Noyaux, SVM

Cours 6
ARF Master DAC

Nicolas Baskiotis
nicolas.baskiotis@lip6.fr
http://webia.lip6.fr/~baskiotisn

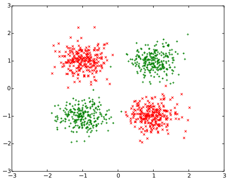
équipe MLIA, Laboratoire d'Informatique de Paris 6 (LIP6)
Université Pierre et Marie Curie (UPMC)

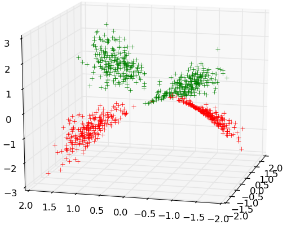
S2 (2016-2017)

Plan

1. Introduction
2. Support Vector Machine : principe
3. SVM : les maths - Intro à l'optimisation quadratique
4. The Kernel Trick - le tour de passe-passe non linéaire

Données non séparables linéairement





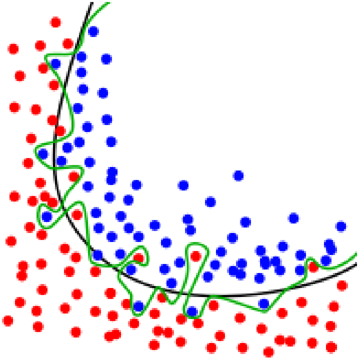
Solutions

- Utiliser des fonctions non-linéaires (réseau de neurones)
- Augmenter les dimensions : projection des données dans un espace de dimension supérieure

Projection des données

Oui mais ...

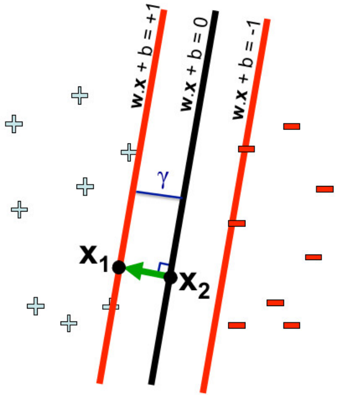
- Quelle projection ?
- Et le sur-apprentissage ?
- Et les “mauvaises” données (le bruit) ?



Plan

1. Introduction
2. Support Vector Machine : principe
3. SVM : les maths - Intro à l'optimisation quadratique
4. The Kernel Trick - le tour de passe-passe non linéaire

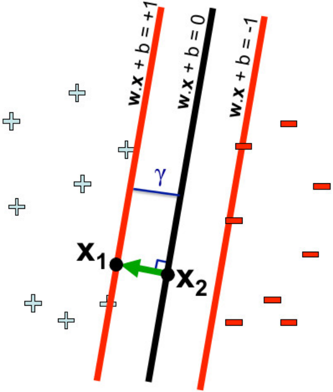
Se donner de la marge



Si séparable

- γ : distance entre l'hyperplan (frontière) et le point le plus proche
- Mise à l'échelle telle que en ce point $\mathbf{wx} + b = \pm 1$

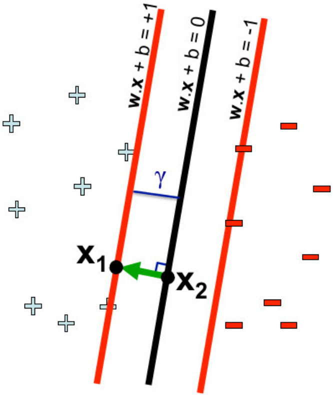
Se donner de la marge



Si séparable

- γ : distance entre l'hyperplan (frontière) et le point le plus proche
- Mise à l'échelle telle que en ce point $\mathbf{wx} + b = \pm 1$
- $\mathbf{x}_1 - \mathbf{x}_2 = \gamma \frac{\mathbf{w}}{\|\mathbf{w}\|}$
- $1 = \mathbf{w}(\mathbf{x}_1 - \mathbf{x}_2) = \gamma \frac{\mathbf{ww}}{\|\mathbf{w}\|} = \gamma \|\mathbf{w}\|$

Se donner de la marge



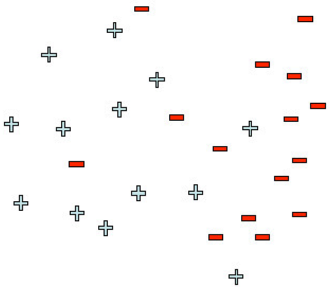
Si séparable

- γ : distance entre l'hyperplan (frontière) et le point le plus proche
- Mise à l'échelle telle que en ce point $\mathbf{wx} + b = \pm 1$
- $\mathbf{x}_1 - \mathbf{x}_2 = \gamma \frac{\mathbf{w}}{\|\mathbf{w}\|}$
- $1 = \mathbf{w}(\mathbf{x}_1 - \mathbf{x}_2) = \gamma \frac{\mathbf{ww}}{\|\mathbf{w}\|} = \gamma \|\mathbf{w}\|$

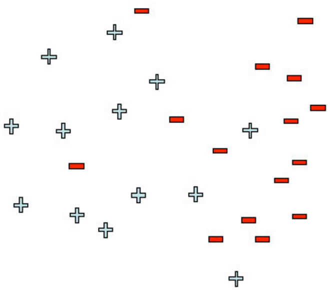
$\Rightarrow$ Maximiser la marge $\Leftrightarrow$ minimiser $\|\mathbf{w}\|$!

- Nouvelle formulation :
  minimiser $\|\mathbf{w}\|$ tel que $(\mathbf{w}x^i + b)y^i \geq 1$
  Problème d'optimisation quadratique convexe

Et si les données sont bruitées ?



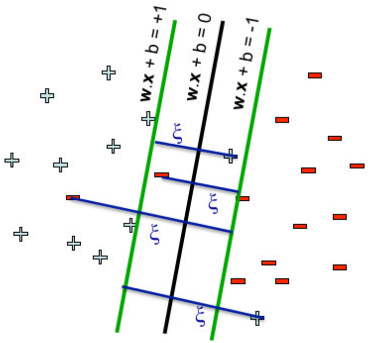
Et si les données sont bruitées ?



Prendre en compte les erreurs

- Minimiser $\|\mathbf{w}\| + K\#$Erreurs
  tel que $(\mathbf{w}x^i + b)y^i \geq 1$
- Problème NP difficile (et les problèmes inhérents au coût 0-1).

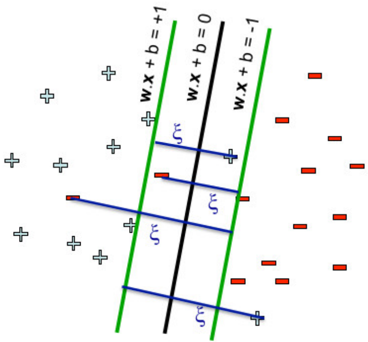
Et si les données sont bruitées ?



Approche Support Vector Machine

- Introduire des variables “ressorts” (slack)
- Minimiser $\|\mathbf{w}\|^2 + K\sum \xi_j$
  tel que $(\mathbf{w}\mathbf{x}^i + b)y^i \geq 1 - \xi_i,\, \xi_i \geq 0$
- Si la marge est plus grande que 1 $\rightarrow$ pas de coût, sinon coût linéaire :
  $$\begin{cases} \xi_i = 0 & \text{si } (\mathbf{w}\mathbf{x}^i + b)y^i \geq 1 \\ \xi_i = 1 - (\mathbf{w}\mathbf{x}^i)y^i & \text{si } (\mathbf{w}\mathbf{x}^i + b)y^i < 1 \end{cases}$$
- $\xi_i = max(0, 1 - (\mathbf{w}\mathbf{x}^i)y^i)$
- Pourquoi la constante K ? Comment la choisir ?

Et si les données sont bruitées ?



Approche Support Vector Machine

- Introduire des variables “ressorts” (slack)
- Minimiser $\|\mathbf{w}\|^2 + K \sum \xi_j$
  tel que $(\mathbf{w}\mathbf{x}^i + b)y^i \geq 1 - \xi_i,\ \xi_i \geq 0$
- Si la marge est plus grande que 1 $\rightarrow$ pas de coût, sinon coût linéaire :
  $$\begin{cases} \xi_i = 0 & \text{si } (\mathbf{w}\mathbf{x}^i + b)y^i \geq 1 \\ \xi_i = 1 - (\mathbf{w}\mathbf{x}^i)y^i & \text{si } (\mathbf{w}\mathbf{x}^i + b)y^i < 1 \end{cases}$$
- $\xi_i = max(0, 1 - (\mathbf{w}\mathbf{x}^i)y^i)$
- Pourquoi la constante K ? Comment la choisir ?

Formulation

- Minimiser : $\|\mathbf{w}\|^2 + K \sum \ell(y^i, \mathbf{w}\mathbf{x}^i + b)$
- Avec $\ell(y, \hat{y}) = max(0, 1 - y\hat{y}) \rightarrow$ Hinge Loss !
- Et $\|\mathbf{w}\|^2 \rightarrow$ terme de régularisation pour controler le sur-apprentissage.

Plan

1 Introduction

2 Support Vector Machine : principe

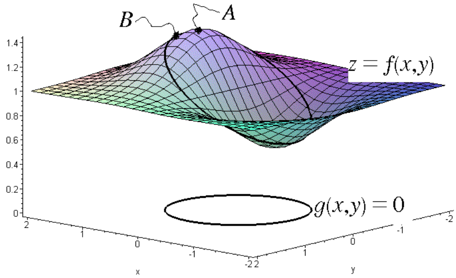
3 SVM : les maths - Intro à l'optimisation quadratique

4 The Kernel Trick - le tour de passe-passe non linéaire

Un outil utile d'optimisation : le lagrangien

Permet de résoudre les problèmes de type :

minimiser$_x$ $f(x)$ avec un ensemble de contraintes $c_i(x) \leq 0$



Un outil utile d'optimisation : le lagrangien

Permet de résoudre les problèmes de type :

minimiser$_x$ $f(x)$ avec un ensemble de contraintes $c_i(x) \leq 0$

Fonction lagrangienne

$L(x, \alpha) = f(x) + \sum_i \alpha_i c_i(x)$

Condition d'optimalité de Karush Kuhn Tucker

- $\partial_x L(x, \alpha) = \partial_x f(x) + \sum_i \alpha_i \partial_x c_i(x) = 0$
- $\forall\ i\ \alpha_i c_i(x) = 0$
- $\forall\ i\ c_i(x) \leq 0$
- $\forall\ i\ \alpha_i \geq 0$

Résoudre pour x et réinjecter dans L

optimiser$_\alpha L(x(\alpha), \alpha)$

La recette magique

Dans le cas simple (sans variables slack)

Le problème primal

$\text{minimiser}_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|^2$ tel que $y^i(\mathbf{w}\mathbf{x}^i + b) \geq 1$

Fonction de Lagrange

- $L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_i \alpha_i(y^i(\mathbf{w}\mathbf{x}^i + b) - 1)$
- Si $\mathbf{w}, b$ sont des minimums, alors il existe $\alpha_i \geq 0$ tel que le gradient du lagrangien soit nul.
- Condition d'optimalité (Karush Kuhn Tucker) :

$$\alpha_i(y^i(\mathbf{w}\mathbf{x}^i + b) - 1) = 0 \rightarrow \begin{cases} \alpha_i = 0 \\ \alpha_i > 0 \Rightarrow (y^i(\mathbf{w}\mathbf{x}^i + b) - 1) = 0 \end{cases}$$

Résolution

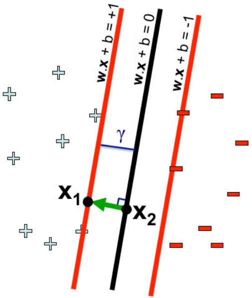
Lagrangien

- $L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_i \alpha_i(y^i(\mathbf{w}\mathbf{x}^i + b) - 1)$

Dérivées

- $\partial_{\mathbf{w}} L(\mathbf{w}, b, \alpha) = \mathbf{w} - \sum_i \alpha_i y^i \mathbf{x}^i = 0$
- $\partial_b L(\mathbf{w}, b, \alpha) = \sum_i \alpha_i y^i = 0$
- maximiser$_\alpha$ $\quad -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y^i y^j < \mathbf{x}^i, \mathbf{x}^j > + \sum_i \alpha_i$
  tel que $\sum \alpha_i y^i = 0$ et $\alpha_i \geq 0$.

Remarques importantes



- $\mathbf{w} = \sum_i \alpha_i y^i \mathbf{x}^i$: le vecteur de poids est une combinaison linéaire des exemples d'apprentissage !
- Il y a (même beaucoup) de α_i qui sont nuls $\Rightarrow$ exemples non pris en compte (normal ?)
- La solution ne fait intervenir que des produits scalaires

Dans le cas compliqué (avec slack)

Le problème primal

minimiser$_{w,b}$ $\quad \frac{1}{2}\|\mathbf{w}\|^2 + K\sum_i \xi_i$ tel que $y^i(\mathbf{w}\mathbf{x}^i + b) \geq 1 - \xi_i$ et $\xi_i \geq 0$

Lagrangien

- $L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 + K\sum_i \xi_i - \sum_i \alpha_i(y^i(\mathbf{w}\mathbf{x}^i + b) + \xi_i - 1) - \sum_i \eta_i \xi_i$

Dérivées

- $\partial_{\mathbf{w}} L(\mathbf{w}, b, \alpha, \xi, \eta) = w - \sum_i \alpha_i y^i x^i = 0$
- $\partial_b L(\mathbf{w}, b, \alpha, \xi, \eta) = \sum_i \alpha_i y^i = 0$
- $\partial_\xi L(\mathbf{w}, b, \alpha, \xi, \eta) = K - \alpha_i - \eta_i = 0$

$\Rightarrow$ maximiser$_\alpha$ $\quad -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y^i y^j < \mathbf{x}^i, \mathbf{x}^j > + \sum_i \alpha_i$
tel que $\sum \alpha_i y^i = 0$ et $\alpha_i \in [0, K]$.

- Condition d'optimalité (KKT) :

$$\begin{cases} \alpha_i(y^i(\mathbf{w}\mathbf{x}^i + b) + \xi_i - 1) = 0 \\ \eta_i \xi_i = 0 \end{cases} \rightarrow \begin{cases} \alpha_i = 0 \Rightarrow y^i(wx^i + b) \geq 1 \\ 0 < \alpha_i < K \Rightarrow (y^i(\mathbf{w}\mathbf{x}^i + b) - 1) = 1 \\ \alpha_i = K \Rightarrow (y^i(\mathbf{w}\mathbf{x}^i + b) - 1) \leq 1 \end{cases}$$

Plan

1 Introduction

2 Support Vector Machine : principe

3 SVM : les maths - Intro à l'optimisation quadratique

4 The Kernel Trick - le tour de passe-passe non linéaire

LE détail important

Dans toutes les formulations

- minimiser$_{\mathbf{w},b}$ $\frac{1}{2}\|\mathbf{w}\|^2 + K\sum_i \xi_i$ tel que $y^i(\mathbf{w}\mathbf{x}^i + b) \geq 1 - \xi_i$ et $\xi_i \geq 0$
- maximiser$_\alpha$ $-\frac{1}{2}\sum_{i,j} \alpha_i\alpha_j y^i y^j < \mathbf{x}^i, \mathbf{x}^j > + \sum_i \alpha_i$
  tel que $\sum \alpha_i y^i = 0$ et $\alpha_i \in [0, K]$.
- $f(\mathbf{x}) = \sum_i \alpha_i y_i < \mathbf{x}^i, \mathbf{x} > +b$

LE détail important

Dans toutes les formulations

- minimiser$_{\mathbf{w},b}$ $\frac{1}{2}\|\mathbf{w}\|^2 + K\sum_i \xi_i$ tel que $y^i(\mathbf{w}\mathbf{x}^i + b) \geq 1 - \xi_i$ et $\xi_i \geq 0$
- maximiser$_\alpha$ $-\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y^i y^j < \mathbf{x}^i, \mathbf{x}^j > +\sum_i \alpha_i$
  tel que $\sum \alpha_i y^i = 0$ et $\alpha_i \in [0, K]$.
- $f(\mathbf{x}) = \sum_i \alpha_i y_i < \mathbf{x}^i, \mathbf{x} > +b$

- Ce qui importe c'est le produit scalaire !

Pourquoi donc ?

Non linéarité $\rightarrow$ projection

- On veut considérer une projection $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^{n'}, \quad n << n'$
- Mais n' peut être vraiment très grand ! (projection polynomiale, gaussienne)

$\Rightarrow$ Problèmes :
  - Computationels
  - Sur-apprentissage
- Est-on obligé de calculer explicitement $\phi(x)$?

Le produit scalaire

Pour le SVM, on a besoin :

- $< \mathbf{w}, \mathbf{x} > \Rightarrow < \mathbf{w}', \phi(\mathbf{x}) >$, mais en fait $\mathbf{w}$ s'exprime à partir de $\phi(\mathbf{x})$
- $< \mathbf{x}, \mathbf{x} > \Rightarrow < \phi(\mathbf{x}), \phi(\mathbf{x} >)$
- et c'est tout !

$\Rightarrow$ Peut-on calculer directement ce produit scalaire ?

Exemple : projection polynomiale

- $\phi(\mathbf{x}) = (1, 2x_1, 2x_2, \ldots, 2x_D, x_1x_1, x_1x_2, \ldots, x_Dx_D)$
- $< \phi(\mathbf{x})\phi(\mathbf{x}') >= 1 + 2\sum_i x_i x_i' + \sum_i \sum_j x_i x_i' x_j x_j' = 1 + 2\mathbf{x}\mathbf{x}' + (\mathbf{x}\mathbf{x}')^2 = (1 + \mathbf{x}\mathbf{x}')^2$

Les noyaux

Définition

- Forme généralisée de produit scalaire : $k(\mathbf{x}, \mathbf{x}') =< \phi(\mathbf{x}), \phi(\mathbf{x}') >$
- Noyaux admissibles : tous ceux qui peuvent se mettre sous la forme d'un produit scalaire de deux projections (il existe ϕ tel que ...).
- Mathématiquement : fonction semi-définie positive : pour toute fonction f carré intégrable, $\int_{x,x'} f(x)k(x,x')f(x')dxdx' > 0$.
- Ou, sur un échantillon $\{x^1, \ldots, x^n\}$, si k est symétrique et pour tout $c_i \in \mathbb{R}$, $\sum_{i,j} c_i c_j k(x_i, x_j) \geq 0$.

Opération

Si k, k' sont des noyaux, alors sont aussi des noayux:

- $k(x,x') + k'(x,x')$
- $k(x,x') * k'(x,x')$
- $k(f(x), f'(x)$
- $f(k(x,x'))$ pour f polynome
- $exp(k(x,x'))$

Quelques exemples

- Noyau gaussien : $k(x,x') = exp(-\|x-x'\|^2/\sigma^2)$
- Bag of words pour une phrase
- Noyau de convolution : $k(x,x') = \sum_{w\in x}\sum_{w'\in x'} k'(w,w')$
- Noyau sur les arbres, les graphes ...
- Penser aux noyaux comme une mesure de similarité entre deux objets !
  - si éloignés $\rightarrow$ 0 (produit scalaire orthogonal)
  - si proche $\rightarrow$ valeur maximale (vecteurs alignés)

Conclusion

- Mythe : Les SVMs fonctionnent parce que l'on projette en très haute dimension

$\Rightarrow$ alors on aurait besoin de bien plus de données

- Combiné à la contrainte de marge.
- On retrouve une forme générique des problèmes d'apprentissage : $R(f) = \sum \ell(f(x^i), y^i) + \Omega(f)$,
  avec ℓ une fonction de coût (risque empirique) et Ω une régularisation sur la complexité de la fonction f.
- Permet de régler le sur-apprentissage (ou de manière équivalent de contraindre la classe de fonction considérée).