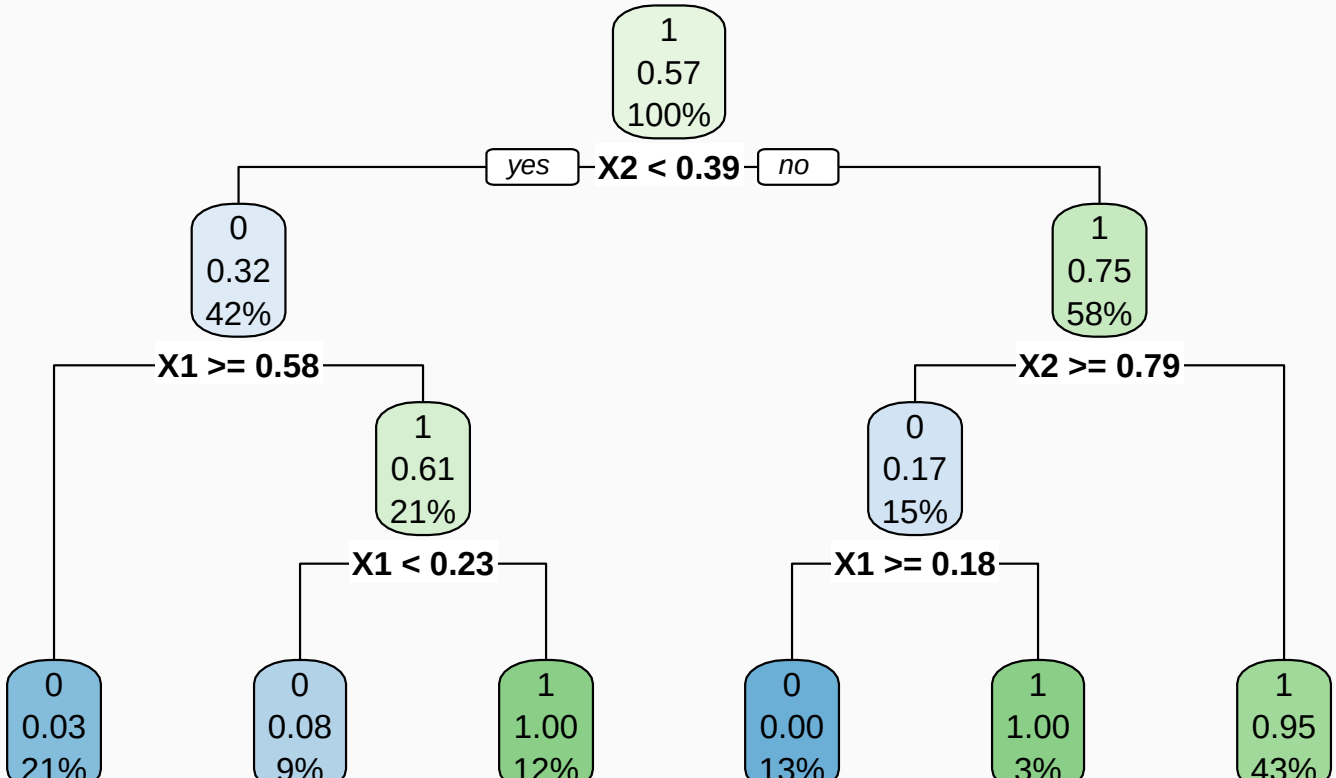
```
> cp_opt <- as_tibble(arbre$cptable) %>% arrange(xerror) %>%
+   slice(1) %>% select(CP) %>% as.numeric()
> cp_opt
## [1] 0.01538462
```

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

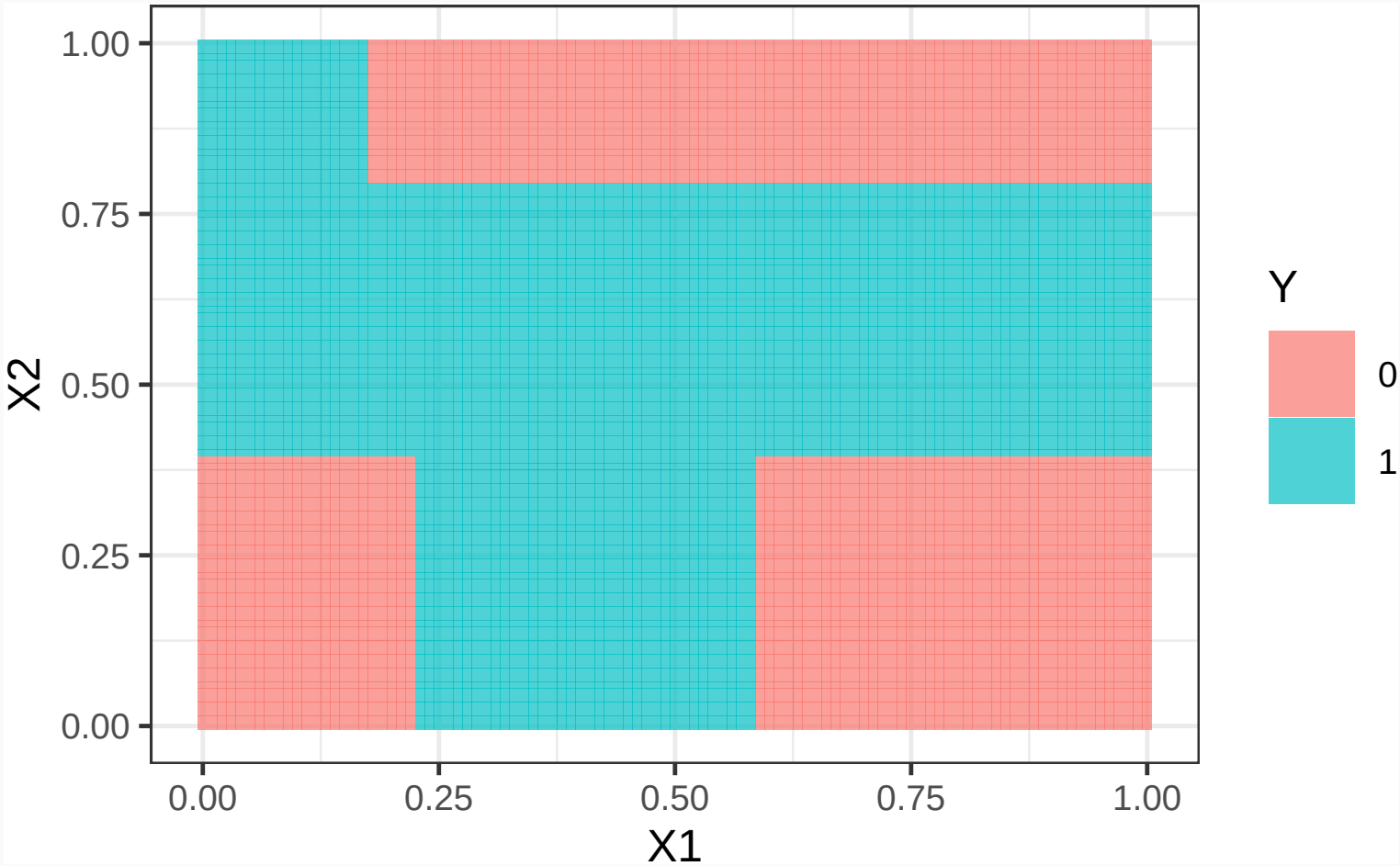
```
> cp_opt <- as_tibble(arbre$cptable) %>% arrange(xerror) %>%
+   slice(1) %>% select(CP) %>% as.numeric()
> cp_opt
## [1] 0.01538462
```

- et à le visualiser :

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



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



Prévision

- Nouvel individu :

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

Prévision

- 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

- 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
```

- 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 apparaitre explicitement dans l'arbre !

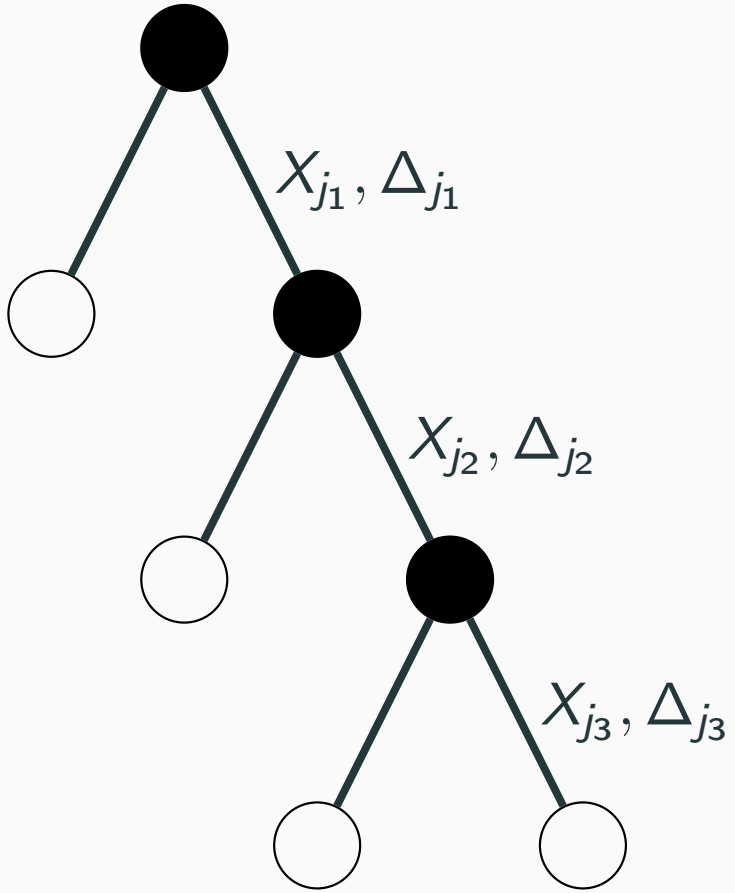
- La visualisation de l'arbre peut donner une idée sur l'importance des variables dans l'algorithme.
- Pas suffisant ! Il se peut en effet que des variables possèdent une grande importance sans pour autant apparaitre explicitement dans l'arbre !
  - Difficile de quantifier l'importance juste en regardant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

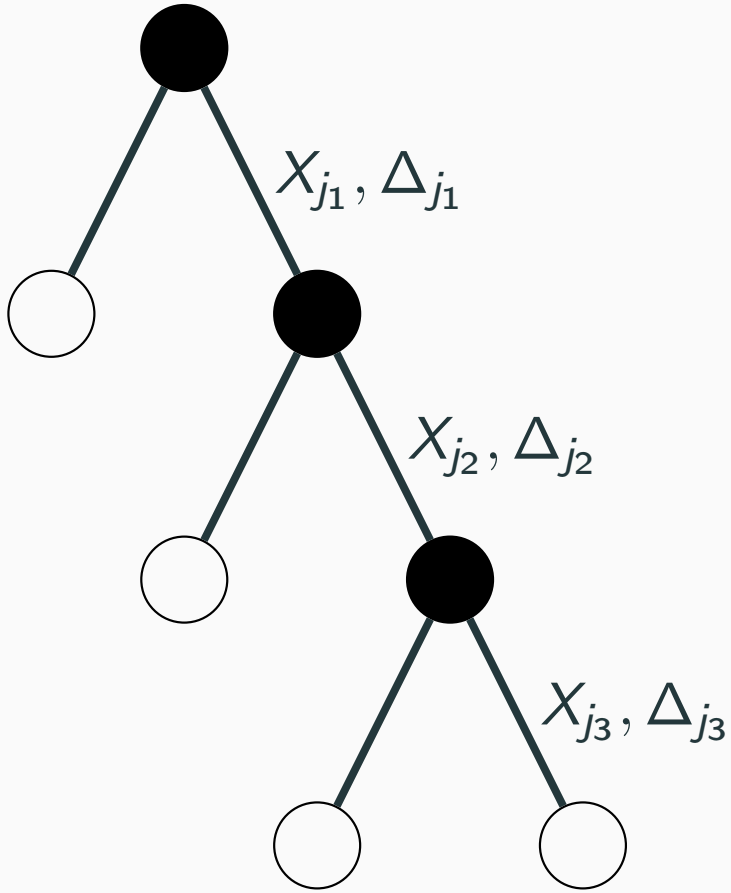
Mesure d'importance d'un arbre

Basée sur le gain d'impureté des nœuds internes.

- Nœuds internes $\Longrightarrow$
  $N_t, t = 1, \ldots, J-1$;
- Variables de coupure $\Longrightarrow X_{j_t}$;
- Gain d'impureté $\Longrightarrow i^2_{j_t}$.



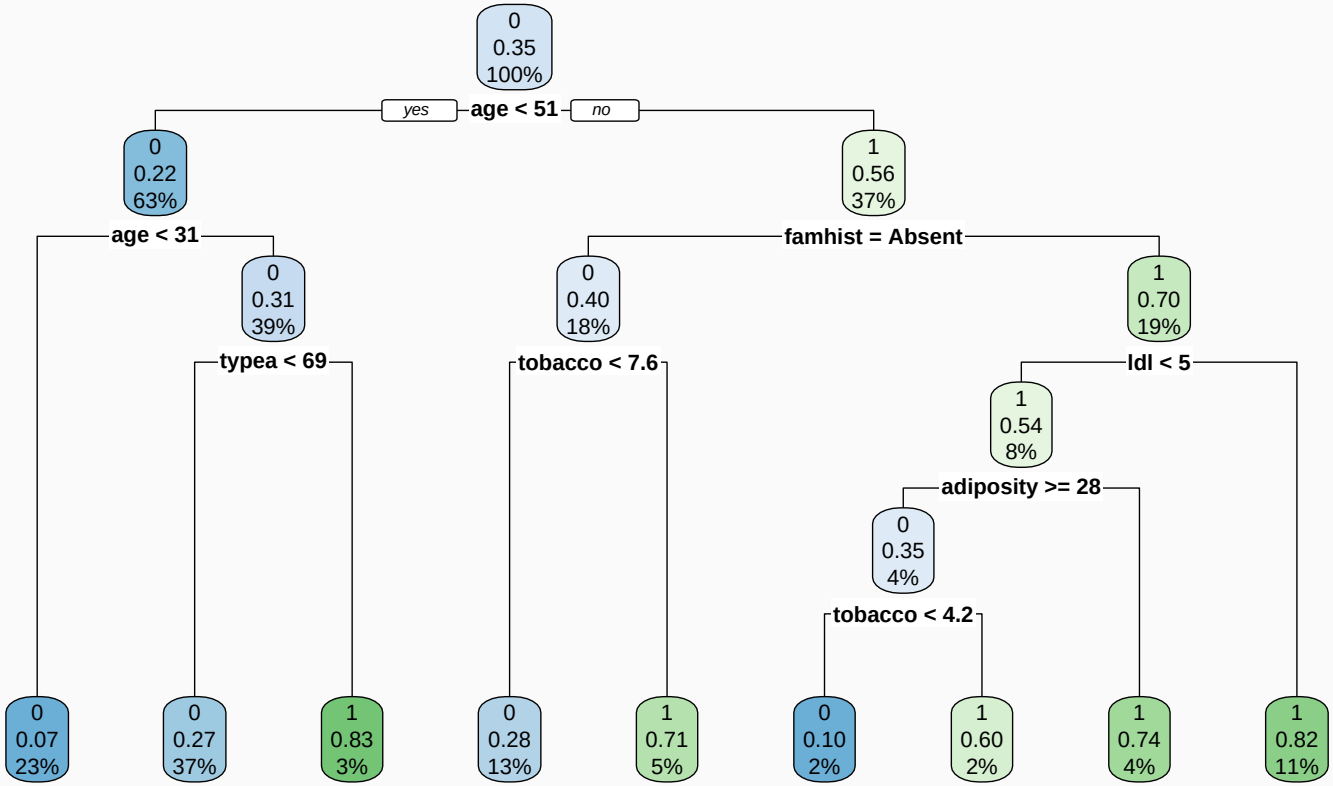
- Nœuds internes $\Longrightarrow$ $N_t, t = 1, \ldots, J-1$;
- Variables de coupure $\Longrightarrow X_{j_t}$;
- Gain d'impureté $\Longrightarrow i^2_{j_t}$.



Mesure d'impureté de la variable ℓ

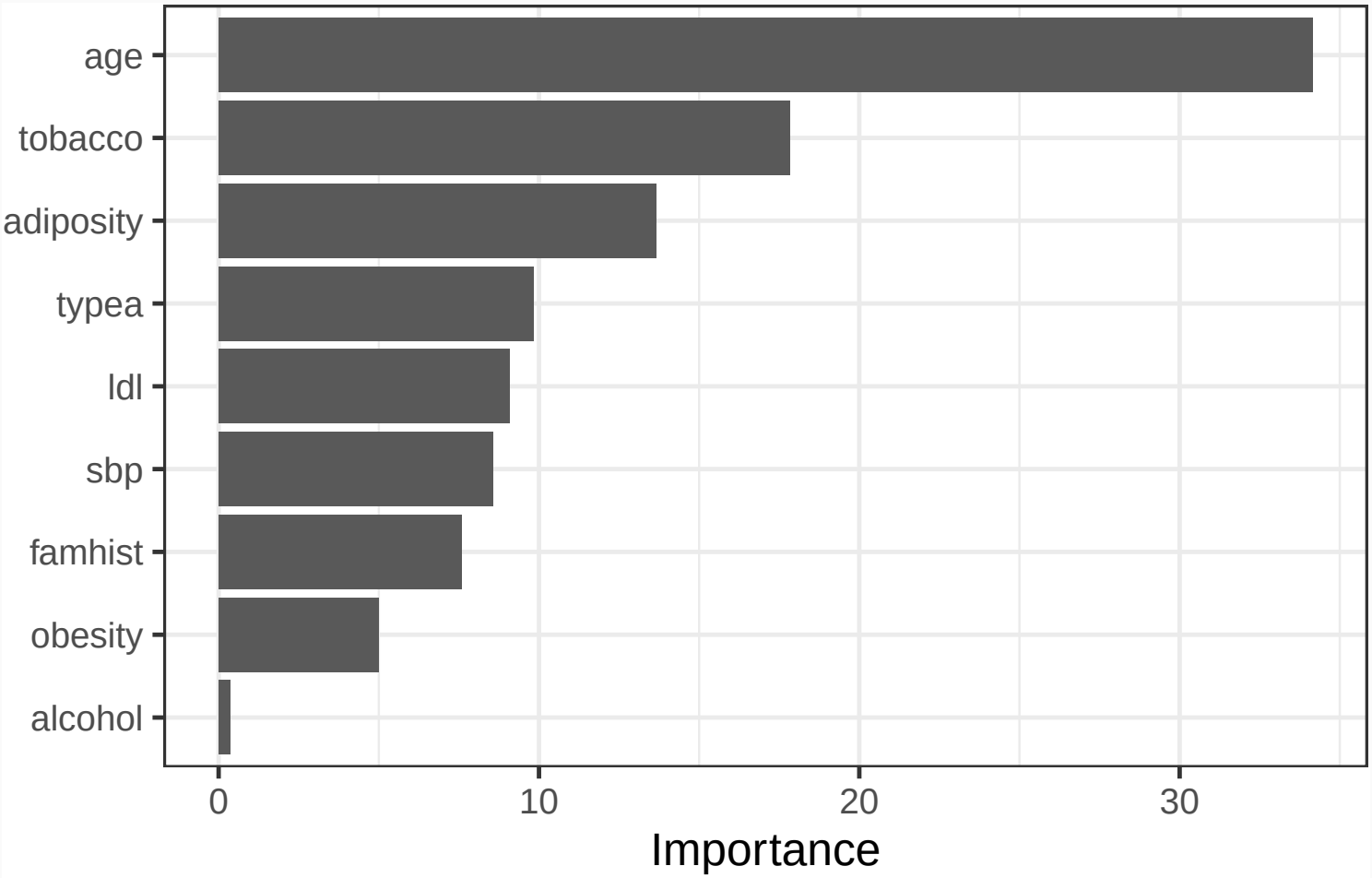
$$\mathcal{I}_\ell(T) = \sum_{t=1}^{|T|-1} \Delta_t 1_{j_t=\ell}.$$

Exemple



- Visualisation des importance à l'aide de vip :

```
> library(vip)
> vip(arbre)
```


age
tobacco
adiposity
typea
ldl
sbp
famhist
obesity
alcohol
0
10
20
30
Importance

Bilan

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).

Bilan

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.
     $\Longrightarrow$ Cet inconvénient sera un avantage pour des agrégations bootstrap
     $\Longrightarrow$ forêts aléatoires.

Références i


Breiman, L., Friedman, J., Olshen, R., and Stone, C. (1984).
***Classification and regression trees*.**
Wadsworth & Brooks.


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.

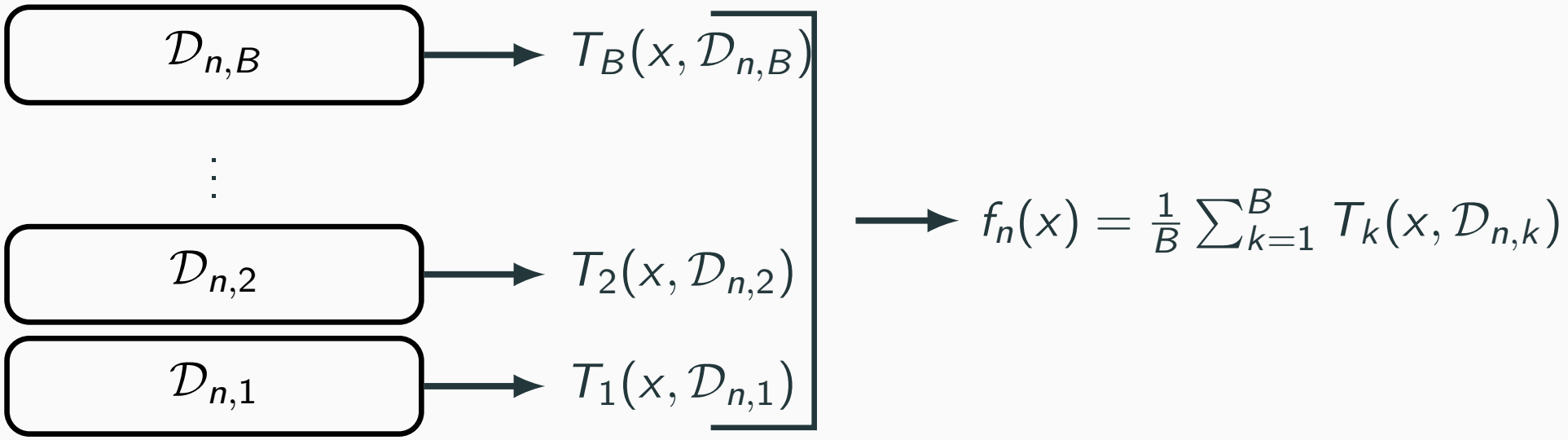
Références ii


Rumelhart, D. E., Hinton, G. E., and R. J. Williams, R. J. (1986).
Learning representations by back-propagating errors.
Nature, pages 533–536.

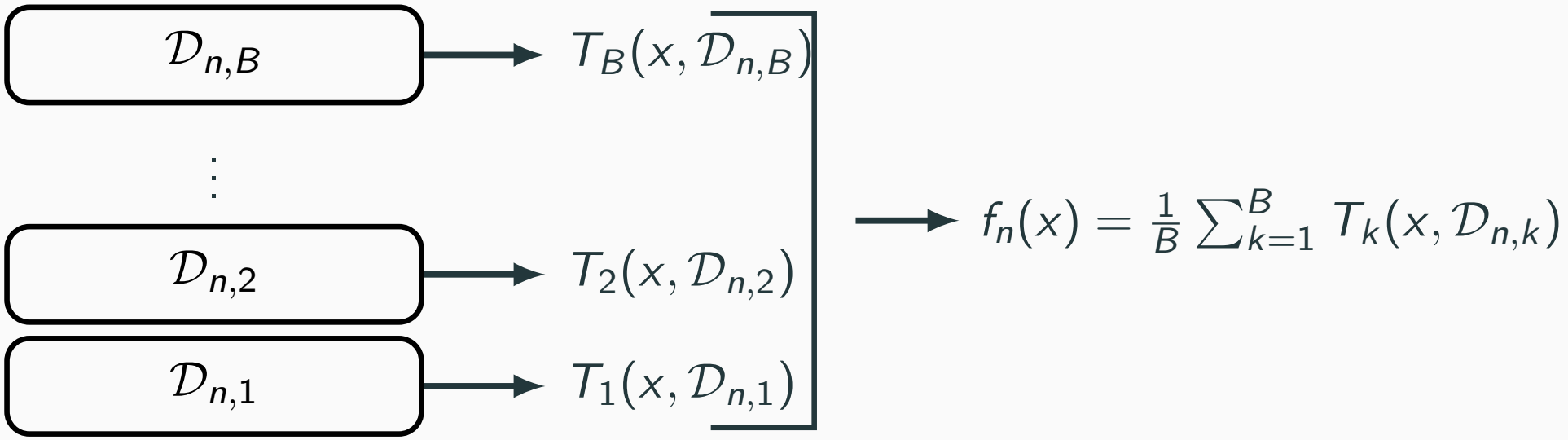
Troisième partie III

Agrégation

- 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. ...

Cadre

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

Cadre

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

- 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 multiclasses.

Cadre

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

- 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 multiclasses.

- Notations :
  - (X, Y) un couple aléatoire à valeurs dans $\mathbb{R}^d \times \mathbb{R}$.
  - $\mathcal{D}_n = (X_1, Y_1), \ldots, (X_n, Y_n)$ 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 $T_1, \ldots, T_b$ sont identiquement distribuées.

- Un algorithme de la forme :

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

- Hypothèse : les $T_1, \ldots, T_b$ sont identiquement distribuées.

Propriété

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

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

- Un algorithme de la forme :

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

- Hypothèse : les $T_1, \ldots, T_b$ sont identiquement distribuées.

Propriété

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

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

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.

- Le bagging désigne un ensemble de méthodes introduit par Léo Breiman [Breiman, 1996].

- Bagging : vient de la contraction de **B**ootstrap **Agg**regat**ing**.

- Idée : plutôt que de constuire 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
	⋮								⋮	
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 $\{1, \ldots, n\}$. On note θ_b l'ensemble des indices sélectionnés et $\mathcal{D}^{\star}_{n,b} = \{(x_i, y_i), i \in \theta_b\}$ l'échantillon bootstrap associé.
2. Entraîner l'algorithme T sur $\mathcal{D}^{\star}_{n,b} \Longrightarrow T(., \theta_b, \mathcal{D}_n)$.

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...

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

Un algorithme pas forcément aléatoire

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

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

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...

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

Conséquence

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

Choix de T

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

Choix de T

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

Conclusion

- Bagger ne modifie pas le biais.

Choix de T

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

Conclusion

- Bagger ne modifie pas le biais.
- B grand $\Longrightarrow \mathrm{V}[f_n(x)] \approx \rho(x)\mathrm{V}[T_1(x)]$

Choix de T

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

Conclusion

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

Choix de T

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

Conclusion

- Bagger ne modifie pas le biais.
- B grand $\Longrightarrow \mathrm{V}[f_n(x)] \approx \rho(x)\mathrm{V}[T_1(x)] \Longrightarrow$ 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.

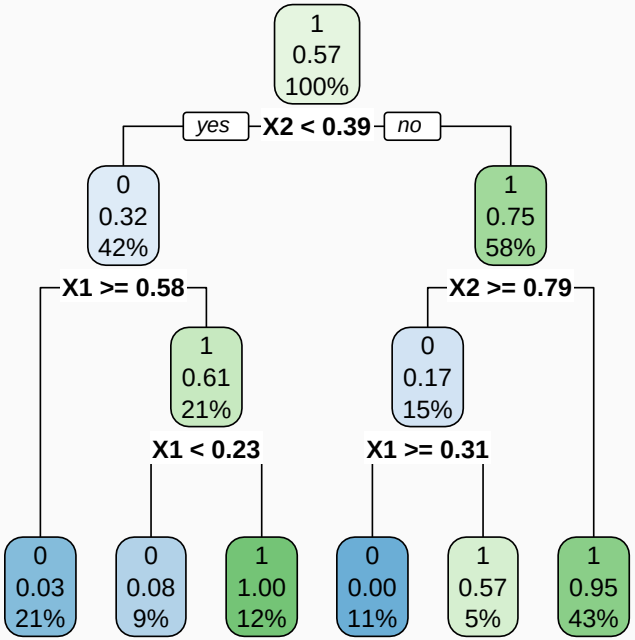
Choix de T

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

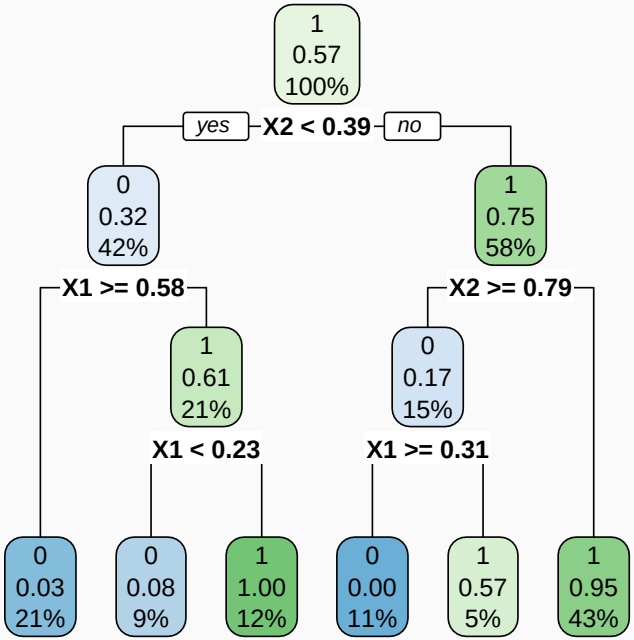
Conclusion

- Bagger ne modifie pas le biais.
- B grand $\Longrightarrow \mathrm{V}[f_n(x)] \approx \rho(x)\mathrm{V}[T_1(x)] \Longrightarrow$ 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.

Rappels sur les arbres



Rappels sur les arbres



Complexité

Profondeur

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

Définition

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

Définition

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

Définition

Soit $T_k(x), k = 1, \ldots, B$ des prédicteurs par arbre ($T_k : \mathbb{R}^d \to \mathbb{R}$). 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

- Forêts aléatoires = collection d'abres.

Forêts aléatoires

- Forêts aléatoires = collection d'abres.
- 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'abres.
- 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.

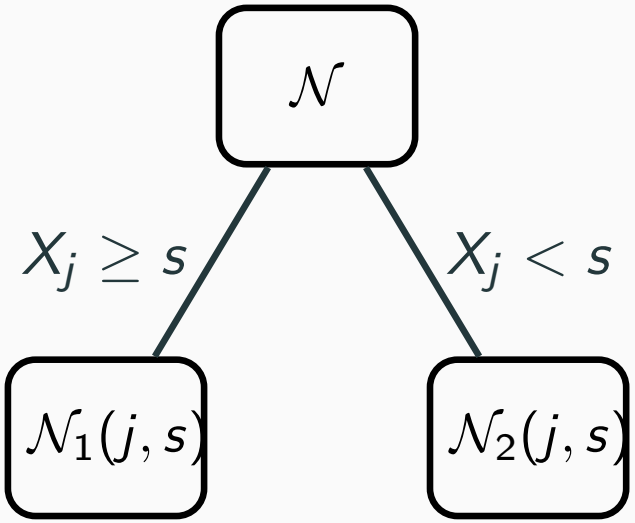
Forêts aléatoires

- Forêts aléatoires = collection d'abres.
- 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].

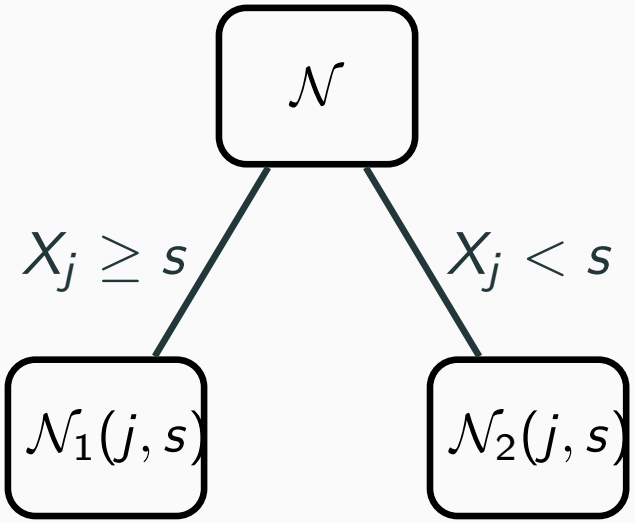
Coupures "aléatoires"



Arbres pour forêt

- Breiman propose de sélectionner la "meilleure" variable dans un ensemble composé uniquement de mtry 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 mtry 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 $\{1, \ldots, n\}$. On note $\mathcal{I}_b$ l'ensemble des indices sélectionnés et $\mathcal{D}^\star_{n,b} = \{(x_i, y_i), i \in \mathcal{I}_b\}$ l'échantillon bootstrap associé.
2. Construire un arbre CART à partir de $\mathcal{D}^\star_{n,b}$ 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 $T(., \theta_b, \mathcal{D}_n)$ 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) : $T(x, \theta_b, \mathcal{D}_n) \in \{1, \ldots, K\}$ et

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

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) : $T(x, \theta_b, \mathcal{D}_n) \in \{1, \ldots, K\}$ et

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

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

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

Le coin R

- 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 %
```

- B réglé $\Longrightarrow$ 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 $\Longrightarrow$ le plus grand possible.
  En pratique on pourra s'assurer que le courbe d'erreur en fonction du nombre d'arbres est stabilisée.
- Pour les autres paramètres on étudie à nouveau :

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

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 $\Longrightarrow$ le plus grand possible.
  En pratique on pourra s'assurer que le courbe d'erreur en fonction du nombre d'arbres est stabilisée.
- Pour les autres paramètres on étudie à nouveau :

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

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 $\Longrightarrow$ le plus grand possible.
  En pratique on pourra s'assurer que le courbe d'erreur en fonction du nombre d'arbres est stabilisée.
- Pour les autres paramètres on étudie à nouveau :

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

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 mtry

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

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.

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* $\searrow$

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

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 $\Longrightarrow$ 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 $\Longrightarrow$ les arbres sont de plus en plus différents $\Longrightarrow$ $\rho(x) \searrow \Longrightarrow$ 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* $\searrow$
  1. tendance à se rapprocher d'un choix "aléatoire" des variables de découpe des arbres $\Longrightarrow$ les arbres sont de plus en plus différents $\Longrightarrow$ $\rho(x) \searrow \Longrightarrow$ la variance de la forêt diminue.
  2. mais... le biais des arbres $\nearrow$

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 $\Longrightarrow$ les arbres sont de plus en plus différents $\Longrightarrow$ $\rho(x) \searrow \Longrightarrow$ la variance de la forêt diminue.
  2. mais... le biais des arbres $\nearrow \Longrightarrow$ 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 $\Longrightarrow$ les arbres sont de plus en plus différents $\Longrightarrow$ $\rho(x) \searrow \Longrightarrow$ la variance de la forêt diminue.
  2. mais... le biais des arbres $\nearrow \Longrightarrow$ le biais de la forêt $\nearrow$.
- Inversement lorsque *mtry* $\nearrow$ (risque de sur-ajustement).

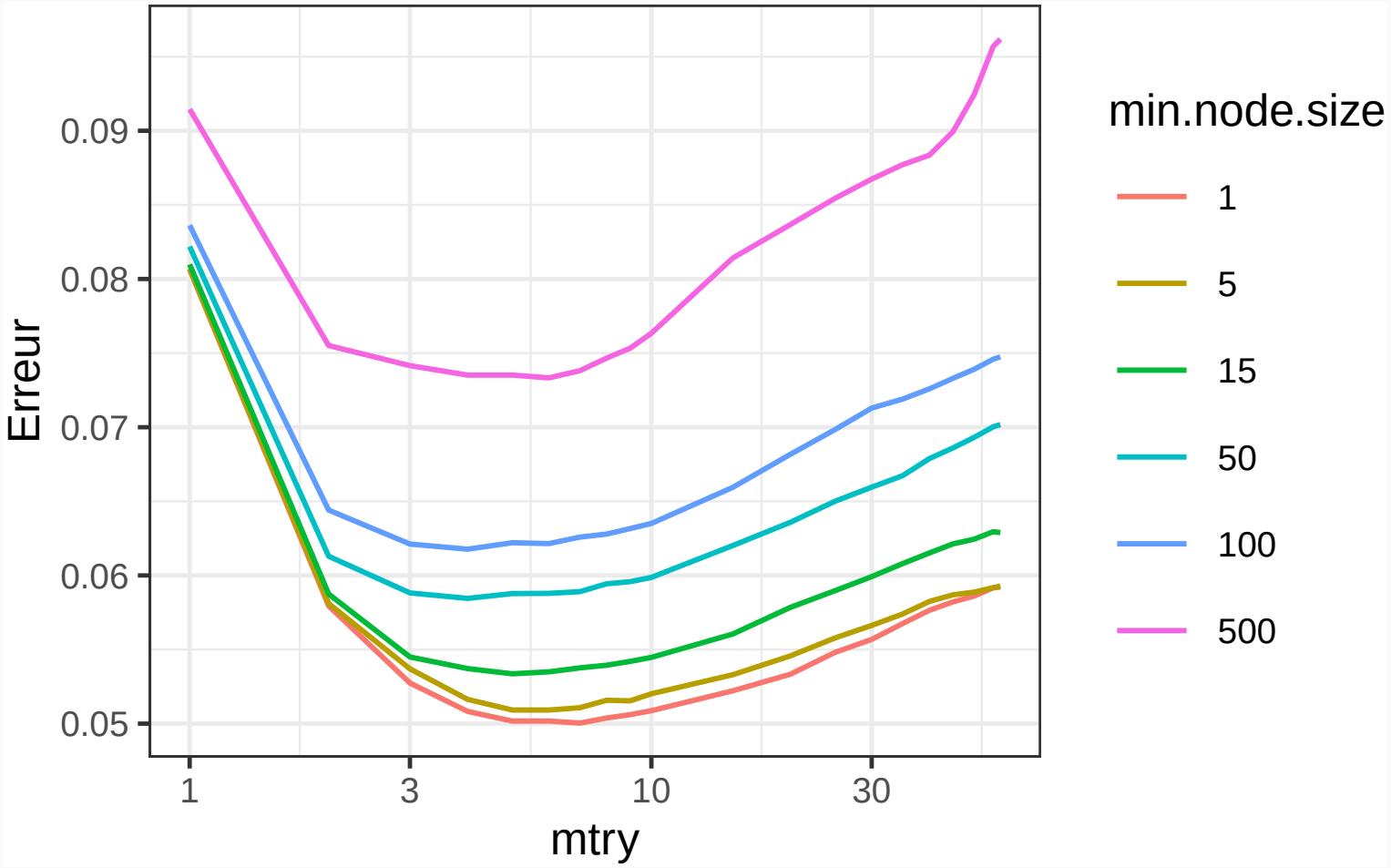
Choix de mtry

- 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 $\Longrightarrow$ les arbres sont de plus en plus différents $\Longrightarrow$ $\rho(x) \searrow \Longrightarrow$ la variance de la forêt diminue.
  2. mais... le biais des arbres $\nearrow \Longrightarrow$ le biais de la forêt $\nearrow$.
- Inversement lorsque *mtry* $\nearrow$ (risque de sur-ajustement).

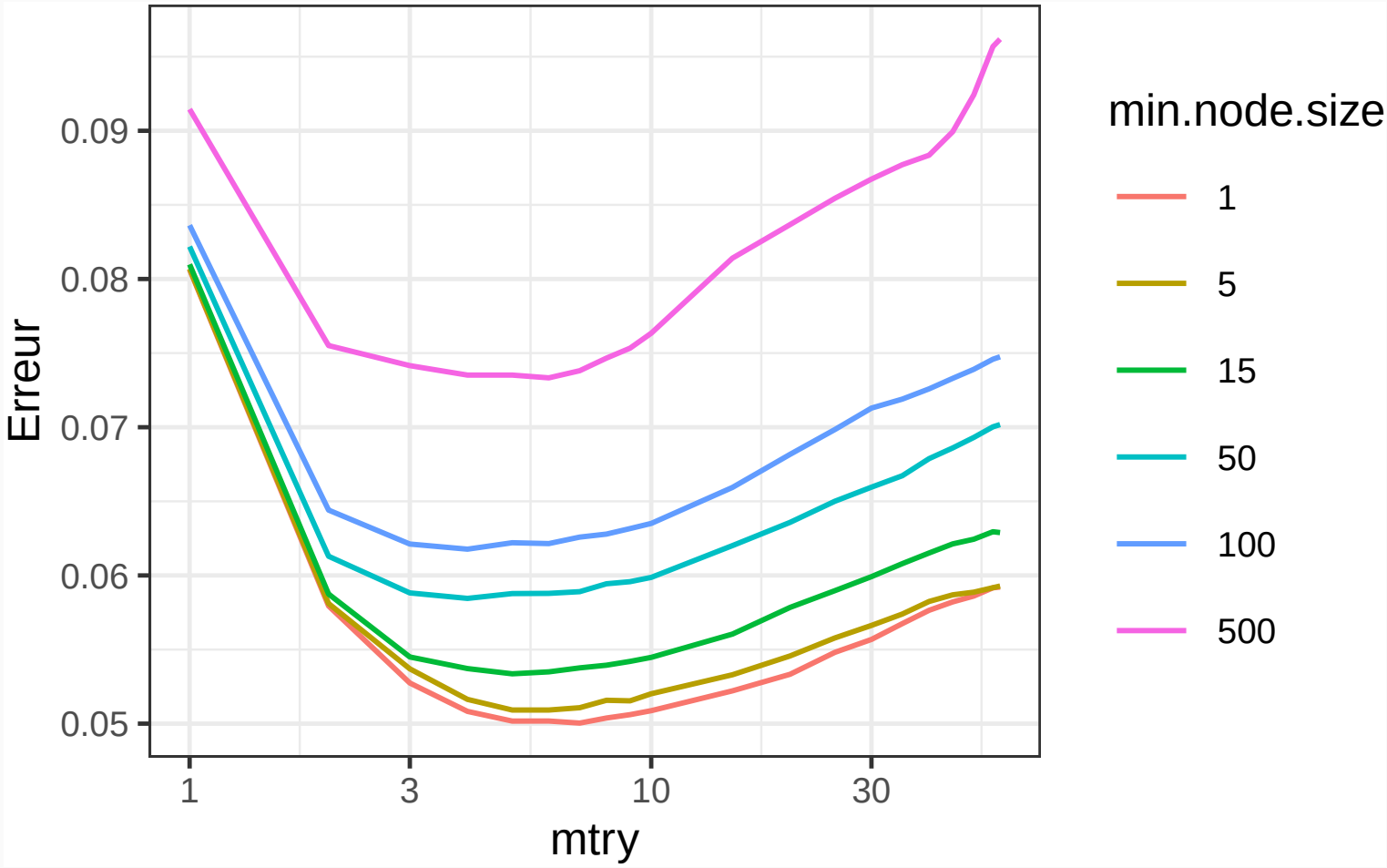
Conclusion

- 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.

En pratique

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

En pratique

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

En pratique

- 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)
```

- 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 $\Longrightarrow \hat{y}_2, \ldots, \hat{y}_n$.

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 $\Longrightarrow \hat{y}_2, \ldots, \hat{y}_n$.
- 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 $i = 1, \ldots, n$ on note

$$\mathrm{OOB}(i) = \{b \leq B : i \notin \mathcal{I}_b\}$$

l'ensemble des tirages bootstrap qui ne contiennent pas i et

$$f_{n,\mathrm{OOB}(i)}(x_i) = \frac{1}{|\mathrm{OOB}(i)|} \sum_{b \in \mathrm{OOB}(i)} T(x_i, \theta_b, \mathcal{D}_n)$$

la prévision de la forêt en ne considérant que les arbres pour lesquels i n'est pas dans le tirage bootstrap.

OOB définition

- Pour $i = 1, \ldots, n$ on note

$$\mathrm{OOB}(i) = \{b \leq B : i \notin \mathcal{I}_b\}$$

l'ensemble des tirages bootstrap qui ne contiennent pas i et

$$f_{n,\mathrm{OOB}(i)}(x_i) = \frac{1}{|\mathrm{OOB}(i)|} \sum_{b \in \mathrm{OOB}(i)} T(x_i, \theta_b, \mathcal{D}_n)$$

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 au valeurs observées, par exemple

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f_{n,\mathrm{OOB}(i)}(x_i))^2 \quad \text{ou} \quad \frac{1}{n}\sum_{i=1}^{n} 1_{f_{n,\mathrm{OOB}(i)}(x_i) \neq y_i}.$$

OOB définition

- Pour $i = 1, \ldots, n$ on note

$$\mathrm{OOB}(i) = \{b \leq B : i \notin \mathcal{I}_b\}$$

l'ensemble des tirages bootstrap qui ne contiennent pas i et

$$f_{n,\mathrm{OOB}(i)}(x_i) = \frac{1}{|\mathrm{OOB}(i)|} \sum_{b \in \mathrm{OOB}(i)} T(x_i, \theta_b, \mathcal{D}_n)$$

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 au valeurs observées, par exemple

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - f_{n,\mathrm{OOB}(i)}(x_i))^2 \quad \text{ou} \quad \frac{1}{n}\sum_{i=1}^{n} 1_{f_{n,\mathrm{OOB}(i)}(x_i) \neq y_i}.$$

$\Longrightarrow$ 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.

  $\Longrightarrow$ Idée : Si X_j est importante ces erreurs doivent êtres 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

$$\text{OOB}_b = \{i \leq n : i \notin \mathcal{I}_b\}.$$

$\Longrightarrow$ 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

$$\text{OOB}_b = \{i \leq n : i \notin \mathcal{I}_b\}.$$

$\Longrightarrow$ 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.

$$\begin{bmatrix} x_{11} & \dots & x_{1j} & \dots & x_{1d} \\ x_{21} & \dots & x_{2j} & \dots & x_{2d} \\ x_{51} & \dots & x_{3j} & \dots & x_{3d} \\ x_{41} & \dots & x_{4j} & \dots & x_{4d} \\ x_{51} & \dots & x_{5j} & \dots & x_{5d} \end{bmatrix} \Longrightarrow \begin{bmatrix} x_{11} & \dots & x_{3j} & \dots & x_{1d} \\ x_{21} & \dots & x_{5j} & \dots & x_{2d} \\ x_{51} & \dots & x_{1j} & \dots & x_{3d} \\ x_{41} & \dots & x_{2j} & \dots & x_{4d} \\ x_{51} & \dots & x_{4j} & \dots & x_{5d} \end{bmatrix}$$

- On note $\tilde{x}_i^j$ les individus de l'échantillon OOB_b permuté et on calcule

$$\text{Err}(\text{OOB}_b^j) = \frac{1}{|\text{OOB}_b|} \sum_{i \in \text{OOB}_b} (y_i - T(\tilde{x}_i^j, \theta_b, \mathcal{D}_n))^2.$$

$$\begin{bmatrix} x_{11} & \dots & x_{1j} & \dots & x_{1d} \\ x_{21} & \dots & x_{2j} & \dots & x_{2d} \\ x_{51} & \dots & x_{3j} & \dots & x_{3d} \\ x_{41} & \dots & x_{4j} & \dots & x_{4d} \\ x_{51} & \dots & x_{5j} & \dots & x_{5d} \end{bmatrix} \Longrightarrow \begin{bmatrix} x_{11} & \dots & x_{3j} & \dots & x_{1d} \\ x_{21} & \dots & x_{5j} & \dots & x_{2d} \\ x_{51} & \dots & x_{1j} & \dots & x_{3d} \\ x_{41} & \dots & x_{2j} & \dots & x_{4d} \\ x_{51} & \dots & x_{4j} & \dots & x_{5d} \end{bmatrix}$$

- On note $\tilde{x}_i^j$ les individus de l'échantillon OOB_b permuté et on calcule

$$\text{Err}(\text{OOB}_b^j) = \frac{1}{|\text{OOB}_b|} \sum_{i \in \text{OOB}_b} (y_i - T(\tilde{x}_i^j, \theta_b, \mathcal{D}_n))^2.$$

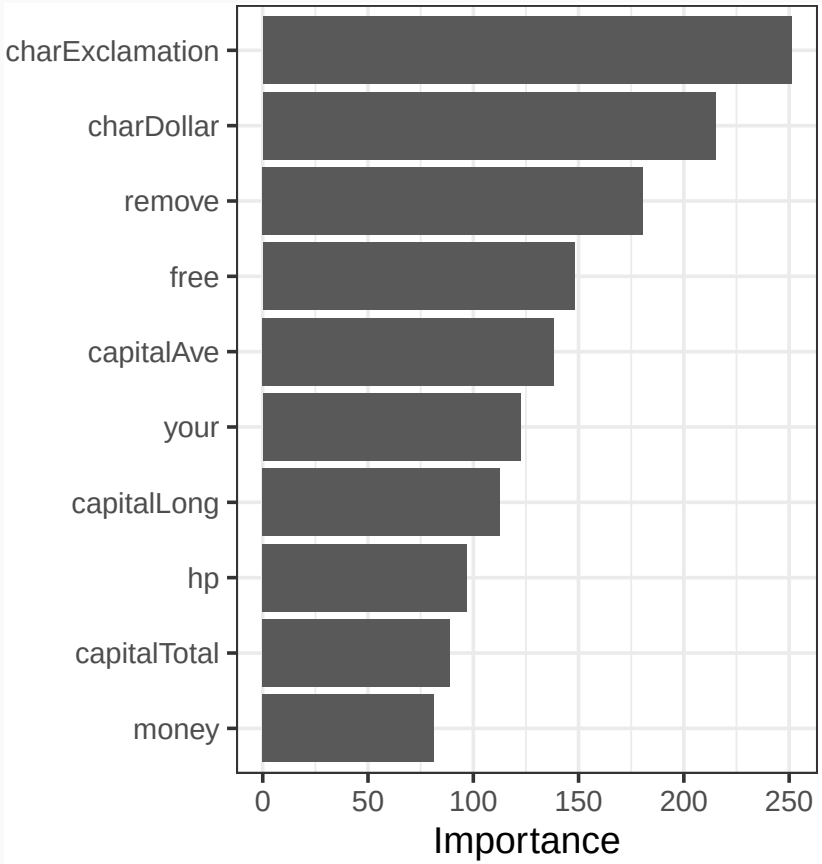
Importance par permutation

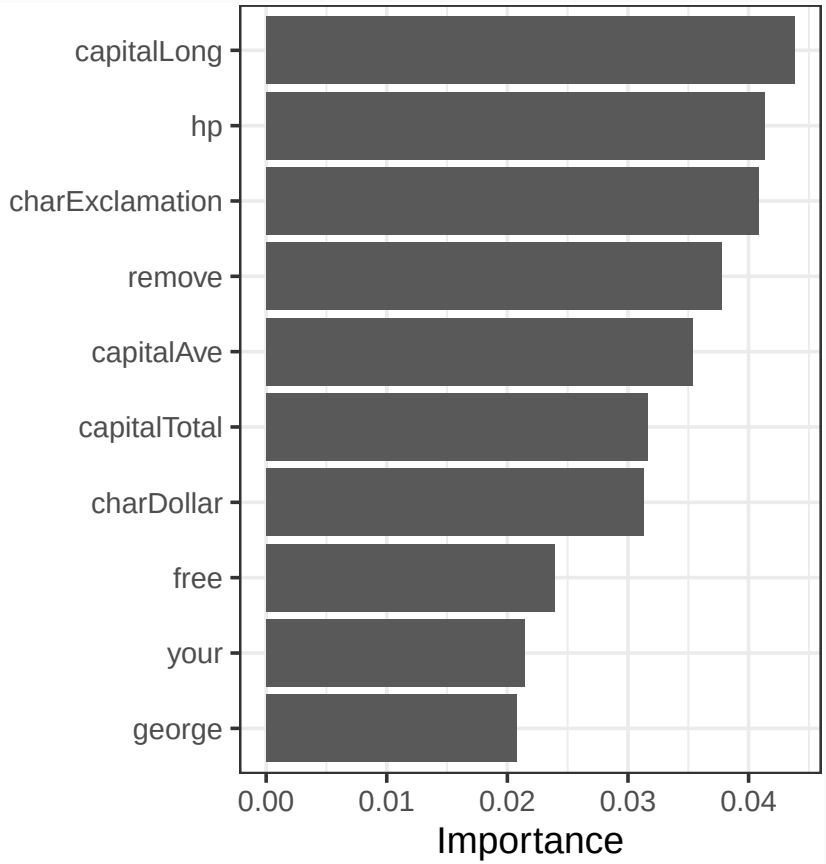
$$\mathcal{I}_j^{\text{perm}} = \frac{1}{B} \sum_{b=1}^{B} (\text{Err}(\text{OOB}_b^j) - \text{Err}(\text{OOB}_b)).$$

Le coin R

- 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)
```





Conclusion

Beaucoup d'avantages

- Bonnes performances prédictives $\Longrightarrow$ souvent parmi les algorithmes de tête dans les compétitions [Fernández-Delgado et al., 2014].
- Facile à calibrer.

Conclusion

Beaucoup d'avantages

- Bonnes performances prédictives $\Longrightarrow$ 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...)

Références i


Breiman, L. (1996).
Bagging predictors.
Machine Learning, 26(2) :123–140.


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*.

Références ii


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.

Références iii

Hastie, T., Tibshirani, R., and Friedman, J. (2009).
***The Elements of Statistical Learning : Data Mining, Inference, and Prediction*.**
Springer, second edition.

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..)
  $\Longrightarrow$ performance $\nearrow$ réseaux pré-entrainés $\Longrightarrow$ apprentissage profond/deep learning.