Apprentissage automatique numérique

Classification supervisée : exemples

- *Reconnaissance des visages, des chiffres, ...* : reconnaître un code postal (ou une personne à partir d'une photo).

  Données : des chiffres écrits sur une rétine de 16x16 pixels, associés à une classe parmi $\{0, 1, \ldots, 9\}$.

  Problème de *reconnaissance des formes* (*pattern recognition*).

- *Email/Spam* : écarter automatiquement les annonces publicitaires et autres messages non sollicités.

  Données : des messages réguliers et des SPAMs.

  Objectif : construire un *classifieur*, capable d'attribuer une de ces deux classes à un nouveau document.

Classification binaire, *classification n-aire*, *classification multi-étiquettes*.

Apprentissage automatique numérique

1. Classification linéaire
2. Machines à vecteurs support et méthodes à noyaux
3. Réseaux de neurones

Quelques ouvrages de référence

- *Apprentissage artificiel*, d'Antoine Cornuejols et Laurent Miclet.
- *Machine Learning* de Tom Mitchell.
- *The elements of statistical Learning* de Hastie, Tibshirani et Friedman.
- *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations*, de Witten et Frank, auteurs de Weka.
- *An introduction to SVM*, N. Cristianini et J. Shawe-Taylor.
- *Learning with Kernels*, B. Schölkopf, A. J. Smola.

Modélisation : classification supervisée

X : domaine des données. $X = X_1 \times \ldots \times X_n$ où chaque X_i est fini, égal à $\mathbb{R}$ ou à Σ^* pour un certain alphabet Σ fini.

Y : un ensemble fini de classes.

Les observations (les *exemples*) sont les réalisations (x, y) d'une variable aléatoire fixe mais inconnue de loi de distribution $P(x, y)$ définie sur $X \times Y$.

Un *échantillon* S est un ensemble fini d'exemples $\{(x_1, y_1), \ldots, (x_l, y_l)\}$ i.i.d. selon P.

On cherche une fonction $f : X \to Y$ qui approche au mieux les réponses du superviseur.

Fonction de perte (*loss function*)

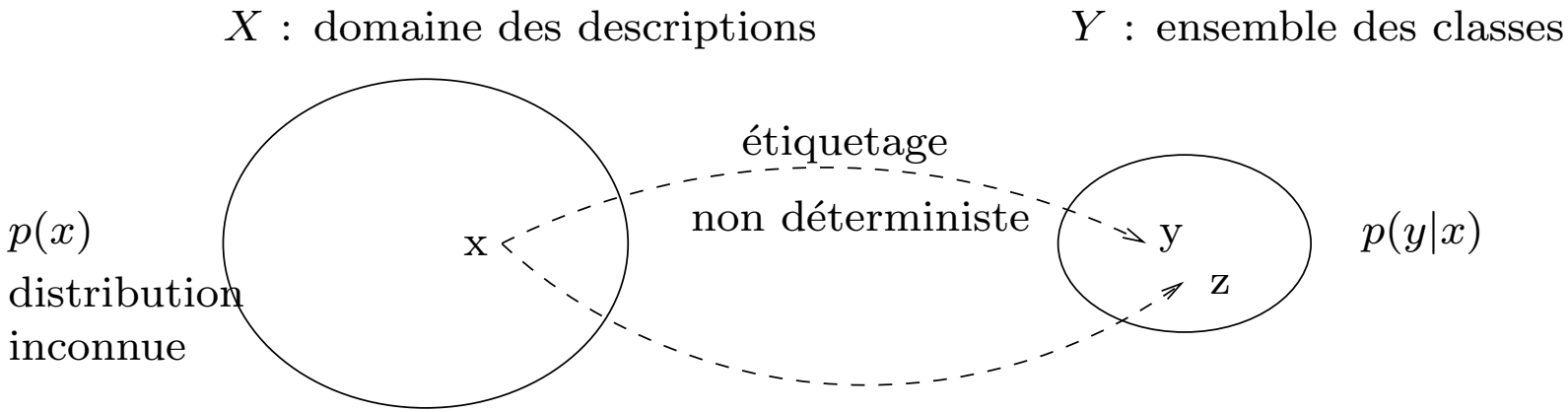
$$L(y, f(x)) = \begin{cases} 0 \text{ si } y = f(x) \\ 1 \text{ sinon.} \end{cases}$$

La fonction *risque* (ou *erreur*) : espérance mathématique de la fonction de perte.

$$R(f) = \int L(y, f(x)) dP(x, y) = \int_{y \neq f(x)} dP(x, y) = P(y \neq f(x)).$$

Le problème général de la classification supervisée :

étant donné un échantillon $S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$,
trouver une fonction f *qui minimise le risque* $R(f)$.



$S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ tiré selon $p(x, y) = p(x)p(y|x)$

Objectif : trouver $f : X \to Y$ dont l'erreur $R(f) = P(y \neq f(x))$ soit la plus petite possible.

Quelques règles de classification :

- La *règle majoritaire* : retourner la classe y_{maj} pour laquelle $P(y)$ est maximum : pour tout $x \in X$,

$$f_{maj}(x) = ArgMax_y P(y) = y_{maj} \text{ et } R(f_{maj}) = 1 - P(y_{maj}).$$

- La *règle du maximum de vraisemblance* (*maximum likelihood*) : retourner pour chaque instance x la classe y pour laquelle x est la valeur la plus observée.

$$f_{mv}(x) = ArgMax_y P(x|y).$$

- La *règle de Bayes* : retourner pour chaque instance x, la classe y dont l'observation est la plus probable, ayant observé x.

$$f_{Bayes}(x) = ArgMax_y P(y|x).$$

Théorème : La règle de décision de Bayes est la règle de risque minimal.

Preuve : soit f un classifieur et f_{Bayes} le classifieur de Bayes ; soit $x \in X$.

$$P(f_{Bayes}(x)|x) \geq P(f(x)|x) \Rightarrow P(x, f_{Bayes}(x)) \geq P(x, f(x))$$

$$\begin{aligned} P(y \neq f_{Bayes}(x)) &= P(\{x\} \times Y) - P(x, f_{Bayes}(x)) \\ &\leq P(\{x\} \times Y) - P(x, f(x)) \\ &= P(y \neq f(x)) \end{aligned}$$

Donc,

$$R(f_{Bayes}) \leq R(f).$$

L'apprentissage en pratique

On dispose d'un échantillon qu'on suppose i.i.d. ; on cherche un *classifieur* f dont le risque $R(f)$ soit le plus faible possible.

Il existe toujours une meilleure solution f_{Bayes} ... inaccessible !

Dans la pratique, on cherche une solution dans un ensemble de fonctions $\mathcal{F}$ particulier : *arbres de décision*, *réseaux de neurones*, *fonctions linéaires*, etc.

Soit f_{opt} une fonction de risque minimal dans $\mathcal{F}$

L'ensemble $\mathcal{F}$ doit avoir deux qualités :

1. contenir des fonctions f_{opt} dont le risque n'est pas trop éloigné de $R(f_{min})$
2. permettre d'approcher un classifieur de risque minimal f_{opt} au moyen des informations dont on dispose.

Le principe de minimisation du risque empirique

Une idée naturelle : sélectionner une fonction dans $\mathcal{F}$ qui décrit au mieux les données de l'échantillon d'apprentissage.

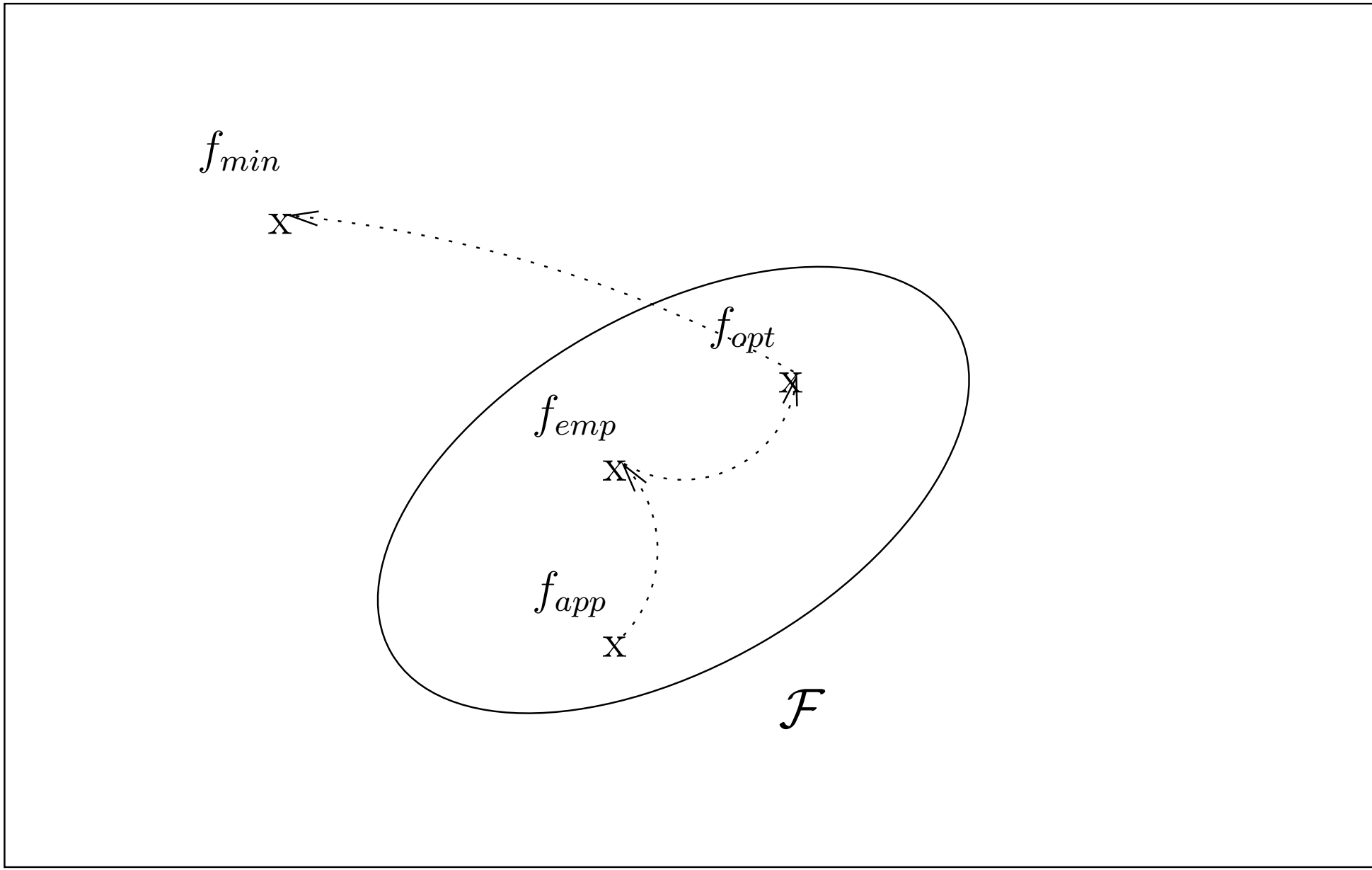
Le *risque empirique* $R_{emp}(f)$ d'une fonction f sur l'échantillon $S = \{(x_1, y_1), \ldots (x_l, y_l)\}$ est la moyenne de la fonction de perte calculée sur S :

$$R_{emp}(f) = \frac{1}{l} \sum_{i=1}^{l} L(y_i, f(x_i)) = \frac{Card\{i | f(x_i) \neq y_i\}}{l}.$$

$R_{emp}(f)$ est la moyenne du nombre d'erreurs de prédiction de f sur les éléments de S ; c'est une estimation du risque réel $R(f)$ de f.

Principe inductif de minimisation du risque empirique (ERM) :

trouver une fonction $f \in \mathcal{F}$ qui minimise $R_{emp}(f)$.


f_{min}
f_{opt}
f_{emp}
f_{app}
$\mathcal{F}$

Classification linéaire binaire

$X \subset \mathbb{R}^n$, $Y = \{-1, 1\}$, $S = \{(x_1, y_1), \ldots, (x_l, y_l)\} \subset (X \times Y)^l$: échantillon d'apprentissage.

Définition. Un classifieur linéaire est une fonction de la forme $f(x) = sgn(\langle w, x \rangle + b)$ où $w \in \mathbb{R}^n$, $b \in \mathbb{R}$ et

$$sgn(x) = \begin{cases} 1 \text{ si } x \geq 0 \\ -1 \text{ sinon.} \end{cases} .$$

Interprétation géométrique : $\langle w, x \rangle + b = 0$ est l'équation d'un hyperplan qui sépare X en deux demi-espaces correspondant aux deux classes.

Un échantillon S est *linéairement séparable* s'il existe un classifieur linéaire qui classe correctement tous les exemples de S.

Remarques :

- Séparer des données séparables est un problème qui peut être résolu en temps polynomial par un algorithme de programmation linéaire : chaque exemple (x_i, y_i) fournit une inéquation linéaire $y_i(\langle w, x_i \rangle + b) > 0$ et il suffit de chercher (w, b) qui les satisfasse toutes.
- Il existe une infinité d'hyperplans séparant un échantillon séparable, qui ne sont pas équivalent du point de vue de l'apprentissage.

Nous allons étudier deux méthodes :

- L'algorithme du perceptron (Rosenblatt, 1958)
- Le classifieur de marge maximale (redécouvert plusieurs fois).

Algorithme d'apprentissage du Perceptron

Entrée : $S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, un échantillon linéairement séparable de $\mathbb{R}^n \times \{-1, 1\}$

$w_0 = 0, b_0 = 0, k = 0$

$R = max_{1 \leq i \leq l} ||x_i||$

Répéter

  Pour $i = 1$ **à** l

    Si $y_i(\langle w_k, x_i \rangle + b_k) \leq 0$ **alors**

      $w_{k+1} = w_k + y_i x_i$

      $b_{k+1} = b_k + y_i R^2$

      $k = k + 1$

Jusqu'à ce qu'il n'y ait plus d'erreurs

Sortie : (w_k, b_k)

C'est une procédure *on-line*, par *correction d'erreurs (mistake-driven)*.

On peut montrer que si S est séparable, l'algorithme converge.

Théorème (Novikoff) : Soit $S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ un échantillon d'apprentissage non trivial (i.e. $\exists i, j \; y_i y_j = -1$) et soit $R = max_{1 \leq i \leq l} ||x_i||$.

Supposons qu'il existe $w, b, \gamma > 0$ tels que

$$y_i(\langle w, x_i \rangle + b) \geq \gamma$$

pour $1 \leq i \leq l$. Alors, le nombre d'erreurs $(y_i(\langle w_k, x_i \rangle + b_k) \leq 0)$ commises par l'algorithme est au plus égal à $(2R/\gamma)^2$.

Remarques : le rapport R/γ est invariant par homothétie ; $y(\langle w, x \rangle + b)$ est la *marge fonctionnelle* du classifieur relativement à l'exemple (x, y).

Remarque : l'hypothèse finale est une combinaison linéaire des exemples d'apprentissage.

$$w = \sum_{i=1}^{l} \alpha_i y_i x_i.$$

Les nombres α_i sont positifs et égaux au nombre de fois où une mauvaise classification de x_i a entraîné une mise à jour du perceptron. Ils peuvent être vus comme une représentation duale de la solution :

$$f(x) = sgn(\langle w, x \rangle + b) = sgn\left(\sum_{i=1}^{l} \alpha_i y_i \langle x_i, x \rangle + b\right).$$

Algorithme d'apprentissage du Perceptron : forme duale

entrée : $S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, un échantillon séparable

$\alpha = 0, b = 0$

$R = max_{1 \leq i \leq l} ||x_i||$

répéter

  Pour $i = 1$ **à** l

    Si $y_i(\sum_{j=1}^{l} \alpha_j \langle x_j, x_i \rangle + b) \leq 0$ **alors**

      $\alpha_i = \alpha_i + 1$

      $b = b + y_i R^2$

      $k = k + 1$

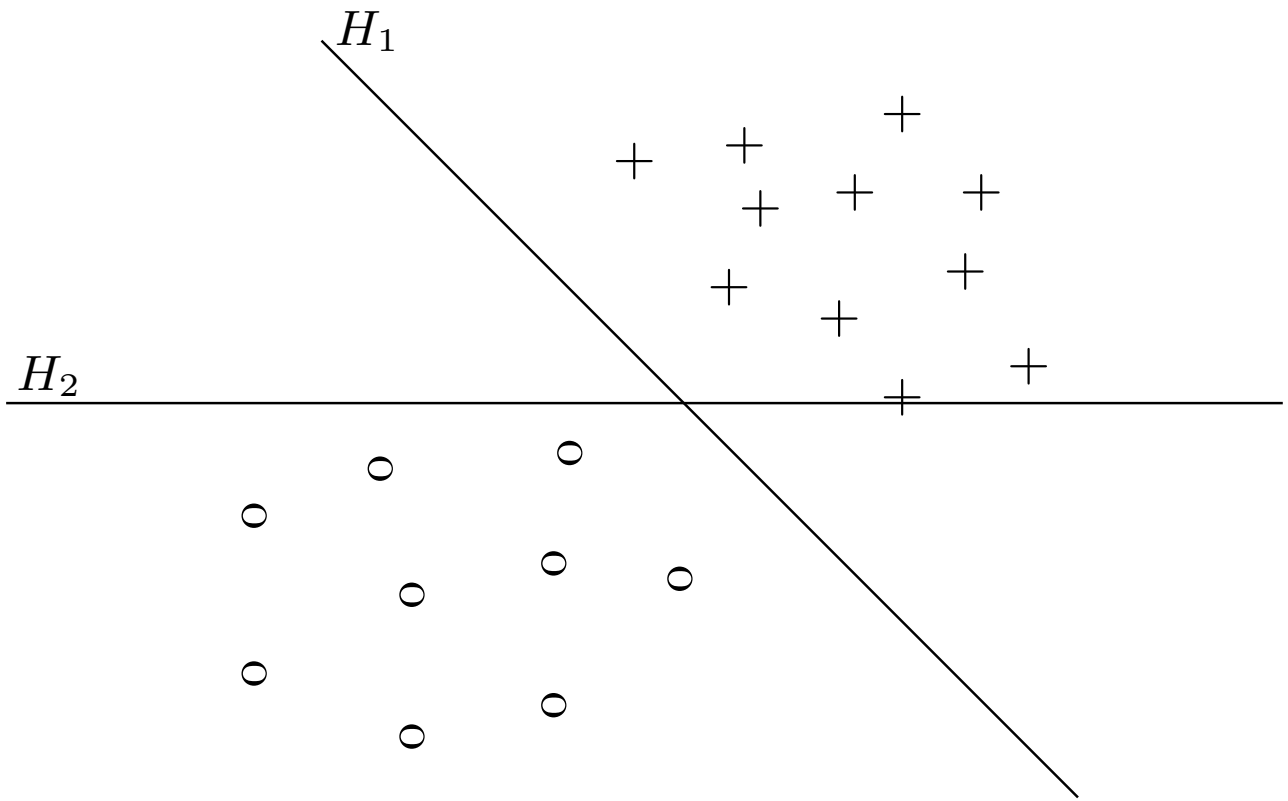
Jusqu'à ce qu'il n'y ait plus d'erreurs

Sortie : (α, b)

Remarques :

- dans la représentation duale, le nombre de paramètres de la solution ne dépend pas de la dimension de l'espace dans lequel les x_i sont plongés,
- les exemples d'apprentissage ne sont pris en compte par l'algorithme que par l'intermédiaire de leurs produits scalaires.
- On appelle **Matrice de Gram** la matrice $G = (\langle x_i, x_j \rangle)_{1 \leq i,j \leq l}$.

Entre ces deux solutions, laquelle est la meilleure ?

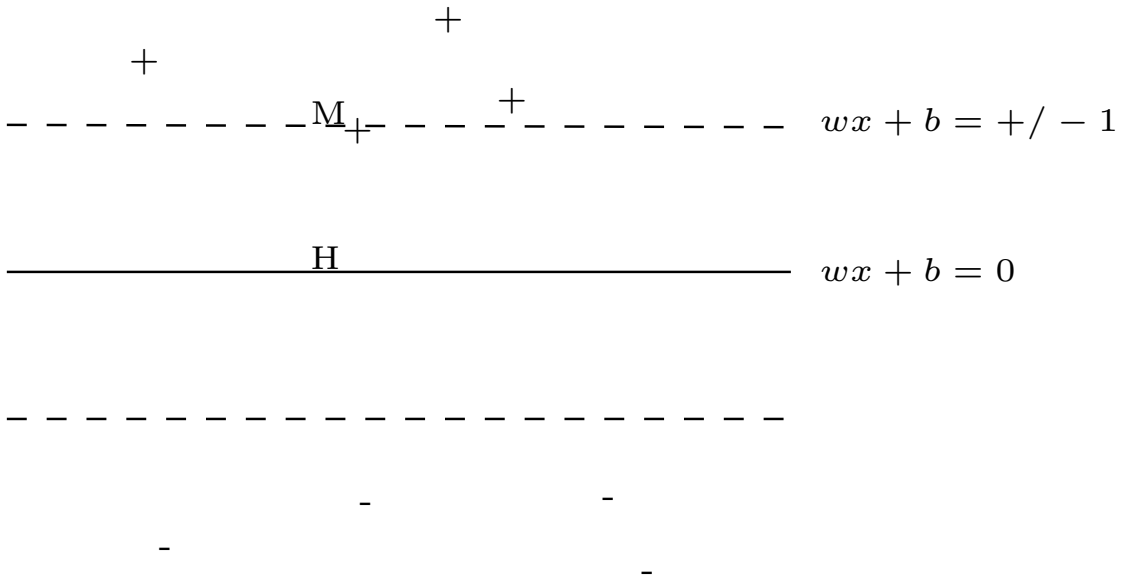


Intuitivement : H_1 est meilleur que H_2 parce que la **marge** d'erreur semble plus grande.

La notion de marge

Soit S un échantillon linéairement séparable et soit H un hyperplan séparateur, d'équation $\langle w, x \rangle + b = 0$.

On peut choisir w et b tel que le point M le plus proche de H satisfasse: $\langle w, x_M \rangle + b = +/-1$.



$wx_M + b = +/-1$
$wx_H + b = 0$
$marge = HM$

$\langle w \cdot (x_M - x_H) \rangle = +/-1$
$|x_M - x_H| = 1/||w||$
$marge = 1/||w||$

Hyperplans optimaux

Soit $S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, $x_i \in R^n$ et $y_i \in \{-1, 1\}$ un échantillon séparable.

Il existe un unique hyperplan de marge maximale qui est solution du problème d'optimisation quadratique convexe suivant :

$$\begin{cases} f(x) = \langle w, x \rangle + b \\ y_i f(x_i) \geq 1 \text{ pour tout } i = 1 \ldots l \\ \text{ Minimiser } ||w||^2. \end{cases}$$

Multiplicateurs de Lagrange : $\alpha_i \geq 0$

$$L(w, b, \alpha) = \frac{1}{2} w^2 - \sum_{i=1}^{l} \alpha_i [y_i(\langle w, x_i \rangle + b) - 1]$$

Hyperplans optimaux : propriétés de la solution

$$L(w, b, \alpha) = \frac{1}{2}w^2 - \sum_{i=1}^{l} \alpha_i [y_i(\langle w, x_i \rangle + b) - 1]$$

La solution (*unique* grâce à la convexité) (w^*, b^*, α^*) est un *point selle* du Lagrangien : $L(w, b, \alpha^*)$ est minimal en (w^*, b^*) et $L(w^*, b^*, \alpha)$ est maximal en α^*.

Propriétés

- $w^* = \sum_{i=1}^{l} \alpha_i^* y_i x_i$
- $\sum_{i=1}^{l} \alpha_i^* y_i = 0$
- Si $\alpha_i^* \neq 0$ alors $y_i(\langle w^*, x_i \rangle + b^*) = 1$

Hyperplans optimaux : vecteurs supports

Si $\alpha_i^* \neq 0$ alors $y_i(\langle w^*, x_i \rangle + b^*) = 1$

On appelle *vecteur support* toute donnée x_i telle que $\alpha_i^* \neq 0$.

Soit VS l'ensemble des vecteurs supports de S.

Propriétés

- Si $x_i \in SV$, $\langle w^*, x_i \rangle + b^* = y_i$
- Si $x_i \notin SV$, la contrainte correspondante n'est pas active,
- Les problèmes d'optimisation associés à S et à VS ont la même solution.
- Le rapport $\#SV/\#S$ est une borne supérieure de l'estimateur *leave-one-out* de l'erreur en généralisation de la solution.

Hyperplans optimaux : problème dual

Maximiser

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

sous les contraintes $\sum_{i=1}^{l} \alpha_i y_i = 0$ et $\alpha_i \geq 0$.

Remarques

- Seuls les produits scalaires $\langle x_i, x_j \rangle$ sont nécessaires pour trouver l'hyperplan optimal.
- Retrouver w^* à partir des α_i^* : $w^* = \sum_{i=1}^{l} \alpha_i^* y_i x_i$
- Calcul de b^* à partir d'un vecteur support : $\langle w^*, x_i \rangle + b^* = y_i$.
- La différence des fonctions à optimiser (*duality gap*) $||w||^2/2 - W(\alpha)$ est ≥ 0 et nulle au point solution uniquement.

Exercice

Soit $S = \{((4,3),1),((0,2),1),((0,0),-1)\}$.

1. Donnez l'équation de l'hyperplan calculé par l'algorithme d'apprentissage du Perceptron.
2. Donnez l'équation de l'hyperplan optimal, solution du problème "direct".
3. Résolvez le problème dual.

Et lorsque les données ne sont pas séparables ?

- Trouver le classifieur linéaire qui minimise le nombre d'erreurs est un problème NP-complet.
- L'algorithme du Perceptron oscille, changeant d'hypothèses à chaque présentation d'un contre-exemple sans se stabiliser sur une solution intéressante.

Idée : se soucier davantage de la confiance avec laquelle la plupart des exemples sont correctement classés plutôt que du nombre d'exemples mal classés.

$\rightarrow$ maximiser la marge d'un séparateur linéaire en tolérant un nombre limité d'exceptions : notion de marge *souple* (*soft margin*).

Soft margin optimisation

On introduit des variables de relachement (*slack variable*).

$$\begin{cases} f(x) = w, x + b \\ y_i f(x_i) \geq 1 - \xi_i \text{ et } \xi_i \geq 0 \text{ pour tout } i = 1, \ldots, l \\ \text{Minimiser } ||w||^2 + C \sum_{i=1}^{l} \xi \text{ (ou } ||w||^2 + C \sum_{i=1}^{l} \xi^2). \end{cases}$$

Problème d'optimisation dual :

Maximiser

$$\sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j \langle x_i, x_j \rangle$$

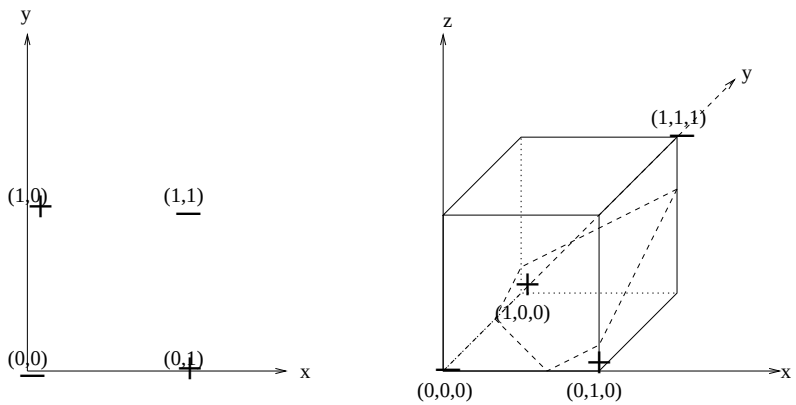
sous les contraintes

$$\sum_{i=1}^{l} \alpha_i y_i = 0 \text{ et } C \geq \alpha_i \geq 0$$

Expressivité des discriminants linéaires

Aucune raison que les exemples “naturels” se situent sagement de part et d’autre d’un hyperplan

Une idée : plonger les données dans un espace plus grand.



(x,y) −−−> (x,y,xy)

C : −x −y + 2xy +1/3 =0 <−−− P : −x−y+2z+1/3=0

Est-ce une bonne idée de plonger les données dans un espace plus grand ?

Pour séparer les données d'apprentissage : sans doute.

Pour apprendre à classer de nouvelles données : ce n'est pas évident.

Minimiser $R_{emp}(f)$ permet-il toujours de minimiser $R(f)$?

La *Théorie de l'apprentissage statistique* traite de ce problème (Vapnik).

Théorème : (Vapnik) Soit $P(,,,)$ une distribution de probabilité définie sur $\mathbb{R}^n \times \{-1, 1\}$, soit $\delta > 0$, soit l un entier tq $l > n$, soit S un ensemble de l exemples tirés indépendamment selon P et soit f un classifieur linéaire. Alors, avec une probabilité supérieure à $1 - \delta$, on a

$$R(f) \leq R_{emp}(f) + \sqrt{\frac{n+1}{l}\left(1 + \log\frac{2l}{n+1}\right) + \frac{1}{l}\log\frac{4}{\delta}}$$

Application 1 : dans $\mathbb{R}^2$, un échantillon de 100 exemples i.i.d. selon une distribution P fixe mais inconnue est séparée linéairement par le classifieur f. Peut-on majorer l'erreur de prédiction (le risque) de f avec une confiance de 95 % ?

$$R(f) \leq R_{emp}(f) + \sqrt{\frac{n+1}{l}\left(1 + \log\frac{2l}{n+1}\right) + \frac{1}{l}\log\frac{4}{\delta}}$$

Application 2 : dans $\mathbb{R}^2$, le classifieur linéaire f commet 10 erreurs de classification sur un échantillon de 100 exemples. Un plongement $\mathbb{R}^2$ permet de trouver un classifieur g qui sépare les données. Devra t-on utiliser f plutôt que g ? Où l'inverse ?

Application 3 : principe de minimisation du risque structurel. Une suite de plongement dans des espaces de dimensions $n_1 < \ldots < n_h$ et dans chacun de ces espaces, une "séparation" linéaire commettant $k_1 \geq \ldots \geq k_h$ erreurs de classification. Choisir le plongement qui minimise

$$k_i + \sqrt{\frac{n_i+1}{l}\left(1 + \log\frac{2l}{n_i+1}\right) + \frac{1}{l}\log\frac{4}{\delta}}$$

Ce principe est généralisable à toutes classes de classifieur.

$$R(f) \leq R_{emp}(f) + \sqrt{\frac{n+1}{l}\left(1 + \log\frac{2l}{n+1}\right) + \frac{1}{l}\log\frac{4}{\delta}}$$

Est-ce une bonne idée de plonger les données dans un espace plus grand ?

Réponse : pas à n'importe quel prix ! Il faut que l'augmentation de la dimension soit compensée par le nombre d'exemples bien classés.

Théorème : (Vapnik, Bartlett, Shawe-Taylor 99)

Soit $P(,,,)$ une distribution de probabilité définie sur $\mathbb{R}^n \times \{-1, 1\}$, soit $\delta > 0$, soit l un entier, soit S un ensemble de l exemples tirés indépendamment selon P et soit $R > 0$ tel que $\forall (x, y) \in S$, $||x|| < R$. Soit f un classifieur linéaire et soit $\rho > 0$. On note $R_\rho(f)$ la proportion des exemples de S mal classés ou situés à une distance inférieure à ρ de l'hyperplan d'équation $f(x) = 0$. Alors, avec une probabilité supérieure à $1 - \delta$, on a

$$R(f) \leq R_\rho(f) + \sqrt{\frac{c}{l}\left(\frac{R^2}{\rho^2}\ln^2\frac{l}{\rho} + \ln\frac{1}{\delta}\right)}$$

où c est une constante.

$$R(f) \leq R_\rho(f) + \sqrt{\frac{c}{l}\left(\frac{R^2}{\rho^2}\ln^2\frac{l}{\rho} + \ln\frac{1}{\delta}\right)}$$

Commentaires :

- la dimension de l'espace de plongement ne figure plus dans la borne !
- c'est R/ρ, c'est-à-dire la marge de séparation, qui permet de contrôler la qualité de la prédiction ; il est donc fondé de rechercher l'hyperplan optimal (de marge maximale);
- le principe de minimisation du risque empirique n'est pas le seul disponible.

  une méthode d'apprentissage théoriquement fondée : plonger les données dans un espace de dimension quelconque (éventuellement infinie) en contrôlant la marge de l'hyperplan optimal.

Comment réaliser un plongement des données ?

- c'est la marge de l'hyperplan de séparation qui permet de contrôler la qualité de la prédiction ;
- étant donnés un plongement $\Phi : X \to Y$ et un échantillon $S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, seule la matrice de Gram $G = (\langle \Phi(x_i), \Phi(x_j) \rangle)_{1 \leq i, j_l eql}$ est nécessaire pour calculer l'équation d'un hyperplan de marge maximale, que les données soient séparables ou non (soft margin).

Il suffit donc d'utiliser une fonction $k : X \times X \to \mathbb{R}$ qui pourra être interprétée comme produit scalaire des images dans un plongement Φ :

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle.$$

Noyaux

Définition : étant donné un domaine X, on appelle *noyau* toute application $k : X \times X \to \mathbb{R}$ telle qu'il existe un espace de Hilbert $\mathcal{H}$ et une fonction de plongement $\Phi : X \to \mathcal{H}$ tel que pour tout $x, y \in X$, $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$.

Exemples de noyaux :

- $X = \mathbb{R}^n, k(x, y) = \left(\sum_{i=1}^n x_i y_i\right)^d$
  On a $\Phi : X \to \mathbb{R}^N$, les coordonnées de $\Phi(x)$ étant tous les nombres de la forme $x_{i_1} \dots x_{i_d}$ où $1 \leq i_1 \leq \dots i_d \leq n$.
- $X = \mathbb{R}^n, k(x, y) = \left(1 + \sum_{i=1}^n x_i y_i\right)^d$.
- **Noyau gaussien :**

$$k(x, y) = exp\left(-\frac{||x - y||^2}{2\sigma^2}\right)$$

Espaces de Hilbert

Définitions : étant donné un espace vectoriel réel $\mathcal{H}$

- un *produit scalaire* sur $\mathcal{H}$ est une application bilinéaire symétrique définie positive $\langle , , , \rangle \to \mathbb{R}$,
- la *norme* de x est alors définie par $||x|| = \sqrt{\langle x, x \rangle}$,
- une suite $(x_n)_{n \in \mathbb{N}}$ de $\mathcal{H}$ est une *suite de Cauchy* si pour tout $\epsilon > 0$ il existe N tel que pour tout $m, n \geq N$, $||x_m - x_n|| < \epsilon$,
- un espace vectoriel $\mathcal{H}$ muni d'un produit scalaire est *complet* si toute suite de Cauchy est convergente dans $\mathcal{H}$,
- un *espace de Hilbert* est un espace vectoriel muni d'un produit scalaire et complet.

Exemples d'espaces de Hilbert :

- $\mathbb{R}^n$ pour tout entier n,
- l_2 : espace des suites $(x_n)_{n\in\mathbb{N}}$ de nombres réelles telles que $\sum_n x_n^2$ est une série convergente, muni du produit scalaire $\langle (x_n)_{n\in\mathbb{N}}, (y_n)_{n\in\mathbb{N}} \rangle = \sum_n x_n y_n$.

A retenir : un espace de Hilbert est un espace euclidien de dimension éventuellement infinie, c.-à-d. possèdant une base orthonormale qui peut comporter un nombre dénombrable de vecteurs.

Caractérisation des noyaux

Théorème : soit X un domaine quelconque et soit $k : X \times X \to \mathbb{R}$. k est un noyau ssi pour tout m-uplet $x_1, \ldots, x_m$ d'éléments de X, la matrice de Gram $k(x_i, x_j)_{1 \leq i,j \leq m}$ est définie positive, c'est-à-dire que pour tous réels $c_1, \ldots, c_m$,

$$\sum_{i,j} c_i c_j k(x_i, x_j) \geq 0.$$

A retenir : on sait (théoriquement) caractériser les fonctions noyaux et déterminer un plongement correspondant.

Kernel trick

Chaque fois que l'on dispose d'un algorithme formulé en terme de produits scalaires, on peut construire un autre algorithme en remplaçant ce produit scalaire par une fonction noyau k.

Remarque : le plongement est virtuel, il ne sert qu'à légitimer les calculs qui sont effectuées sur les données initiales dans leur espace d'origine.

Application : les SVMs

étant donné un noyau k *bien choisi*, déterminer le classifieur linéaire correspondant à l'hyperplan optimal (avec *soft margin*) dans l'espace de plongement.

Exercices

1. Montrer que l'application $k(x, y) = x_1^2 y_1^2 + x_2^2 y_2^2 + 2x_1 y_1 x_2 y_2$ est un noyau de $\mathbb{R}^2$. Montrer que ce noyau permet de séparer les points intérieurs à l'ellipse d'équation $x_1^2 + 2x_2^2 = 1$ des points qui lui sont extérieurs.

2. Montrer que l'application $k(x, y) = x_1 y_1 + x_2 y_2 + x_1 y_1 x_2 y_2$ est un noyau de $\mathbb{R}^2$ qui permet de séparer les points $\{((0,0),0),((0,1),1),((1,0),1),((1,1),0)\}$. Quel est l'équation de l'hyperplan optimal dans l'espace de plongement ? A quelle courbe correspond-il dans l'espace initial ?

Récapitulation

Etant donné un échantillon
$S = \{(x_1, y_1), \ldots, (x_l, y_l)\} \subset X \times \{-1, 1\}$, on souhaite résoudre le problème d'optimisation

$$\begin{cases} \text{Minimiser } ||w||^2/2 + C \sum_{i=1}^{l} \xi_i \text{ (1-norm soft margin)} \\ \text{sous les contraintes } y_i(< w, \phi(x_i)) > +b) \geq 1 - \xi_i \text{ et } \xi_i \geq 0 \text{ pour } i = 1, \ldots, l. \end{cases}$$

dans l'espace de plongement. Il est équivalent au problème d'optimisation dual :

$$\begin{cases} \text{Maximiser } W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j k(x_i, x_j) \\ \text{sous les contraintes } \sum_{i=1}^{l} \alpha_i y_i = 0 \text{ et } C \geq \alpha_i \geq 0 \end{cases}$$

pour une fonction noyau k et une valeur de C bien choisies ($C = \infty \equiv$ hard-margin).

Propriétés

- La fonction associée à la solution est

$$f(x) = \sum_{i=1}^{l} y_i \alpha_i^* k(x_i, x) + b^*$$

- La solution vérifie les (Karush-Kuhn-Tucker) conditions :

$$\alpha_i^* [y_i f(x_i) - 1 + \xi_i^*] = 0 \text{ et } \xi_i^* (\alpha_i^* - C) = 0$$

- Les exemples tq $\alpha_i^* \neq 0$ sont ceux pour lesquels $y_i f(x_i) \leq 1$. Ce sont les *vecteurs supports*.

  Si $0 < \alpha_i^* < C$, l'exemple (x_i, y_i) vérifie $y_i f(x_i) = 1$.

  Si $\alpha_i^* = C$, l'exemple (x_i, y_i) est bien classé mais avec une marge insuffisante, ou mal classé : on doit avoir $\xi_i^* > 0$.

Propriétés (suite)

- Pour calculer b^* connaissant α^* : utiliser les exemples pour lesquels $0 < \alpha_i^* < C$. On a $y_i f(x_i) = 1$.
- Pour calculer ξ^* connaissant α^* : si $\alpha_i < C$ alors $\xi_i = 0$; sinon, on doit avoir $y_i f(x_i) - 1 + \xi_i^* = 0$.
- La différence des fonctions à optimiser (*duality gap*) $||w||^2/2 + C \sum_{i=1}^{l} \xi_i - W(\alpha)$ est ≥ 0 et nulle au point solution uniquement, où $||w||^2 = \sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i; x_j)$.
- Soit VS l'ensemble des vecteurs supports de S. Les problèmes d'optimisation associés à S et à VS ont la même solution.
- Le rapport $\#SV/\#S$ est une borne supérieure de l'estimateur *leave-one-out* de l'erreur en généralisation de la solution.

Comment déterminer C ?

Méthode classique pour fixer un ou des paramètres d'un algorithme d'apprentissage : par *validation croisée*.

Soit $\mathcal{C}$ un ensemble fini de valeurs possible pour C. L'échantillon de travail S est partitionné aléatoirement en k (par exemple $k = 10$) parties approximativement de même cardinal : $S = S_1 \cup \ldots S_k$ et $i \neq j \Rightarrow S_i \cap S_j \neq 0$. On peut veiller à ce que la répartition des classes soit équilibrée dans chaque partie (échantillonnage stratifié). Soit $\overline{S}_i = S \setminus S_i$ pour $i = 1; \ldots, k$.

- Pour chaque $c \in \mathcal{C}$, pour chaque $i = 1, \ldots, k$, entraîner le classifieur sur $\overline{S}_i$ et calculer son erreur e_i^c sur l'échantillon test S_i.
- Choisir la valeur c pour laquelle $\sum_{i=1}^{k} e_i^c$ est minimale.

Entraîner le classifieur sur l'échantillon S en utilisant cette valeur du paramètre.

Comment déterminer C (suite)

Inconvénients de la méthode

- Utiliser le même ensemble pour déterminer le paramètre et pour apprendre peut entraîner de l'*overfitting*.
- Les paramètres optimaux pour l données ou $\frac{9}{10}l$ données peuvent ne pas coïncider.
- Il peut même y avoir des phénomènes de *transitions de phase* : saut brusque de l'erreur de généralisation au voisinage de certaines valeurs.
- *Having said all this, practitioners often do not care about these theoretical precautions, and use the unchanged parameters with excellent results.* (Schölkopf, Smola)

Techniques d'implémentation (généralités)

- *Stratégie générale :* à partir d'un point α qui satisfait les contraintes, accroître itérativement la valeur de la fonction $W(\alpha)$ sans violer les contraintes, jusqu'à ce qu'un critère d'arrêt soit satisfait.

- *Heuristique :* ne travailler que sur de petits sous ensembles de données à chaque étape. Permet d'éviter le chargement de la matrice de Gram $K(x_i, x_j)$ dans son intégralité.

- *Critères d'arrêt :*
  - lorsque $W(\alpha)$ semble se stabiliser (pas très bon),
  - lorsque le *duality gap* est proche de 0,
  - ...

Sequential Minimal Optimization (SMO)

- Ne considérer que deux points à chaque itération. Avantage : existence d'une solution analytique dans ce cas.
- Le choix des deux points (points actifs) se fait selon une heuristique.
- Cette méthode ne nécessite pas le stockage en mémoire la matrice de Gram.
- De très nombreuses itérations mais des performances globales excellentes.

SMO : solution analytique pour deux points

Soient α_1 et α_2 les points choisis, α_1^{old} et α_2^{old} les valeurs courantes, α_1^{new} et α_2^{new} les valeurs à déterminer.

Comme $\sum_{i=1}^{l} y_i \alpha_i = 0$, on doit avoir $y_1 \alpha_1^{new} + y_2 \alpha_2^{new}$ $= y_1 \alpha_1^{old} + y_2 \alpha_2^{old} = Cte$: α_1^{new} est donc déterminé par α_2^{new}.

- Si $y_1 \neq y_2$, soient

$$U = max(0, \alpha_2^{old} - \alpha_1^{old}) \text{ et } V = min(C, C - \alpha_1^{old} + \alpha_2^{old})$$

- Si $y_1 = y_2$, soient

$$U = max(0, \alpha_2^{old} + \alpha_1^{old} - C) \text{ et } V = min(C, \alpha_1^{old} + \alpha_2^{old})$$

Pour que $0 \leq \alpha_1^{new}, \alpha_2^{new} \leq C$, on doit avoir $U \leq \alpha_2^{new} \leq V$.

Si $C = \infty$, on pose $U = max(0, \alpha_2^{old} - \alpha_1^{old})$ si $y_1 \neq y_2$ (pas de V) ; $U = 0$ et $V = \alpha_1^{old} + \alpha_2^{old}$ si $y_1 = y_2$.

SMO : solution analytique pour deux points (suite)

Soit $E_i = f(x_i) - y_i = \left(\sum_{j=1}^{l} \alpha_j y_j k(x_i, x_j) + b\right) - y_i, i = 1, 2$ et soit

$$\kappa = k(x_1, x_1) + k(x_2, x_2) - 2k(x_1, x_2) = ||\phi(x_1) - \phi(x_2)||^2$$

où ϕ est la fonction de plongement dans l'espace associé au noyau.

Théorème : lorsque seuls α_1 et α_2 sont variables, la fonction W est maximale pour les valeurs de α_1 et α_2 définies par

$$\alpha_2^{new,unc} = \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\kappa}$$

$$\alpha_2^{new} = \begin{cases} V \text{ si } \alpha_2^{new,unc} > V \\ U \text{ si } \alpha_2^{new,unc} < U \\ \alpha_2^{new,unc} \text{ sinon.} \end{cases},$$

$$\alpha_1^{new} = \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha_2^{new}).$$

SMO : heuristiques pour le choix des points

Idée : choisir des points dont la mise à jour est importante.

Choix du premier point x_1 : un point parmi ceux qui violent les conditions de KKT.

Choix du second point : maximiser $|E_1 - E_2|$.

Choix du biais b : il n'intervient pas dans le calcul de $|E_1 - E_2|$. Prendre $b = 0$ jusqu'à convergence.

Exercice : Soit $z_1 = ((0,0), -1), z_2 = ((0,1), 1), z_3 = ((1,2), 1)$ et $S = \{z_1, z_2, z_3\}$. Appliquer l'algorithme SMO en considérant les points z_1, z_2,puis z_1, z_3 puis z_2, z_3.

Classification multi-classe : $Y = \{1, \ldots, M\}$.

Méthode 1 : un contre tous.

Soit $S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ l'échantillon de travail. Pour tout $i = 1, \ldots, M$, on considère l'échantillon
$S^j = \{(x_1, \phi^j(y_1)), \ldots, (x_l, \phi^j(y_l))\}$ où $\phi^j(y) = 1$ si $y = j$ et -1 sinon.

Soit $g^j(x) = \sum_{i=1}^{l} y_i \alpha_i^j k(x, x_i) + b^j$ la fonction linéaire définissant le SVM calculé à partir de l'échantillon S^j.

On pose

$$f(x) = argmax_{j=1,\ldots,M} g^j(x).$$

Inconvénient : il n'est pas clair si les fonctions réelles $g^j(x)$ sont à la même échelle ! Que faire en particulier si plusieurs classes sont choisies avec une marge importante ou si au contraire, toutes les classes sont rejetées !

Classification multi-classe : $Y = \{1, \ldots, M\}$ (suite).

Méthode 2 : comparaisons par paires.

Soit $S = \{(x_1, y_1), \ldots, (x_l, y_l)\}$ l'échantillon de travail. Pour tout $i < j \in \{1, \ldots, M\}$, on considère l'échantillon

$$S^{i,j} = \{(x, \phi^{i,j}(y)) | (x, y) \in S, y \in \{i, j\}\}$$

où $\phi^{i,j}(y) = 1$ si $y = i$ et -1 sinon.

Le classifieur f est obtenu par un vote des $M(M-1)/2$ classifieurs entraînés sur les échantillons $S^{i,j}$.

Il existe d'autres méthodes plus sophistiquées.

The *US Postal Service (USPS)* database.

- 9298 chiffres manuscrits (7291 pour apprendre, 2007 pour tester) provenant d'enveloppes postées ou reçues à Buffalo ;
- Chaque chiffre est une image 16×16 représenté par un vecteur de $[-1, 1]^{256}$;
- Les jeu de tests est difficile : $2,5\%$ d'erreur en moyenne pour un humain ;
- Données disponibles à

  http://www.kernel-machines.org.data.html.

Résultats (1) (Schölkopf, Smola)

Classification par SVMs sur les données USPS.

Noyau polynomial : $k(x, y) = (\langle x, y \rangle / 256)^d$.

Construction de 10 classifieurs (*méthode un contre tous*).

d	1	2	3	4	5	6	7
erreur %	8,9	4,7	4,0	4,2	4,5	4,5	4,7
Nb. moyen de VSs	282	237	274	321	374	422	491

Résultats (2) (Schölkopf, Smola)

Classification par SVMs sur les données USPS.

Noyau gaussien : $k(x, y) = exp(-||x - y||^2/(256c))$.

Construction de 10 classifieurs (*méthode un contre tous*).

c	4,0	2,0	1,2	0,8	0,5	0,2	0,1
erreur %	5,3	5,0	4,9	4,3	4,4	4,4	4,5
Nb. moyen de VSs	266	240	233	235	251	366	722

Résultats (3) (Schölkopf, Smola)

Classification par SVMs sur les données USPS.

Tangente hyperbolique : $k(x, y) = tanh(2\langle x, y\rangle/256 + \theta)$.

Construction de 10 classifieurs (*méthode un contre tous*).

$-\theta$	0,8	0,9	1,0	1,1	1,2	1,3	1,4
erreur %	6,3	4,8	4,1	4,3	4,3	4,4	4,8
Nb. moyen de VSs	206	242	254	267	278	289	296

Comparaison des résultats (Schölkopf, Smola)

USPS+ : USPS + un ensemble de chiffres imprimés.

Classifieur	Ens. d'App.	Erreur sur ens. test
Decision tree, C4.5	USPS	16,2
Best two layer neural network	USPS	5,9
Five-layer network	USPS	5,1
Linear SVM	USPS	8,9
Hard margin SVM	USPS	4,6
SVM	USPS	4,0
Virtual SVM	USPS	3,2
Virtual SV, local kernel	USPS	3,0
Nearest neighbor	USPS+	5,9
Boosted neural net	USPS+	2,6
Human performance		2,5

Universalité des vecteurs supports

Pourcentage des VS des colonnes contenus dans l'ensemble des VS des lignes (10 classifieurs).

	Polynomial	RBF	Sigmoid
Polynomial	100	93	94
RBF	83	100	87
Sigmoid	90	93	100

Pourcentage des VS des colonnes contenus dans l'ensemble des VS des lignes (classifieur binaire pour le chiffre 7).

	Polynomial	RBF	Sigmoid
Polynomial	100	84	93
RBF	89	100	92
Sigmoid	93	86	100

Universalité des vecteurs supports (suite)

Entraîner un classifieur sur les VS d'autres classifieurs conduit à des résultats équivalents à ceux obtenus en utilisant l'ensemble des données.

Nombre d'erreurs sur l'échantillon test (de cardinal 2007) pour des classifieurs binaires séparant le 7 des autres chiffres. A comparer aux résultats obtenus sur un sous ensemble aléatoire de taille 200.

trained on:	poly-SVs	RBF-SVs	Tanh-SVs	full db	rnd. subset
size	178	189	177	7291	200
Polynomial	13	13	12	13	23
RBF	17	13	17	15	27
Sigmoid	15	13	13	15	25