IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

UVF3B403 MS IABDA
Support Vector Machines

Yannis Haralambous (IMT Atlantique)

12 janvier 2017

Préliminaires

- Soit D la droite donnée par l'équation $ax + by + c = 0$.
- Une manière de voir cette équation est de dire que l'on a un vecteur $\alpha = (a, b)$ et d'écrire l'équation comme

$$\langle \alpha, (x, y) \rangle = -c.$$

- Alors α est un vecteur perpendiculaire à D (pourquoi ?).

Préliminaires

- Réponse : parce que (a, b) est perpendiculaire à $(-b, a)$, qui est le vecteur directeur de la droite.
- Et pourquoi $(-b, a)$ est-il vecteur directeur de la droite $ax + by + c = 0$?
- Prenons les intersections de la droite et des axes : $(0, -\frac{c}{b})$ et $(-\frac{c}{a}, 0)$. Le vecteur directeur sera proportionnel à la différence de ces deux points :

$$(-\frac{c}{a}, 0) - (0, -\frac{c}{b}) = (-\frac{c}{a}, \frac{c}{b}) = \frac{c}{ab}(-b, a) = \lambda(-b, a).$$

Préliminaires

Proposition

Soit M un point extérieur à D. La distance entre M et D est donnée par la formule

$$d(M,D) = \frac{|aM_x + bM_y + c|}{\|\alpha\|}.$$

Démonstration.

Soit $z_0 = (x_0, y_0)$ un point quelconque de la droite. Alors $|\langle \overrightarrow{Mz_0}, \alpha\rangle| = |a(M_x - x_0) + b(M_y - y_0)| = |aM_x + bM_y - ax_0 - by_0| = |aM_x + bM_y + c|$.
Mais aussi, en prenant la projection sur α, $|\langle \overrightarrow{Mz_0}, \alpha\rangle| = \|\mathrm{proj}_\alpha(Mz_0)\| \cdot \|\alpha\|$ et comme cette projection est égale à la distance euclidienne (car $\alpha \perp D$), ceci est égal à $d(M,D) \cdot \|\alpha\|$. □

Préliminaires

- On peut donc dire que la distance est égale à

$$d(M,D) = \frac{|\langle \alpha, M\rangle + c|}{\|\alpha\|}$$

où α est ce fameux vecteur de coordonnées a et b, qui est orthogonal à D.

- L'avantage de cette formulation est qu'elle se généralise aux hyperplans.

Définition

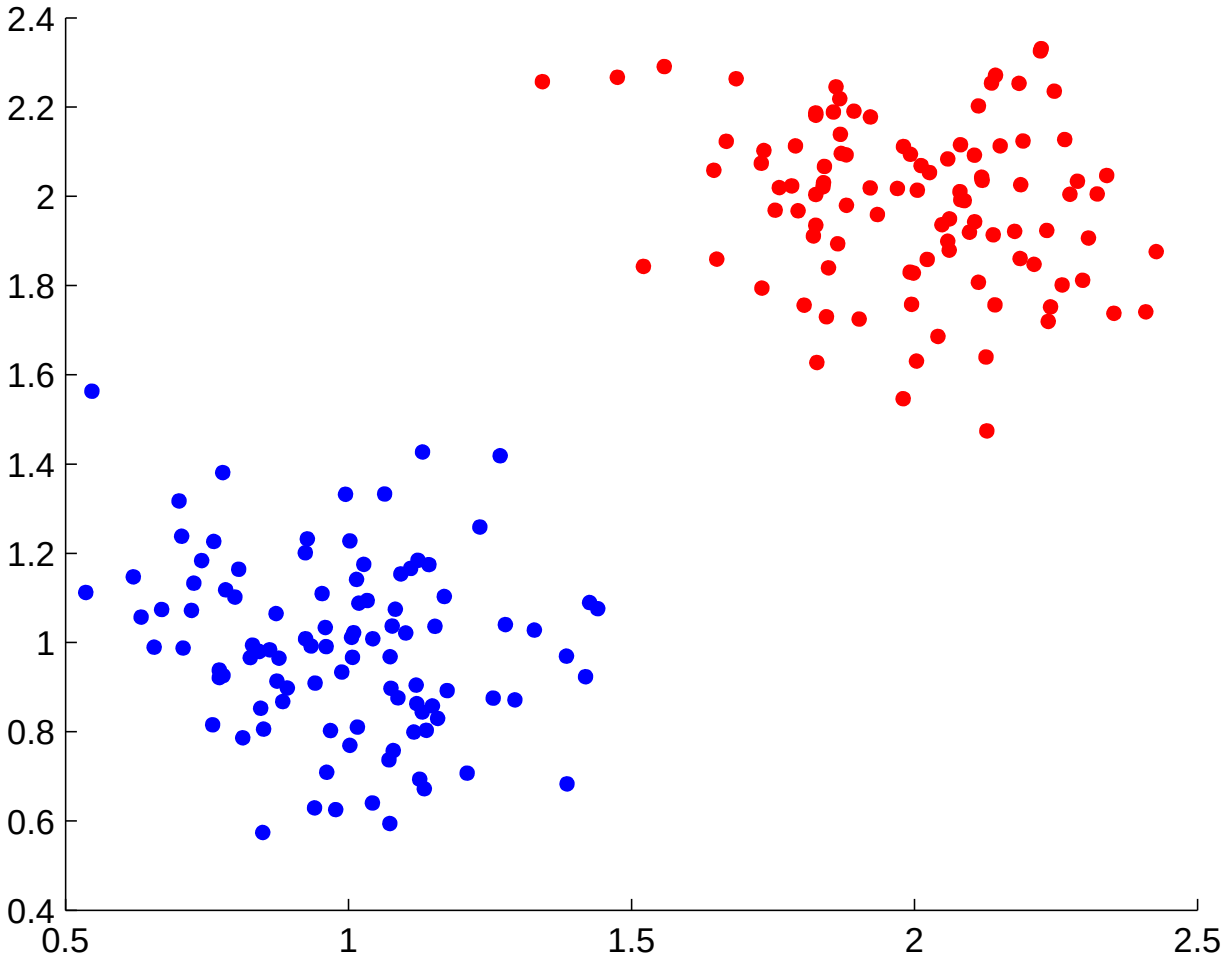
Soient E, E' des espaces affines, E' de dimension 1. Une application $f : E \to E'$ est une *forme affine* si pour tout $x \in E$, $\vec{f}_x$ définie par $\vec{f}_x(\vec{u}) = \overrightarrow{f(x)f(x+u)}$ est une application linéaire $\vec{f}_x : \vec{E} \to \vec{E'}$.

- On peut dire que $\varphi(z) = \langle \alpha, z\rangle + c$ est une forme affine, qui est nulle quand on est sur la droite et non nulle sinon.

Classification

Yannis Haralambous
(IMT Atlantique)

Soit un nuage de points coloriés de $\mathbb{R}^2$ correspondant à deux classes.

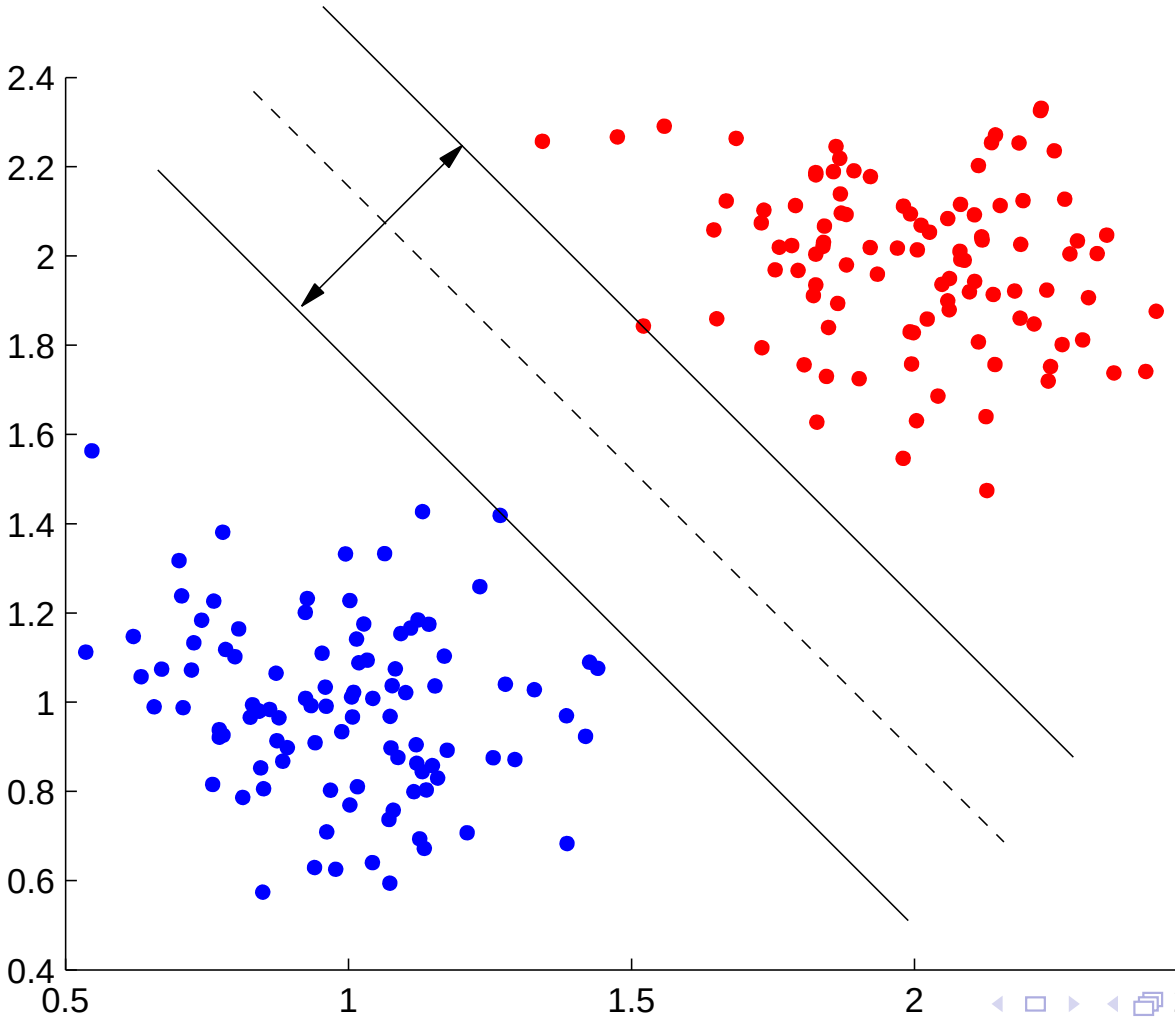


(Figures tirées de *Introduction aux SVM* de J.-O. Moussafir, 2005.)

Classification

Yannis Haralambous
(IMT Atlantique)

On cherche la droite D qui sépare au mieux les classes, donc celle pour laquelle on ait atteint le max $\min_{\text{tous les points z}} d(z, D)$.



Classification

- Supposons que les classes des points z_i soient représentées par des valeurs $f(z_i) = +1$ et -1. D'autre part, la droite ne change pas si on multiplie (a, b) et c par une même valeur. On peut donc s'arranger pour que les points bleu et rouge les plus proches de la droite aient les valeurs $\langle \alpha, z \rangle + c = +1$ et -1.
- Alors le problème revient à trouver α et c tels que

$$\begin{cases} \max\limits_{\alpha \in \mathbb{R}^2, c \in \mathbb{R}} \dfrac{1}{\|\alpha\|} \\ \langle \alpha, z_i \rangle + c \geq +1 \quad \text{quand } f(z_i) = +1 \\ \langle \alpha, z_i \rangle + c \leq -1 \quad \text{quand } f(z_i) = -1 \end{cases}$$

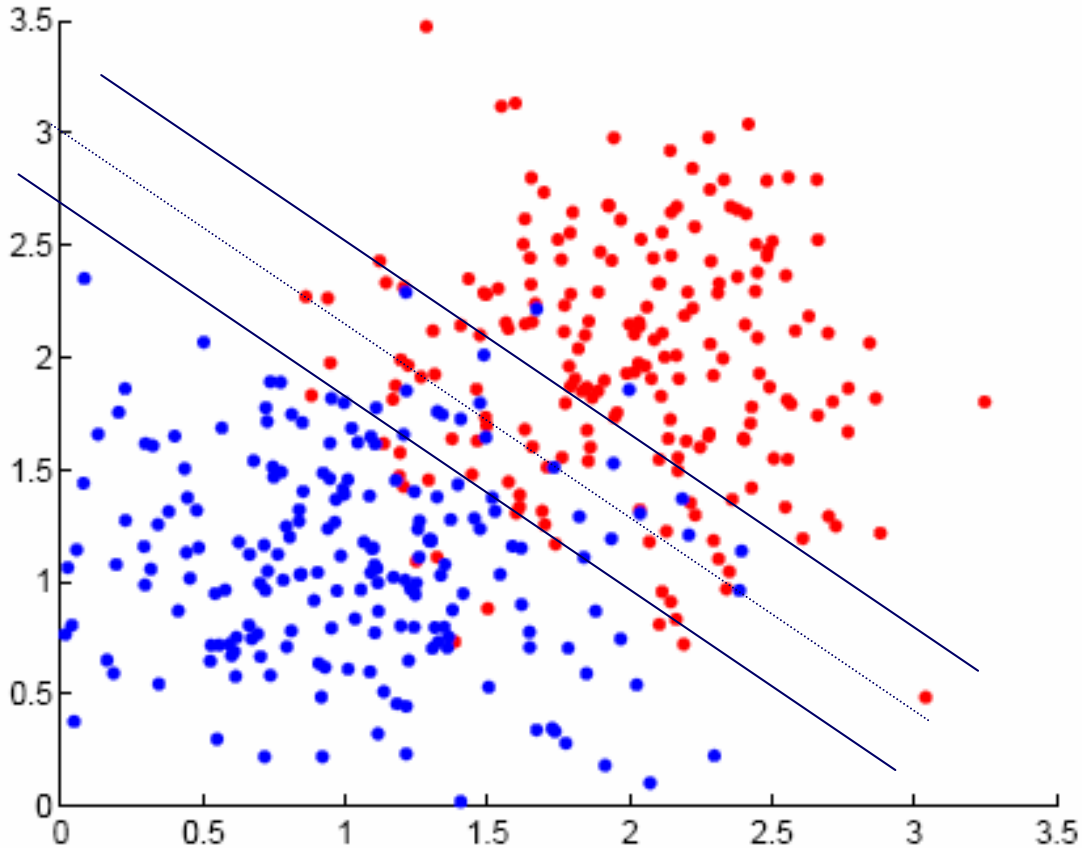
- et donc α et c tels que

$$\begin{cases} \min\limits_{\alpha \in \mathbb{R}^2, c \in \mathbb{R}} \|\alpha\| \\ f(z_i)\big(\langle \alpha, z_i \rangle + c\big) \geq 1 \quad \text{pour tout } i. \end{cases}$$

- C'est un problème d'optimisation classique.

Classes non séparables, première approche

- Que faire dans le cas où les deux nuages ne sont pas séparables ?



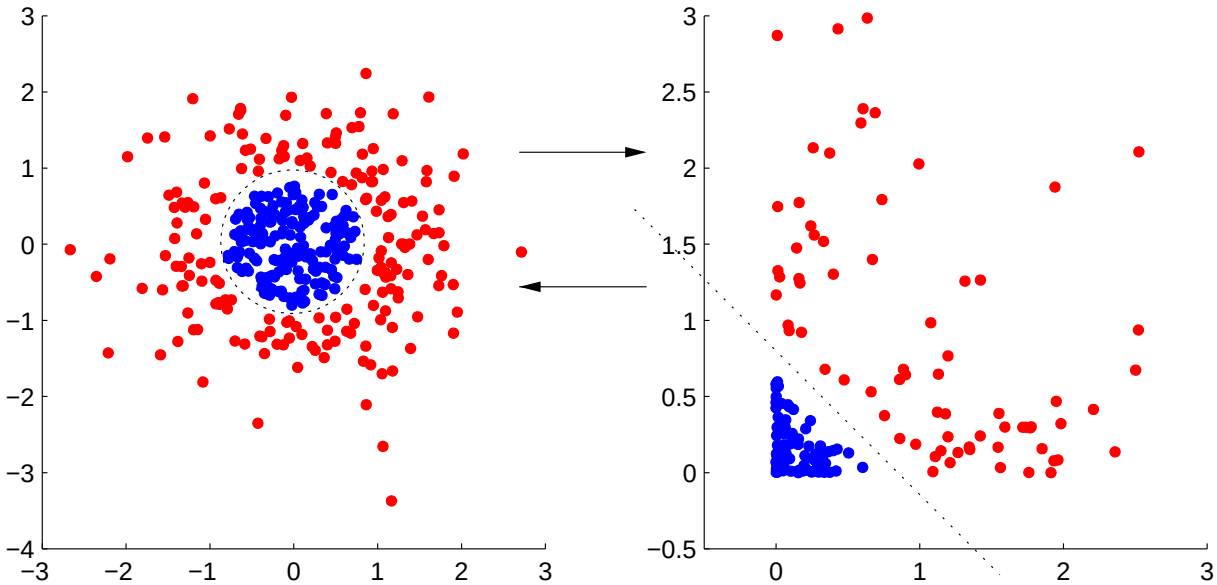
Classes non séparables, première approche

- Première approche : on fait au mieux avec ce qu'on a.
- On ajoute au système précédent des ξ_i positifs qui représentent l'écart entre un point z_i et la marge, quand il dépasse (et zéro sinon). On cherche à minimiser la somme des écarts :

$$\begin{cases} \min\limits_{\alpha\in\mathbb{R}^2, c\in\mathbb{R}, \xi\in\mathbb{R}^n} (\|\alpha\| + C\sum \xi_i) & \\ f(z_i)\big(\langle \alpha, z_i\rangle + c\big) \geq 1 - \xi_i & \text{pour tout } i \\ \xi_i \geq 0 & \text{pour tout } i. \end{cases}$$

Classes non séparables, deuxième approche

- Deuxième approche : on *triche* pour rendre la situation séparable. Dans la figure



on prend les points du diagramme de gauche et on leur applique la fonction $\chi(x, y) \mapsto (x^2, y^2)$.

- Alors les nuages deviennent séparables. Mais est-ce toujours possible ?
- La réponse est oui, *si l'espace vers lequel on envoie nos points est de dimension suffisamment grande, voire très très grande.*

Rappels d'analyse

- Un espace *préhilbertien* est un espace vectoriel E muni d'un produit scalaire $\langle\,.\,,\,.\,\rangle$.
- Un *produit scalaire* $\langle\,.\,,\,.\,\rangle : E \times E \longrightarrow \mathbb{R}$ est une application
  1. bilinéaire ($\langle \lambda x, y\rangle = \lambda\langle x, y\rangle$, $\langle x, \lambda y\rangle = \lambda\langle x, y\rangle$),
  2. symétrique ($\langle x, y\rangle = \langle y, x\rangle$),
  3. positive ($\langle x, x\rangle \geq 0$),
  4. définie ($\langle x, x\rangle = 0 \Rightarrow x = 0$).
- Exemple : le $\mathbb{R}$-espace vectoriel des fonctions continues $[a, b] \longrightarrow \mathbb{R}$ est préhilbertien si on le munit du produit scalaire $\langle f, g\rangle = \int_a^b f \cdot g$. Cet espace est de dimension infinie non dénombrable.

Retour à la classification

- Exemple : prenons comme espace d'arrivée un espace de fonctions, et pas n'importe lequel, l'espace des fonctions $k_\sigma(x,.)$ où

$$k_\sigma(x,.) : x' \mapsto k_\sigma(x,x') = \frac{1}{(2\pi\sigma^2)^{d/2}} e^{-\frac{\|x-x'\|}{2\sigma^2}}$$

  qui est la gaussienne de dimension d centrée en x. Notons que, par symétrie, $k_\sigma(.,x) = k_\sigma(x,.)$ dans l'espace des fonctions.
- Soit l'espace vectoriel $\mathcal{H}$ des fonctions $k_\sigma(x,.)$, il est préhilbertien pour le produit scalaire suivant :

$$\left\langle \sum_i \lambda_i k_\sigma(x_i,.), \sum_i \lambda'_i k_\sigma(x'_i,.) \right\rangle = \sum_i \lambda_i \lambda'_i k_\sigma(x_i,x'_i).$$

- Le changement de coordonnées sera donc

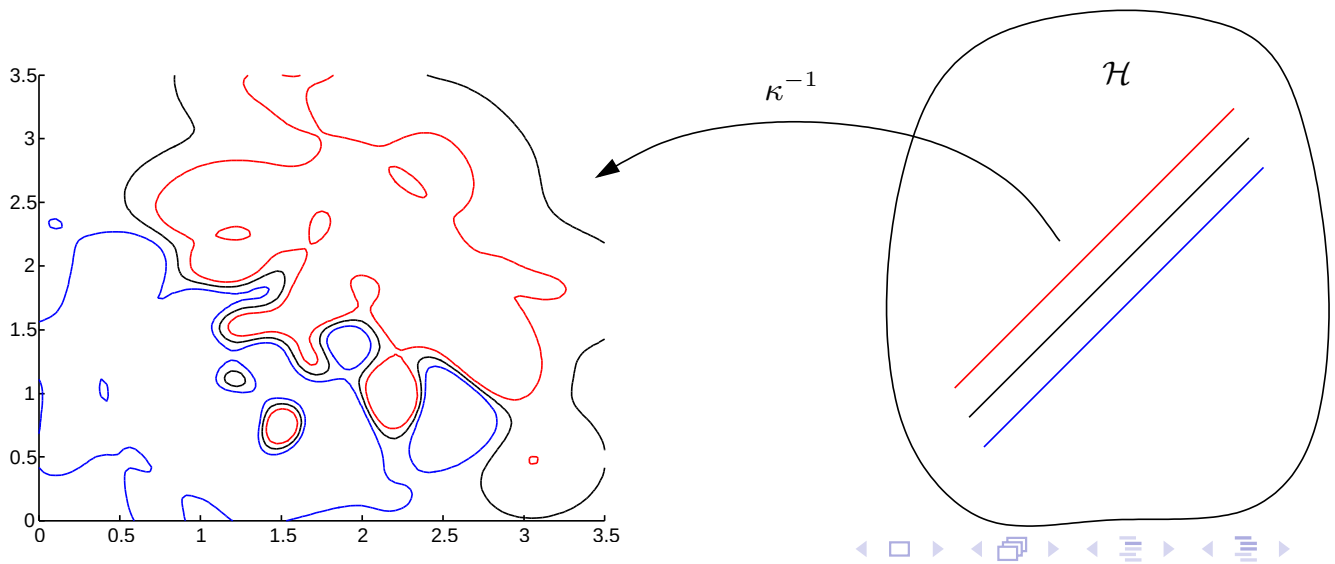
$$\kappa : \begin{array}{c} \mathbb{R}^n \to \mathcal{H} \\ x \mapsto k_\sigma(x,.). \end{array}$$

Classification

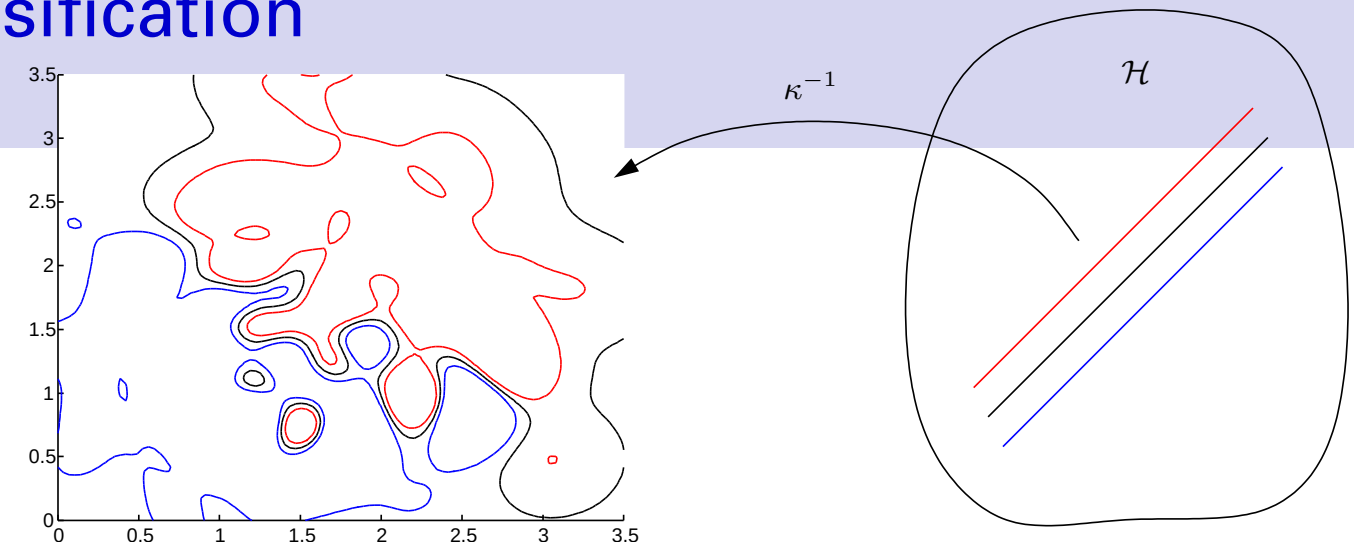
- Prenons un vecteur ω de $\mathcal{H}$, il s'écrit $\omega = \sum_i \lambda_i k_\sigma(x_i, .)$. Soit un hyperplan H de $\mathcal{H}$ donné par l'équation $\langle \omega, h \rangle = c$, quelle est son image réciproque dans $\mathbb{R}^n$?
- On a

$$\kappa^{-1}(H) = \{z \in \mathbb{R}^n \mid \sum_i \lambda_i k_\sigma(x_i, z) = c\}.$$

c'est-à-dire que c'est la surface de niveau c d'une somme pondérée de gaussiennes autour des points de nos nuages :



Classification



- Si $f(z_i)$ est la valeur du point z_i ($+1$ ou -1), si on prend $\omega = \sum_i f(z_i) k_\sigma(z_i, .)$ dans $\mathcal{H}$ et l'hyperplan H_0 tel que $\langle \omega, h \rangle = 0$ alors $\kappa^{-1}(H_0)$ sépare les deux nuages.
- Et le problème d'optimisation devient

$$\begin{cases} \min_{\alpha \in \mathcal{H}, c \in \mathbb{R}} \|\alpha\| \\ f(z_i)\big(\langle \alpha, \kappa(z_i) \rangle + c\big) \geq 1 \quad \text{pour tout } i. \end{cases}$$

ou alors

$$\begin{cases} \min_{\alpha \in \mathcal{H}, c \in \mathbb{R}, \xi \in \mathbb{R}^n} \big(\|\alpha\| + C \sum_i \xi_i\big) & \\ f(z_i)\big(\langle \alpha, \kappa(z_i) \rangle + c\big) \geq 1 - \xi_i & \text{pour tout } i \\ \xi_i \geq 0 & \text{pour tout } i. \end{cases}$$

En passant par le lagrangien

Théorème

Soit un problème de recherche d'extremum local avec contrainte : $\min \phi(x)$ avec $\psi(x) = 0$. Alors si x_0 est extremum local, il existe λ (un *multiplicateur de Lagrange*) tel que la différentielle de $L(x, \lambda) = \phi(x) + \lambda\psi(x)$ soit nulle en x_0.

En passant par le lagrangien

- Dans notre cas, cela donne :

$$L(\alpha, c, \lambda) = \frac{1}{2}\|\alpha\|^2 + \sum_i \lambda_i (1 - f(z_i)(\langle \alpha, z_i \rangle + c)).$$

- On a

$$\frac{\partial L}{\partial \alpha} = \alpha - \sum_i f(z_i)\lambda_i z_i$$
$$\frac{\partial L}{\partial c} = \sum_i \lambda_i z_i$$

et donc quand on annule les dérivées partielles,

$$\alpha = \sum_i f(z_i)\lambda_i z_i$$
$$\sum_i \lambda_i z_i = 0$$

Le problème dual

- Posons $\vartheta(\lambda) = \min_{\alpha,c} L(\alpha, c, \lambda)$.
- Après avoir placé les deux conditions de minimalité du slide précédent dans le lagrangien, la théorie nous dit que le problème d'optimisation est équivalent au problème dual

$$\begin{cases} \max_\lambda \vartheta(\lambda) = \max_\lambda -\frac{1}{2}\sum_{i,j} \lambda_i\lambda_j f(z_i)f(z_j)\langle z_i, z_j\rangle \\ \lambda \geq 0. \end{cases}$$

- qui devient comme suit, quand on passe par un noyau k_σ:

$$\begin{cases} \max_\lambda \vartheta(\lambda) = \max_\lambda -\frac{1}{2}\sum_{i,j} \lambda_i\lambda_j f(z_i)f(z_j)k_\sigma(z_i, z_j) \\ \lambda \geq 0. \end{cases}$$

L'astuce du noyau

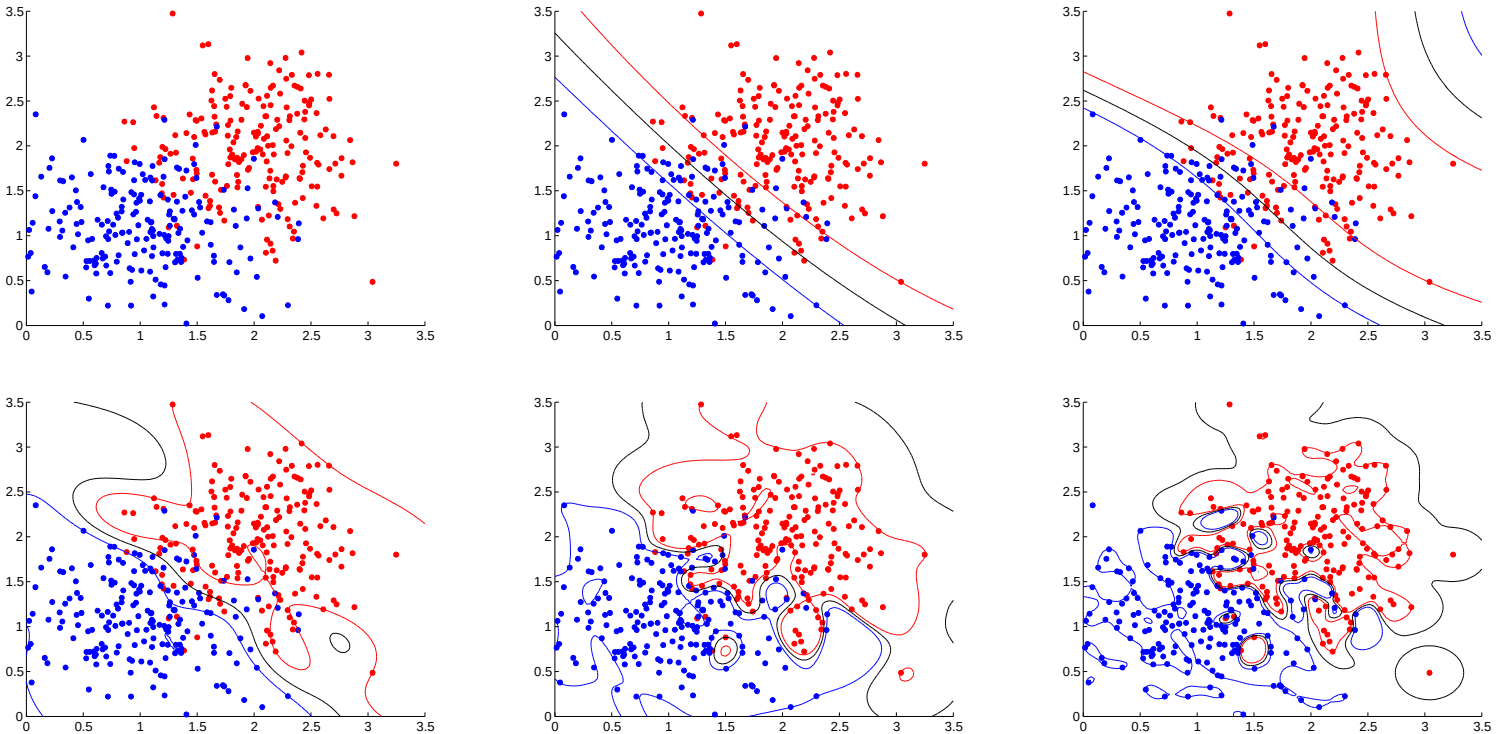
- En lisant la formule suivante que constatez-vous de crucial ?

$$\begin{cases} \max_\lambda -\frac{1}{2}\sum_{i,j}\lambda_i\lambda_j f(z_i)f(z_j)k_\sigma(z_i,z_j) \\ \lambda \geq 0. \end{cases}$$

- On appelle cela l'*astuce du noyau* (*the kernel trick*), et c'est grâce à cette astuce que l'on peut mener à terme des calculs qui sinon nous placeraient dans des espaces de dimension infinie.
- Citation : *One interesting property of support vector machines and other kernel-based systems is that, once a valid kernel function has been selected, one can practically work in spaces of any dimension without any significant additional computational cost, since feature mapping is never effectively performed. In fact, one does not even need to know which features are being used.*

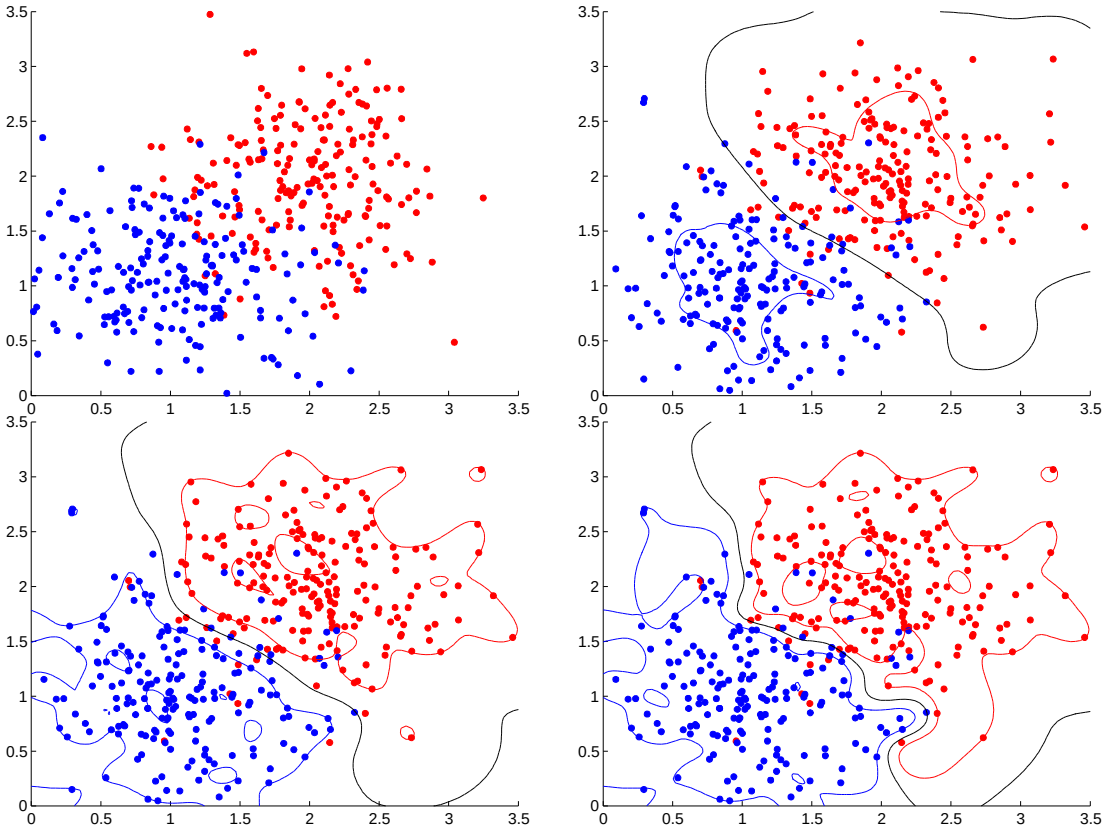
L'effet du paramètre σ sur le SVM

Yannis Haralambous
(IMT Atlantique)



Dans ces graphiques l'écart-type σ décroît.

L'effet du paramètre C sur le SVM



Dans ces graphiques le coût C croît.

Petite démo bricolée de toutes pièces

- Les données : maladie chronique des reins, Inde, 2015. 400 individus, 25 propriétés, valeurs numériques et modales, présence de valeurs manquantes. `http://archive.ics.uci.edu/ml/datasets/Chronic_Kidney_Disease`
- Les classes : malade (1), non malade (-1).
- Récupérer, comprendre le format ARFF, nettoyer les données (blancs, tabulations parasites). Données nettoyées sur Moodle (`donnees`$_r$*eins.arff*).
- L'outil : *SVMLight* `https://www.cs.cornell.edu/people/tj/svm_light/index.html#References`
- Comprendre le format demandé par l'outil, convertir les données.

Petite démo bricolée de toutes pièces

- Premier programme (`convert_data.pl`) : convertir les données ARFF en SVMLight, passer les valeurs modales en numériques, remplacer les valeurs manquantes par les moyennes des valeurs existantes.
- format ARFF :

```
@attribute 'age' numeric
@attribute 'bp'  numeric
@attribute 'sg' {1.005,1.010,1.015,1.020,1.025}
@attribute 'al' {0,1,2,3,4,5}
@attribute 'su' {0,1,2,3,4,5}
@attribute 'rbc' {normal,abnormal}
...
@data
48,80,1.020,1,0,?,normal,notpresent,notpresent,121,36,1.2,?,
...
```

- format *SVMLight* :

```
1 1:48 2:80 3:3 4:1 5:0 6:0.189516129032258 7:0 8:1 9:1 10:12
```

```
#!/usr/bin/perl
$index_attr=-1;
$data_line=-1;
while (<>) {
if (m/\@attribute/) {
$index_attr++;
if (m/^\@attribute '[^']+'[ ]+numeric/) {
   $CHANGE[$index_attr]="numeric";
   }
elsif (m/^\@attribute '[^']+' \{(.+?)\}/) {
   @VALUES=split /,/,$1;
   $CHANGE[$index_attr]="modal";
   $index_val=-1;
   foreach $x (@VALUES) {
      $index_val++;
      ${"CHANGE".$index_attr}{$x}=$index_val;
      }
   }
```

```
}
elsif (m/\@data/) { $READ_DATA=1; }
elsif ($READ_DATA==1 and m/./) {

$data_line++;
chomp;
@DATA=split /,/, $_;
$index=-1;
foreach $x (@DATA) {

$index++;
if ($CHANGE[$index] eq "numeric") {
   if ($x eq "?") {
      ${"DATALINE".$data_line}[$index]="XXX"; }
   else {
      ${"DATALINE".$data_line}[$index]=$x;
      ${"MEAN".$index}+=$x;
      ${"LINES".$index}++; }
```

```
    }
else {
    if ($x eq "?") {
        ${"DATALINE".$data_line}[$index]="XXX"; }
    else {
        ${"DATALINE".$data_line}[$index]=
            ${"CHANGE".$index}{$x};
        ${"MEAN".$index}+=${"CHANGE".$index}{$x};
        ${"LINES".$index}++; }
}}
$MAX_index=$index;}}

# calculer moyennes
foreach $index (0 .. $MAX_index) {
if (${"LINES".$index} > 0) {
$MEAN[$index]=${"MEAN".$index} / ${"LINES".$index};
}}
$MAX_LINES=$data_line;
```

```
# écrire le résultat
foreach $i (0 .. $MAX_LINES) {
@TMP=();
$index=-1;

if (@{"DATALINE".$i}[$MAX_index]==0) { print "1 "; }
if (@{"DATALINE".$i}[$MAX_index]==1) { print "-1 "; }

foreach $index (0 .. $MAX_index-1) {
if (@{"DATALINE".$i}[$index] eq "XXX") {
push @TMP, ($index+1).":".$MEAN[$index]
} else {
push @TMP, ($index+1).":".@{"DATALINE".$i}[$index]
}
}
print join(" ",@TMP)."\n";
}
```

Petite démo bricolée de toutes pièces

- Deuxième programme (`split_donnees.pl`) : préparer la validation 10-croisée.

```
use List::Util qw(shuffle);
open IN, "donnees.dat";
while (<IN>) { push @ORIG, $_; }
close IN;
@BIG = shuffle @ORIG;
$taille = $#BIG;
$part=int(($taille+1)/10);

foreach $I (0 .. 9) {
open TEST, ">test$I"; open TRAI, ">trai$I";
foreach $Z (0 .. $taille) {
   if ($Z >= $I * $part and $Z < ($I+1) * $part) {
   print TEST $BIG[$Z];
   } else { print TRAI $BIG[$Z]; }
}
close TEST; close TRAI;
```

Petite démo bricolée de toutes pièces

- Troisième programme (`cross-validation.pl`) : lancer *SVMLight* autant de fois que nécessaire et calculer les moyennes des résultats.

```
foreach $I (0 .. 9) {
print "Iteration $I...\n";
system("svm_learn -z c -c 1.0 -t 1 -d 2 trai$I model > res$I");
open IN, "res$I"; while (<IN>) {
if (m/Runtime in cpu-seconds: ([0-9.]+)/) { $sec += $1; }
} close IN;
system("svm_classify test$I model out$I > res$I");
open IN, "res$I";
while (<IN>) {
if (m/Accuracy on test set: ([0-9.]+)\%/) { $acc += $1; }
if (m|Precision/recall on test set: ([0-9.]+)\%/([0-9.]+)\%|) {
   $prec += $1; $rec += $2; }
} close IN; }
$acc /= 10; $prec /= 10; $rec /= 10;
print "Time: $sec sec, Accuracy: $acc, Precision: $prec, Recall:
```

Rappel : justesse, précision, rappel

- Justesse (*accuracy*) :

$$\frac{\text{vrais positifs} + \text{faux négatifs}}{\text{total}}$$

- Précision :

$$\frac{\text{vrais positifs}}{\text{vrais positifs} + \text{faux positifs}}$$

- Rappel :

$$\frac{\text{vrais positifs}}{\text{vrais positifs} + \text{vrais négatifs}}$$

- (positif/négatif = prédiction, vrai/faux = réalité).