MDI343 - Cours 5: Support Vector Machines

Florence d'Alché–Buc
Institut Mines-Télécom, Télécom ParisTech, LTCI
florence.dalche@telecom-paristech.fr

Outline

1. Rappels
2. SVM linéaires
3. Passage au cas non linéaire et noyaux
4. Support Vector Regression
5. References

Classification binaire supervisée

Cadre probabiliste et statistique 1/2

- Soit X un vecteur aléatoire de $\mathcal{X} = \mathbb{R}^p$
- Exemple: X décrit les caractéristiques ("features") d'un message ou document
- Y une variable aléatoire discrète $\mathcal{Y} = \{-1, 1\}$
- Soit $\mathbb{P}$ la loi de probabilité jointe de (X,Y)
- Soit $\mathcal{S}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, i.i.d. sample from $\mathbb{P}$.

Classification binaire supervisée

Cadre probabiliste et statistique 2/2

- Soit $h : \mathbb{R}^p \rightarrow \{-1, +1\}$ une fonction de classification binaire
- Soit $\ell : \{\mathbb{R}^p, -1, +1\} \times \{-1, +1\} \rightarrow \mathbb{R}$ une fonction de perte ou coût
- Risque empirique $R_n(h) = \frac{1}{n}\sum_i \ell(y_i, h(x_i))$ et un terme régularisateur $\Omega(h)$ qui mesure la *complexité* de h.
- On cherche : $\hat{h} = \arg\min_{h \in \mathcal{H}} R_n(h) + \lambda\Omega(h)$

Et en pratique, comment fait-on ?

Méthodologie pour développer une approche discriminante

- Définir
  - l'**espace de représentation** des entrées

Et en pratique, comment fait-on ?

Méthodologie pour développer une approche discriminante

- Définir
  - l'**espace de représentation** des entrées
  - la **classe des fonctions** de classification binaire considérées

Et en pratique, comment fait-on ?

Méthodologie pour développer une approche discriminante

- Définir
  - l'**espace de représentation** des entrées
  - la **classe des fonctions** de classification binaire considérées
  - la **fonction de coût** à minimiser pour obtenir le meilleur classifieur dans cette classe

Et en pratique, comment fait-on ?

Méthodologie pour développer une approche discriminante

- Définir
  - l'**espace de représentation** des entrées
  - la **classe des fonctions** de classification binaire considérées
  - la **fonction de coût** à minimiser pour obtenir le meilleur classifieur dans cette classe
  - l'**algorithme de minimisation** de cette fonction de coût

Et en pratique, comment fait-on ?

Méthodologie pour développer une approche discriminante

- Définir
  - l'**espace de représentation** des entrées
  - la **classe des fonctions** de classification binaire considérées
  - la **fonction de coût** à minimiser pour obtenir le meilleur classifieur dans cette classe
  - l'**algorithme de minimisation** de cette fonction de coût
  - une **méthode de sélection de modèle** pour définir les hyperparamètres

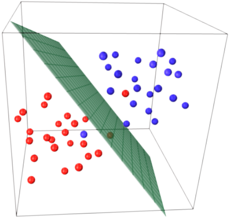
Outline

1. Rappels
2. SVM linéaires
3. Passage au cas non linéaire et noyaux
4. Support Vector Regression
5. References

Séparateur linéaire

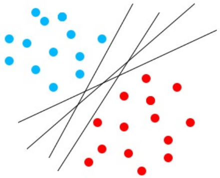
Définition

Soit $\mathbf{x} \in \mathbb{R}^p$
$h(\mathbf{x}) = \text{signe}(\mathbf{w}^T\mathbf{x} + b)$
L'équation : $\mathbf{w}^T\mathbf{x} + b = 0$ définit un hyperplan dans l'espace euclidien $\mathbb{R}^p$



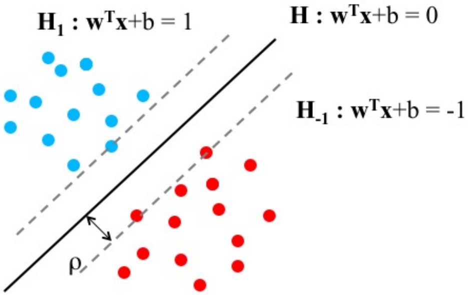
Exemple: données d'apprentissage en 3D et séparateur linéaire

Cas de données linéairement séparables



Exemple en 2D: quelle droite choisir ?

Critère de marge



Critère de marge

Notion de marge géométrique

- Pour séparer les données, on considère un triplet d'hyperplans:
  - H: $\mathbf{w}^T\mathbf{x} + b = 0$, $H_1 : \mathbf{w}^T\mathbf{x} + b = 1$, $H_{-1} : \mathbf{w}^T\mathbf{x} + b = -1$
- On appelle *marge géométrique*, $\rho(\mathbf{w})$ la plus petite distance entre les données et l'hyperplan H, ici donc la moitié de la distance entre H_1 et H_{-1}
- Un calcul simple donne : $\rho(\mathbf{w}) = \frac{1}{||\mathbf{w}||}$.

Nouvelle fonction de coût à optimiser

Comment déterminer w et b ?

- Maximiser la marge $\rho(\mathbf{w})$ tout en séparant les données de part et d'autre de H_1 et H_{-1}
- Séparer les données bleues ($y_i = 1$) : $\mathbf{w}^T\mathbf{x}_i + b \geq 1$
- Séparer les données rouges ($y_i = -1$) : $\mathbf{w}^T\mathbf{x}_i + b \leq -1$

SVM linéaire: cas séparable

Optimisation dans l'espace primal

$$\underset{\mathbf{w},b}{\text{minimiser}} \quad \frac{1}{2}\|\mathbf{w}\|^2$$
$$\text{sous la contrainte} \quad y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b}) \geq 1, \; i = 1, \ldots, n.$$

Référence

Boser, B. E.; Guyon, I. M.; Vapnik, V. N. (1992). "A training algorithm for optimal margin classifiers". Proceedings of the fifth annual workshop on Computational learning theory - COLT '92. p. 144.

Programmation quadratique sous contraintes inégalités

Problème du type (attention les notations changent !)

- un problème d'optimisation $(\mathcal{P})$ est défini par

$$\begin{array}{rl} \text{minimiser sur } \mathbb{R}^n & J(\mathbf{x}) \\ \text{avec} & h_i(\mathbf{x}) = 0\,,\, 1 \leq i \leq p \\ & g_j(\mathbf{x}) \leq 0\,,\, 1 \leq j \leq q \end{array}$$

- rappel de vocabulaire :
  - les h_i sont les **contraintes d'égalité** (notées $\mathbf{h}(\mathbf{x}) = 0$)
  - les g_j sont les **contraintes d'inégalité** (notées $\mathbf{g}(\mathbf{x}) \leq 0$)
  - l'**ensemble des contraintes** est

$$\mathcal{C} = \{\mathbf{x} \in \mathbb{R}^n | h_i(\mathbf{x}) = 0\,,\, 1 \leq i \leq p \text{ et } g_j(\mathbf{x}) \leq 0\,,\, 1 \leq j \leq q\}$$

ensemble des points admissibles ou **réalisables**

Programming under inequality constraints

Problem of the following kind:
$\min_x f(x)$
s.c. $g(x) \leq 0$

- Here: $g(x)$: linear constraints
- f is strictly convex

1. Lagrangian: $J(x, \lambda) = f(x) + \lambda g(x)$, $\lambda \geq 0$

Programmation quadratique sous contraintes inégalités

$$\begin{array}{ll} \underset{\mathbf{w},b}{\text{minimiser}} & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{sous la contrainte} & 1 - y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b}) \leq 0,\ i = 1, \ldots, n. \end{array}$$

Lagrangien

$$\mathcal{L}(\mathbf{w}, b, \alpha) = \frac{1}{2}||\mathbf{w}||^2 + \sum_i \alpha_i(1 - y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b}))$$

$$\forall i, \alpha_i \geq 0$$

Conditions de Karush-Kuhn-Tucker

En l'extremum, on a

$$
\begin{aligned}
\nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}) = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0 \\
\nabla_b \mathcal{L}(b) = -\sum_{i=1}^{n} \alpha_i y_i = 0 \\
\forall i, \alpha_i \geq 0 \\
\forall i, \alpha_i [1 - y_i(\mathbf{w}^T \mathbf{x}_i + b)] = 0
\end{aligned}
$$

Obtention des α_i : résolution dans l'espace dual

$$\mathcal{L}(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j)$$

- Maximiser $\mathcal{L}$ sous les contraintes $\alpha_i \geq 0$ et $\sum_i \alpha_i y_i = 0, \forall i = 1, \ldots, n$
- Faire appel à un solveur quadratique

SVM linéaires ou Optimal Margin Hyperplan

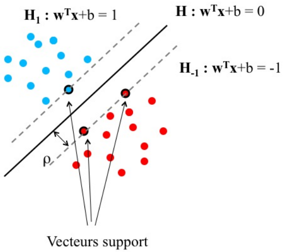
Supposons que les multiplicateurs de Lagrange α_i soient déterminés :

Equation d'un SVM linéaire

$$f(\mathbf{x}) = \text{signe}(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b)$$

Pour classer une donnée $\mathbf{x}$, ce classifier combine linéairement les valeurs de classe y_i des données support avec des poids du type $\alpha_i \mathbf{x}_i^T \mathbf{x}$ dépendant de la ressemblance entre $\mathbf{x}$ et les données support au sens du produit scalaire.

Vecteurs "supports"



Les données d'apprentissage $\mathbf{x}_i$ telles que $\alpha_i \neq 0$ sont sur l'un ou l'autre des hyperplans H_1 ou H_{-1}. Seules ces données dites *vecteur de support* comptent dans la définition de $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$

NB : b est obtenu en choisissant une donnée support ($\alpha_i \neq 0$)

Cas réaliste: SVM linéaire dans le cas données non séparables

Introduire une variable d'écart ξ_i pour chaque donnée:

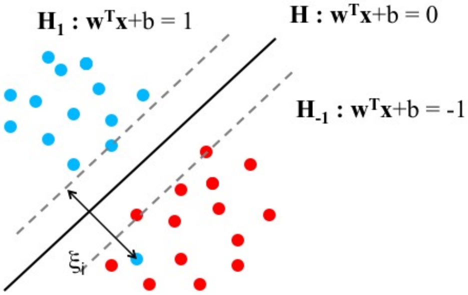
Problème dans le primal

$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{sous les contraintes} \quad y_i(\mathbf{w}^T\mathbf{x}_i + \mathrm{b}) \geq 1 - \xi_i \; i = 1, \ldots, n.$$

$$\xi_i \geq 0 \; i = 1, \ldots, n.$$

Cas réaliste: SVM linéaire dans le cas données non séparables



Cas réaliste: SVM linéaire dans le cas données non séparables

Problème dans le dual

$$
\begin{aligned}
\max_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\
\text{sous les contraintes} \quad & 0 \leq \alpha_i \leq C \; i = 1, \ldots, n. \\
& \sum_i \alpha_i y_i \; i = 1, \ldots, n.
\end{aligned}
$$

Conditions de Karush-Kuhn-Tucker (KKT)

Soit α^* la solution du problème dual:

$$\forall i, [y_i f_{w^*,b^*}(x_i) - 1 + \xi_i^*] \leq 0 \tag{1}$$
$$\forall i, \alpha_i^* \geq 0 \tag{2}$$
$$\forall i, \alpha_i^*[y_i f_{w^*,b^*}(x_i) - 1 + \xi_i^*] = 0 \tag{3}$$
$$\forall i, \mu_i^* \geq 0 \tag{4}$$
$$\forall i, \mu_i^* \xi_i^* = 0 \tag{5}$$
$$\forall i, \alpha_i^* + \mu_i^* = C \tag{6}$$
$$\forall i, \xi_i^* \geq 0 \tag{7}$$
$$\mathbf{w}^* = \sum_i \alpha_i^* y_i \mathbf{x}_i \tag{8}$$
$$\sum_i \alpha_i^* y_i = 0 \tag{9}$$
$$\tag{10}$$

Différents cas de figure

Soit α^* la solution du problème dual:

- si $\alpha_i^* = 0$, alors $\mu_i^* = C > 0$ et donc, $\xi_i^* = 0$: x_i est bien classé
- si $0 < \alpha_i^* < C$ alors $\mu_i^* > 0$ et donc, $\xi_i^* = 0$: x_i est tel que : $y_i f(x_i) = 1$
- si $\alpha_i^* = C$, alors $\mu_i^* = 0$, $\xi_i^* = 1 - y_i f_{w^*,b^*}(x_i)$

NB : on calcule b^* en utilisant un i tel que $0 < \alpha_i^* < C$

Cas réaliste: SVM linéaire dans le cas données non séparables

Quelques remarques

- certaines données support peuvent donc être de l'autre côté des hyperplans H_1 ou H_{-1}
- C est un hyperparamètre qui contrôle le compromis entre la complexité du modèle et le nombre d'erreurs de classification du modèle.

SVM : approche par régularisation

Optimisation dans l'espace primal

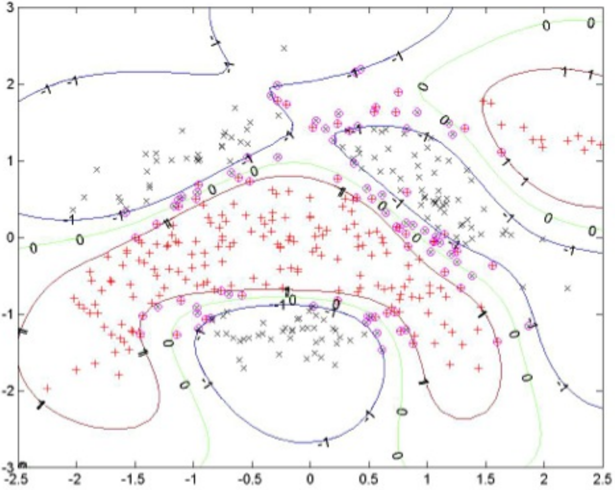
$$\min_{\mathbf{w},b} \quad \sum_{i=1}^{n} (1 - y_i(\mathbf{w}^T \mathbf{x}_i + b))_+ + \lambda \frac{1}{2} \|\mathbf{w}\|^2$$

Avec: $(z)_+ = max(0, z)$
$f(\mathbf{x}) = \text{signe}(h(\mathbf{x}))$
Fonction de coût: $L(\mathbf{x}, y, h(\mathbf{x})) = (1 - yh(\mathbf{x}))_+$
$yh(\mathbf{x})$ est appelée marge du classifieur

Outline

1. Rappels
2. SVM linéaires
3. Passage au cas non linéaire et noyaux
4. Support Vector Regression
5. References

Support Vector Machine : le cas non linéaire



Remarque

Le problème de l'hyperplan de marge optimale ne fait intervenir les données d'apprentissage qu'à travers de produits scalaires.

$$
\begin{array}{ll}
\max\limits_{\alpha} & \displaystyle\sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\
\text{sous les contraintes} & 0 \leq \alpha_i \leq C \; i = 1, \ldots, n. \\
 & \displaystyle\sum_i \alpha_i y_i \; i = 1, \ldots, n.
\end{array}
$$

Remarque 1: apprentissage

Si je transforme les données à l'aide d'une fonction ϕ (non linéaire) et si je sais calculer les produits scalaires $\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$, je peux apprendre une fonction de séparation non linéaire.

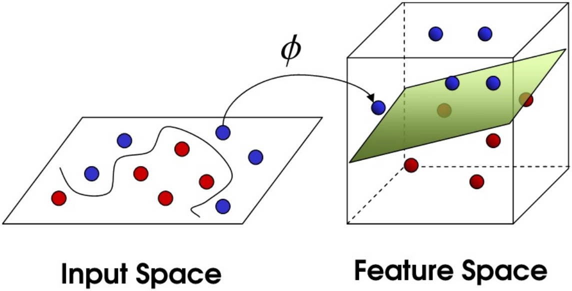
$$\begin{aligned}
\max_{\alpha} \quad & \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) \\
\text{sous les contraintes} \quad & 0 \leq \alpha_i \leq C \; i = 1, \ldots, n. \\
& \sum_i \alpha_i y_i \; i = 1, \ldots, n.
\end{aligned}$$

Pour classer une nouvelle donné $\mathbf{x}$, je n'ai besoin que de savoir calculer $\phi(\mathbf{x})^T\phi(\mathbf{x}_i)$.

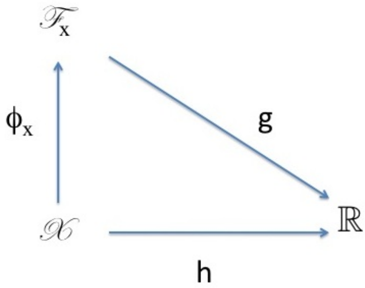
Astuce du noyau

Si je remplace $\mathbf{x}_i^T \mathbf{x}_j$ par l'image par une fonction k : $k(\mathbf{x}_i, \mathbf{x}_j)$ telle qu'il existe un espace de caractérisques $\mathcal{F}$ et une fonction de caractéristique (feature map) $\phi : \mathcal{X} \to \mathcal{F}$ et $\forall (\mathbf{x}, \mathbf{x}') \in \mathcal{X}, k(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$, alors je peux appliquer le même algorithme d'optimisation (résolution dans le dual) et j'obtiens :
$f(\mathbf{x}) = \text{signe}(\sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b)$
Des telles fonctions existent et sont appelées *noyaux*.

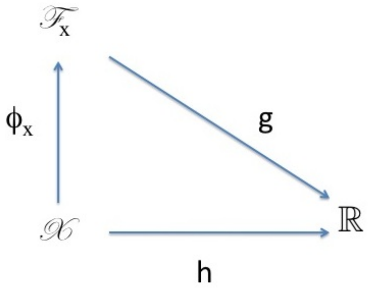
Astuce du noyau et fonction de redescription 1/2



Astuce et fonction de redescription 2/2



Astuce et fonction de redescription 2/2



Fonction h du type: $h(\mathbf{x}) = \sum_{i=1}^{n} \beta_i \phi(x)^T \phi(x_i) = \sum_{i=1}^{n} \beta_i k(x, x_i)$,
avec $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ un noyau positif défini.

Noyaux

Définition

Soit $\mathcal{X}$ un ensemble. Soit k:$\mathcal{X} \times \mathcal{X} \to \mathbb{R}$, une fonction symétrique. La fonction k est appelée *noyau* positif défini si et seulement si quel que soit le sous-ensemble fini $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ de $\mathcal{X}$ et le vecteur colonne **c** de $\mathbb{R}^m$,
$\mathbf{c}^T K \mathbf{c} = \sum_{i,j=1}^m c_i c_j k(x_i, x_j) \geq 0$

N.B.: on impose donc que toute matrice construite à partir d'un nombre fini d'éléments de $\mathcal{X}$ soit semi-définie positive.

Propriété des noyaux

Théorème de Moore-Aronzajn (simplifié)

Soit K un noyau positif défini. Alors, il existe un espace de Hilbert $\mathcal{F}$, appelé *espace de redescription* et une fonction $\phi : \mathcal{X} \to \mathcal{F}$, appelée fonction de redescription (en anglais, feature map) telle que:
$\langle \phi(x), \phi(x') \rangle_{\mathcal{F}} = k(x, x')$.
N.B.: pour un noyau k, il peut exister plusieurs couples $(\mathcal{F}, \phi)$.

Propriété des noyaux

Théorème de Moore-Aronzajn 2 (plus précis)

Il existe une unique fonction de redescription et un unique espace de Hilbert tels que: $\langle \phi(x), f \rangle_{\mathcal{F}} = f(x)$. Il s'agit de la fonction définie par $\phi(x) = k(\cdot, x)$ et de l'espace de Hilbert à noyau autoreproduisant qui est l'espace fonctionnel engendré par $\{\sum_\ell k(\cdot, z_\ell)\beta_\ell, z_\ell \in \mathcal{X}\}$ et completé par les limites des suites de Cauchy de ces fonctions.

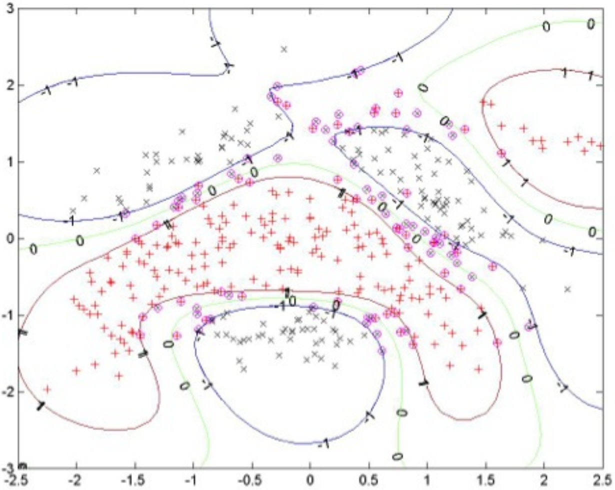
Noyaux

Noyaux entre vecteurs

$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^p$

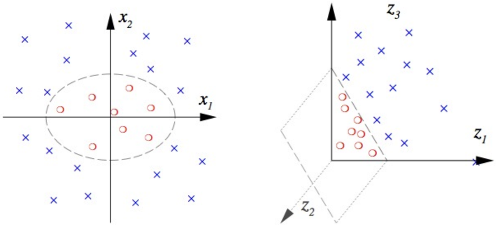
- Noyau linéaire : $k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'$
- Noyau polynomial : $k(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^T \mathbf{x}' + c)^d$
- Noyau gaussien : $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2)$

Support Vector Machine : séparateur non linéaire par noyau gaussien



Exemple : noyau polynomial

$$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$



Exemple : noyau polynomial

Astuce du noyau

On remarque que $\phi(\mathbf{x}_1)^T\phi(\mathbf{x}')$ peut se calculer sans travailler dans $\mathbb{R}^3$
Je peux définir $k(\mathbf{x},\mathbf{x}') = \phi(\mathbf{x})^T\phi(\mathbf{x}') = (\mathbf{x}^T\mathbf{x}')^2$

Autre usage des noyaux

On peut construire des noyaux pour des données structurées : grpahes, séquences, arbres et appliquer les SVM !

- Classifier des molécules
- Classifier des documents structurés
- Traiter des séquences biologiques
- ...

Kernel Design

- Use closure properties to build new kernels from existing ones
- Kernels can be defined for various objects:
  - **Structured objects**: (sets), graphs, trees, sequences, ...
  - Unstructured data with underlying structure: texts, images, documents, signal, biological objects
- **Kernel learning**:
  - Hyperparameter learning: see Chapelle et al. 2002
  - Multiple Kernel Learning: given $k_1, \ldots, k_m$, learn a convex combination $\sum_i \beta_i k_i$ of kernels (see SimpleMKL Rakotomamonjy et al. 2008, unifying view in Kloft et al. 2010)

Quel noyau pour notre détecteur de spams?

On peut prendre soit:

- le noyau linéaire
- le noyau gaussien ou une de ses variantes
  - Connaissance *a priori* d'une matrice de similarité sémantique A entre mots
  - Appliquer A au vecteur **x** revient à faire apparaître des mots proches sémantiquement
  - $k(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||A(\mathbf{x} - \mathbf{x}')||^2)$
  - équivalent à: $k(\mathbf{x}, \mathbf{x}') = \exp[-\gamma(A(\mathbf{x} - \mathbf{x}'))^T(A(\mathbf{x} - \mathbf{x}'))]$
  - soit : $k(\mathbf{x}, \mathbf{x}') = \exp[-\gamma(\mathbf{x} - \mathbf{x}')^T A^T A(\mathbf{x} - \mathbf{x}')]$

Outline

1. Rappels
2. SVM linéaires
3. Passage au cas non linéaire et noyaux
4. Support Vector Regression
5. References

Régression

Cadre probabiliste et statistique

Soit X un vecteur aléatoire de $\mathcal{X} = \mathbb{R}^p$
Y une variable aléatoire continue $\mathcal{Y} = \mathbb{R}$
Soit P la loi de probabilité jointe de (X,Y), loi fixée mais inconnue
Supposons que $S_{app} = \{(x_i, y_i), i = 1, \ldots, n\}$ soit un échantillon i.i.d. tiré de la loi P

Régression

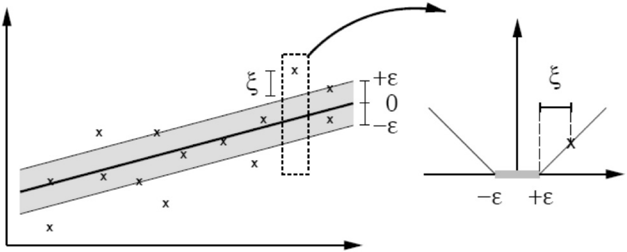
Cadre probabiliste et statistique

- A partir de S_{app}, déterminer la fonction $f \in \mathcal{F}$ qui minimise $R(f) = \mathbb{E}_P[\ell(X, Y, f(X))]$
- ℓ étant une fonction de coût local qui mesure à quel point la vraie cible et la prédiction par le classifieur sont différentes

Pb: la loi jointe n'est pas connue : on ne peut pas calculer $R(f)$

Support Vector Regression

- Extend the idea of maximal soft margin to regression
- Impose an ε-tube : perte ε-insensible
  $|y' - y|_{\varepsilon} = max(0, |y' - y| - \varepsilon)$



Support Vector Regression

SVR in the primal space

Given C and ε

$\min_{w,b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i(\xi_i + \xi_i^*)$

s.c.

$\forall i = 1, \ldots n, y_i - f(x_i) \leq \varepsilon + \xi_i$

$\forall i = 1, \ldots n, f(x_i) - y_i \leq \varepsilon + \xi_i^*$

$\forall i = 1, \xi_i \geq 0, \xi_i^* \geq 0$

with $f(x) = w^T\phi(x) + b$

General case : ϕ is a feature map associated with a positive definite kernel k.

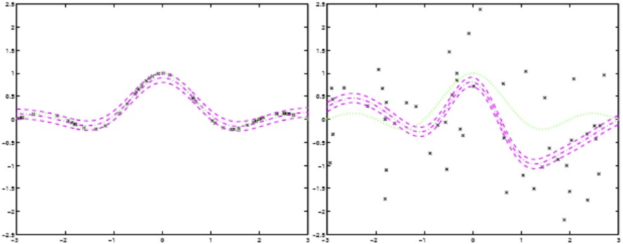
Solution in the dual

$\min_{\alpha,\alpha^*} \sum_{i,j}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j) + \varepsilon \sum_i(\alpha_i + \alpha_i^*) - \sum_i y_i(\alpha_i - \alpha_i^*)$
s.c. $\sum_i(\alpha_i - \alpha_i^*) = 0$ and $0 \leq \alpha_i \leq C$ and $0 \leq \alpha_i^* \leq C$
$w = \sum_{i=1}^n(\alpha_i - \alpha_i^*)\phi(x_i)$

Solution

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

Support Vector Regression: example in 1D



Identical machine parameters ($\varepsilon = 0.2$), but different amounts of noise in the data.

B. Schölkopf, Canberra, February 2002

Outline

1. Rappels
2. SVM linéaires
3. Passage au cas non linéaire et noyaux
4. Support Vector Regression
5. References

References

- BOSER, Bernhard E., Isabelle M. GUYON, and Vladimir N. VAPNIK, 1992. A training algorithm for optimal margin classifiers. In: COLT âĂŹ92: Proceedings of the Fifth Annual Workshop on Computational Learning Theory. New York, NY, USA: ACM Press, pp. 144-152.
- CORTES, Corinna, and Vladimir VAPNIK, 1995. Support-vector networks. Machine Learning, 20(3), 273âĂŞ297.
- Article vraiment sympa, complet (un peu de maths) : A tutorial review of RKHS methods in Machine Learning, Hofman , Schoelkopf, Smola, 2005 (`https://www.researchgate.net/publication/228827159_A_Tutorial_Review_of_RKHS_Methods_in_Machine_Learning`)