Apprentissage Statistique

Magalie Fromont

Master de Statistique appliquée, Université Rennes 2

2016 - 2017

Plan du cours

Plan du cours

Encyclopedia Universalis

La Statistique est l'activité qui consiste à recueillir, traiter et interpréter un ensemble de données d'observations.

- Statistique descriptive
- Statistique inférentielle : paramétrique / semi, non paramétrique

Développement de moyens informatiques

↙ ↘

données en masse
de nature et d'origine variées

⇓

data mining

⇓

big data

↗

méthodes d'analyse
faisant appel au calcul intensif

⇓

apprentissage (statistique)

De nombreux types d'apprentissage :

- non supervisé : classification/segmentation
- supervisé :
  - régression
  - discrimination/classement/reconnaissance de forme
- semi-supervisé...

De nombreuses applications possibles en :

marketing, finance, économie, informatique, biologie, médecine, psychologie, physique, géophysique, écologie, géologie, archéologie, mais aussi disciplines littéraires, artistiques...

Plan du cours

2. Apprentissage statistique supervisé
   - De quoi s'agit-il ?
   - Modèle statistique non paramétrique
   - Qualité de prédiction
   - Estimation du risque moyen

De quoi s'agit-il ?

Données observées de type entrée-sortie : $d_1^n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ avec $x_i \in \mathcal{X}$ quelconque (souvent égal à $\mathbb{R}^d$), $y_i \in \mathcal{Y}$ pour $i = 1 \ldots n$.

Objectif : prédire la sortie y associée à une nouvelle entrée x, sur la base de d_1^n.

sorties quantitatives	sorties qualitatives
$\mathcal{Y} \subset \mathbb{R}^{d'}$	$\mathcal{Y}$ fini
$\downarrow$	$\downarrow$
régression	**discrimination, classement, reconnaissance de forme**

Partie théorique portant essentiellement sur la régression réelle ($\mathcal{Y} \subset \mathbb{R}$) et la discrimination binaire ($\mathcal{Y} = \{-1, 1\}$)

Modèle statistique non paramétrique

On suppose que d_1^n est l'observation d'un n-échantillon $D_1^n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ d'une loi conjointe P sur $\mathscr{X} \times \mathscr{Y}$, totalement inconnue.
On suppose que x est une observation de la variable X, (X, Y) étant un couple aléatoire de loi conjointe P indépendant de D_1^n.

Une **règle de prédiction (régression ou discrimination)** est une fonction (mesurable) $f : \mathscr{X} \to \mathscr{Y}$ qui associe la sortie $f(x)$ à l'entrée $x \in \mathscr{X}$.

Qualité de prédiction

Soit $l : \mathscr{Y} \times \mathscr{Y} \to \mathbb{R}_+$ une fonction de perte (i.e. $l(y,y) = 0$ et $l(y,y') > 0$ pour $y \neq y'$), par exemple :

- $l(y,y') = |y - y'|^q$ en régression réelle
  (perte absolue si $q = 1$, perte quadratique si $q = 2$)
- $l(y,y') = \mathbb{1}_{y \neq y'} = \frac{|y-y'|}{2} = \frac{(y-y')^2}{4}$ en discrimination binaire

Le **risque** - ou l'**erreur de généralisation** - d'une règle de prédiction f est défini par

$$R_P(f) = \mathbb{E}_{(X,Y)\sim P}[l(Y, f(X))].$$

Si $\mathscr{F}$ désigne l'ensemble des règles de prédiction possibles, quelles sont les règles de prédiction optimales au sens de la minimisation du risque sur $\mathscr{F}$ i.e. les règles f^* telles que $R_P(f^*) = \inf_{f \in \mathscr{F}} R_P(f)$?

Régression réelle et discrimination

On appelle **fonction de régression** la fonction $\eta^* : \mathscr{X} \to \mathscr{Y}$ définie par $\eta^*(x) = \mathbb{E}[Y|X = x]$.

Cas de la régression réelle

$$\underline{\mathscr{Y} = \mathbb{R}, \quad l(y, y') = (y - y')^2}$$

Théorème

La fonction de régression η^* vérifie $R_P(\eta^*) = \inf_{f \in \mathscr{F}} R_P(f)$.

$$\underline{\mathscr{Y} = \mathbb{R}, \quad l(y, y') = |y - y'|}$$

Théorème

La règle de régression définie par $\mu^*(x) = mediane[Y|X = x]$ vérifie $R_P(\mu^*) = \inf_{f \in \mathscr{F}} R_P(f)$.

Cas de la discrimination binaire

$$\mathscr{Y} = \{-1, 1\}, \quad l(y, y') = \mathbb{1}_{y \neq y'} = |y - y'|/2 = (y - y')^2/4$$

On appelle **règle de Bayes** toute fonction ϕ^* de $\mathscr{F}$ telle que $\forall x \in \mathscr{X}$, $\mathbb{P}(Y = \phi^*(x)|X = x) = \max_{y \in \mathscr{Y}} \mathbb{P}(Y = y|X = x)$.

Théorème

Si ϕ^* est une règle de Bayes, alors $R_P(\phi^*) = \inf_{f \in \mathscr{F}} R_P(f)$.

On appelle **règle de discrimination plug in** toute fonction ϕ_η de $\mathscr{F}$ telle que : $\forall x \in \mathscr{X}$, $\phi_\eta(x) = signe(\eta(x))$, η étant une règle de régression.

Théorème

Pour toute règle de discrimination plug in ϕ_η,

$$\mathbb{E}_{(X,Y)\sim P}[\mathbb{1}_{Y \neq \phi_\eta(X)} - \mathbb{1}_{Y \neq \phi_{\eta^*}(X)}] \leq \mathbb{E}_{X \sim P_X}[|\eta(X) - \eta^*(X)|]$$
$$\leq \left(\mathbb{E}_{(X,Y)\sim P}[(Y - \eta(X))^2 - (Y - \eta^*(X))^2]\right)^{1/2}$$

Théorème

La règle de discrimination plug in définie par
$\phi_{\eta^*}(x) = signe(\eta^*(x)) = \mathbb{1}_{\eta^*(x)\geq 0} - \mathbb{1}_{\eta^*(x)<0}$ *est une règle de Bayes.*

✘ Dans tous les cas, ces règles optimales dépendent de P !

↪ Nécessité de construire des règles - ou **algorithmes** - de prédiction qui ne dépendent pas de P, mais de D_1^n.

Le **risque moyen** d'un algorithme de prédiction $f_{D_1^n}$ construit sur D_1^n est défini par $\mathscr{R}(f_{D_1^n}) = \mathbb{E}[l(Y, f_{D_1^n}(X))] = \mathbb{E}_{D_1^n \sim P^{\otimes n}} \mathbb{E}_{(X,Y)\sim P}[l(Y, f_{D_1^n}(X))]$.

✘ Le risque moyen d'un algorithme de prédiction dépend de P !

↪ Nécessité, pour juger de la qualité d'un algorithme de prédiction et/ou ajuster ses paramètres, de donner :

- Des résultats théoriques : asymptotiques (consistance) ou non (inégalités oracles).
- Une estimation du risque moyen.

Estimation du risque moyen

Le **risque empirique** (ou **risque apparent**) d'un algorithme de prédiction $f_{D_1^n}$ construit sur D_1^n est défini par

$$\widehat{\mathscr{R}_n}(f_{D_1^n}) = \frac{1}{n}\sum_{i=1}^{n} l\Big(Y_i, f_{D_1^n}(X_i)\Big).$$

⚠ Sous-estimation du risque moyen

⇒ Minimisation du risque empirique = sur-apprentissage (overfitting) !

Méthodes de validation croisée et de bootstrap

- Validation croisée hold-out (apprentissage / validation)
- Validation croisée leave-p-out

L' **estimateur Leave-One-Out** du risque moyen de $f_{D_1^n}$ est défini par $\frac{1}{n}\sum_{i=1}^{n} l(Y_i, f_{D_{(i)}}(X_i))$, où $D_{(i)}$ est l'échantillon D_1^n privé de son $i$ème point.

- Validation croisée K fold
- Bootstrap 0.632 et variantes

Algorithme de validation croisée hold-out

Variable

$\mathscr{A}$: *sous-ensemble de* $\{1,\dots,n\}$, *définissant l'ensemble d'apprentissage*

$\mathscr{V}$: *sous-ensemble de* $\{1,\dots,n\}$, *définissant l'ensemble de validation*

Début

Construire l'algorithme de prédiction $f_{D_{\mathscr{A}}}$ sur $D_{\mathscr{A}} = \{(X_i, Y_i),\ i \in \mathscr{A}\}$

Retourner $\frac{1}{\#\mathscr{V}} \sum_{i \in \mathscr{V}} l(Y_i, f_{D_{\mathscr{A}}}(X_i))$

Fin

Algorithme de validation croisée leave p out

Variable

p : *entier inférieur à* n

Début

Construire $\mathcal{V}_1, \ldots, \mathcal{V}_{\binom{n}{p}}$ parties à p éléments de $\{1, \ldots, n\}$

Pour k **variant de** 1 **à** $\binom{n}{p}$

Déterminer $\mathcal{A}_k = \{1, \ldots, n\} \setminus \mathcal{V}_k$

Construire $f_{D_{\mathcal{A}_k}}$ sur $D_{\mathcal{A}_k} = \{(X_i, Y_i),\ i \in \mathcal{A}_k\}$

Calculer $R_k = \frac{1}{p} \sum_{i \in \mathcal{V}_k} I\left(Y_i, f_{D_{\mathcal{A}_k}}(X_i)\right)$

FinPour

Retourner $\binom{n}{p}^{-1} \sum_{k=1}^{\binom{n}{p}} R_k$

Fin

Algorithme de validation croisée K fold

Variable

K : *entier diviseur de* n

Début

Construire $\mathcal{V}_1, \ldots, \mathcal{V}_K$ partition de $\{1, \ldots, n\}$

Pour k **variant de** 1 **à** K

Déterminer $\mathcal{A}_k = \{1, \ldots, n\} \setminus \mathcal{V}_k$

Construire $f_{D_{\mathcal{A}_k}}$ sur $D_{\mathcal{A}_k} = \{(X_i, Y_i),\ i \in \mathcal{A}_k\}$

Calculer $R_k = \frac{K}{n} \sum_{i \in \mathcal{V}_k} l\left(Y_i, f_{D_{\mathcal{A}_k}}(X_i)\right)$

FinPour

Retourner $\frac{1}{K} \sum_{k=1}^{K} R_k$

Fin

L'**estimateur Leave-One-Out bootstrap** (théorique) est défini par :

$$\mathscr{R}^*(f_{D_1^n}) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{E}\left[I\left(Y_i, f_{D^*_{(i)}}(X_i)\right) \middle| D_1^n \right],$$

où $D^*_{(i)}$ est un échantillon bootstrap de taille n associé à $D_{(i)}$.

Approximation de Monte Carlo

Un échantillon bootstrap $D^*_{(i)}$ de taille n associé à $D_{(i)}$ est aussi un échantillon bootstrap D^* associé à D_1^n ne contenant pas (X_i, Y_i).

D^* est simulé en tirant au hasard avec remise n éléments dans d_1^n.

$\mathscr{R}^*(f_{D_1^n})$ peut donc être approché par une méthode de Monte Carlo.

Algorithme de Leave-One-Out bootstrap

Variable

B : *entier assez grand*

Début

Pour b **variant de** 1 **à** B

Générer d^{*b} échantillon bootstrap associé à d_1^n

Calculer $I_i^b = 1$ si $(x_i, y_i) \notin d^{*b}$, 0 sinon.

FinPour

Retourner $\frac{1}{n}\sum_{i=1}^{n} \frac{\sum_{b=1}^{B} I_i^b I\left(y_i, f_{d^{*b}}(x_i)\right)}{\sum_{b=1}^{B} I_i^b}$

Fin

LOO bootstrap 0.632 : $\mathscr{R}^{*0.632}(f_{D_1^n}) = 0.632\mathscr{R}^*(f_{D_1^n}) + 0.368\widehat{\mathscr{R}_n}(f_{D_1^n})$.

$\hookrightarrow$ Pour tout $j \neq i$, $P\left((X_j, Y_j) \in D^*_{(i)}\right) = 1 - \left(1 - \frac{1}{n}\right)^n \to_{n\to\infty} 1 - e^{-1} \simeq 0.632$.

⚠ Si $\widehat{\mathscr{R}_n}(f_{D_1^n}) \simeq 0$, correction trop forte $\Rightarrow$ LOO bootstrap 0.632+ (Efron et Tibshirani 1997) et variantes.

En conclusion...

Pour chaque approche : de nouveaux paramètres à choisir...

✘ Pas de choix et/ou d'ajustement des paramètres complètement automatique (c.f. "No free lunch theorems").

✘ ✔ Implication nécessaire de l'utilisateur

Plan du cours

3. Algorithmes de moyennage local
   - Définition
   - Consistance universelle
   - Algorithme des k p.p.v.
   - Algorithme par noyau d'approximation
   - Algorithme par partition
   - Le "fléau" de la dimension

Définition

Soit $\{W_{n,i}, 1 \leq i \leq n\}$ une famille de poids positifs tels que pour tout $n \geq 1$, $x, x_1, \ldots, x_n \in \mathscr{X}$, $\sum_{i=1}^{n} W_{n,i}(x, x_1, \ldots, x_n) = 1$.

Un **algorithme de moyennage local** est un algorithme défini, pour $d_1^n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, par :

- $\eta_{d_1^n} : x \in \mathscr{X} \mapsto \sum_{i=1}^{n} W_{n,i}(x, x_1, \ldots, x_n) y_i$ en régression réelle,
- $\phi_{\eta_{d_1^n}} : x \in \mathscr{X} \mapsto signe\left(\sum_{i=1}^{n} W_{n,i}(x, x_1, \ldots, x_n) y_i\right)$ en discrimination binaire (règle de discrimination plug in).

Consistance universelle

On suppose que $\mathscr{X} = \mathbb{R}^d$, $\mathscr{Y} \subset \mathbb{R}$ et $\mathbb{E}[Y^2] < +\infty$.

Théorème (Stone 1977)

On suppose que quelle que soit la loi marginale P_X de X,

(i) $\exists c > 0$, $\forall f : \mathscr{X} \to \mathbb{R}_+$ *telle que* $\mathbb{E}_{P_X}[f(X)] < \infty$, $\forall n$,

$$\mathbb{E}_{P_X^{\otimes(n+1)}}\left[\sum_{i=1}^n W_{n,i}(X, X_1, \ldots, X_n) f(X_i)\right] \leq c\mathbb{E}_{P_X}[f(X)],$$

(ii) $\forall a > 0$, $\mathbb{E}_{P_X^{\otimes(n+1)}}\left[\sum_{i=1}^n W_{n,i}(X, X_1, \ldots, X_n) \mathbb{1}_{\|X_i - X\| > a}\right] \to 0$,

(iii) $\mathbb{E}_{P_X^{\otimes(n+1)}}\left[\sum_{i=1}^n W_{n,i}^2(X, X_1, \ldots, X_n)\right] \to 0.$

- *Si* $\mathscr{Y} \subset \mathbb{R}$, $l(y, y') = (y - y')^2$,
  $\sup_P \lim_{n \to +\infty} \left\{\mathbb{E}_{D_1^n \sim P^{\otimes n}} \mathbb{E}_{(X,Y)\sim P}[l(Y, \eta_{D_1^n}(X))] - \inf_{f \in \mathscr{F}} R_P(f)\right\} = 0.$
- *Si* $\mathscr{Y} = \{-1, 1\}$, $l(y, y') = \mathbb{1}_{y \neq y'}$,
  $\sup_P \lim_{n \to +\infty} \left\{\mathbb{E}_{D_1^n \sim P^{\otimes n}} \mathbb{E}_{(X,Y)\sim P}[l(Y, \phi_{\eta_{D_1^n}}(X))] - \inf_{f \in \mathscr{F}} R_P(f)\right\} = 0.$

Algorithme des k p.p.v.

On appelle **algorithme des k plus proches voisins** un algorithme par moyennage local dont les poids vérifient :

$$W_{n,i}(x, x_1, \ldots, x_n) = \begin{cases} 1/k & \text{si } x_i \text{ fait partie des } k \text{ p.p.v. de } x \\ & \text{dans } \{x_1, \ldots, x_n\} \\ 0 & \text{sinon.} \end{cases}$$

$\hookrightarrow$ En cas d'égalité, on peut procéder à un tirage aléatoire.

Théorème

Si $\mathcal{X} = \mathbb{R}^d$, $(k_n)_{n \geq 1}$ est une suite d'entiers tels que $k_n \to +\infty$ et $k_n/n \to 0$, alors l'algorithme des k_n p.p.v. pour une distance associée à une norme quelconque de $\mathbb{R}^d$ est universellement consistant.

$\hookrightarrow$ k_n aléatoire : il suffit que les hypothèses soient vérifiées en proba.

Théorème (Cover and Hart 1967)

L'algorithme du plus proche voisin ($k = 1$) n'est pas universellement consistant.

Algorithme par noyau d'approximation

On appelle **algorithme par noyau d'approximation (ou de lissage)** un algorithme de moyennage local dont les poids sont de la forme :
$W_{n,i}(x, x_1, \ldots, x_n) = K\left(\frac{x_i - x}{h}\right) / \sum_{j=1}^{n} K\left(\frac{x_j - x}{h}\right)$,
où K est une fonction (un noyau d'approximation) à valeurs dans $\mathbb{R}_+$, h un réel > 0 (largeur du noyau) avec la convention $0/0 = 0$.

Noyaux usuels ($\mathscr{X} = \mathbb{R}^d$ muni de la norme euclidienne $\|.\|$) :

- noyau fenêtre $K(x) = \mathbb{1}_{\|x\| \leq 1}$,
- noyau gaussien ou radial $K(x) = e^{-\|x\|^2}$.

Soit $\mathscr{B}(O, r)$ la boule euclidienne de $\mathscr{X} = \mathbb{R}^d$ de centre O de rayon r.

Théorème (Devroye and Wagner / Spiegelman and Sacks 1980)

S'il existe $0 < r \leq R$ et b tels que $b\mathbb{1}_{\mathscr{B}(O,r)} \leq K \leq \mathbb{1}_{\mathscr{B}(O,R)}$, si $(h_n)_{n \geq 1}$ vérifie $h_n \to 0$ et $nh_n^d \to +\infty$ alors l'algorithme par noyau d'approximation de largeur h_n est universellement consistant.

Algorithme par partition

Etant donnée une partition $V_1, V_2, \ldots$ finie ou dénombrable de $\mathscr{X}$, pour $x \in \mathscr{X}$, on note $V(x)$ l'élément de la partition contenant x. On appelle **algorithme par partition** un algorithme de moyennage local dont les poids sont de la forme : $W_{n,i}(x, x_1, \ldots, x_n) = \mathbb{1}_{x_i \in V(x)} / \sum_{j=1}^{n} \mathbb{1}_{x_j \in V(x)}$ $(0/0 = 0)$.

Théorème

Soit $(V_{1,n}, V_{2,n}, \ldots)_{n \geq 1}$ une suite de partitions dénombrables de $\mathscr{X} = \mathbb{R}^d$. En notant $diam(V_{k,n}) = \sup_{x,x' \in \mathscr{X}} \|x - x'\|$, si pour tout $r > 0$,

- $|\{k, V_{k,n} \cap \mathscr{B}(O,r)\}| \neq \emptyset\}|/n \to 0$,
- $\sup_{k, V_{k,n} \cap \mathscr{B}(O,r)} diam(V_{k,n}) \to 0$,

alors l'algorithme de partition défini sur $V_{1,n}, V_{2,n}, \ldots$ est u. consistant.

✘ Attention : effets de bords.

↪ partition régulière de pas h_n lorsque $h_n \to 0$ et $nh_n^d \to +\infty$

↪ cas d'une partition aléatoire (arbres de décision)

Le "fléau" de la dimension (curse of dimensionality, Bellman 1961)

Illustration 1 : P_X = loi uniforme sur l'hypercube unité de $\mathbb{R}^d$.
Sélectionner une proportion p de données d'observation = sélectionner un hypercube de côté moyen $p^{1/d}$.
$\hookrightarrow$ $d = 10$, $p = 1\% \Rightarrow p^{1/d} = 0.63$; $p = 10\% \Rightarrow p^{1/d} = 0.80$ (proche de 1 !).

Illustration 2 : P_X = loi uniforme sur la boule unité de $\mathbb{R}^d$.
Distance médiane de O à l'entrée X_i la plus proche :

$$D(n,d) = \left(1 - \frac{1}{2^{1/n}}\right)^{1/d}.$$

$\hookrightarrow$ $D(500,10) \simeq 0.52$ (proche de la frontière !)

Le terme "local" a-t-il encore un sens en grande dimension ?

Plan du cours

4. Algorithmes de minimisation du risque empirique
   - Rappels
   - Définition, exemples
   - Minimisation structurelle du risque
   - Choix de modèle : pénalisation ou régularisation
   - Convexification

Rappels

- Le risque d'une règle de prédiction f est défini par $R_P(f) = \mathbb{E}_{(X,Y)\sim P}[l(Y, f(X))]$.
- Les règles "optimales" f^* au sens de la minimisation du risque dépendent de P inconnue.

$\hookrightarrow$ Approche non paramétrique (Théorème de Glivenko-Cantelli) : P approchée par P_n, mesure empirique associée à D_1^n.

Le **risque empirique** ou **apparent** (associé à $D_1^n = \{(X_i, Y_i), 1 \leq i \leq n\}$) d'une règle de prédiction $f \in \mathscr{F}$ est défini par

$$\widehat{\mathscr{R}_n}(f) = R_{P_n}(f) = \mathbb{E}_{(X,Y)\sim P_n}[l(Y, f(X))] = \frac{1}{n}\sum_{i=1}^{n} l(Y_i, f(X_i)).$$

Définition, exemples

Minimisation du risque empirique : Vapnik et Chervonenkis (1971).

Etant donné un sous-ensemble F de $\mathscr{F}$ (un **modèle**), l'**algorithme de minimisation du risque empirique sur** F est défini par :

$$f_{D_1^n,F} \in \operatorname{argmin}_{f \in F} \widehat{\mathscr{R}_n}(f).$$

Exemples avec $F = \mathscr{F}$:

- L'algorithme qui attribue la sortie Y_i à une entrée $x = X_i$, et une sortie quelconque à une entrée x différente des X_i.
- L'algorithme du p.p.v.

⚠ F trop "riche" ou "complexe" = risque de sur-apprentissage

Exemples avec $F \neq \mathscr{F}$:

- Méthodes statistiques classiques : régression linéaire multiple, discrimination linéaire de Fisher, régression logistique (entropie croisée)...
- Réseaux de neurones.

Un **réseau de neurones à une couche cachée** est un algorithme de minimisation du risque empirique sur

$$\left\{ x \in \mathbb{R}^d \mapsto \sum_{j=1}^{k} c_j \sigma(a_j (1, x_1, \ldots, x_d)^T) + c_0, \right.$$
$$\left. k \leq k_n, a_j \in \mathbb{R}^{d+1}, c_j \in \mathbb{R}, \sum_{j=0}^{d} |c_j| \leq \beta_n \right\},$$

où $\sigma : \mathbb{R} \to \mathbb{R}$ est une fonction **sigmoïde** c'est-à-dire croissante et telle que $\lim_{t \to -\infty} \sigma(t) = 0$ et $\lim_{t \to +\infty} \sigma(t) = 1$.

Minimisation structurelle du risque

Minimisation structurelle du risque en discrimination binaire : Vapnik et Chervonenkis (1974).

$\hookrightarrow$ Une nouvelle "mesure" de la complexité du modèle F.

- La **dimension de Vapnik-Chervonenkis** d'une classe $\mathscr{C}$ de sous-ensembles de $\mathscr{X}$ est définie par :

$$V(\mathscr{C}) = +\infty \text{ si } \forall k \geq 1, \max_{x_1,...,x_k \in \mathscr{X}} |\{\{x_1,...,x_k\} \cap C, C \in \mathscr{C}\}| = 2^k,$$

$$= \sup \left\{ k, \max_{x_1,...,x_k \in \mathscr{X}} |\{\{x_1,...,x_k\} \cap C, C \in \mathscr{C}\}| = 2^k \right\} \text{ sinon.}$$

- $\mathscr{C}$ est une **classe de Vapnik-Chervonenkis** si $V(\mathscr{C}) < +\infty$.

$\hookrightarrow$ Exemples.

Théorème

Soit $\mathscr{C}_1, \mathscr{C}_2, \ldots$ une suite de classes de Vapnik-Chervonenkis de $\mathscr{X}$, telle que si $F_i = \{2\mathbb{1}_C - 1,\ C \in \mathscr{C}_i\}$, pour toute loi P,

$$\lim_{i \to +\infty} \inf_{f \in F_i} R_P(f) = R_P(f^*).$$

Si $k_n \to +\infty$ et $V(\mathscr{C}_{k_n})/n \to 0$, alors $f_{D_1^n, F_{k_n}}$ est universellement consistant.

✘ Deux questions :

- Choix de F (adapté aux données) ?
- Détermination effective du minimiseur du risque empirique en discrimination binaire ?

Choix de modèle : pénalisation ou régularisation

Soit une collection de modèles $\{F_\lambda, \lambda \in \Lambda\}$, les algorithmes de MRE associés $\left\{f_{D_1^n,\lambda}, \lambda \in \Lambda\right\}$, et une fonction $pen : \Lambda \to \mathbb{R}_+$, qui à λ associe une quantité rendant compte de la complexité de F_λ.

L'**algorithme de minimisation du risque empirique pénalisé** est défini par $f_{D_1^n,\widehat{\lambda}}$, où $\widehat{\lambda} \in \operatorname{argmin}_{\lambda\in\Lambda} \left\{\widehat{\mathscr{R}_n}(f_{D_1^n,\lambda}) + pen(\lambda)\right\}$.

- Régression ridge,
- LASSO,
- Pénalisation K-fold, bootstrap...

✔ Résultats théoriques non asymptotiques (inégalités oracles).

Convexification

✘ En discrimination binaire, le problème de minimisation du risque empirique n'est pas en général un problème d'optimisation convexe.

✔ Une solution récente : la **convexification du risque empirique**.

Chercher dans F une fonction $f_{D_1^n,F}$ qui minimise $\widehat{\mathscr{R}_n}$ $\leftrightarrow$ chercher dans un ensemble convexe G de fonctions mesurables : $\mathscr{X} \to \mathbb{R}$ une fonction $g_{D_1^n,G}$ qui minimise $\sum_{i=1}^n \Phi_{0-1}(Y_i g(X_i))$, avec $\Phi_{0-1}(t) = \mathbb{1}_{t\leq 0}$, et poser $f_{D_1^n,F} = signe(g_{D_1^n,G})$ (plug in).

Convexifier le risque empirique consiste à remplacer Φ_{0-1} par une fonction convexe Φ telle que Φ majore Φ_{0-1} et $\Phi(t) \to_{t\to+\infty} 0$.

Le Φ**-risque** d'une fonction $g : \mathcal{X} \to \mathbb{R}$ est défini par

$$R_{\Phi,P}(g) = \mathbb{E}_{(X,Y)\sim P}[\Phi(Yg(X))].$$

On utilise par ex. des fonctions Φ convexes telles que $R_P(signe(g)) - \inf_f R_P(f) \leq \kappa \left(R_{\Phi,P}(g) - \inf_g R_{\Phi,P}(g)\right)^{\alpha}$, pour $\kappa,\ \alpha > 0$ ("conservation" de la consistance universelle).

Exemples :

- perte charnière (hinge loss) : $\Phi(t) = (1-t)_+$ (S.V.M.),
- perte logit : $\Phi(t) = \log_2(1+e^{-t})$ (boosting),
- perte exponentielle : $\Phi(t) = e^{-t}$ (boosting)...

Références bibliographiques

Ouvrages

- G. Biau, L. Devroye, Lectures on the Nearest Neighbor Method, Springer, 2015.
- L. Devroye, L. Györfi, and G. Lugosi, A Probabilistic Theory of Pattern Recognition. Springer, New York, 1996.
- T. Hastie, R. Tibshirani, and J. Friedman, The elements of statistical learning. Springer, New York, 2001.
- L. Györfi, M. Kohler, A. Krzyzak, and H. Walk, A distribution-free theory of non-parametric regression. Springer, New York, 2002.
- V. Vapnik, Statistical Learning Theory. John Wiley, 1998.

Polycopiés, supports de cours

- Polycopiés et notes de cours en accès sur la page de Sylvain Arlot.

Plan du cours

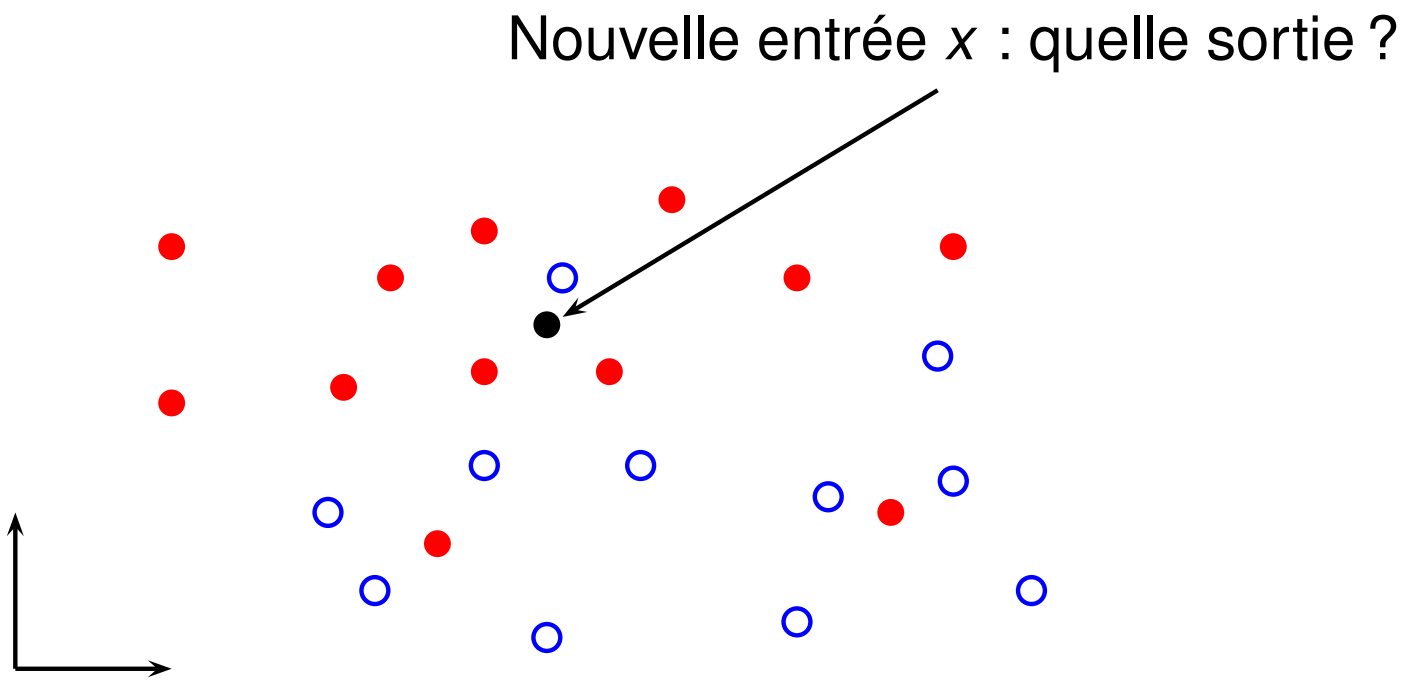
5 Méthodes à noyaux : SVM, SVR

- Les SVM : quoi ? pourquoi ?
- SVM linéaire pour des données séparables
- SVM linéaire pour des données non séparables
- SVM non linéaire : astuce du noyau
- Éléments de théorie
- Conclusion / questions ouvertes
- SVM pour la régression (SVR) : un aperçu

Les SVM : quoi ? pourquoi ?

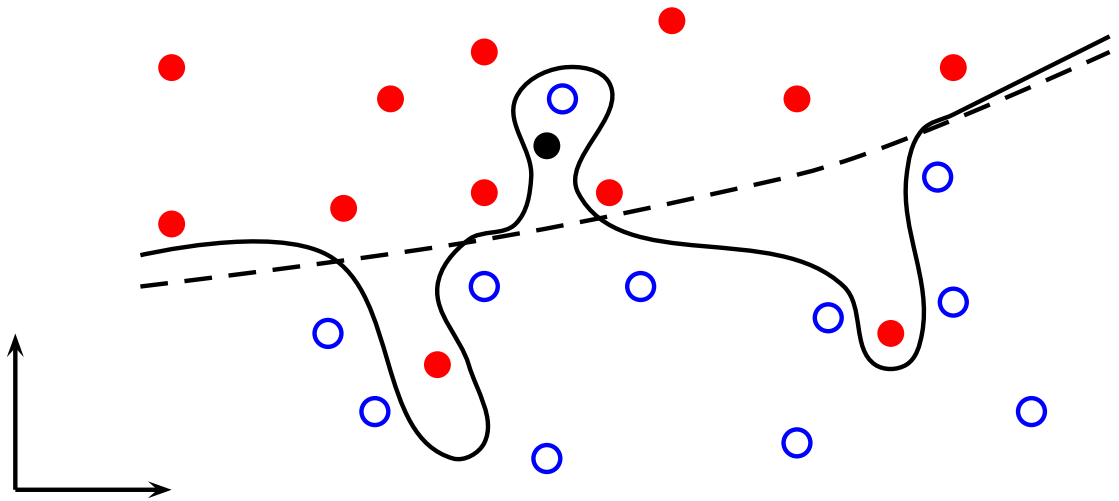
Une **SVM (Support Vector Machine) ou Machine à Vecteurs Supports** est une famille d'algorithmes d'apprentissage supervisé pour des problèmes de discrimination (ou de régression).

Exemple type : discrimination binaire en dimension 2



✔ Fondements mathématiques solides ⇒ bonnes **propriétés de généralisation** i.e. bon compromis la discrimination des données et la prédiction de la sortie associée à une nouvelle entrée x.

Exemple d'une règle de discrimination n'ayant pas de bonnes propriétés de généralisation



⚠ Phénomène de **sur-apprentissage (ou overfitting)** particulièrement présent en grande dimension ou pour des règles de discrimination non linéaires complexes.

SVM linéaire pour des données séparables

Données linéairement séparables

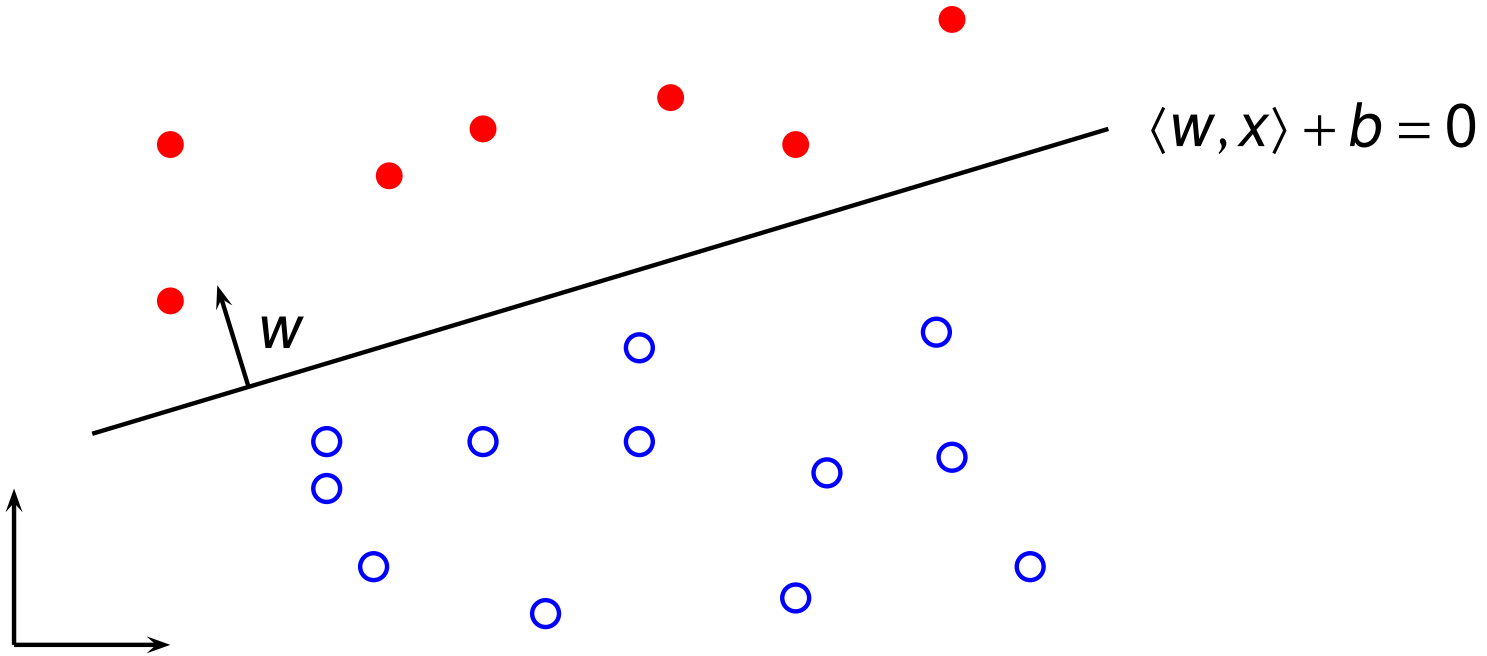
On considère $\mathscr{X} = \mathbb{R}^d$, muni du produit scalaire usuel $\langle .,. \rangle$.

Les données observées $d_1^n = (x_1, y_1), \ldots, (x_n, y_n)$ sont dites **linéairement séparables** s'il existe (w, b) tel que pour tout i,
- $y_i = 1$ si $\langle w, x_i \rangle + b > 0$,
- $y_i = -1$ si $\langle w, x_i \rangle + b < 0$,

c'est-à-dire

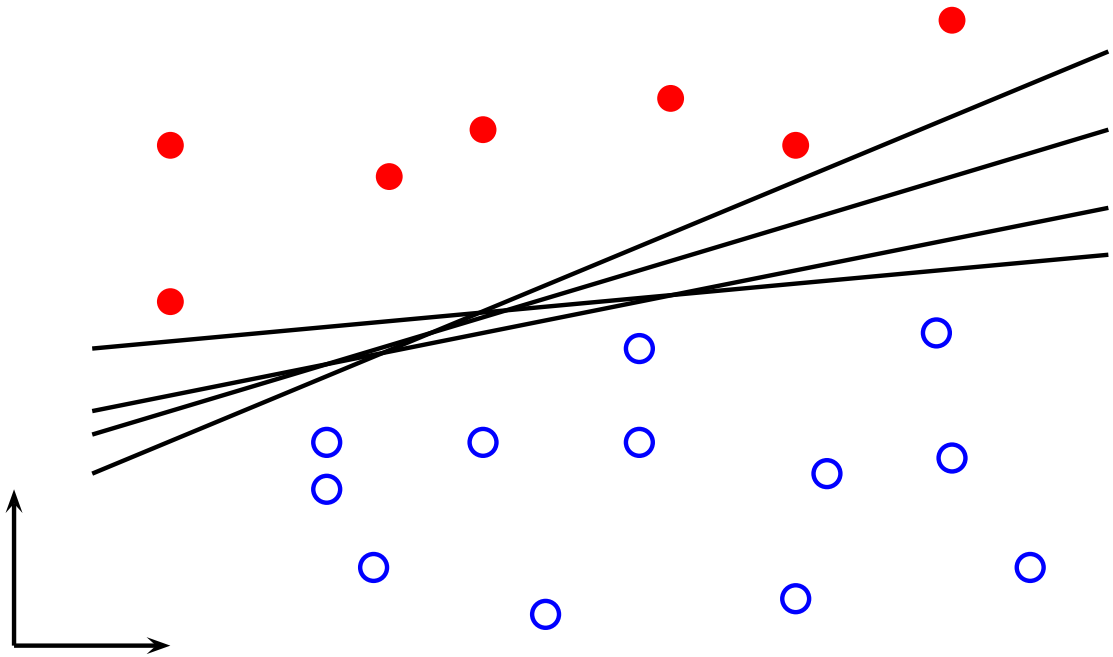
$$\forall i = 1, \ldots, n \quad y_i \left(\langle w, x_i \rangle + b \right) > 0$$

L'équation $\langle w, x \rangle + b = 0$ définit un hyperplan séparateur de vecteur orthogonal w.



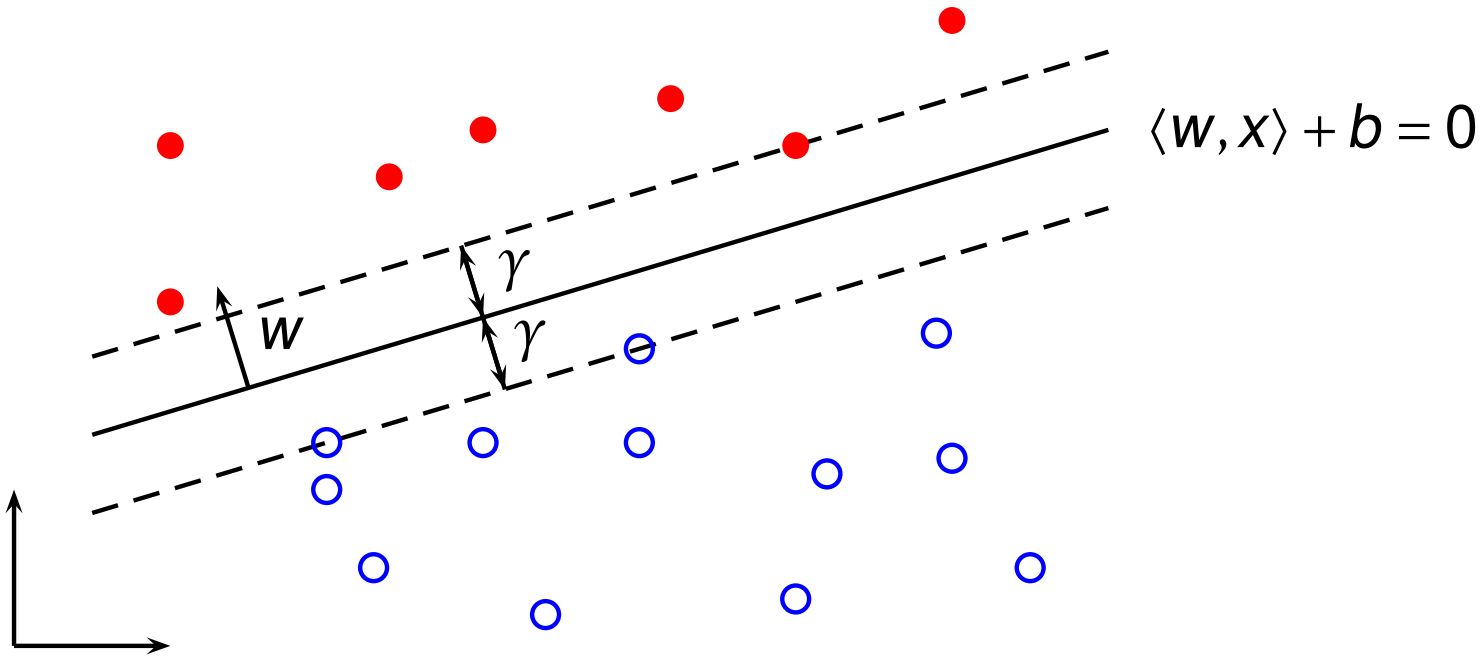
La fonction $\phi_{w,b}(x) = \mathbb{1}_{\langle w,x \rangle + b \geq 0} - \mathbb{1}_{\langle w,x \rangle + b < 0}$ est une règle de discrimination linéaire potentielle.

✘ **Problème** : une infinité d'hyperplans séparateurs ⇒ une infinité de règles de discrimination linéaires potentielles !



✘ Laquelle choisir ?

✔ **La réponse (Vapnik)** : La règle de discrimination linéaire ayant les meilleures propriétés de généralisation correspond à l'hyperplan séparateur de **marge maximale** γ.



La marge maximale

Soit deux entrées de l'ensemble d'apprentissage (re)notées x_1 et x_{-1} de sorties respectives 1 et -1, se situant sur les frontières définissant la marge. L'hyperplan séparateur correspondant se situe à mi-distance entre x_1 et x_{-1}.
La marge s'exprime donc :

$$\gamma = \frac{1}{2} \frac{\langle w, x_1 - x_{-1} \rangle}{\|w\|}.$$

Remarque : pour tout $\kappa \neq 0$, les couples $(\kappa w, \kappa b)$ et (w, b) définissent le même hyperplan.

L'hyperplan $\langle w, x \rangle + b = 0$ est dit en forme canonique relativement à un ensemble de vecteurs $x_1, \ldots, x_k$ si $\min_{i=1\ldots k} |\langle w, x_i \rangle + b| = 1$.

L'hyperplan séparateur est en forme canonique relativement aux vecteurs $\{x_1, x_{-1}\}$ s'il est défini par (w, b) avec $\langle w, x_1 \rangle + b = 1$ et $\langle w, x_{-1} \rangle + b = -1$. On a alors $\langle w, x_1 - x_{-1} \rangle = 2$, d'où

$$\gamma = \frac{1}{\|w\|}.$$

Problème d'optimisation "primal"

Trouver l'hyperplan séparateur de marge maximale revient à trouver le couple (w, b) tel que

$$\|w\|^2 \text{ ou } \frac{1}{2}\|w\|^2 \text{ soit minimal}$$
$$\text{sous la contrainte}$$
$$y_i(\langle w, x_i \rangle + b) \geq 1 \text{ pour tout } i.$$

✔ Problème d'optimisation convexe sous contraintes linéaires

✔ Existence d'un optimum global, obtenu par résolution du problème "dual" (méthode des multiplicateurs de Lagrange).

Multiplicateurs de Lagrange : problème primal

Minimiser pour $u \in \mathbb{R}^2$ $h(u)$ sous les contraintes $g_i(u) \geq 0$ pour $i = 1 \dots n$, h fonction quadratique, g_i fonctions affines.

Le **Lagrangien** est défini sur $\mathbb{R}^2 \times \mathbb{R}^n$ par $L(u, \alpha) = h(u) - \sum_{i=1}^{n} \alpha_i g_i(u)$.

Les variables α_i sont appelées les **variables duales**.

Soit pour tout $\alpha \in \mathbb{R}^n$,

- $u_\alpha = \text{argmin}_{u \in \mathbb{R}^2} L(u, \alpha)$,
- $\theta(\alpha) = L(u_\alpha, \alpha) = \min_{u \in \mathbb{R}^2} L(u, \alpha)$ (**fonction duale**).

Multiplicateurs de Lagrange : problème dual

Maximiser $\theta(\alpha)$ sous les contraintes $\alpha_i \geq 0$ pour $i = 1 \dots n$.

La solution du problème dual α^* donne la solution du problème primal : $u^* = u_{\alpha^*}$.

Multiplicateurs de Lagrange : conditions de Karush-Kuhn-Tucker

- $\alpha_i^* \geq 0$ pour tout $i = 1 \dots n$.
- $g_i(u_{\alpha^*}) \geq 0$ pour tout $i = 1 \dots n$.
- Retour sur le problème dual :

  On doit minimiser $L(u, \alpha) = h(u) - \sum_{i=1}^{n} \alpha_i g_i(u)$ par rapport à u et maximiser $L(u_\alpha, \alpha)$ correspondant par rapport aux variables duales α_i.
  $\Rightarrow$ Si $g_i(u_{\alpha^*}) > 0$, alors nécessairement $\alpha_i^* = 0$.

  Condition complémentaire de Karush-Kuhn-Tucker qui s'exprime sous la forme $\alpha_i^* g_i(u_{\alpha^*}) = 0$.

Multiplicateurs de Lagrange pour la SVM en discrimination binaire

Lagrangien : $L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^n \alpha_i \left(y_i\left(\langle w, x_i\rangle + b\right) - 1\right)$

Fonction duale :

$$\begin{cases} \frac{\partial L}{\partial w}(w,b,\alpha) &= w - \sum_{i=1}^n \alpha_i y_i x_i = 0 \Leftrightarrow w = \sum_{i=1}^n \alpha_i y_i x_i \\ \frac{\partial L}{\partial b}(w,b,\alpha) &= -\sum_{i=1}^n \alpha_i y_i = 0 \Leftrightarrow \sum_{i=1}^n \alpha_i y_i = 0 \end{cases}$$

$$\theta(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle.$$

La solution du problème d'optimisation primal est donnée par :

- $w^* = \sum_{i=1}^n \alpha_i^* y_i x_i$,
- $b^* = -\frac{1}{2}\left\{\min_{y_i=1}\langle w^*, x_i\rangle + \min_{y_i=-1}\langle w^*, x_i\rangle\right\}$,

où $\alpha^* = (\alpha_1^*, \ldots, \alpha_n^*)$ est la solution du problème d'optimisation dual :

$$\text{Maximiser } \theta(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle$$
$$\text{s. c. } \sum_{i=1}^n \alpha_i y_i = 0 \text{ et } \alpha_i \geq 0 \ \forall i.$$

✔ La solution α^* du problème dual est indépendante de la dimension d : la SVM linéaire ne souffre pas du "fléau de la dimension".

Conditions de Karush-Kuhn-Tucker :

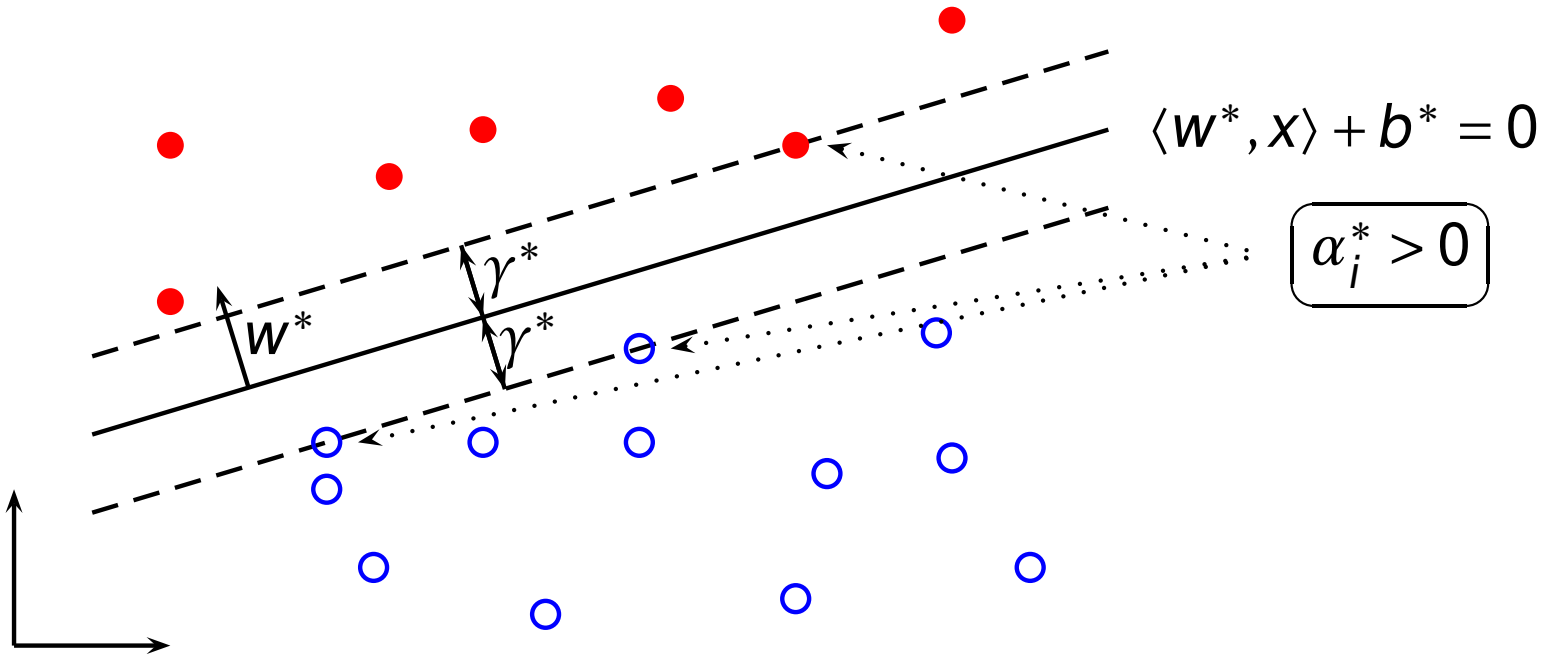
- $\alpha_i^* \geq 0 \ \forall i = 1 \dots n.$
- $y_i \left(\langle w^*, x_i \rangle + b^* \right) \geq 1 \ \forall i = 1 \dots n.$
- $\alpha_i^* \left(y_i \left(\langle w^*, x_i \rangle + b^* \right) - 1 \right) = 0 \ \forall \ i = 1 \dots n.$
  (condition complémentaire)

✔ Le nombre de $\alpha_i^* > 0$ peut être petit : on dit que la solution du problème dual est **parcimonieuse** (**sparse**).

✔ Efficacité algorithmique.

Les x_i tels que $\alpha_i^* > 0$ sont appelés les **vecteurs supports**. Ils sont situés sur les frontières définissant la marge maximale i.e. $y_i \left(\langle w^*, x_i \rangle + b^* \right) = 1$ (c.f. condition complémentaire de KKT).

Représentation des vecteurs supports



Pour conclure, l'algorithme est défini par :

$$\phi_{d_1^n}(x) = \mathbb{1}_{\langle w^*, x\rangle + b^* \geq 0} - \mathbb{1}_{\langle w^*, x\rangle + b^* < 0},$$

avec

- $w^* = \sum_{i=1}^n \alpha_i^* y_i x_i$,
- $b^* = -\frac{1}{2}\left\{\min_{y_i=1} \langle w^*, x_i\rangle + \min_{y_i=-1} \langle w^*, x_i\rangle\right\}$,

ou encore :

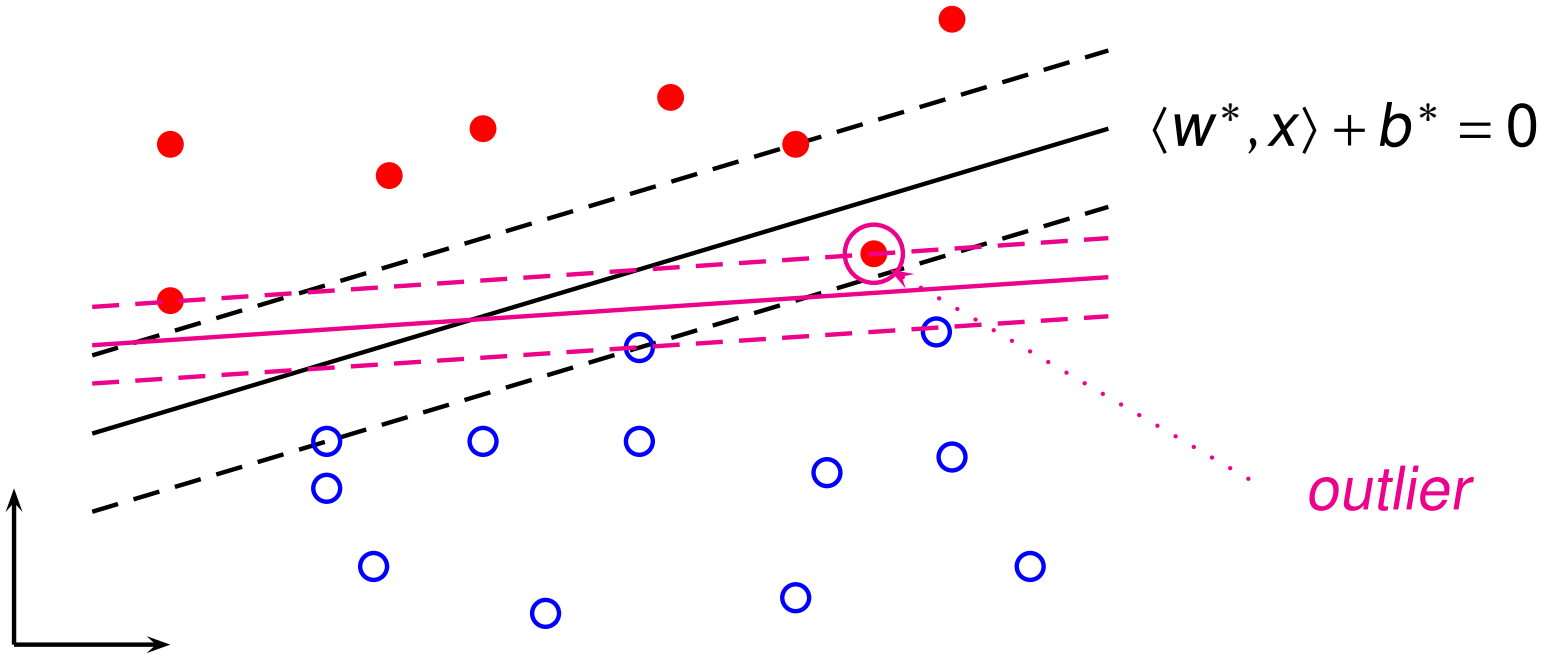
$$\phi_{d_1^n}(x) = \mathbb{1}_{\sum_{x_i V.S.} \alpha_i^* y_i \langle x_i, x\rangle + b^* \geq 0} - \mathbb{1}_{\sum_{x_i V.S.} \alpha_i^* y_i \langle x_i, x\rangle + b^* < 0}.$$

La marge maximale vaut $\gamma^* = \frac{1}{\|w^*\|} = \left(\sum_{i=1}^n (\alpha_i^*)^2\right)^{-1/2}$.

SVM linéaire pour des données non séparables

✘ Méthode précédente non applicable si les données ne sont pas linéairement séparables

✘ Méthode très sensible aux "outliers"



✔ La solution : autoriser quelques vecteurs à être bien classés mais dans la région définie par la marge, voire mal classés.

La contrainte $y_i(\langle w, x_i \rangle + b) \geq 1$ devient $y_i(\langle w, x_i \rangle + b) \geq 1 - \xi_i$, avec $\xi_i \geq 0$.

$\xi_i \in [0, 1] \leftrightarrow$ bien classé, mais région définie par la marge.
$\xi_i > 1 \leftrightarrow$ mal classé.

On parle de **marge souple** ou marge relaxée.

Les variables ξ_i sont appelées les **variables ressort** (**slacks**).

✘ Un nouveau problème : les contraintes relaxées ne peuvent pas être utilisées sans contrepartie sous peine d'obtenir une marge maximale infinie (en prenant des valeurs de ξ_i suffisamment grandes).

✔ La nouvelle solution : pénaliser les grandes valeurs de ξ_i.

Problème d'optimisation primal

$$\text{Minimiser en } (w,b,\xi) \qquad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^n \xi_i$$
$$\text{s. c. } \begin{cases} y_i\left(\langle w, x_i\rangle + b\right) \geq 1 - \xi_i \ \forall\ i \\ \xi_i \geq 0 \end{cases}$$

$\hookrightarrow$ $C > 0$ paramètre (**constante de tolérance**) à ajuster.

La solution du problème d'optimisation primal est donnée par :

- $w^* = \sum_{i=1}^n \alpha_i^* y_i x_i$,
- b^* tel que $y_i\left(\langle w^*, x_i\rangle + b^*\right) = 1 \ \forall x_i,\ 0 < \alpha_i^* < C$,

où $\alpha^* = (\alpha_1^*, \ldots, \alpha_n^*)$ est la solution du **problème d'optimisation dual** :

$$\text{Maximiser } \theta(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i,j=1}^n \alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle$$
$$\text{s. c. } \sum_{i=1}^n \alpha_i y_i = 0 \text{ et } 0 \leq \alpha_i \leq C \ \forall i.$$

Conditions de Karush-Kuhn-Tucker :

- $0 \leq \alpha_i^* \leq C \ \forall i = 1 \dots n.$
- $y_i \left(\langle w^*, x_i \rangle + b^* \right) \geq 1 - \xi_i^* \ \forall i = 1 \dots n.$
- $\alpha_i^* \left(y_i \left(\langle w^*, x_i \rangle + b^* \right) + \xi_i^* - 1 \right) = 0 \ \forall \ i = 1 \dots n.$
- $\xi_i^* (\alpha_i^* - C) = 0.$

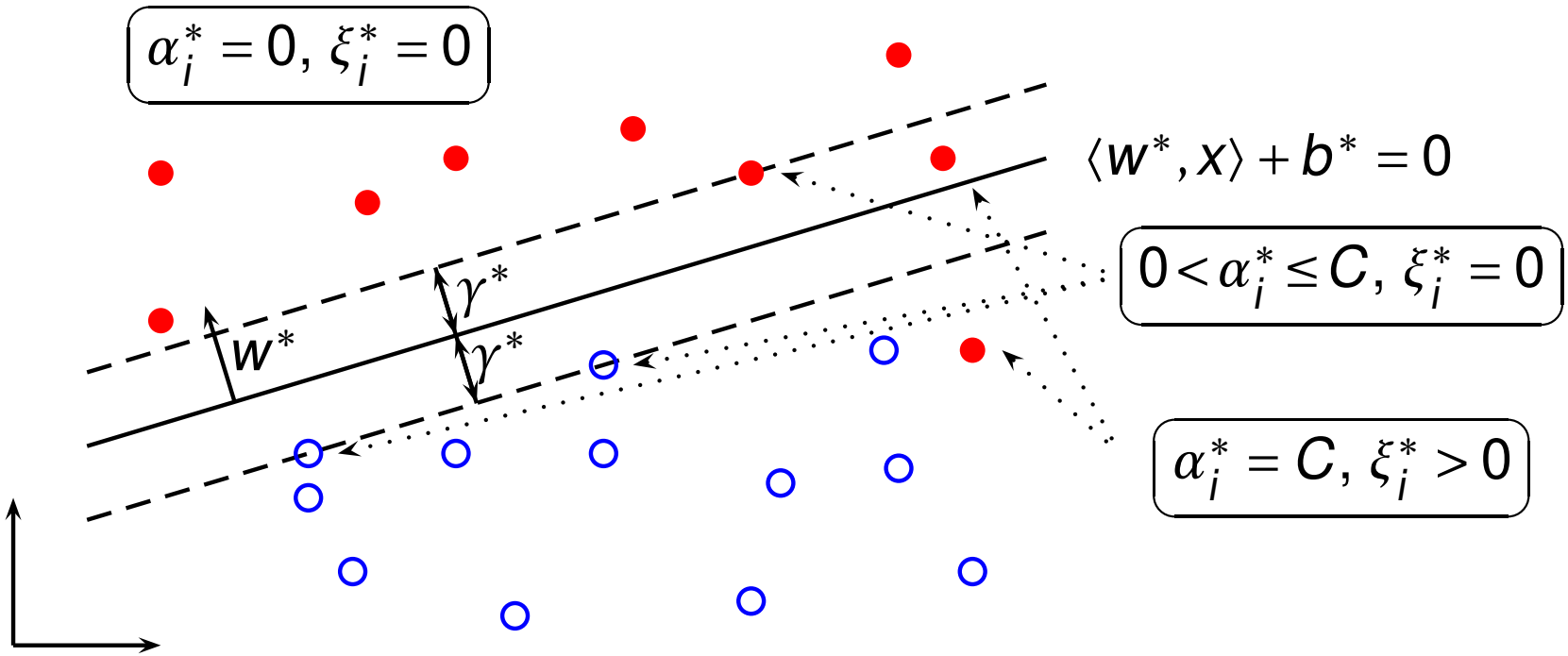
Les x_i tels que $\alpha_i^* > 0$ sont les **vecteurs supports**.

Deux types de vecteurs supports :

- Les vecteurs correspondant à des variables ressort nulles. Ils sont situés sur les frontières de la région définissant la marge.
- Les vecteurs correspondant à des variables ressort non nulles : $\xi_i^* > 0$ et dans ce cas $\alpha_i^* = C$.

Les vecteurs qui ne sont pas supports vérifient $\alpha_i^* = 0$ et $\xi_i^* = 0$.

Représentation des vecteurs supports



Pour conclure, la règle de discrimination de la SVM linéaire est définie par :

$$\phi_{d_1^n}(x) = \mathbb{1}_{\langle w^*, x\rangle + b^* \geq 0} - \mathbb{1}_{\langle w^*, x\rangle + b^* < 0},$$

avec

- $w^* = \sum_{i=1}^n \alpha_i^* x_i y_i$,
- b^* tel que $y_i\left(\langle w^*, x_i\rangle + b^*\right) = 1 \ \forall x_i,\ 0 < \alpha_i^* < C$,

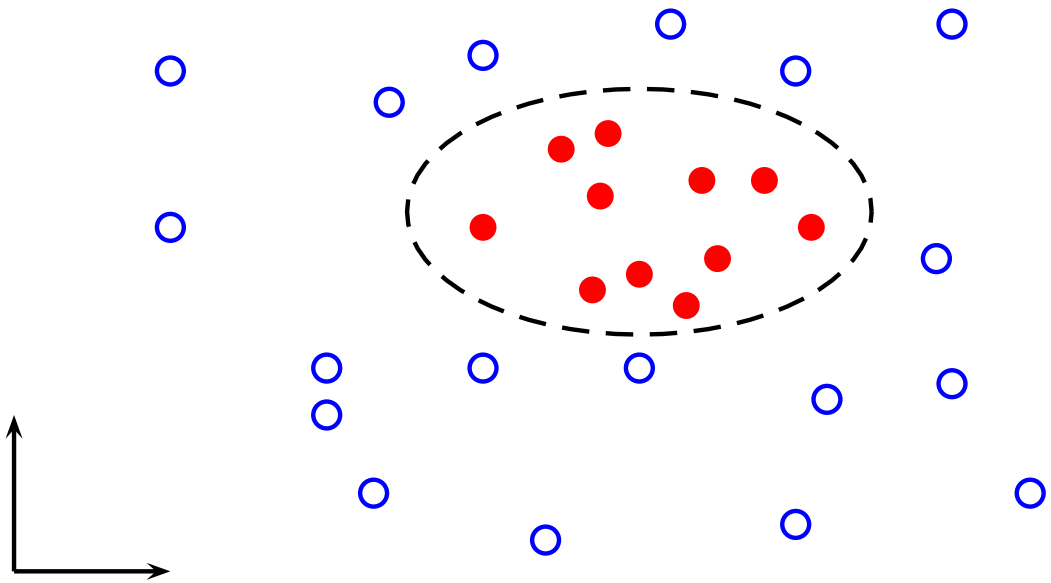
ou encore :

$$\phi_{d_1^n}(x) = \mathbb{1}_{\sum_{x_i V.S.} \alpha_i^* y_i \langle x_i, x\rangle + b^* \geq 0} - \mathbb{1}_{\sum_{x_i V.S.} \alpha_i^* y_i \langle x_i, x\rangle + b^* < 0}.$$

La marge maximale vaut $\gamma^* = \frac{1}{\|w^*\|} = \left(\sum_{i=1}^n (\alpha_i^*)^2\right)^{-1/2}$.

SVM non linéaire : astuce du noyau

Exemple de données difficiles à discriminer linéairement :



✘ Une SVM linéaire donnera une très mauvaise discrimination avec un nombre de vecteurs supports très élevé ⇒ SVM non linéaire ?

✔ L'idée (Boser, Guyon, Vapnik, 1992) :

Envoyer les entrées $\{x_i, i = 1 \dots n\}$ dans un espace de Hilbert $\mathscr{H}$ (muni du produit scalaire $\langle ., . \rangle_{\mathscr{H}}$), de grande dimension, voire de dimension infinie, via une fonction φ, et appliquer une SVM linéaire aux nouvelles données $\{(\varphi(x_i), y_i), i = 1 \dots n\}$.
La sortie attribuée à l'entrée x est celle attribuée à son image $\varphi(x)$.

La fonction φ est appelée la **fonction de représentation (feature function)**.
L'espace $\mathscr{H}$ est appelé l'**espace de représentation (feature space)**.

Dans l'exemple précédent, pour $\varphi(x) = (x_1^2, x_2^2, x_1, x_2)$, les données deviennent linéairement séparables dans $\mathbb{R}^4$.

✘ Problème : Comment choisir $\mathcal{H}$ et φ ?

La règle de discrimination de la SVM non linéaire est définie par :

$$\phi_{d_1^n}(x) = \mathbb{1}_{\sum y_i \alpha_i^* \langle \varphi(x_i), \varphi(x) \rangle_{\mathcal{H}} + b^* \geq 0} - \mathbb{1}_{\sum y_i \alpha_i^* \langle \varphi(x_i), \varphi(x) \rangle_{\mathcal{H}} + b^* < 0},$$

$\alpha^* = (\alpha_1^*, \ldots, \alpha_n^*)$ étant solution du problème dual dans $\mathcal{H}$:

$$\text{Maximiser } \theta(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \langle \varphi(x_i), \varphi(x_j) \rangle_{\mathcal{H}}$$
$$\text{s. c. } \sum_{i=1}^n \alpha_i y_i = 0 \text{ et } 0 \leq \alpha_i \leq C \ \forall i.$$

Remarque fondamentale :
La règle de discrimination de la SVM non linéaire ne dépend de φ qu'à travers des produits scalaires de la forme $\langle \varphi(x_i), \varphi(x) \rangle_{\mathcal{H}}$ ou $\langle \varphi(x_i), \varphi(x_j) \rangle_{\mathcal{H}}$.

✔ **Astuce du noyau (kernel trick)** : la connaissance seule de la fonction k définie par $k(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathcal{H}}$ permet de lancer la SVM dans $\mathcal{H}$, sans déterminer explicitement $\mathcal{H}$ et φ.

Une fonction $k : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$ telle que $k(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathscr{H}}$ pour une fonction $\varphi : \mathscr{X} \to \mathscr{H}$ donnée est appelée un **noyau**.

Un noyau est souvent plus facile à calculer que la fonction φ.

Par exemple, si pour $x = (x_1, x_2) \in \mathbb{R}^2$, $\varphi(x) = (x_1^2, \sqrt{2}x_1 x_2, x_2^2)$, alors $k(x, x') = \langle x, x' \rangle^2$.

Quelques noyaux classiques pour $\mathscr{X} = \mathbb{R}^d$

- Noyau **polynomial** : $k(x, x') = (\langle x, x' \rangle + c)^p$
  $\hookrightarrow \varphi(x) = (\varphi_1(x), \ldots, \varphi_k(x))$ avec $\varphi_i(x)$=monôme de degré inférieur à p de certaines composantes de x.
- Noyau **gaussien** ou **radial** (RBF) : $k(x, x') = e^{-\frac{\|x - x'\|^2}{2\sigma^2}}$
  $\hookrightarrow \varphi$ à valeurs dans un espace de dimension infinie.
- Noyau **laplacien** : $k(x, x') = e^{-\frac{\|x - x'\|}{\sigma}}$.

Agrégation de noyaux

Soit k_1 et k_2 des noyaux, f une fonction : $\mathbb{R}^d \to \mathbb{R}$, $\psi : \mathbb{R}^d \to \mathbb{R}^{d'}$, B une matrice définie positive, P un polynôme à coefficients positifs, $\lambda \geq 0$.

La fonction définie par $k(x,x') = k_1(x,x') + k_2(x,x')$, $\lambda k_1(x,x')$, $k_1(x,x')k_2(x,x')$, $f(x)f(x')$, $k_1(\psi(x),\psi(x'))$, $x^T Bx'$, $P(k_1(x,x'))$, ou $e^{k_1(x,x')}$ est encore un noyau.

Noyaux pour $\mathcal{X} \neq \mathbb{R}^d$

Quelques noyaux ont été proposés pour d'autres types d'objets comme des

- ensembles,
- arbres,
- graphes,
- chaînes de symboles,
- documents textuels...

Éléments de théorie

Minimiser $\frac{1}{2}\|w\|^2 + C\sum_{i=1}^n \xi_i$ s. c. $\begin{cases} y_i\left(\langle w, x_i\rangle + b\right) \geq 1 - \xi_i \ \forall\ i \\ \xi_i \geq 0 \end{cases}$

est équivalent à minimiser

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^n \left(1 - y_i\left(\langle w, x_i\rangle + b\right)\right)_+,$$

ou encore

$$\frac{1}{n}\sum_i^n \left(1 - y_i\left(\langle w, x_i\rangle + b\right)\right)_+ + \frac{1}{2Cn}\|w\|^2.$$

$\gamma(w, b, x_i, y_i) = \left(1 - y_i\left(\langle w, x_i\rangle + b\right)\right)_+$ est un majorant convexe de l'erreur empirique $\mathbb{1}_{y_i(\langle w, x_i\rangle + b) \leq 0}$

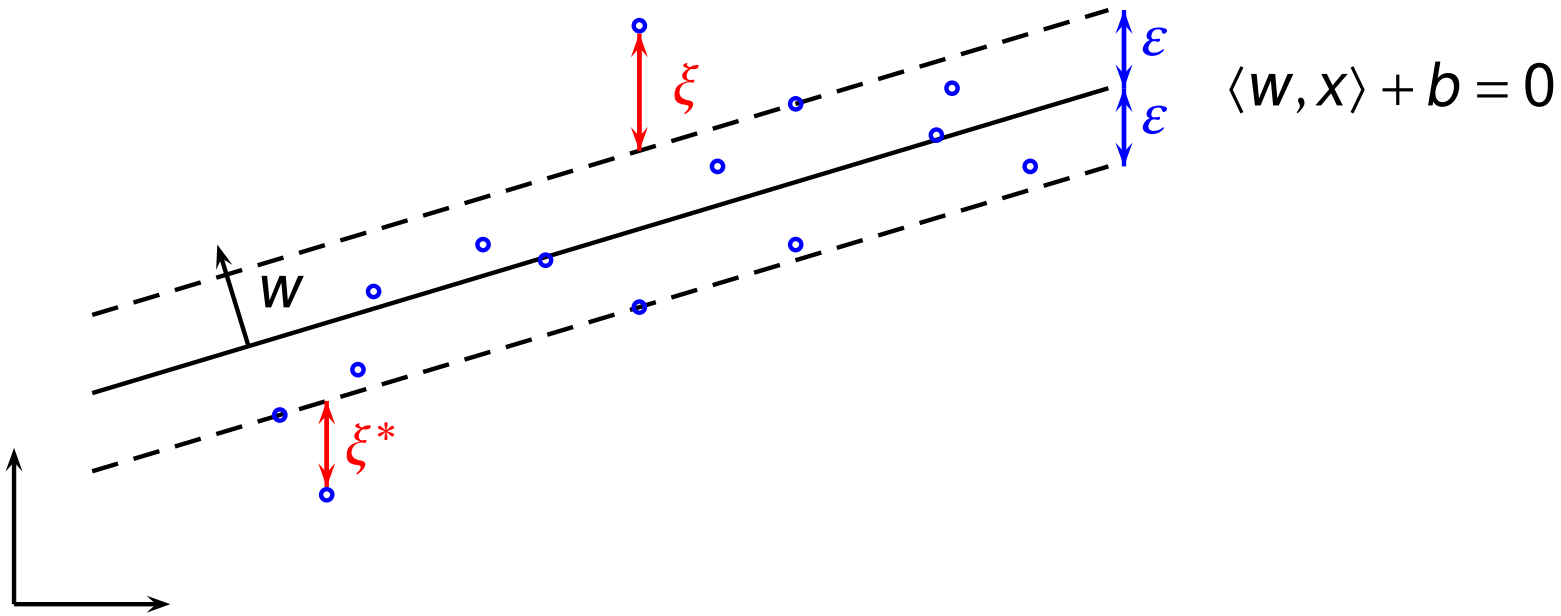
$\hookrightarrow$ SVM = Minimisation de risque empirique convexifié régularisé

Conclusion / questions ouvertes

- Le rééquilibrage des données
- La renormalisation des données
- Les règlages à effectuer :
  - Le noyau et ses paramètres (validation croisée, bootstrap ?)
  - La constante de tolérance C (validation croisée, bootstrap ?)
  - L'algorithme d'optimisation (SMO ou variantes par exemple)
- La sélection de variables
- Généralisation à la discrimination multiclasses : one-versus-all, one-versus-one ?

SVM pour la régression (SVR) : un aperçu

La ε-SVR linéaire en dimension 2



Le principe

Trouver la règle de régression linéaire la plus "plate" possible, sous certaines contraintes.

Problème d'optimisation primal pour la ε-SVR linéaire

$$\text{Minimiser en } (w, b, \xi, \xi^*) \qquad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\left(\xi_i + \xi_i^*\right)$$

$$\text{s. c.} \begin{cases} y_i - \langle w, x_i\rangle - b \leq \varepsilon + \xi_i \ \forall\ i \\ \langle w, x_i\rangle + b - y_i \leq \varepsilon + \xi_i^* \ \forall\ i \\ \xi_i,\ \xi_i^* \geq 0 \ \forall\ i \end{cases}$$

$\hookrightarrow$ $C > 0$ paramètre à ajuster.

$\hookrightarrow$ SVR non linéaire : astuce du noyau !

Le coin du UseR

Package `e1071`

```
svm(formula, data, scale = TRUE, kernel = "radial", degree
= 3, gamma = if (is.vector(x)) 1 else 1 / ncol(x), coef0 =
0, cost = 1, epsilon = 0.1, cross = 0, fitted = TRUE,
..., subset, na.action =na.omit)
```

Package `kernlab`

```
ksvm(formula,data, scaled = TRUE, kernel ="rbfdot", kpar =
"automatic", C = 1, epsilon = 0.1, cross = 0, fit = TRUE,
..., subset, na.action = na.omit)
```

Package généraliste `caret`

Références bibliographiques

Ouvrages

- N. Cristianini and J. Shawe-Taylor, An introduction to support vector machines. Cambridge University Press, Cambridge, UK, 2000.
- B. Schölkopf and A. Smola. Learning with Kernels. MIT Press, Cambridge, MA, 2002.

Polycopiés, supports de cours

- `http://www.svms.org/tutorials/` (celui de Burges notamment).
- Notes de cours en accès sur la page de Jean-Philippe Vert.

Plan du cours

6 Méthodes d'agrégation : Bagging et forêts aléatoires

- Arbres de décision
- Agrégation d'algorithmes de prédiction
- Bagging
- Forêts aléatoires
- Ajustement des paramètres / erreur Out Of Bag
- Avantages / inconvénients
- Importance des variables

Arbres de décision

Un arbre binaire de décision **CART - Classification And Regression Tree -** est un algorithme de moyennage local par partition (moyenne ou vote à la majorité sur les éléments de la partition), dont la partition est construite par divisions successives au moyen d'hyperplans orthogonaux aux axes de $\mathscr{X} = \mathbb{R}^d$, dépendant des (X_i, Y_i).

Les éléments de la partition d'un arbre sont appelés les **nœuds terminaux** ou les **feuilles** de l'arbre.

L'ensemble $\mathscr{X} = \mathbb{R}^d$ constitue le **nœud racine**. Puis chaque division définit deux nœuds, les **nœuds fils à gauche et à droite**, chacun soit terminal, soit interne, par le choix conjoint :

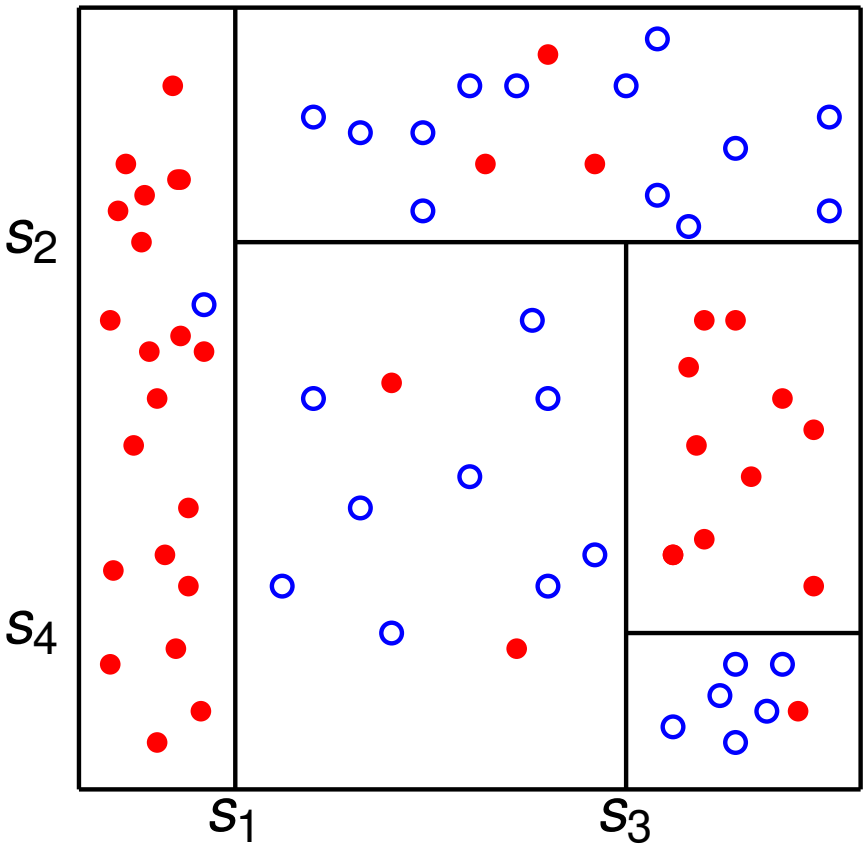
- d'une variable explicative X^j $(j = 1 \dots d)$,
- d'une valeur seuil pour cette variable.

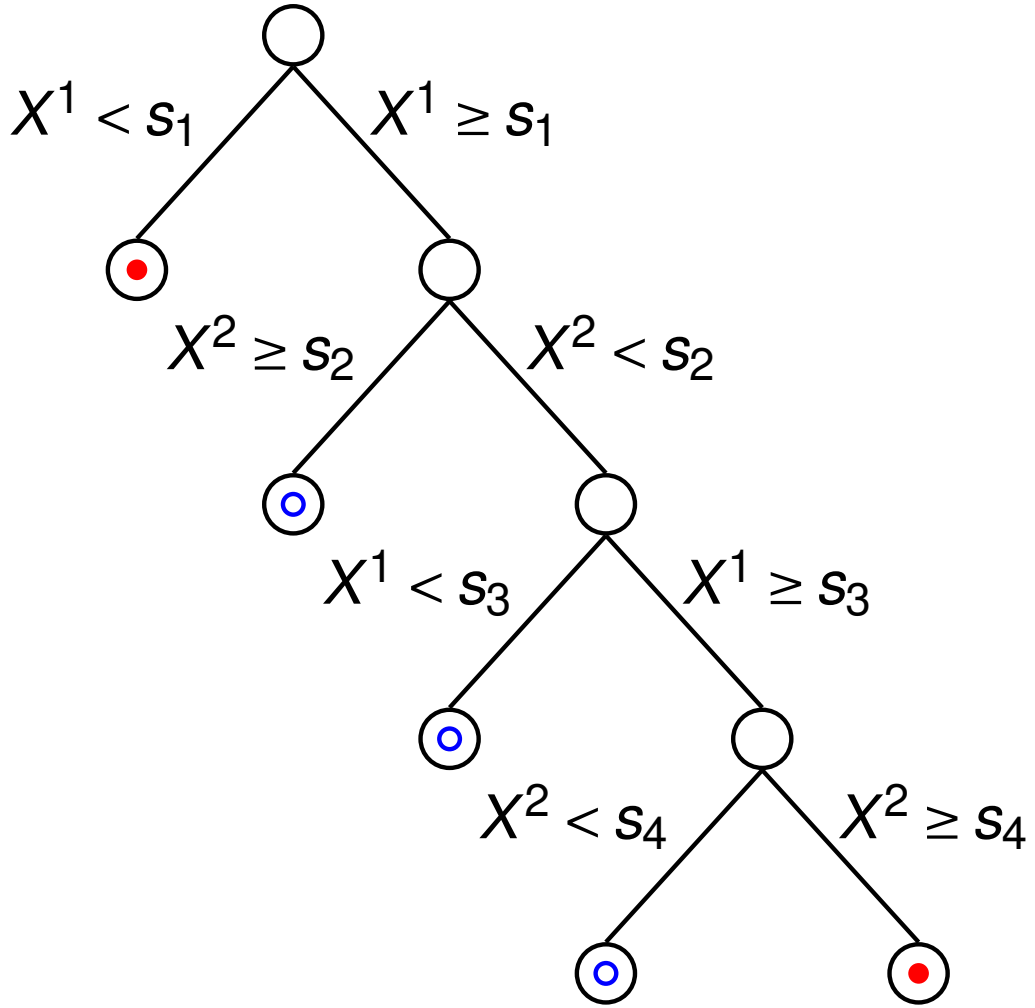
Ce choix se fait par maximisation du gain d'homogénéité, défini à l'aide d'une **fonction d'hétérogénéité** H, sur les observations de la variable à expliquer.

Pour un nœud k, si k_g, k_d désignent les nœuds fils à gauche et à droite issus de la division de ce nœud, on choisit la variable explicative et le seuil de la variable explicative maximisant :

- En régression : $H_k - (H_{k_g} + H_{k_d})$, avec $H_k =$ la variance empirique des y_i du nœud k,
- En discrimination binaire : $H_k - (p_{k_g} H_{k_g} + p_{k_d} H_{k_d})$, avec p_k la proportion d'observations dans le nœud k, et $H_k = p_k^1(1-p_k^1) + p_k^{-1}(1-p_k^{-1}) = 1 - (p_k^1)^2 - (p_k^{-1})^2$, où p_k^δ est la proportion de y_i du nœud k égaux à $\delta \rightarrow$ Indice de Gini

Arbres de décision : représentation





Arbres de décision : sélection

Étape 1 : construction de l'arbre maximal T_{max}. T_{max} correspond à la partition qui ne peut plus être subdivisée, soit parce que ses parties contiennent moins d'observations qu'un nombre fixé au départ (entre 1 et 5), soit parce qu'elles ne contiennent que des observations de même réponse (homogènes).

Étape 2 : élagage. De la suite d'arbres qui a conduit à l'arbre maximal, on extrait une sous-suite d'arbres emboîtés à l'aide du critère pénalisé suivant : pour $\alpha \geq 0$, $crit_\alpha(T) = \widehat{\mathscr{R}}_n(\widehat{\phi}_T) + \alpha|T|/n$, où $\widehat{\mathscr{R}}_n(\widehat{\phi}_T)$ est le risque apparent de la règle de régression ou de discrimination associée à l'arbre T (taux de mal classés en discrimination, erreur quadratique moyenne empirique en régression) et $|T|$ est la taille de l'arbre c'est-à-dire son nombre de feuilles.

Théorème (Breiman *et al.*)

Il existe une suite finie $\alpha_0 = 0 < \alpha_1 < \ldots < \alpha_M$ et une suite associée de sous-arbres emboîtés telles que : pour $\alpha \in [\alpha_m, \alpha_{m+1}[$,

$$\operatorname{argmin}_T crit_\alpha(T) = T_m.$$

Étape 3 : sélection du meilleur arbre dans la suite d'arbres obtenue par élagage, par estimation du risque de chaque arbre de la suite.

$\hookrightarrow$ Cette estimation se fait par validation croisée hold-out ou K blocs.

Algorithme de validation croisée K fold (pruning)

Variable

K : *entier diviseur de* n

Début

Construire $\mathcal{V}_1, \ldots, \mathcal{V}_K$ partition de $\{1, \ldots, n\}$

Pour k **variant de** 1 **à** K

Pour $\mathcal{A}_k = \{1, \ldots, n\} \setminus \mathcal{V}_k$, construire sur $D_{\mathcal{A}_k} = \{(X_i, Y_i),\ i \in \mathcal{A}_k\}$:

l'arbre maximal et la sous-suite d'arbres emboîtés $(T_p^{(k)})_p \leftrightarrow (\alpha_p^{(k)})_p$

Calculer pour tout p, $R_p^{(k)} = \frac{K}{n} \sum_{i \in \mathcal{V}_k} l\left(Y_i, \widehat{\phi}_{T_p^{(k)}}(X_i)\right)$

FinPour

Pour m **variant de** 1 **à** M

Trouver pour $k = 1 \ldots K$, $p_m^{(k)}$ tel que $\alpha^{(k)}_{p_m^{(k)}} \leq \sqrt{\alpha_m \alpha_{m+1}} < \alpha^{(k)}_{p_m^{(k)}+1}$

Estimer le risque de T_m par la moyenne R_m des $R^{(k)}_{p_m^{(k)}}$

FinPour

Retourner $(R_m)_m$

Fin

Arbres de décision : le coin du UseR

> `library(rpart)`

> `tree=rpart(formula,data=,method="class"/"anova", control=rpart.control(minspilt=1,cp=0,xval=10))`

> `plot(tree)` trace l'arbre

> `printcp(tree)` affiche la table des α_m

> `plotcp(tree)` estimations par validation croisée du risque des T_m

> `alpha=tree$cptable[which.min(tree$cptable[,"xerror"]),"CP"]` sélectionne $\alpha_{\widehat{m}}$

> `ptree=prune(tree,cp=alpha)` donne l'arbre élagué avec le $\alpha_{\widehat{m}}$ choisi

> `predict(ptree,newdata=)` prédit les sorties associées à de nouvelles entrées

Arbres de décision : interprétation

La représentation graphique de l'arbre permet une interprétation facile de l'algorithme de prédiction construit, et sa construction est algorithmiquement efficace, ce qui fait le succès de la méthode CART d'un point de vue métier.

Mises en garde

- L'arbre sélectionné ne dépend que de quelques variables explicatives, et est souvent interprété (à tort) comme une procédure de sélection de variables.
- L'arbre souffre d'une grande instabilité (fléau de la dimension, sensibilité à l'échantillon).
- La qualité de prédiction d'un arbre est souvent médiocre comparée à celle d'autres algorithmes.

$\hookrightarrow$ Agrégation d'arbres !

Agrégation d'algorithmes de prédiction

Les **méthodes d'agrégation** d'algorithmes de prédiction se décrivent de la façon suivante.

- Construction d'un grand nombre d'algorithmes de prédiction simples $\widehat{f}_b$, $b = 1 \dots B$.
- Agrégation ou combinaison de ces algorithmes sous la forme : $\widehat{f} = \sum_{b=1}^{B} w_b \widehat{f}_b$ ou $signe\left(\sum_{b=1}^{B} w_b \widehat{f}_b\right)$.

$\hookrightarrow$ En particulier : agrégation par bagging/boosting.

Le **bagging** s'applique à des algorithmes instables, de variance forte.

Le **boosting** s'applique à des algorithmes fortement biaisés, mais de faible variance.

Bagging (Breiman 1996)

Le bagging regroupe un ensemble de méthodes s'appliquant à des problèmes de régression ou de discrimination, introduites par Leo Breiman en 1996. Le terme bagging provient de la contraction de **B**ootstrap **agg**regat**ing**.

Rappels des notations :

Données observées de type entrée-sortie :
$d_1^n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ avec $x_i \in \mathbb{R}^d$, $y_i \in \mathscr{Y}$ ($\mathscr{Y} = \mathbb{R}$ en régression, $\mathscr{Y} = \{-1, 1\}$ en discrimination binaire)
$D_1^n = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$, (X_i, Y_i) i.i.d. $\sim P$ (totalement inconnue).

Objectif : prédire la sortie y associée à une nouvelle entrée x, où x est une observation de la variable X, $(X, Y) \sim P$ indépendant de D_1^n.

On note

- η^* la fonction de régression définie par $\eta^*(x) = \mathbb{E}[Y|X = x]$ (minimisant le risque quadratique)
- ϕ^* la règle **plug-in** associée (règle de Bayes) : $\phi^*(x) = signe(\eta^*(x))$.

Considérons un algorithme de régression $\widehat{\eta}$ et l'algorithme de discrimination plug-in correspondant $\widehat{\phi} = signe(\widehat{\eta})$, construits sur D_1^n.

$$\mathbb{E}\left[\mathbb{1}_{Y\neq\widehat{\phi}(X)} - \mathbb{1}_{Y\neq\phi^*(X)}\right] \leq \left(\mathbb{E}\left[(\widehat{\eta}(X) - \eta^*(X))^2\right]\right)^{1/2}$$

En régression, le bagging consiste à agréger un ensemble d'algorithmes $\widehat{\eta}_1, \ldots, \widehat{\eta}_B$ sous la forme $\widehat{\eta}(x) = \frac{1}{B}\sum_{b=1}^{B} \widehat{\eta}_b$.

Décomposition biais/variance

Pour $x \in \mathbb{R}^d$, $\mathbb{E}\left[(\widehat{\eta}(x) - \eta^*(x))^2\right] \leq (\mathbb{E}[\widehat{\eta}(x)] - \eta^*(x))^2 + \mathrm{Var}(\widehat{\eta}(x)))$.

Si les algorithmes de régression $\widehat{\eta}_1, \ldots, \widehat{\eta}_B$ étaient i.i.d. on aurait :

$$\mathbb{E}[\widehat{\eta}(x)] = \mathbb{E}[\widehat{\eta}_b(x)] \text{ et } \mathrm{Var}(\widehat{\eta}(x)) = \mathrm{Var}(\widehat{\eta}_b(x))/B$$

$\hookrightarrow$ biais identique, mais variance diminuée.

En pratique, les algorithmes ne peuvent pas être i.i.d. puisqu'ils sont construits sur le même échantillon D_1^n !

Si les algorithmes sont seulement identiquement distribués, si $\rho(x)$ est le cœfficient de corrélation entre $\widehat{\eta}_b(x)$ et $\widehat{\eta}_{b'}(x)$,

$$\mathrm{Var}(\widehat{\eta}(x)) = \rho(x)\mathrm{Var}(\widehat{\eta}_b(x)) + \frac{1-\rho(x)}{B}\mathrm{Var}(\widehat{\eta}_b(x)) \to_{B\to+\infty} \rho(x)\mathrm{Var}(\widehat{\eta}_b(x))$$

Comment construire des algorithmes peu corrélés entre eux, alors qu'ils sont a priori construits sur le même échantillon ?

✔ **Solution de Breiman** : construire un même algorithme de base sur des échantillons bootstrap de D_1^n.

Considérons :

- un algorithme de régression $D \mapsto \eta_D$, ou de discrimination $D \mapsto \phi_D$
- un nombre B (grand) d'échantillons bootstrap de D_1^n : $D_{m_n}^{*1} \dots D_{m_n}^{*B}$, de taille $m_n \leq n$, indépendants les uns des autres conditionnellement à D_1^n.

Pour $b = 1 \dots B$,

$$\widehat{\eta}_b = \eta_{D_{m_n}^{*b}} \quad \text{ou} \quad \widehat{\phi}_b = \phi_{D_{m_n}^{*b}}.$$

L'algorithme de bagging consiste à agréger les algorithmes $\widehat{\eta}_1, \dots, \widehat{\eta}_B$ ou $\widehat{\phi}_1, \dots, \widehat{\phi}_B$ de la façon suivante :

- $\widehat{\eta} = \frac{1}{B} \sum_{b=1}^{B} \widehat{\eta}_b$ en régression
- $\widehat{\phi} = signe(\sum_{b=1}^{B} \widehat{\phi}_b)$ (vote à la majorité) en discrimination binaire

Exemple : algorithme du ppv

Théorème (Biau, Devroye, Lugosi 2010)

Bagger un nombre infini de fois l'algorithme du ppv (non universellement consistant), avec $m_n \to +\infty$, $m_N = o(n)$, le rend universellement consistant.

L'algorithme de base sera choisi comme étant sensible aux perturbations de l'échantillon : calculé sur deux échantillons bootstrap, il donnera deux algorithmes peu corrélés entre eux ($\rho(x) < 1$).

✔ L'exemple des arbres de décision CART est fondamental $\Rightarrow$ forêts aléatoires.

Forêts aléatoires (Breiman and Cutler 2005)

Le terme de forêt aléatoire se rapporte initialement à l'agrégation, au sens large, d'arbres de régression ou de discrimination.
Désormais, il désigne le plus souvent une forêt aléatoire **Random Input**, méthode introduite par Breiman et Cutler (2005)
`http://www.stat.berkeley.edu/users/breiman/RandomForests/`

$\hookrightarrow$ Bagging d'arbres maximaux construits sur des échantillons bootstrap de taille $m_n = n$, par une variante de la méthode CART consistant, pour chaque nœud, à

- tirer au hasard un sous-échantillon de taille $m < p$ de variables explicatives,
- partitionner le nœud en un nœud fils à gauche et un nœud fils à droite sur la base de la "meilleure" de ces m variables explicatives (sélectionnée par les critères de la méthode CART).

✔ Diminution encore plus importante de la corrélation entre les arbres.

Algorithme Random Forest RI

Variable

x : *l'entrée dont on veut prédire la sortie*

d_1^n : *l'échantillon observé*

m : *le nombre de variables explicatives sélectionnées à chaque nœud*

B : *le nombre d'itérations*

Début

Pour b **variant de** 1 **à** B

Tirer un échantillon bootstrap d^{*b} de d_1^n

Construire un arbre maximal $\widehat{\eta}_b$ ou $\widehat{\phi}_b$ sur l'échantillon bootstrap d^{*b} par la variante de CART :

Pour chaque nœud variant de 1 **à** N_b

Tirer un sous-échantillon de m variables explicatives

Partitionner le nœud à partir de la "meilleure" de ces m variables

FinPour

FinPour

Retourner $\frac{1}{B}\sum_{b=1}^{B}\widehat{\eta}_b(x)$ ou $signe\left(\sum_{b=1}^{B}\widehat{\phi}_b(x)\right)$

Fin

Ajustement des paramètres / erreur Out Of Bag

On remarque que si m diminue, la variance diminue (la corrélation entre les arbres diminue), mais le biais augmente (les arbres ont une moins bonne qualité d'ajustement).

Compromis biais/variance $\Rightarrow$ choix optimal de m lié aussi au nombre d'observations dans les nœuds terminaux.

$\hookrightarrow$ Ajustement par validation croisée hold-out ou K fold ou par l'estimation Out Of Bag du risque

L'**erreur Out Of Bag** d'une forêt aléatoire RI est définie par

- $\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \sum_{b=1}^{B} I_i^b \widehat{\eta}_b(x_i) \Big/ \sum_{b=1}^{B} I_i^b\right)^2$ en régression,
- $\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{signe\left(\sum_{b=1}^{B} I_i^b \widehat{\eta}_b(x_i)\right) \neq y_i}$ en discrimination binaire,

où $I_i^b = 1$ si l'observation $i \notin d^{*b}$, 0 sinon.

Avantages / inconvénients

Avantages

✔ Bonne qualité de prédiction
✔ Implémentation facile
✔ Adaptée à la PARALLÉLISATION...

Inconvénients

✘ Du point de vue métier, on perd l'interprétation facile d'un arbre (effet "boîte noire")

↪ mesures d'importance des variables, même si on perd l'interprétation avec des seuils sur ces variables.

Importance des variables

Méthode rudimentaire - non retenue par Breiman et Cutler - consistant à regarder la fréquence des variables explicatives sélectionnées pour découper les arbres de la forêt.

Méthode recommandée par Breiman et Cutler : pour chaque variable explicative X^j et pour tout b :

- Calculer l'erreur Out Of Bag de l'arbre $\widehat{\eta}_b$ ou $\widehat{\phi}_b$ (sur l'échantillon Out Of Bag correspondant) :
  $OOB_b = \frac{1}{\sum_{i=1}^n I_i^b} \sum_{i=1}^n I_i^b (\widehat{\eta}_b(x_i) - y_i)^2$ ou $\frac{1}{\sum_{i=1}^n I_i^b} \sum_{i=1}^n I_i^b \mathbb{1}_{\widehat{\phi}_b(x_i) \neq y_i}$
- Créer un échantillon Out Of Bag permuté (en permutant aléatoirement les valeurs de la variable explicative X^j dans l'échantillon Out Of Bag) et calculer l'erreur Out Of Bag OOB_b^j de l'arbre $\widehat{\eta}_b$ ou $\widehat{\phi}_b$ sur cet l'échantillon Out Of Bag permuté.

L'importance de la variable X^j est finalement mesurée par

$$\frac{1}{B} \sum_{b=1}^{B} (OOB_b^j - OOB_b).$$

Le coin du UseR

Package `randomForest`

> `model=randomForest(formula .,data=, )`

> `model$err.rate` donne l'erreur Out Of Bag de la forêt

> `order(importance(model),decreasing=TRUE)` donne l'ordre d'importance des variables explicatives

> `predict(model,newdata=)` prédit les nouvelles sorties

✘ Par défaut, `randomForest` prend :

- un nombre d'observations dans les nœuds terminaux égal à 5 en régression, 1 en discrimination.
- un nombre de variables explicatives p égal à $d/3$ en régression, $\sqrt{d}$ en discrimination.

Packages généralistes : `caret` et `h2o`

Références bibliographiques

Articles

L. Breiman (1996). Bagging predictors.
L. Breiman (2001). Random Forests.
G. Biau, L. Devroye, G. Lugosi (2008). Consistency of random forests and other averaging classifiers.

Ouvrages

L. Breiman, J. Friedman, C. J. Stone, R. A. Olshen (1984). Classification and regression trees.
T. Hastie, R. Tibshirani, and J. Friedman (2001). The elements of statistical learning.