

# TECHNIQUES AVANCÉES POUR L'APPRENTISSAGE — COURS 2

CES DATA SCIENTIST, TÉLÉCOM PARISTECH

---

**Aurélien Bellet**

Inria Lille

March 12, 2016

1. Le perceptron revisité
2. Support Vector Machines linéaires
3. Cas non linéaire et noyaux
4. Support Vector Regression

## LE PERCEPTRON REVISITÉ

---

# APPRENTISSAGE SUPERVISÉ : CADRE PROBABILISTE ET STATISTIQUE

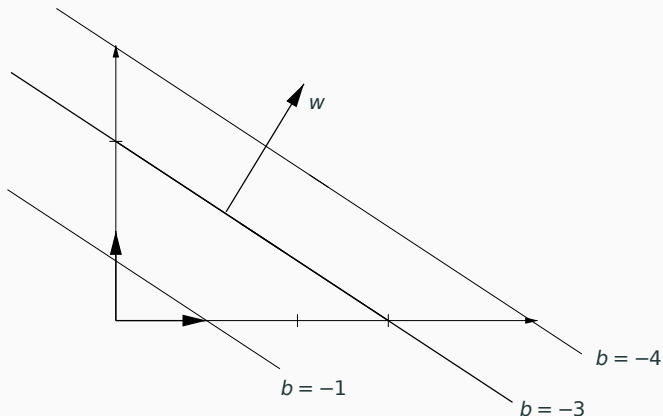
- $X$  : variable explicative, vecteur aléatoire dans  $\mathcal{X} = \mathbb{R}^p$
- $Y$  : variable à prédire, aléatoire dans  $\mathcal{Y} = \{1, \dots, C\}$  (classification) ou  $\mathcal{Y} = \mathbb{R}$  (régression)
- $P$  : loi de probabilité jointe de  $(X, Y)$ , fixée mais **inconnue**
- $\mathcal{S} = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^n$  : échantillon i.i.d. tiré selon la loi  $P$
- $\mathcal{H}$  : collection de classifieurs / modèles,  $h \in \mathcal{H}$
- $L$  : perte mesurant les erreurs d'un classifieur / modèle
  - Exemple (classification) :  $L(y, h(x)) = \begin{cases} 1 & \text{si } h(x) \neq y \\ 0 & \text{sinon} \end{cases}$
  - Exemple (régression) :  $L(y, h(x)) = (y - h(x))^2$
- **Objectif** : déterminer à partir de  $\mathcal{S}$  la fonction  $h \in \mathcal{H}$  qui minimise  $R(h) = \mathbb{E}_{(X,Y) \sim P}[L(Y, h(X))]$

- Classification binaire :  $\mathcal{Y} = \{-1, 1\}$
- On choisit  $\mathcal{H}$  comme étant la classe des **séparateurs linéaires** dans  $\mathcal{X} = \mathbb{R}^p$
- Un séparateur linéaire  $h \in \mathcal{H}$  est défini par un **vecteur de poids**  $w \in \mathbb{R}^p$  et un **biais**  $b \in \mathbb{R}$  :

$$\begin{aligned}h(x) &= \text{sign}(w^T x + b) \\ &= \text{sign}\left(\sum_{i=1}^p w^i x^i + b\right)\end{aligned}$$

# CLASSIFIEURS LINÉAIRES

- $h \in \mathcal{H}$  définit un hyperplan d'équation  $w^T x + b = 0$  séparant  $\mathbb{R}^p$  en deux régions
- Exemple avec  $w = [1, 3/2]^T$  et différentes valeurs de  $b$  :



- On a  $p + 1$  paramètres à apprendre ( $w$  et  $b$ )
- Un algorithme classique : le **perceptron** de Rosenblatt

## Algorithme du perceptron

**entrée :** Ensemble d'apprentissage  $\mathcal{S}$

Initialiser  $w = [0, \dots, 0]$ ,  $b = 0$ ,  $R = \max_{1 \leq i \leq n} \|x_i\|_2$

**tant que** il y a au moins une erreur **faire**

**pour**  $i = 1, 2, \dots, n$  **faire**

**si**  $y_i(w^T x_i + b) \leq 0$  **alors**

$w = w + y_i x_i$

$b = b + y_i R^2$

**fin si**

**fin pour**

**fin tant que**

retourner  $(w, b)$

- L'algorithme s'arrête si  $\mathcal{S}$  est **linéairement séparable**

- **Marge de l'observation**  $(x_i, y_i)$  par rapport à un séparateur linéaire

$$\gamma_i = y_i(w^T x_i + b)$$

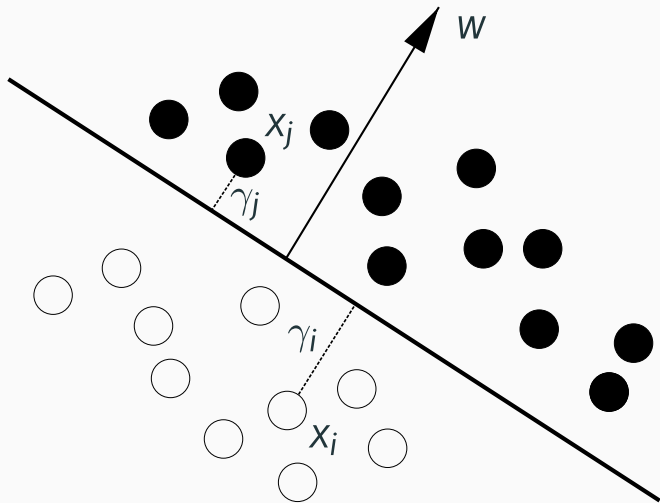
- Si  $\gamma_i > 0$ ,  $(x_i, y_i)$  est bien classifié
- **Marge du séparateur** par rapport à un ensemble  $\mathcal{S}$

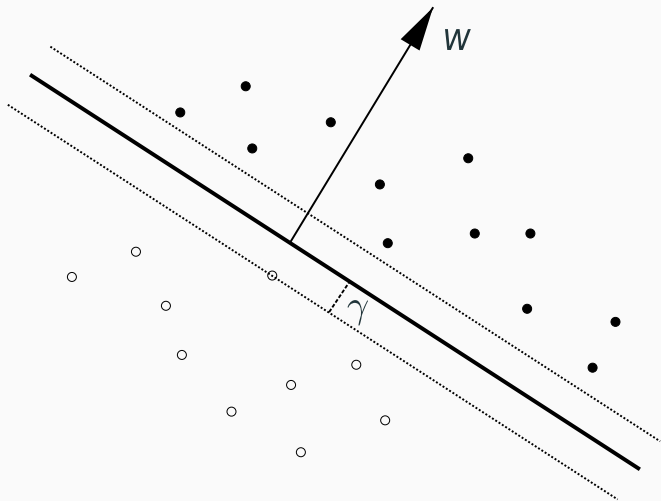
$$\gamma = \min_{1 \leq i \leq n} \gamma_i$$

- Les marges correspondent aux **marges géométriques** (distance Euclidienne entre les observations et le séparateur) quand on considère le séparateur normalisé

$$w = \frac{1}{\|w\|_2} w, \quad b = \frac{1}{\|w\|_2} b$$







## Convergence de l'algorithme du perceptron

Soit  $R = \max_{1 \leq i \leq n} \|x_i\|_2$ . S'il existe un hyperplan  $(w, b)$  tel que

$$\|w\| = 1,$$

$$y_i(w^T x_i + b) \geq \gamma \quad \forall i \in \{1, \dots, n\},$$

alors l'algorithme du perceptron fait au plus  $(2R/\gamma)^2$  erreurs.

- L'algorithme du perceptron forme  $w$  en ajoutant (resp. retirant) itérativement du vecteur initial les observations positives (resp. négatives) mal classées
- On a ainsi une représentation dans un autre espace (dit **dual**) du classifieur appris

$$h(x) = \text{sign} \left( \underbrace{\sum_{i=1}^n \alpha_i y_i x_i^T}_w x + b \right)$$

- $\alpha_i$  est positif et **proportionnel au nombre de fois où  $x_i$  a été mal classifié** → les exemples “difficiles” ont un  $\alpha_i$  élevé

- Algorithme équivalent apprenant  $\alpha$  directement

## Algorithme dual du perceptron

entrée : Ensemble d'apprentissage  $\mathcal{S}$

Initialiser  $\alpha = [0, \dots, 0]$ ,  $b = 0$ ,  $R = \max_{1 \leq i \leq n} \|x_i\|_2$

**tant que** il y a au moins une erreur **faire**

**pour**  $i = 1, 2, \dots, n$  **faire**

**si**  $y_i(\sum_{j=1}^n \alpha_j y_j x_j^T x_i + b) \leq 0$  **alors**

$\alpha_i = \alpha_i + 1$

$b = b + y_i R^2$

**fin si**

**fin pour**

**fin tant que**

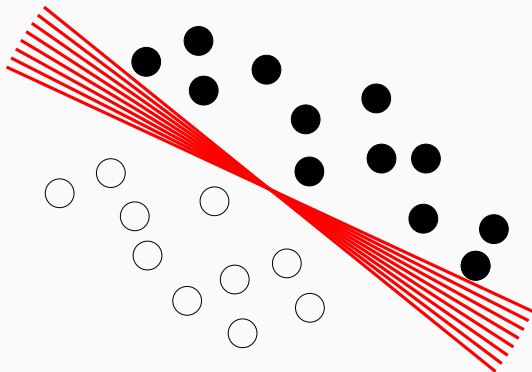
retourner  $(w, b)$

# SVM LINÉAIRES

---

## CHOIX DU SÉPARATEUR LINÉAIRE

- Pour l'instant, supposons que  $\mathcal{S}$  est linéairement séparable
- Quel hyperplan séparateur choisir ?



- Principe : fixer la marge fonctionnelle à 1 et maximiser la marge géométrique
- Une marge fonctionnelle à 1 pour un exemple positif  $x^+$  et un exemple négatif  $x^-$  signifie

$$\begin{cases} w^T x^+ + b = 1 \\ -(w^T x^- + b) = 1 \end{cases} \iff \begin{cases} w^T x^+ = 1 - b \\ w^T x^- = -b - 1 \end{cases}$$

- La marge géométrique est alors

$$\begin{cases} \gamma = \frac{1}{\|w\|_2} w^T x^+ + \frac{1}{\|w\|_2} b \\ \gamma = -\frac{1}{\|w\|_2} w^T x^- - \frac{1}{\|w\|_2} b \end{cases} \implies \gamma = \frac{1}{\|w\|_2}$$

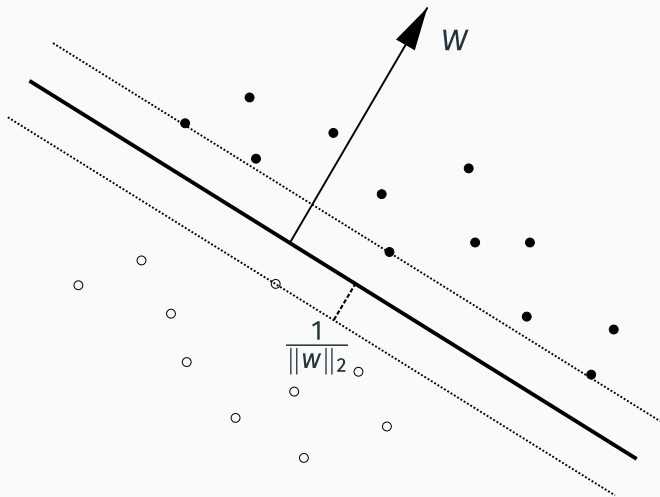


## Formulation primale du SVM linéaire (cas séparable)

$$\begin{aligned} \min_{w \in \mathbb{R}^p, b \in \mathbb{R}} \quad & \frac{1}{2} \|w\|_2^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 \quad \forall i \in \{1, \dots, n\} \end{aligned}$$

- Assure une **solution unique** : il s'agit du séparateur maximisant la marge géométrique

## SVM LINÉAIRE : CAS SÉPARABLE

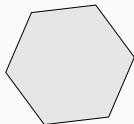


## Définition : ensemble convexe

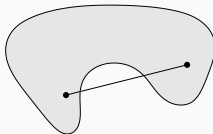
Un ensemble  $C$  est *convexe* si il contient le segment reliant tout couple de points de  $C$  :

$$x_1, x_2 \in C, \quad 0 \leq \alpha \leq 1 \quad \Rightarrow \quad \alpha x_1 + (1 - \alpha)x_2 \in C$$

Convexe



Non convexe

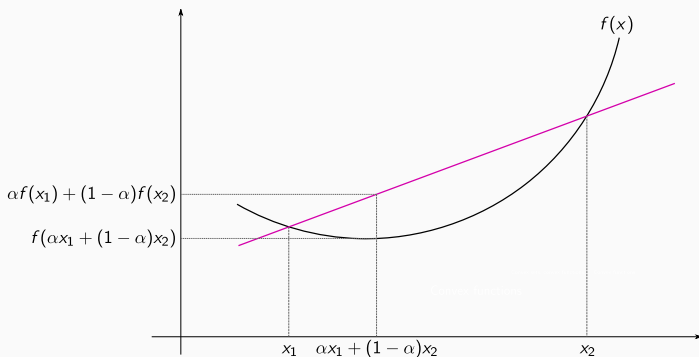


# INTERLUDE : CONVEXITÉ

## Définition : fonction convexe

La fonction  $f : \mathbb{R}^p \rightarrow \mathbb{R}$  est *convexe* si

$$x_1, x_2 \in \mathbb{R}^p, \quad 0 \leq \alpha \leq 1 \quad \implies \quad f(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha f(x_1) + (1 - \alpha)f(x_2)$$



## Formulation primale du SVM linéaire (cas séparable)

$$\begin{aligned} \min_{w \in \mathbb{R}^p, b \in \mathbb{R}} \quad & \frac{1}{2} \|w\|_2^2 \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 \quad \forall i \in \{1, \dots, n\} \end{aligned}$$

- La **fonction objective** est quadratique et convexe en  $w$
- Les **contraintes** sont linéaires en  $w$  et  $b$
- C'est ce que l'on appelle un **problème d'optimisation quadratique convexe**
- Il existe beaucoup d'algorithmes d'optimisation numérique permettant de résoudre efficacement cette classe de problème

- **Lagrangien** du problème convexe  $\min_x f(x)$  s.t.  $g(x) \leq 0$

$$L(x, \alpha) = f(x) + \alpha g(x), \quad \alpha \geq 0$$

- **Problème dual** équivalent au primal

$$\max_{\alpha \geq 0} \inf_x L(x, \alpha)$$

- Dans notre cas, on a

$$L(w, b, \alpha) = \frac{1}{2} \|w\|_2^2 + \sum_{i=1}^n \alpha_i [1 - y_i(w^T x_i + b)], \quad \alpha_1, \dots, \alpha_n \geq 0$$

### Conditions d'optimalité de Karush-Kuhn-Tucker (KKT)

Au point extremum, on a

1.  $\frac{\partial L(w,b,\alpha)}{\partial w} = w - \sum_{i=1}^n \ell_i \alpha_i x_i = 0$
2.  $\frac{\partial L(w,b,\alpha)}{\partial b} = \sum_{i=1}^n \ell_i \alpha_i = 0$
3.  $\alpha_i \geq 0, \quad \forall i \in \{1, \dots, n\}$
4.  $\alpha_i [1 - y_i(w^T x_i + b)] = 0, \quad \forall i \in \{1, \dots, n\}$

- On obtient le problème dual en remplaçant la valeur de  $w$  donnée par (KKT 1) dans le Lagrangien  $L$ , en développant puis en utilisant (KKT 2)

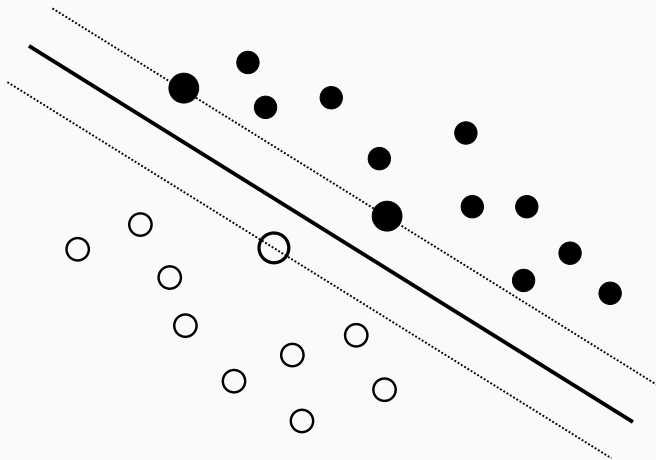
## Formulation duale du SVM linéaire (cas séparable)

$$\begin{aligned} \max_{\alpha \in \mathbb{R}_+^n} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{s.t.} \quad & \sum_{i=1}^n \alpha_i y_i = 0 \end{aligned}$$

- C'est encore un problème quadratique convexe
- Le vecteur de poids optimal a la forme  $w = \sum_{i=1}^n \alpha_i y_i x_i$  (KKT 1)
- Pour  $1 \leq i \leq n$ , on remarque par (KKT 4)
  - Cas 1 :  $y_i(w^T x_i + b) > 1 \Rightarrow x_i$  n'est pas sur la marge et  $\alpha_i = 0$
  - Cas 2 :  $y_i(w^T x_i + b) = 1 \Rightarrow \alpha_i \neq 0$  et on en déduit  $b$
- Le  $w$  optimal est défini par les points sur la marge : ce sont les fameux **vecteurs supports**



## SVM LINÉAIRE : CAS SÉPARABLE



## SVM LINÉAIRE : CAS NON SÉPARABLE

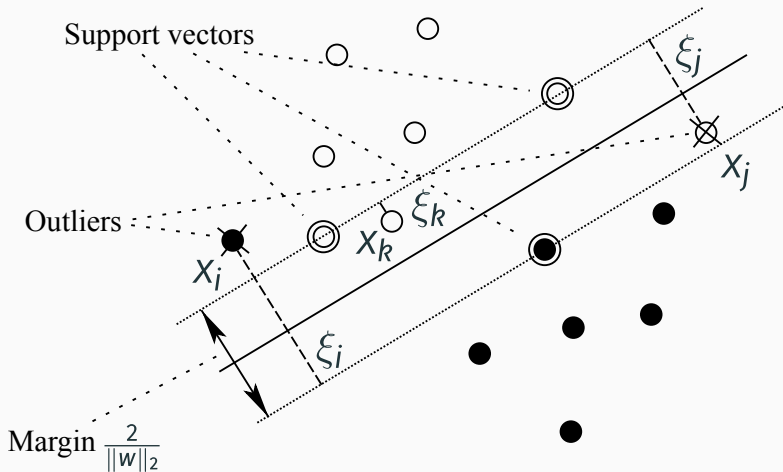
- Comment traiter le cas non séparable ?
- Autoriser la violation des contraintes de marge, mais infliger une pénalité quand cela arrive

### Formulation primale du SVM linéaire (cas non séparable)

$$\begin{aligned} \min_{w \in \mathbb{R}^p, b \in \mathbb{R}, \xi \in \mathbb{R}^n} \quad & \frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^n \xi_i \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 - \xi_i \quad \forall i \in \{1, \dots, n\} \\ & \xi_i \geq 0 \quad \forall i \in \{1, \dots, n\} \end{aligned}$$

- L'hyperparamètre  $C$  permet de régler le compromis entre maximiser la marge (régularisation) et minimiser les violations
- $C = +\infty$  permet de retrouver la formulation des marges strictes

## SVM LINÉAIRE : CAS NON SÉPARABLE

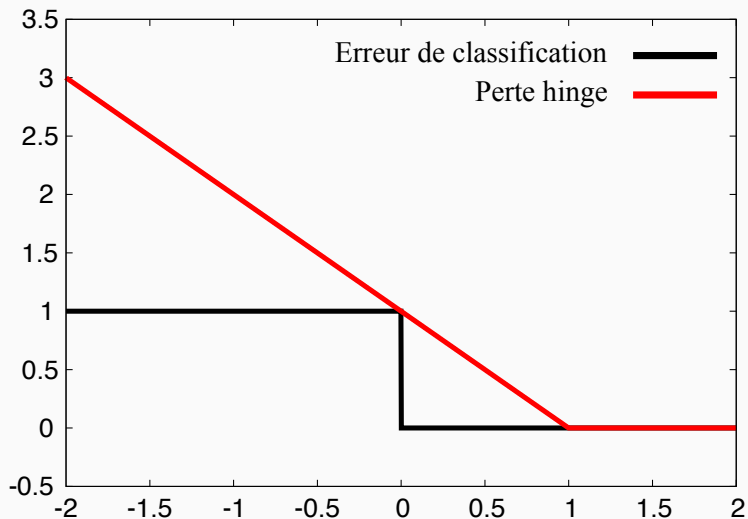


### Formulation primale équivalente du SVM linéaire (non séparable)

$$\min_{w \in \mathbb{R}^p, b \in \mathbb{R}} \quad \frac{1}{2} \|w\|_2^2 \quad + \quad c \sum_{i=1}^n [y_i(w^T x_i + b)]_+$$

- La fonction  $[a]_+ = \max(0, 1 - a)$  est la **perte hinge**
- Note : cette formulation sans contrainte se prête à un algorithme d'optimisation de type **gradient stochastique** adapté au cas où  $n$  est grand

## PERTE HINGE



# FORMULATION DUALE (CAS NON SÉPARABLE)

## Formulation duale du SVM linéaire (cas non séparable)

$$\begin{aligned} \max_{\alpha \in \mathbb{R}^n} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{s.t.} \quad & \sum_{i=1}^n \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad \forall i \in \{1, \dots, n\} \end{aligned}$$

- La forme duale est presque la même que dans le cas séparable
- Intuition pour la nouvelle contrainte : elle empêche de mettre trop de poids sur un exemple difficile pour bien le classifier

## CAS NON LINÉAIRE ET NOYAUX

---

- En pratique, on a souvent affaire à des données qui ont une **structure non linéaire**
- Une solution serait de travailler avec une classe de modèles plus complexes
- Mais on perdra la **simplicité des modèles linéaires**
  - Simples et efficaces à entraîner
  - Prédiction rapide
  - Robustesse au sur-apprentissage
- Approche SVM : apprendre un classifieur linéaire dans un espace obtenu par projection non linéaire !



## EXEMPLE DE PROJECTION UTILE

- La loi de la gravitation universelle de Newton

$$f(m_1, m_2, r) = G \frac{m_1 m_2}{r^2}$$

où  $m_1, m_2$  sont les masses,  $r$  la distance entre le centre des masses, et  $G$  la constante de gravitation

- Cette fonction est non linéaire : on ne peut donc pas l'apprendre avec une machine linéaire
- Changement de coordonnées avec la projection non linéaire  $\phi$

$$\phi([m_1, m_2, r]) = [\log m_1, \log m_2, \log r]^T = [x, y, z]^T$$

- On a alors une fonction linéaire de  $x, y, z$

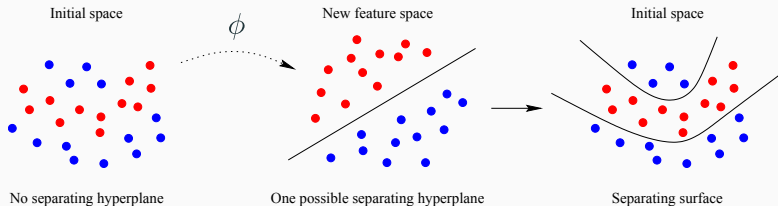
$$\begin{aligned} g(x, y, z) &= \log f(\phi([m_1, m_2, r])) \\ &= \log G + \log m_1 + \log m_2 - 2 \log r = C + x + y - 2z \end{aligned}$$

# SÉPARATEUR LINÉAIRE APRÈS PROJECTION NON LINÉAIRE

- Considérons le classifieur linéaire dans l'espace  $\mathcal{F}$  induit par la projection  $\phi : \mathcal{X} \rightarrow \mathcal{F}$

$$\begin{aligned}h(x) &= \text{sign}(w^T \phi(x) + b) \\ &= \text{sign} \left( \sum_{i=1}^{|\mathcal{F}|} w^i \phi^i(x) + b \right)\end{aligned}$$

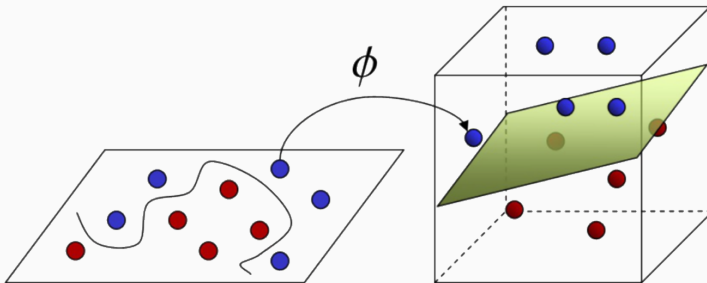
- Celui-ci correspond à un classifieur non linéaire dans  $\mathcal{X}$



- Un simple changement de coordonnées peut ne pas suffire pour rendre les données linéairement séparables
- On peut projeter les données dans un **espace de plus grande dimension**
- Par exemple, supposons que  $\mathcal{X} = \mathbb{R}^2$  mais que des connaissances sur notre problème nous disent que l'on peut l'apprendre parfaitement avec des monômes de degré 2
- C'est alors une bonne idée de travailler dans  $\mathcal{F} = \mathbb{R}^3$  plutôt que  $\mathcal{X} = \mathbb{R}^2$  en utilisant la transformation

$$\phi([x^1, x^2]) = (x^1x^1, x^2x^2, x^1x^2)$$

## AUGMENTER LA DIMENSION



# MALÉDICTION DE LA DIMENSION

- Et si on a besoin de davantage de dimensions, par exemple de monômes de plus grand degré ?
- Nombre de monômes de degré  $d$  à partir de  $p$  variables

$$\binom{p + d - 1}{d}$$

- $p = 5, d = 5 \rightarrow$  need 126 dimensions
- $p = 10, d = 5 \rightarrow$  need 11628 dimensions
- $p = 20, d = 10 \rightarrow$  need 20030010 dimensions
- **Malédiction de la dimension** : l'apprentissage devient très coûteux en temps de calcul si la dimension est trop grande

### Formulation duale du SVM linéaire (cas non séparable)

$$\begin{aligned} \max_{\alpha \in \mathbb{R}^n} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \\ \text{s.t.} \quad & \sum_{i=1}^n \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad \forall i \in \{1, \dots, n\} \end{aligned}$$

- Contrairement à la formulation primale, on n'a pas besoin de manipuler explicitement les  $\phi(x_i)$ , mais seulement les produits scalaires  $\phi(x_i)^T \phi(x_j)$  qui sont les entrées de la **matrice de Gram**  $G$

$$G_{i,j} = \phi(x_i)^T \phi(x_j)$$

- Cette observation est valable pour les autres formulations duales étudiées (perceptron, SVM pour le cas séparable)

## Definition (Fonction noyau)

Une fonction symétrique  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$  est un *noyau* si et seulement si il existe une projection  $\phi : \mathcal{X} \rightarrow \mathcal{H}$  de  $\mathcal{X}$  vers un espace de Hilbert  $\mathcal{H}$  telle que  $K$  s'écrit comme un produit scalaire dans  $\mathcal{H}$  :

$$K(x_1, x_2) = \phi(x_1)^T \phi(x_2), \quad \forall x_1, x_2 \in \mathcal{X}.$$

De manière équivalente,  $K$  est un *noyau* si et seulement si il est semi-défini positif (SDP), c'est-à-dire

$$\sum_{i=1}^n \sum_{j=1}^n c_i c_j K(x_i, x_j) \geq 0$$

pour tous les  $x_1, \dots, x_n \in \mathcal{X}$  et  $c_1, \dots, c_n \in \mathbb{R}$ .

- **Noyau linéaire** :  $K(x_1, x_2) = x_1^T x_2$ 
  - Équivalent au cas linéaire
- **Noyaux polynomiaux** :  $K(x_1, x_2) = (x_1^T x_2 + c)^d$  avec  $c \in \mathbb{R}, d \in \mathbb{N}$ 
  - $\phi(x)$  contient tous les monômes de degré au plus  $d$  (voir plus bas)
- **Noyau Gaussien** :  $K(x_1, x_2) = \exp\left(\frac{\|x_1 - x_2\|^2}{2\sigma^2}\right)$  avec  $\sigma^2 \geq 0$ 
  - $\phi(x)$  est de dimension infinie ! C'est le plus utilisé
- Note : il existe de nombreux noyaux pour les données structurées
  - Par exemple, le **noyau  $k$ -spectrum** :  $\phi(x)$  contient le nombre d'occurrences dans la chaîne de caractères  $x$  de chaque sous-séquence possible de taille  $k$



- On peut combiner plusieurs noyaux de manière à obtenir un nouveau noyau valide
- Par exemple, si  $K_1$  et  $K_2$  sont des noyaux alors les fonctions suivantes sont aussi des noyaux :
  - $K(x_1, x_2) = K_1(x_1, x_2) + K_2(x_1, x_2)$
  - $K(x_1, x_2) = aK_1(x_1, x_2)$  pour  $a \geq 0$
  - $K(x_1, x_2) = K_1(x_1, x_2)K_2(x_1, x_2)$
  - $K(x_1, x_2) = x_1^T B x_2$  pour  $B \in \mathbb{R}^{p \times p}$  semi-définie positive

## NOYAU POLYNOMIAL HOMOGÈNE DE DEGRÉ 2

- Considérons tout d'abord le noyau polynomial homogène de degré 2 :  $K(x_1, x_2) = (x_1^T x_2)^2$

$$\begin{aligned} K(x_1, x_2) &= \left( \sum_{i=1}^p x_1^i x_2^i \right)^2 = \left( \sum_{i=1}^p x_1^i x_2^i \right) \left( \sum_{j=1}^p x_1^j x_2^j \right) \\ &= \sum_{i,j=1}^p x_1^i x_2^i x_1^j x_2^j = \sum_{i,j=1}^p (x_1^i x_1^j) (x_2^i x_2^j) \end{aligned}$$

- $K$  est donc un noyau avec  $\phi(x) = [x^i x^j]_{i,j=1}^p$ , un vecteur de dimension  $\binom{p+1}{2}$  contenant tous les monômes de degré 2

## NOYAU POLYNOMIAL NON-HOMOGENÈNE DE DEGRÉ 2

- Pour le noyau polynomial non-homogène de degré 2 :

$$K(x_1, x_2) = (x_1^T x_2 + c)^2$$

$$\begin{aligned} K(x_1, x_2) &= \left( \sum_{i=1}^p x_1^i x_2^i + c \right) \left( \sum_{j=1}^p x_1^j x_2^j + c \right) \\ &= c^2 + \sum_{i,j=1}^p (x_1^i x_1^j)(x_2^i x_2^j) + \sum_{i=1}^p \sqrt{2c} x_1^i \sqrt{2c} x_2^i \end{aligned}$$

- K est donc un noyau avec  $\phi(x)$  correspondant au vecteur de dimension  $1 + \binom{p+1}{2} + p$  contenant tous les monômes de degré au plus 2, avec des poids contrôlés par c

- Le raisonnement peut être généralisé pour tout degré  $d \in \mathbb{N}$ 
  - $K(x_1, x_2) = (x_1^T x_2)^d$  : les attributs sont les  $\binom{d+p-1}{d}$  monômes de degré  $d$
  - $K(x_1, x_2) = (x_1^T x_2 + c)^d$  : les attributs sont les  $\binom{d+p}{d}$  monômes de degré au plus  $d$
- Note : on peut aussi montrer la validité de ces noyaux par construction

## L'ASTUCE DU NOYAU

- L'astuce du noyau (**kernel trick** en anglais) consiste à remplacer les  $x_i^T x_j$  dans les formulations duales par  $\phi(x)_i^T \phi(x)_j = K(x_i, x_j)$
- On réécrit la forme du classifieur linéaire dans l'espace dual

$$\begin{aligned} h(x) &= \text{sign} \left( \underbrace{\sum_{i=1}^n \alpha_i y_i \phi(x_i)^T \phi(x)}_w + b \right) \\ &= \text{sign} \left( \sum_{i=1}^n \alpha_i y_i K(x_i, x) + b \right) \end{aligned}$$

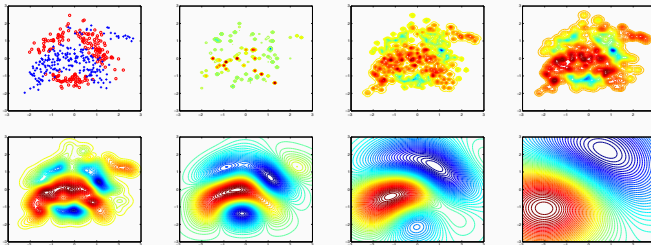
- Avantages
  - Pas besoin de représenter  $\phi(x)$  explicitement
  - Pas d'influence de la dimension de  $\phi(x)$  sur le nombre de paramètres à apprendre (on apprend  $n$  paramètres)
  - Pas d'influence de la dimension de  $\phi(x)$  sur l'évaluation de  $h$  (on a besoin au plus de  $n$  évaluations de la fonction noyau)
  - En fait, **pas besoin de connaître  $\phi$**  : la projection peut être implicite

# L'ASTUCE DU NOYAU : VISUALISATION

- Pour le noyau polynomial, voir par exemple

<http://www.youtube.com/watch?v=3liCbRZPrZA>

- Pour le noyau Gaussien et  $\sigma \in [0.01, 0.05, 0.1, 0.2, 0.5, 1, 2]$



- Démo graphique de LibSVM

<https://www.csie.ntu.edu.tw/~cjlin/libsvm/>

- Le choix du noyau a une influence primordiale sur la performance du modèle
- Quelques règles générales :
  - Pour des données en **grande dimension**, un noyau linéaire est souvent suffisant
  - Pour des données en **grande dimension et creuses** (ex : sacs de mots en texte), essayer le noyau polynomial
  - Le noyau Gaussien donne généralement de très bonnes performances
- Règle d'or : utiliser les principes de **sélection de modèles** pour le choix du noyau et de ses paramètres !
- Multiple Kernel Learning (MKL) : permet d'apprendre une bonne combinaison de noyaux de base

## Theorem (Vapnik)

Soit  $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^n$  un échantillon i.i.d. tiré selon la loi  $P$ . Soit  $h$  le classifieur appris sur  $\mathcal{S}$  avec une marge géométrique  $\gamma$ . Pour  $i = 1, \dots, n$ , on note  $\xi_i = \max(0, \gamma - y_i h(x_i))$ . On suppose  $\|x\|_2 \leq R$  pour tout  $x$  tiré selon  $P$ . Alors on a avec probabilité au moins  $1 - \delta$  :

$$\underbrace{\mathbb{E}_{(X,Y) \sim P} [\mathbb{I}\{h(X) \neq Y\}]}_{\text{erreur en généralisation}} \leq \underbrace{\frac{1}{n} \sum_{i=1}^n \mathbb{I}\{\xi_i > 0\}}_{\text{\# violations dans } \mathcal{S}} + O\left(\sqrt{\frac{R^2 \log n + \log(1/\delta)}{\gamma^2 n}}\right).$$

- On voit que l'erreur en généralisation :
  - ne dépend pas du **nombre d'attributs**
  - dépend du **nombre d'exemple d'apprentissage**, de la **taille de la marge** et des **violations de marge** en apprentissage
- C'est donc une validation théorique des deux principes des SVMs : maximisation de marge + astuce du noyau



# SUPPORT VECTOR REGRESSION

---

## VASTE MARGE POUR LA RÉGRESSION

- Dans le cas de la régression, on a  $\mathcal{Y} = \mathbb{R}$
- On cherche un modèle linéaire : pour  $w \in \mathbb{R}^p, b \in \mathbb{R}$

$$h(x) = w^T x + b$$

- Fonctions de perte classique en régression
  - Moindre carrés :  $(h(x) - y)^2$
  - Valeur absolue :  $|h(x) - y|$
- Dans les SVMs pour la régression, on utilise la **perte  $\epsilon$ -insensible**

$$|h(x) - y|_{\epsilon} = \max(0, |h(x) - y| - \epsilon)$$

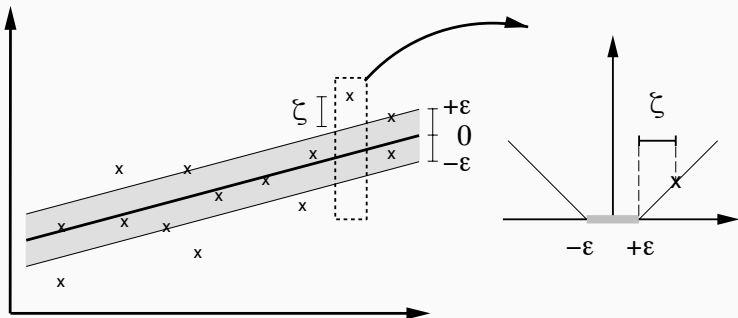
## Formulation primale du SVR linéaire

$$\begin{aligned}
 \min_{w \in \mathbb{R}^p, b \in \mathbb{R}, \xi \in \mathbb{R}^n, \xi' \in \mathbb{R}^n} \quad & \frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^n (\xi_i + \xi'_i) \\
 \text{s.t.} \quad & w^T x_i + b - y_i \leq \epsilon + \xi_i \quad \forall i \in \{1, \dots, n\} \\
 & y_i - w^T x_i + b \leq \epsilon + \xi'_i \quad \forall i \in \{1, \dots, n\} \\
 & \xi_i, \xi'_i \geq 0 \quad \forall i \in \{1, \dots, n\}
 \end{aligned}$$

## Formulation primale équivalente du SVR linéaire

$$\min_{w \in \mathbb{R}^p, b \in \mathbb{R}} \quad \frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^n |w^T x_i + b - y_i|_\epsilon$$

# VASTE MARGE POUR LA RÉGRESSION



## Formulation duale du SVR linéaire

$$\begin{aligned} \min_{\alpha, \alpha' \in \mathbb{R}^n} \quad & \sum_{i,j=1}^n (\alpha_i - \alpha'_i)(\alpha_j - \alpha'_j) x_i^T x_j + \epsilon \sum_{i=1}^n (\alpha_i + \alpha'_i) - \sum_{i=1}^n y_i (\alpha_i - \alpha'_i) \\ \text{s.t.} \quad & \sum_{i=1}^n (\alpha_i - \alpha'_i) = 0 \\ & 0 \leq \alpha_i, \alpha'_i \leq C \quad \forall i \in \{1, \dots, n\} \end{aligned}$$

- Le modèle se réécrit sous la forme  $h(x) = \underbrace{\sum_{i=1}^n (\alpha_i - \alpha'_i) x_i^T}_w x + b$
- La formulation duale n'implique que les produits scalaires  $x_i^T x_j$ .  
On peut donc appliquer l'astuce du noyau !

- Article : **A Tutorial on Support Vector Machines for Pattern Recognition** (C. Burges, Data Mining and Knowledge Discovery 1998)
- Livre : **An introduction to Support Vector Machines and Others Kernel-Based Learning Methods** (N. Cristianini and J. Shawe-Taylor, 2000)
- Livre : **Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond** (B. Scholkopf and A. Smola, 2002)
- Article : **A tutorial on support vector regression** (A. Smola & B. Schölkopf, Statistics and Computing 2004)