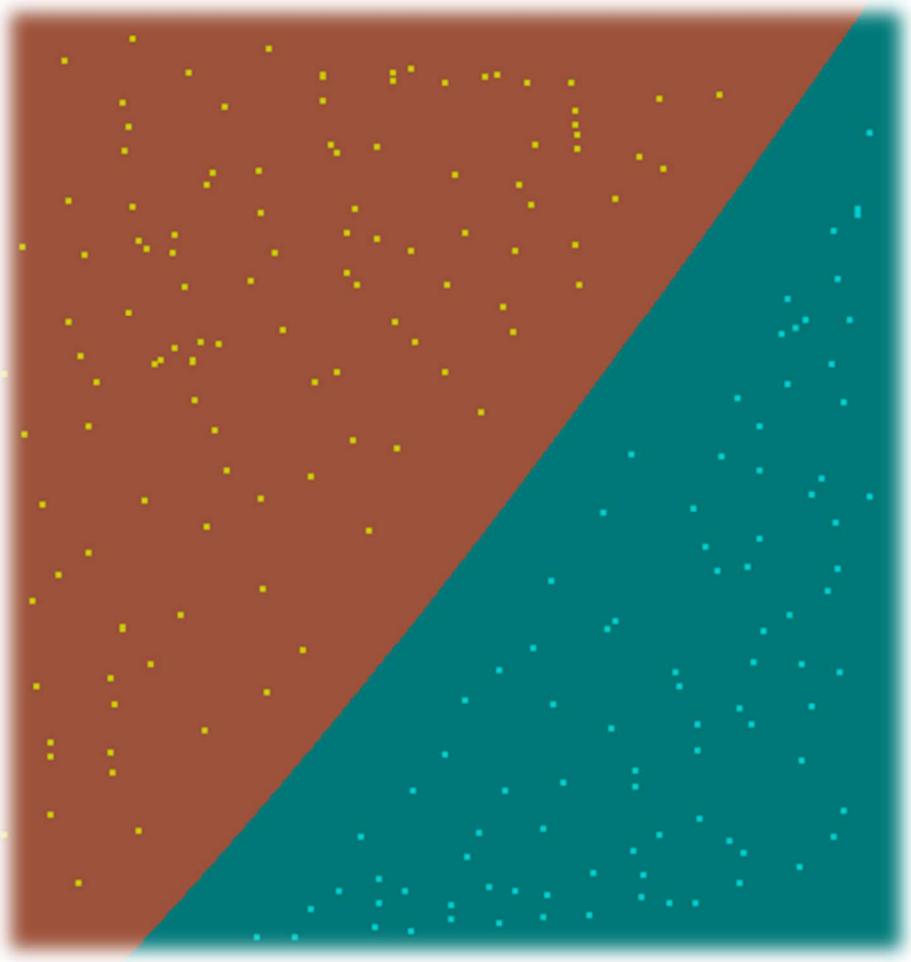
Jacques Cellier

SVM, bases mathématiques

Support Vector Machine

Séparateur à Vaste Marge



Dans les années 1980, la star de l'IA était le langage Prolog fondé sur la logique. En ces jours, où la vague de l'IA déferle, les méthodes se fondent sur l'apprentissage statistique. Parmi ces méthodes, SVM est l'objet de ce petit document. Il existe, bien sûr de nombreuses sources sur le sujet. Mais après consultation de plusieurs d'entre elles, j'ai éprouvé une certaine frustration. Dans la plupart de ces documents, les fondements mathématiques (avec démonstrations) étaient traités assez légèrement au profit de la mise en œuvre pratique. Ce qui m'a conduit à rédiger ce texte.

Bien que ce document ne prétende nullement être un guide pratique de SVM, je fournis 3 exemples en illustration : deux exemples « jouets » à finalité pédagogique et un exemple avec des données réelles. Les outils utilisés sont le Solveur d'Excel, Tanagra et R package e1071.

Les prérequis sont :

- ✓ Le théorème *Karush-Kuhn-Tucker* version optimisation convexe.
- ✓ Des connaissances d'algèbre linéaire, en particulier la manipulation des matrices par blocs.
- ✓ Un minimum de familiarité avec les logiciels précités.

Optimisation quadratique

Soit h et $g_1 \cdots g_n$ fonctions $\mathbb{R}^p \to \mathbb{R}$, on s'intéresse au problème d'optimisation sous contrainte (s.c.) suivant :

$$(P)\begin{cases}\min\limits_{v\in\mathbb{R}^p} h(v)\\ s.c.\ \forall i \in 1\cdots n : g_i(v) - c_i \le 0\end{cases}$$

En vue des applications au SVM, on supposera le problème quadratique :

1. $h(v) = \frac{1}{2} v^t A v + B^t v$ où $A \in \mathcal{M}_{p\times p}(\mathbb{R})$ semi-définie positive et $B \in \mathbb{R}^p$
2. $\forall i \in 1\cdots n : g_i$ est une forme linéaire sur $\mathbb{R}^p$

Remarques :

1. Les conditions $g_i(v) - c_i \le 0$ peuvent s'écrire matriciellement : $Gv - C \le 0$ où $G \in \mathcal{M}_{n\times p}(\mathbb{R})$ et $C \in \mathbb{R}^n$.
2. $V = \{v \in \mathbb{R}^p / Gv - C \le 0\}$, autrement dit l'ensemble des vecteurs qui vérifient les contraintes est un ensemble convexe non vide. Le problème (P) peut donc encore s'écrire : $\min\limits_{v\in V} h(v)$.
3. $\nabla h(v) = Av + B$; $\nabla^2 h(v) = A$.
4. La matrice A étant semi-définie positive, la fonction h est convexe. Tout minimum local est un minimum global sur V.

   DEMONSTRATION :

   Soit $\tilde{v}$ un minimum local de f sur V. Supposons qu'il existe $v \in V$ tel que $f(v) \le f(\tilde{v})$ et posons $y_t = tv + (1-t)\tilde{v}\ t \in]0\,1[$. Alors par convexité de f, $f(y_t) \le t f(v) + (1-t) f(\tilde{v})$. Pour $t \in]0\,1[$, $t f(v) + (1-t) f(\tilde{v}) < f(\tilde{v})$, d'où : $f(y_t) < f(\tilde{v})$.

   Par ailleurs, il existe un voisinage U de $\tilde{v}$ tel que pour tout $x \in U$, $f(\tilde{v}) \le f(x)$. Pour t suffisamment petit, $y_t \in U$ et alors $f(\tilde{v}) \le f(y_t)$, contradiction.
5. Si, de plus, la matrice A est définie positive, la fonction h est strictement convexe et la solution est unique.

   DEMONSTRATION :

   Supposons qu'il existe deux minimums distincts $\tilde{v}_1$ et $\tilde{v}_2$, selon 4., ils sont globaux et $f(\tilde{v}_1) = f(\tilde{v}_2) = m$. Par convexité stricte : $f\left(\frac{\tilde{v}_1 + \tilde{v}_2}{2}\right) < \frac{f(\tilde{v}_1) + f(\tilde{v}_2)}{2} = m$, contradiction.

Karush-Kuhn-Tucker

Dans le cadre de l'optimisation avec h convexe de classe C^2 et des contraintes affines on peut appliquer le théorème *Karush-Kuhn-Tucker* version « forte » :

$$\left|\begin{array}{l} \tilde{v} \text{ solution de } (P) \\ \qquad \Updownarrow \\ \exists \tilde{\alpha} \in \mathbb{R}_+^n \text{ tel que :} \\ \nabla h(\tilde{v}) + \sum_{i=1}^{n} \tilde{\alpha}_i \nabla g_i(\tilde{v}) = 0 \qquad (1) \\ \qquad \text{et} \\ \forall i \in 1 \cdots n : \tilde{\alpha}_i \left(g_i(\tilde{v}) - c_i \right) = 0 \quad (2) \end{array}\right.$$

Les $\tilde{\alpha}_i$ se nomment les multiplicateurs de Lagrange. L'interprétation de cette deuxième égalité est importante. Elle signifie que si $0 < \tilde{\alpha}_i$, alors la contrainte est saturée $\left(g_i(\tilde{v}) - c_i = 0\right)$.

Lagrangien

Le lagrangien du problème est alors défini pour $v \in \mathbb{R}^p, \alpha \in \mathbb{R}_+^n$ par :

$$\mathcal{L}(v, \alpha) = h(v) + \sum_{i=1}^{n} \alpha_i \left(g_i(v) - c_i \right)$$

Or, pour $\alpha \in \mathbb{R}_+^n$ et pour $v \in V$: $g_i(v) - c_i \leq 0$, donc $\sum_{i=1}^{n} \alpha_i \left(g_i(v) - c_i \right) \leq 0$. Il en résulte que :

$$\text{Pour } v \in V, \ \max_{\alpha \in \mathbb{R}_+^n} \mathcal{L}(v, \alpha) = h(v).$$

Dans le cadre de l'optimisation convexe, on a :

$$\min_{v \in \mathbb{R}^p} \max_{\alpha \in \mathbb{R}_+^n} \mathcal{L}(v, \alpha) = \max_{\alpha \in \mathbb{R}_+^n} \min_{v \in \mathbb{R}^p} \mathcal{L}(v, \alpha)$$

Le problème (P), dit primal peut donc s'écrire : $\min_{v \in V} \max_{\alpha \in \mathbb{R}_+^n} \mathcal{L}(v, \alpha)$ et est équivalent au problème (D), dit dual : $\max_{\alpha \in \mathbb{R}_+^n} \min_{v \in \mathbb{R}^p} \mathcal{L}(v, \alpha)$.

Point-selle du Lagrangien

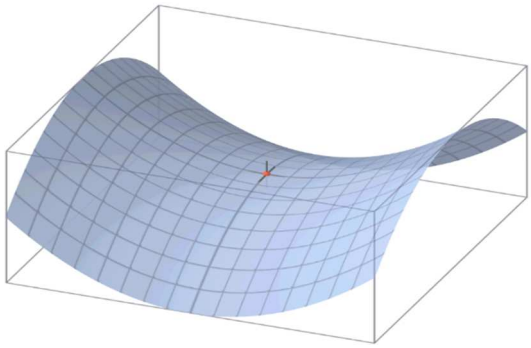
$(\tilde{v}, \tilde{\alpha}) \in \mathbb{R}^p \times \mathbb{R}_+^n$ est un point-selle du lagrangien si et seulement si :

$$\forall v \in \mathbb{R}^p, \forall \alpha \in \mathbb{R}_+^n : \mathcal{L}(\tilde{v}, \alpha) \leq \mathcal{L}(\tilde{v}, \tilde{\alpha}) \leq \mathcal{L}(v, \tilde{\alpha})$$

Autrement dit :

$$\mathcal{L}(\tilde{v}, \tilde{\alpha}) = \max_{\alpha \in \mathbb{R}_+^n} \mathcal{L}(\tilde{v}, \alpha) = \min_{v \in \mathbb{R}^p} \mathcal{L}(v, \tilde{\alpha})$$

Si $(\tilde{v}, \tilde{\alpha})$ est solution de (P) ou (D), c'est alors un point-selle du lagrangien.



Retour au problème quadratique

✓ Si $h(v)=\frac{1}{2}v^tAv+B^tv$, alors : $\nabla h(v)=Av+B$.

✓ Si $g_i(v)=G_{i,\bullet}v$ forme linéaire, alors $\nabla g_i(v)=G_{i,\bullet}^t$ et $\sum_{i=1}^{n}\tilde{\alpha}_i\nabla g_i(v)=G^t\tilde{\alpha}$.

L'égalité (1) s'écrit alors matriciellement : $A\tilde{v}+B+G^t\tilde{\alpha}=0$.

L'égalité (2) s'écrit alors matriciellement : $Diag(\tilde{\alpha})(G\tilde{v}-C)=0$

Le lagrangien s'écrit alors :

$$\mathcal{L}(v,\alpha)=h(v)+\sum_{i=1}^{n}\alpha_i\left(g_i(v)-c_i\right)=\frac{1}{2}v^tAv+B^tv+\alpha^t(Gv-C)$$

Résumé optimisation quadratique $A\in\mathcal{M}_p(\mathbb{R})$ semi-définie positive, $B\in\mathbb{R}^p$, $G\in\mathcal{M}_{n\times p}(\mathbb{R})$	
Primal	Dual
$\begin{cases}\min\limits_{v\in\mathbb{R}^p}\left(\frac{1}{2}v^tAv+B^tv\right)\\ s.c.\ Gv-C\le 0\end{cases}$ Si $\tilde{v}$ solution du primal	$\max\limits_{\alpha\in\mathbb{R}_+^n}\left(\mathcal{L}_D(\alpha)\right)$ Où $\mathcal{L}_D(\alpha)=\min\limits_{v\in\mathbb{R}^p}\left(\frac{1}{2}v^tAv+B^tv+\alpha^t(Gv-C)\right)$ Si $\tilde{\alpha}$ solution du dual
$\tilde{v}$ et $\tilde{\alpha}$ sont liés par les relations : $A\tilde{v}+B+G^t\tilde{\alpha}=0$ (3) et $Diag(\tilde{\alpha})(G\tilde{v}-C)=0$ (4)	

Application à SVM

Deux nuages linéairement séparables, marge rigide (*hard-margin*)

Données :

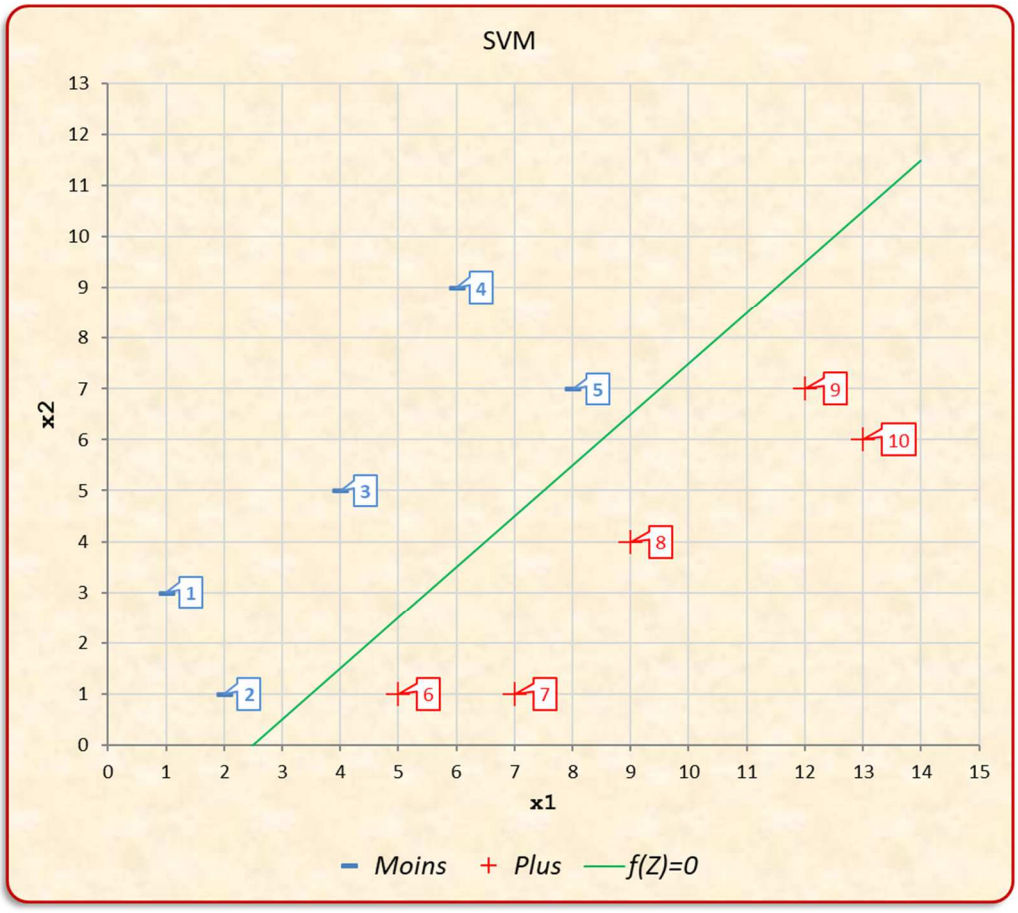
> $X \in M_{n\times p}(\mathbb{R})$ n vecteurs (ou points) dans $(\mathbb{R}^p)^*$
>
> à chaque point $X_{i,\bullet} = (x_{i,1}, \cdots, x_{i,p}) \in (\mathbb{R}^p)^*$ on associe une étiquette $y_i \in \{-1, 1\}$

Chacune des p colonnes de X correspond à une variable numérique, chacune des n ligne, aux valeurs des variables pour un individu.

Avertissement

On a un hiatus entre deux conventions. D'une part l'algèbre linéaire où les éléments de $\mathbb{R}^p$ sont représentés par des vecteurs colonnes. D'autre part la statistique où dans un tableau de données (*data frame*) les vecteurs décrivant les individus sont écrits en ligne. Compte tenu de l'isomorphisme canonique d'espaces euclidiens entre $\mathbb{R}^p$ et son dual $(\mathbb{R}^p)^*$, cela n'a pas de conséquence sauf dans les écritures matricielles du genre $f(Z) = Z\beta + \beta_0 = 0$ où Z doit être un vecteur-ligne et β un vecteur colonne.

On suppose que les données sont linéairement séparables, c'est-à-dire qu'il existe un hyperplan H_0 de $(\mathbb{R}^p)^*$ d'équation $f(Z) = Z\beta + \beta_0 = 0$ tel que $0 < y_i f(X_{i,\bullet})$.



Ensuite on se met en quête du « meilleur » hyperplan séparateur. En langage imagé et en dimension 2, on le définit comme celui qui va définir la « route » la plus large possible entre les deux nuages. Les marges de cette « route » sont définies par les équations $f(Z)=a$ et $f(Z)=-a$ de deux hyperplans H_1 et H_{-1} parallèles à H_0. Comme les coefficients des équations sont définies à une constante multiplicative près, les équations peuvent se ramener à : $f(Z)=1$ et $f(Z)=-1$ Les conditions de séparation s'expriment : $\forall i \in 1\cdots n:\ 1 \le y_i\left(X_{i,\bullet}\beta+\beta_0\right)$.

La distance d'un point M à l'hyperplan H_0 est donné par : $d\left(M,H_0\right)=\dfrac{\left|M\beta+\beta_0\right|}{\|\beta\|}$. Si $M \in H_1$, $M\beta+\beta_0=1$ et $d\left(M,H_0\right)=\dfrac{1}{\|\beta\|}$. La marge qui sépare les deux hyperplans H_1 et H_{-1} est donc $d\left(H_1,H_{-1}\right)=\dfrac{2}{\|\beta\|}$ La maximalisation de cette marge conduit à résoudre le problème d'optimisation suivant :

$$\beta=\begin{pmatrix}\beta_1\\ \vdots\\ \beta_p\end{pmatrix}\in\mathbb{R}^p,\ \beta_0\in\mathbb{R},$$

$$\min_{\beta,\beta_0}\left(\frac{\|\beta\|^2}{2}\right)\ \text{s.c.}\ \forall i\in 1\cdots n:\ 1\le y_i\left(X_{i,\bullet}\beta+\beta_0\right)$$

Il s'agit d'un problème d'optimisation quadratique sous contrainte affine avec :

$$v=\begin{pmatrix}\beta_0\\ \beta\end{pmatrix}\in\mathbb{R}^{1+p}\ ;\ A=Diag\left(0,1\cdots 1\right)\in\mathcal{M}_{(1+p)\times(1+p)}\left(\mathbb{R}\right);\ B=0$$

$$G=-Diag\left(y\right)\left(\vec{1}\mid X\right)\in\mathcal{M}_{n\times(p+1)}\left(\mathbb{R}\right)\ ;\ C=-\vec{1}\ \text{où}\ \vec{1}=\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix},\ y=\begin{pmatrix}y_1\\ \vdots\\ y_n\end{pmatrix}\in\mathbb{R}^n$$

L'égalité (3) donne :

$$A\tilde{v}+B+G^t\tilde{\alpha}=0\Rightarrow\begin{pmatrix}0\\ \tilde{\beta}\end{pmatrix}-\begin{pmatrix}\vec{1}^t\\ X^t\end{pmatrix}Diag\left(y_1,\cdots,y_n\right)\tilde{\alpha}=0\Rightarrow\begin{cases}\displaystyle\sum_{i=1}^n y_i\tilde{\alpha}_i=0\Leftrightarrow y^t\tilde{\alpha}=0\ (5)\\ \tilde{\beta}=\displaystyle\sum_{i=1}^n\tilde{\alpha}_i y_i X_{i,\bullet}^t\Leftrightarrow\tilde{\beta}=X^t Diag(y)\tilde{\alpha}\ (6)\end{cases}$$

L'égalité (2) ou (4) donne :

$$\forall i\in 1\cdots n:\tilde{\alpha}_i\left(y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)-1\right)=0\Leftrightarrow Diag(\alpha)\left(\vec{1}-Diag\left(y\right)\left(\beta_0\vec{1}\mid X\beta\right)\right)=0$$

Lagrangien

Le lagrangien s'écrit :

$$\mathcal{L}(v,\alpha)=\frac{1}{2}\beta^t\beta+\sum_{i=1}^{n}\alpha_i\left(-y_i\left(X_{i,\bullet}\beta+\beta_0\right)+1\right)=\frac{1}{2}\beta^t\beta-\sum_{i=1}^{n}\alpha_i y_i X_{i,\bullet}\beta-\sum_{i=1}^{n}\alpha_i y_i\beta_0+\sum_{i=1}^{n}\alpha_i$$

$$=\frac{1}{2}\beta^t\beta-\alpha^t Diag(y)X\beta-\beta_0\sum_{i=1}^{n}\alpha_i y_i+\alpha^t\vec{1}$$

Pour $(\tilde{v},\tilde{\alpha})$ point-selle du lagrangien :

$$\mathcal{L}(\tilde{v},\tilde{\alpha})==\frac{1}{2}\tilde{\beta}^t\tilde{\beta}-\tilde{\alpha}^t Diag(y)X\tilde{\beta}-\tilde{\beta}_0\underbrace{\sum_{i=1}^{n}\tilde{\alpha}_i y_i}_{=0\ (5)}+\tilde{\alpha}^t\vec{1}.$$

En utilisant (6) : $\mathcal{L}(\tilde{v},\tilde{\alpha})=\frac{1}{2}\tilde{\alpha}^t Diag(y)XX^t Diag(y)\tilde{\alpha}-\alpha^t Diag(y)XX^t Diag(y)\tilde{\alpha}+\alpha^t\vec{1}$. Au final :

$$\mathcal{L}(\tilde{v},\tilde{\alpha})=\tilde{\alpha}^t\vec{1}-\frac{1}{2}\tilde{\alpha}^t Diag(y)XX^t Diag(y)\tilde{\alpha}$$

Or $\mathcal{L}(\tilde{v},\alpha)\le\mathcal{L}(\tilde{v},\tilde{\alpha})$, donc $\tilde{\alpha}=\underset{\alpha\in\mathbb{R}_+^n}{\arg\max}\left(\alpha^t\vec{1}-\frac{1}{2}\alpha^t Diag(y)XX^t Diag(y)\alpha\right)$

Le problème dual s'écrit donc :

$$\boxed{\max_{\alpha\in\mathbb{R}_+^n}\left(\alpha^t\vec{1}-\frac{1}{2}\alpha^t Diag(y)XX^t Diag(y)\alpha\right)\ \text{s.c.}\ \alpha^t y=0}$$

Vecteur support

Les égalités : $\forall i\in 1\cdots n:\tilde{\alpha}_i\left(y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)-1\right)$ montrent que si $0<\tilde{\alpha}_i$, alors la contrainte i est saturée : $y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)=1$. On définit alors $S=\{i\in 1\cdots n/0<\alpha_i\}$. Les $X_{i,\bullet},i\in S$, sont appelés vecteurs supports.

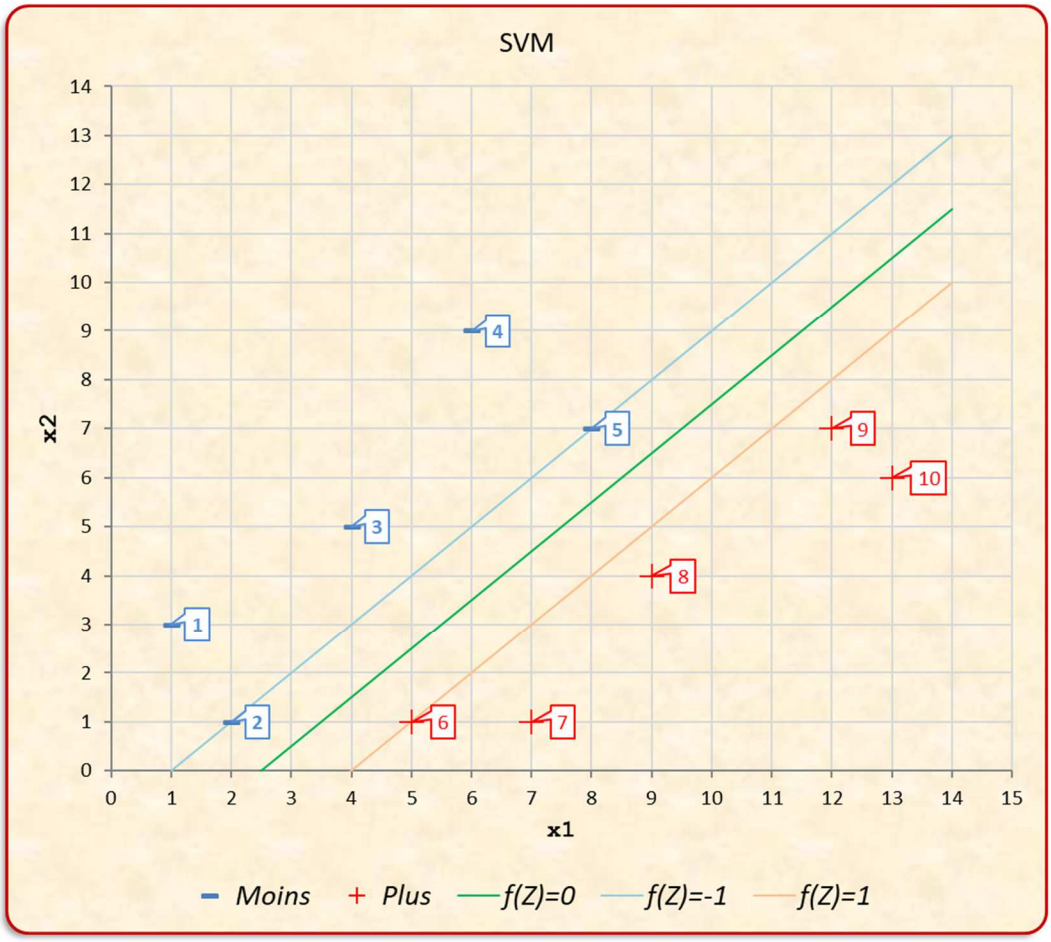
Du dual, retour vers le primal

Soit $\tilde{\alpha}\in\mathbb{R}_+^n$ solution du dual, alors (6) permet de calculer $\tilde{\beta}=\sum_{i=1}^{n}\tilde{\alpha}_i y_i X_{i,\bullet}^t=\sum_{i\in S}\tilde{\alpha}_i y_i X_{i,\bullet}^t$.

Si $i\in S\ y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)=1\Leftrightarrow\tilde{\beta}_0=\frac{1-y_i X_{i,\bullet}\tilde{\beta}}{y_i}=\underbrace{y_i-X_{i,\bullet}\tilde{\beta}}_{\text{car } y_i^2=1}$

Remarques :

- ✓ Le calcul de $\tilde{\beta},\tilde{\beta}_0$ ne fait intervenir que les vecteurs supports.
- ✓ Ce calcul effectué, si $Z=\left(z_1,\cdots,z_p\right)\in\left(\mathbb{R}^p\right)^*$ représente les valeurs des variables pour un nouvel individu, son classement se fait à partir du signe de la valeur de la fonction score $f(Z)=Z\tilde{\beta}+\tilde{\beta}_0$.



Les vecteurs supports sont **6**, **5** *et* **2**

Résumé marge rigide	
Primal	Dual
$\beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_p \end{pmatrix} \in \mathbb{R}^p, \beta_0 \in \mathbb{R}, \quad f(X_{i,\bullet}) = X_{i,\bullet}\beta + \beta_0$ $\min_{\substack{\beta \in \mathbb{R}^p \\ \beta_0 \in \mathbb{R}}} \left(\frac{\|\beta\|^2}{2} \right)$ sous la contrainte : $y_i f(X_{i,\bullet}) \geq 1$	$\max_{\alpha \in \mathbb{R}_+^n} \mathcal{L}_D(\alpha)$ s.c. : $\sum_{i=1}^n \alpha_i y_i = 0$ $\mathcal{L}_D(\alpha) = \alpha^t \vec{1} - \frac{1}{2} \alpha^t Diag(y) X X^t Diag(y) \alpha$ $\mathcal{L}_D(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{i=1}^n \alpha_i y_i \langle X_{i,\bullet} ; X_{j,\bullet} \rangle y_j \alpha_j$ S ensemble des $i \in 1 \cdots n \ \alpha_i \neq 0$ Si $i \in S$, $X_{i,\bullet}$ est un vecteur support $\tilde{\beta} = \sum_{i \in S} \tilde{\alpha}_i y_i X_{i,\bullet}^t$ et $\tilde{\beta}_0 = y_i - X_{i,\bullet}\tilde{\beta}$ $(i \in S)$
Fonction score : $f(Z) = Z\tilde{\beta} + \tilde{\beta}_0 = \sum_{i \in S} \tilde{\alpha}_i y_i \langle X_{i,\bullet} ; Z \rangle + \tilde{\beta}_0$	

Deux nuages non linéairement séparables, marge souple (*soft-margin*)

Avec des vraies données, la séparabilité linéaire est peu probable. Pour y remédier, on introduit des variables d'écart ξ_i (*slack variables* ou variables ressort) et la condition de « bon classement » $1 \le y_i\left(X_{i,\bullet}\beta+\beta_0\right)$ est modifiée en $1\text{-}\xi_i \le y_i\left(X_{i,\bullet}\beta+\beta_0\right)$ afin de tolérer le mauvais classement de certains points :

$$\begin{cases}\xi_i = 0 \text{ point } i \text{ du bon côté de sa marge}\\ 0<\xi_i<1 \text{ point } i \text{ entre les 2 marges}\\ 1\le\xi_i \text{ point } i \text{ mal classé}\end{cases}$$

Le problème s'énonce alors :

$$\beta=\begin{pmatrix}\beta_1\\ \vdots\\ \beta_p\end{pmatrix}\in\mathbb{R}^p,\ \beta_0\in\mathbb{R}, \xi=\begin{pmatrix}\xi_1\\ \vdots\\ \xi_n\end{pmatrix}\in\mathbb{R}^n$$

$$\min_{\beta,\beta_0,\xi}\left(\frac{\|\beta\|^2}{2}+\Gamma\sum_{i=1}^{n}\xi_i\right) \text{ s.c. } \forall i\in 1\cdots n:\ 1\text{-}\xi_i \le y_i\left(X_{i,\bullet}\beta+\beta_0\right) \text{ et } 0\le\xi_i$$

$0\le\Gamma$ [1] est un paramètre dit de coût (*cost*)qui règle la tolérance aux erreurs.

Là encore il s'agit d'un problème d'optimisation quadratique avec :

Rappel de notations : $y=\begin{pmatrix}y_1\\ \vdots\\ y_n\end{pmatrix}, \vec{\mathbf{1}}=\begin{pmatrix}1\\ \vdots\\ 1\end{pmatrix}\in\mathbb{R}^n$

$$v=\begin{pmatrix}\beta_0\\ \hline \beta\\ \hline \xi\end{pmatrix}\in\mathbb{R}^{1+p+n}\ ;\ A=Diag\left(0,\underbrace{1\cdots1}_{p\ fois},\underbrace{0\cdots0}_{n\ fois}\right)\in\mathcal{M}_{(1+p+n)\times(1+p+n)}(\mathbb{R})\ ;\ B=\Gamma\begin{pmatrix}0\\ \vdots\\ 0\\ \hline \vec{\mathbf{1}}\end{pmatrix}\in\mathbb{R}^{1+p+n}$$

Les n premières contraintes s'écrivent :

$$-y_i\beta_0-y_iX_{i,\bullet}\beta-\xi_i\le-1$$

Les n suivantes :

$$-\xi_i\le0$$

Ce qui se traduit par :

$$G=-\left(\begin{array}{c|c|c}y & Diag(y)X & I\\ \hline 0 & 0 & I\end{array}\right)\in\mathcal{M}_{2n\times(n+p+1)}(\mathbb{R})\ ;C=\begin{pmatrix}-\vec{\mathbf{1}}\\ \hline 0\\ \vdots\\ 0\end{pmatrix}\in\mathbb{R}^{2n}$$

Comme on a $2n$ contraintes, on aura $2n$ multiplicateurs de Lagrange que l'on notera : $\begin{pmatrix}\tilde{\alpha}\\ \hline \tilde{\delta}\end{pmatrix}\in\mathbb{R}_+^{2n}$

[1] Ce paramètre est traditionnellement noté $\boldsymbol{C}$, mais il est déjà utilisé dans le texte.

Si $\tilde{v}=\begin{pmatrix}\tilde{\beta}_0\\ \hline \tilde{\beta}\\ \hline \tilde{\xi}\end{pmatrix}$ solution du primal, et $\begin{pmatrix}\tilde{\alpha}\\ \hline \tilde{\delta}\end{pmatrix}=\in \mathbb{R}_+^{2n}$ solution du dual, l'égalité (3) donne :

$$A\tilde{v}+B+G^t\begin{pmatrix}\tilde{\alpha}\\ \hline \tilde{\delta}\end{pmatrix}=0\Rightarrow\begin{pmatrix}0\\ \hline \tilde{\beta}\\ \hline 0\end{pmatrix}+\Gamma\begin{pmatrix}0\\ \hline 0\\ \vdots\\ \hline 0\\ \hline \vec{1}\end{pmatrix}-\left(\begin{array}{c|c}y^t & 0\\ \hline X^tDiag(y) & 0\\ \hline I & I\end{array}\right)\begin{pmatrix}\tilde{\alpha}\\ \hline \tilde{\delta}\end{pmatrix}=0\Rightarrow\begin{cases}y^t\tilde{\alpha}=0\Leftrightarrow\sum_{i=1}^n y_i\tilde{\alpha}_i=0\ (5)\\ \tilde{\beta}=X^tDiag(y)\tilde{\alpha}\Leftrightarrow\tilde{\beta}=\sum_{i=1}^n\tilde{\alpha}_i y_i X_{i,\bullet}^t\ (6)\\ \Gamma\vec{1}=\tilde{\alpha}+\tilde{\delta}\Leftrightarrow\forall i\in 1\cdots n:\tilde{\delta}_i=\Gamma-\tilde{\alpha}_i\ (7)\end{cases}$$

REMARQUES :

- ✓ Il n'est nul besoin de connaitre $\tilde{\delta}$ pour calculer $\tilde{\beta}$.
- ✓ L'égalité (7) impose $\tilde{\alpha}\le\Gamma\vec{1}$.

En appliquant (4) $Diag(\tilde{\alpha})(G\tilde{v}-C)=0$, on obtient :

$$Diag\begin{pmatrix}\tilde{\alpha}\\ \hline \tilde{\delta}\end{pmatrix}\left(-\left(\begin{array}{c|c|c}y & Diag(y)X & I\\ \hline 0 & 0 & I\end{array}\right)\begin{pmatrix}\tilde{\beta}_0\\ \hline \tilde{\beta}\\ \hline \tilde{\xi}\end{pmatrix}+\begin{pmatrix}\vec{1}\\ \hline 0\\ \vdots\\ \hline 0\end{pmatrix}\right)=0\Leftrightarrow\begin{cases}Diag(\tilde{\alpha})\left(-y\tilde{\beta}_0-Diag(y)X\tilde{\beta}-\tilde{\xi}+\vec{1}\right)=0\\ Diag(\tilde{\delta})\tilde{\xi}=0\end{cases}$$

$$\Leftrightarrow\begin{cases}\forall i\in 1\cdots n:\tilde{\alpha}_i\left(y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)-\left(1-\tilde{\xi}_i\right)\right)=0\ (8)\\ \forall i\in 1\cdots n:\tilde{\xi}_i\left(\Gamma-\tilde{\alpha}_i\right)=0\ (9)\end{cases}$$

Lagrangien

$$\begin{aligned}\mathcal{L}(v,\alpha,\delta)&=\frac{1}{2}\beta^t\beta+\Gamma\sum_{i=1}^n\xi_i-\sum_{i=1}^n\alpha_i\left(y_i\left(X_{i,\bullet}\beta+\beta_0\right)-1+\xi_i\right)-\sum_{i=1}^n\delta_i\xi_i\\ &=\frac{1}{2}\beta^t\beta+\Gamma\sum_{i=1}^n\xi_i-\sum_{i=1}^n\alpha_i y_i\left(X_{i,\bullet}\beta\right)-\beta_0\sum_{i=1}^n\alpha_i y_i+\sum_{i=1}^n\alpha_i-\sum_{i=1}^n(\alpha_i+\delta_i)\xi_i\\ &=\frac{1}{2}\beta^t\beta+\Gamma\sum_{i=1}^n\xi_i-\alpha^tDiag(y)X\beta-\beta_0\sum_{i=1}^n\alpha_i y_i+\sum_{i=1}^n\alpha_i-\sum_{i=1}^n(\alpha_i+\delta_i)\xi_i\end{aligned}$$

Pour $(\tilde{v},\tilde{\alpha},\tilde{\delta})$ point-selle du lagrangien, en tenant compte de (7) et (5) :

$$\mathcal{L}(\tilde{v},\tilde{\alpha})=\frac{1}{2}\tilde{\beta}^t\tilde{\beta}+\cancel{\Gamma\sum_{i=1}^n\tilde{\xi}_i}-\tilde{\alpha}^tDiag(y)X\tilde{\beta}-\tilde{\beta}_0\underbrace{\sum_{i=1}^n\tilde{\alpha}_i y_i}_{=0\ (5)}+\sum_{i=1}^n\tilde{\alpha}_i-\cancel{\sum_{i=1}^n\underbrace{(\tilde{\alpha}_i+\tilde{\delta}_i)}_{=\Gamma\ (7)}\tilde{\xi}_i}$$

$$=\tilde{\alpha}^t\vec{1}+\frac{1}{2}\tilde{\beta}^t\tilde{\beta}-\tilde{\alpha}^tDiag(y)X\tilde{\beta}$$

Dès lors il suffit d'utiliser le (6) pour conclure comme dans le cas rigide. Le problème dual s'écrit alors :

$$\boxed{\max_{\alpha\in\mathbb{R}_+^n}\left(\alpha^t\vec{1}-\frac{1}{2}\alpha^tDiag(y)XX^tDiag(y)\alpha\right)\ \text{s.c.}\ \alpha^ty=0\ \text{et}\ \alpha\le\Gamma\vec{1}}$$

Vecteur support

Comme pour le cas rigide, l'égalité (8) permet de définir les vecteurs supports par $0<\tilde{\alpha}_i$, ce qui implique la saturation de la contrainte : $y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)=1-\tilde{\xi}_i$. Les vecteurs supports se partagent en deux types :

- ✓ Type S_I : $\tilde{\alpha}_i<\Gamma$, ce qui implique, selon (9) la nullité de la variable d'écart $\tilde{\xi}_i=0\Rightarrow y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)=1$. Le vecteur se situe sur une marge.
- ✓ Type S_{II} : $\tilde{\alpha}_i=\Gamma$.

Du dual, retour vers le primal

Soit $\tilde{\alpha}\in\mathbb{R}_+^n$ solution du dual, alors (6) permet de calculer $\tilde{\beta}=\sum_{i=1}^{n}\tilde{\alpha}_i y_i X_{i,\bullet}^t=\sum_{i\in S}\tilde{\alpha}_i y_i X_{i,\bullet}^t$.

Pour calculer $\tilde{\beta}_0$, il faut utiliser un vecteur support de type I qui vérifie $y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)=1$.

Pour $Z\in(\mathbb{R}^p)^*$, on a alors la fonction score : $f(Z)=Z\tilde{\beta}+\tilde{\beta}_0.$,

Les $\tilde{\xi}_i$ peuvent se calculer ainsi :

1. Si $X_{i,\bullet}$ est un vecteur support $\tilde{\xi}_i=1-y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)$.
2. Sinon $\tilde{\alpha}_i=0$ et (9) implique $\tilde{\xi}_i=0$.

On a donc : $\xi_i=\max\left(0,1-y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)\right)$

Résumé marge souple	
Primal	Dual
$\min_{\substack{\beta\in\mathbb{R}^p,\\ \beta_0\in\mathbb{R},\\ \xi_i\in\mathbb{R}^n}}\left(\frac{\|\beta\|^2}{2}+\Gamma\sum_{i=1}^{n}\xi_i\right)$ Γ constante à fixer s.c. : $1-\xi_i\le y_i\left(X_{i,\bullet}\beta+\beta_0\right)$ et $0\le\xi_i$ ξ_i : variables d'écart $\xi_i=0$ point i du bon côté de sa marge $0<\xi_i<1$ point i entre les 2 marges $1\le\xi_i$ point i mal classé $\tilde{\xi}_i=\max\left(0,1-y_i\left(X_{i,\bullet}\tilde{\beta}+\tilde{\beta}_0\right)\right)$	$\max_{\alpha\in\mathbb{R}_+^n}\mathcal{L}_D(\alpha)$ s.c. : $0\le\alpha_i\le\Gamma$ et $\sum_{i=1}^{n}\alpha_i y_i=0$ $\mathcal{L}_D(\alpha)=\alpha^t\vec{1}-\frac{1}{2}\alpha^t Diag(y)XX^t Diag(y)\alpha$ $\mathcal{L}_D(\alpha)=\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i y_i\left\langle X_{i,\bullet};X_{j,\bullet}\right\rangle y_j\alpha_j$ S ensemble des $i\in 1\cdots n$ $\alpha_i\neq 0$ Si $i\in S$, $X_{i,\bullet}$ est un vecteur support de type I si $\xi_i=0$, de type II sinon $\tilde{\beta}=\sum_{i\in S}\alpha_i y_i X_{i,\bullet}^t$ et $\tilde{\beta}_0=y_i-X_{i,\bullet}\tilde{\beta}$ $(i\in S_1)$
Fonction score : $f(Z)=Z\tilde{\beta}+\tilde{\beta}_0=\sum_{i\in S}\tilde{\alpha}_i y_i\left\langle X_{i,\bullet};Z\right\rangle+\tilde{\beta}_0$	

Exemple d'après Ricco Rakotomalala (classeur Excel `ex1_dual.xlsx`)

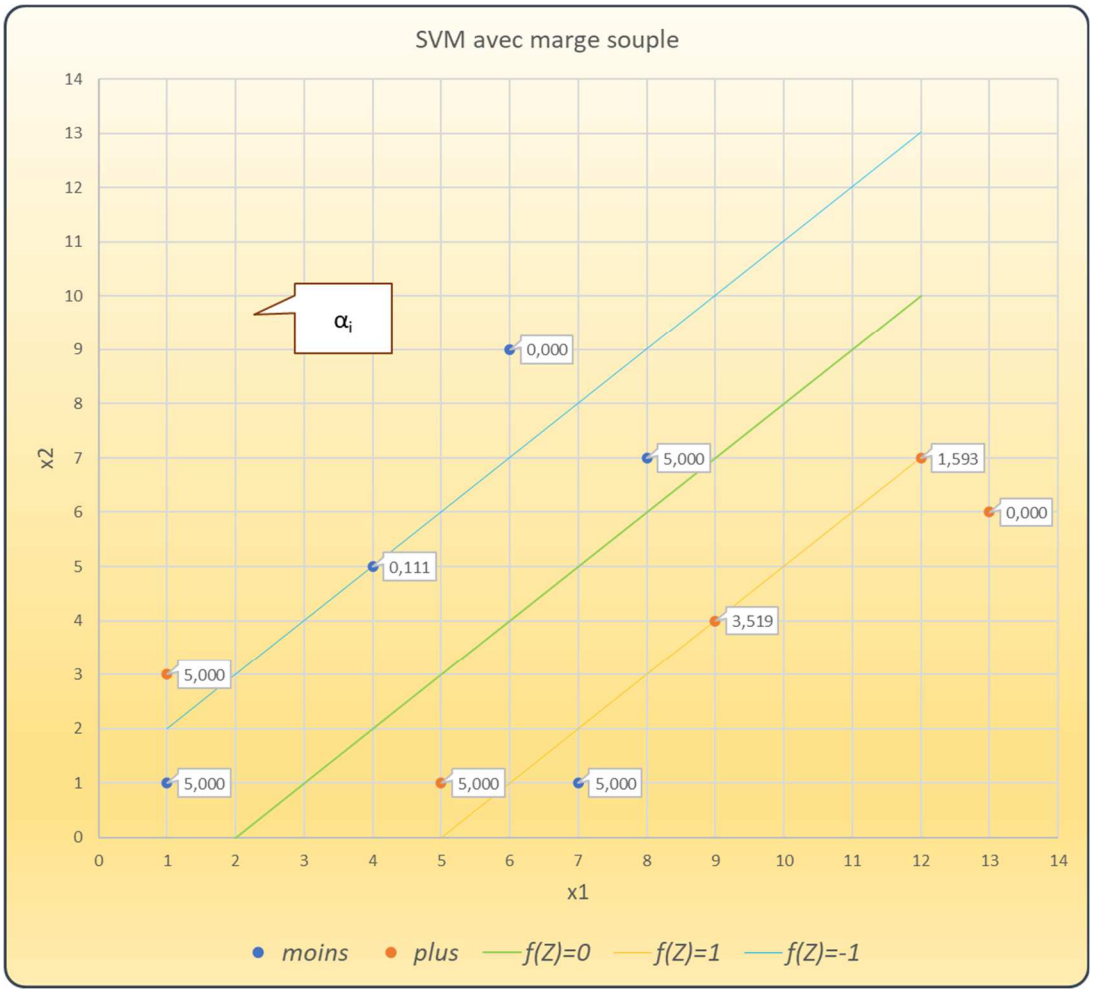
individus	x_1	x_2	y	$\tilde{\alpha}$	$\tilde{\xi}$	$f(x_1, x_2)$
1	1	1	-1	5,000	0,333	-0,667
2	4	5	-1	0,111	0,000	-1,000
3	6	9	-1	0,000	0,000	-1,667
4	8	7	-1	5,000	0,667	-0,333
5	7	1	-1	5,000	2,333	1,333
6	1	3	1	5,000	2,333	-1,333
7	5	1	1	5,000	0,333	0,667
8	13	6	1	0,000	0,000	1,667
9	9	4	1	3,519	0,000	1,000
10	12	7	1	1,593	0,000	1,000

$\Gamma = 5$

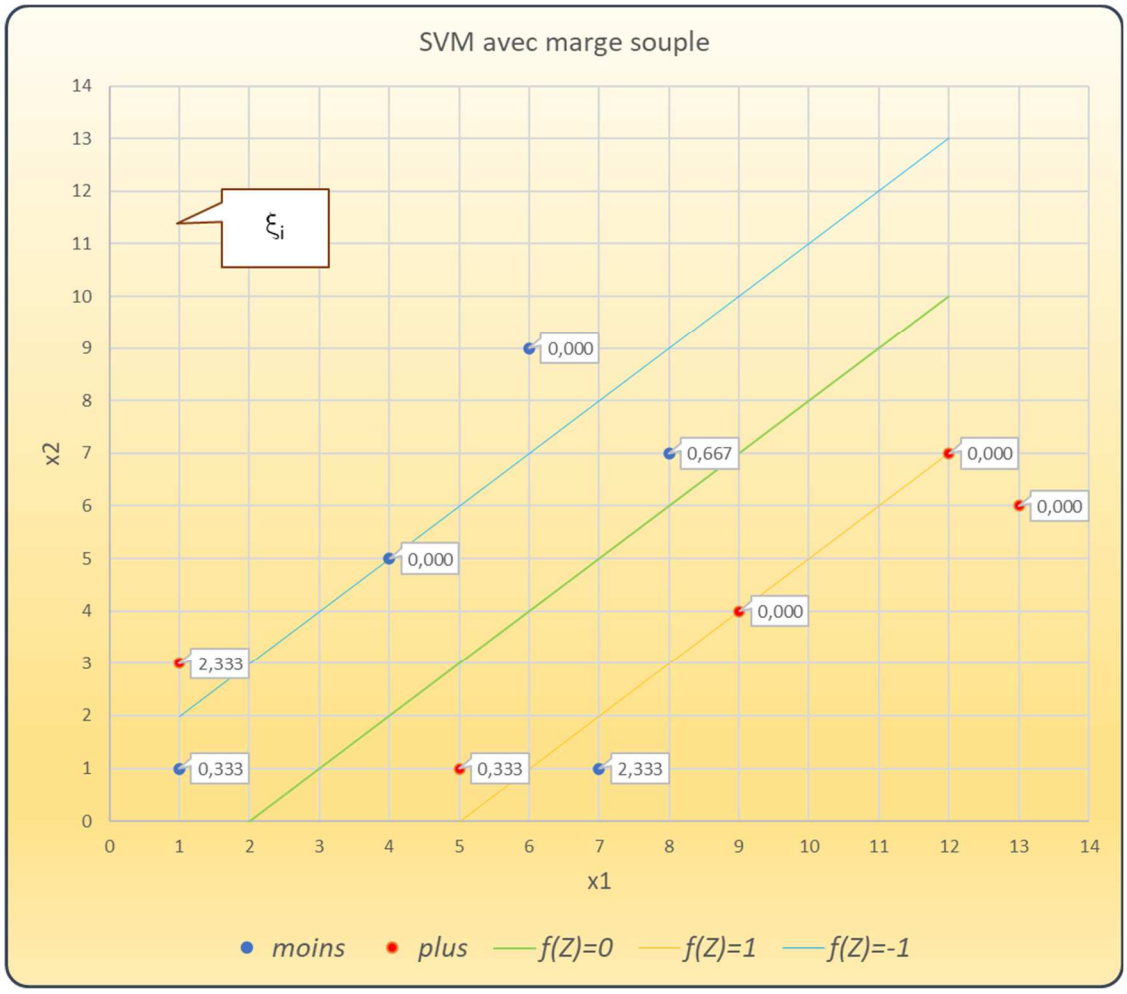
S_I

S_{II}

$\tilde{\beta}_0$	-0,667
$\tilde{\beta}$	0,333
	-0,333



Sauf (6,9) *et* (13,6) *tous les vecteurs sont supports, mais seuls* (4,5), (12,7), (9,4) *sont de type I*



$\xi_i = 0$ point i du bon côté de sa marge

$0 < \xi_i < 1$ point i entre les 2 marges

$1 \leq \xi_i$ point i mal classé

Au sujet du traitement de cet exemple

Les calculs ont été effectués d'une part avec solveur d'Excel[2], de l'autre par le logiciel Tanagra de Ricco Rakotomalala.

	TANAGRA est un logiciel gratuit de data science (data mining, machine learning, statistique) destiné à l'enseignement et à la recherche. Il implémente une série de méthodes de fouilles de données issues du domaine de la statistique exploratoire, de l'analyse de données, de l'apprentissage automatique et des bases de données.
Site :	https://tanagra-machine-learning.blogspot.com/

Un des avantages pratiques de Tanagra est son intégration comme complément d'Excel (`Tanagra.xla`) Après sélection des données, il suffit d'aller dans le menu **Compléments** d'Excel pour lancer Tanagra. Autre avantage, les résultats sont retournés sous forme de page HTML, que l'on peut par un copier/coller intégrer directement dans une feuille Excel ou un document Word. C'est ce qui a été fait pour la page suivante.

[2] Cf. `ex1_dual.xlsx`

Supervised Learning 1 (SVM)

Parameters

SVM Parameters	
Exponent	1
Filter type	NONE
Use polynom space normalization	0
Use RBF kernel	0
Gamma for RBF kernel	0,01
Complexity	5
Calculation parameter	
Epsilon for rounding	1,00E-12
Tolerance for accuracy	1,00E-03

Results

Classifier performances

Error rate			0,2			
Values prediction			**Confusion matrix**			
Value	**Recall**	**1-Precision**		**moins**	**plus**	**Sum**
moins	0,8	0,2	**moins**	4	1	5
plus	0,8	0,2	**plus**	1	4	5
			Sum	5	5	10

Classifier characteristics

Data description

Target attribute	y (2 values)
# descriptors	2

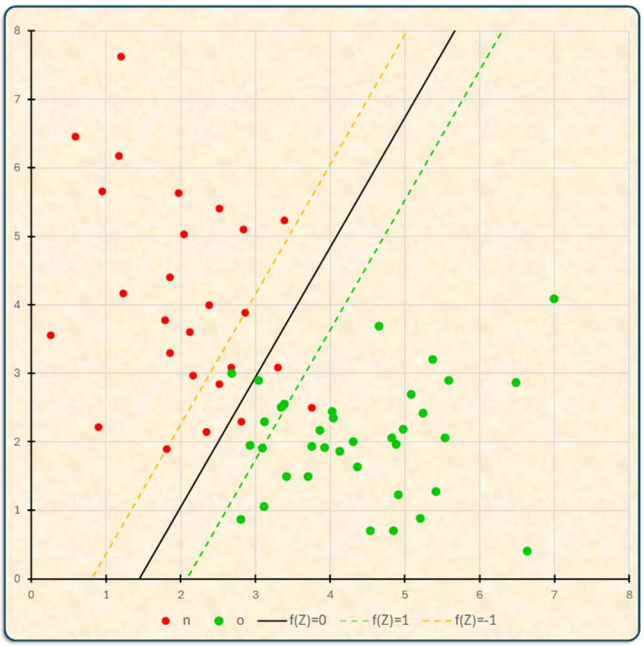
Linear classifier

"Reference" class value : plus

Attribute	Weight	
x1	0,333163	$\tilde{\beta}$
x2	-0,332652	
constant	-0,668626	$\tilde{\beta}_0$

Du choix de la constante Γ (`cost`)

Avec un petit jeu de données fabriqué pour l'occasion[3], voici ce que l'on peut observer :

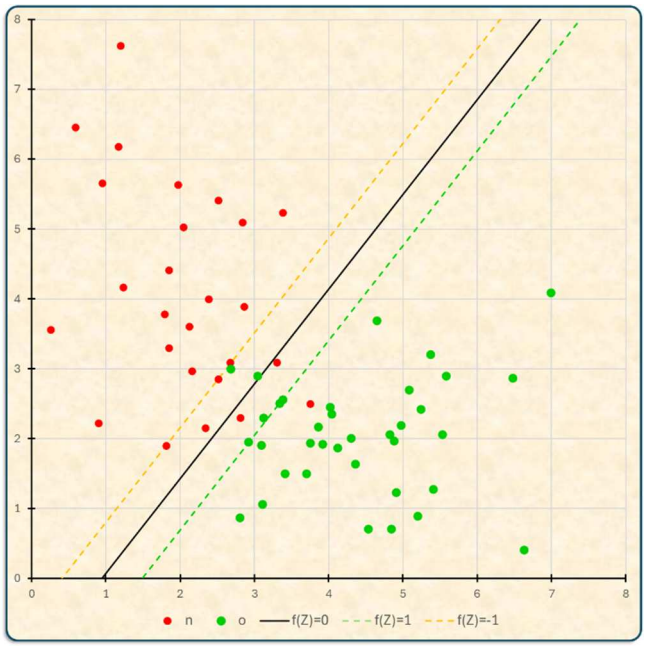


`cost = 1`

$f(Z) \simeq 1{,}57\, z_1 - 0{,}83\, z_2 - 2{,}27$

$marge = \dfrac{2}{\|\beta\|} \simeq 1{,}13 \ ; \ \sum_{i=1}^{n} \xi_i \simeq 9{,}56$

1 ● en zone rouge, 3 ● en zone verte



`cost = 100`

$f(Z) \simeq 1{,}85\, z_1 - 1{,}36\, z_2 - 1{,}76$

$marge = \dfrac{2}{\|\beta\|} \simeq 0{,}87 \ ; \ \sum_{i=1}^{n} \xi_i \simeq 8{,}96$

2 ● en zone rouge, 3 ● en zone verte

Soit, en passant de 1 à 100 :

- ✓ Un changement dans la fonction score.
- ✓ Une réduction de la marge.
- ✓ Une réduction de la somme des écarts (variables ressorts).
- ✓ Une augmentation des mal classés.

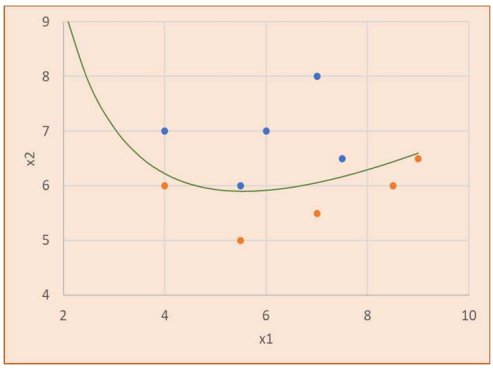
On peut avancer les explications suivantes :

La fonction à minimiser $\dfrac{\|\beta\|^2}{2} + \Gamma \sum_{i=1}^{n} \xi_i$ est la somme de deux termes positifs. Le premier conditionne la marge, le second les écarts. Augmenter Γ, c'est mettre l'accent sur la minimisation de la somme des écarts au détriment de la minimisation de $\|\beta\|^2$, donc de la maximalisation de la marge.

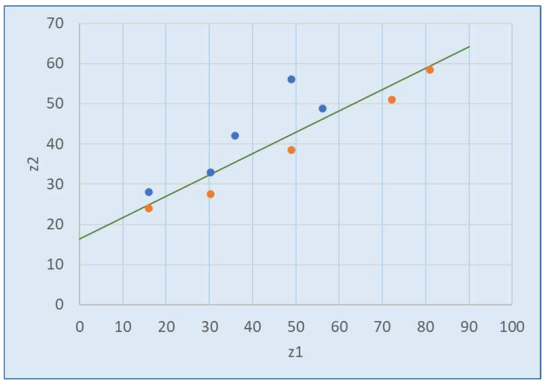
[3] Cf. `ex2_effet_cost.xlsx`

Séparation non linéaire

La première idée est d'utiliser une transformation $\mathbb{R}^p \xrightarrow{\Phi} H$ où H désigne un espace Pré-Hilbertien réel de dimension supérieure à p (voire infini) nommé espace de redescription, avec l'espoir que le nuage de points transformé soit linéairement séparable. Exemple :



$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \overset{\Phi}{\mapsto} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} x_1^2 \\ x_1 x_2 \end{pmatrix}$$



Une question demeure : comment déterminer Φ ?

La fonction à maximiser dans la formulation duale peut s'écrire :

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \left\langle X_{i,\bullet} \,;\, X_{j,\bullet} \right\rangle y_i y_j$$

Si on effectue la transformation Φ, elle s'écrira :

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j \left\langle \Phi\left(X_{i,\bullet}\right) ; \Phi\left(X_{j,\bullet}\right) \right\rangle y_i y_j$$

On définit la fonction « noyau » :

$$\mathbb{R}^p \times \mathbb{R}^p \xrightarrow{K} \mathbb{R} \text{ par } K(U,V) = \left\langle \Phi(U) ; \Phi(V) \right\rangle = \Phi(U)\Phi(V)^t$$

Dans le petit exemple introductif, le noyau est : $K(U,V) = u_1 v_1 \left(u_1 v_1 + u_2 v_2\right)$

La deuxième idée est d'utiliser des fonctions noyaux K pour remplacer le produit scalaire, sans expliciter Φ.

L'astuce du noyau (kernel trick) en détail

Le problème primal transporté dans l'espace de redescription H s'écrit :

$$\min_{\substack{\beta \in H, \\ \beta_0 \in \mathbb{R}, \\ \xi \in \mathbb{R}^n}} \left(\frac{\|\beta\|^2}{2} + \Gamma \sum_{i=1}^{n} \xi_i \right) \quad \text{s.c. : } 1 - \xi_i \leq y_i \left(\left\langle \Phi\left(X_{i,\bullet}\right) ; \beta \right\rangle + \beta_0 \right) \text{ et } 0 \leq \xi_i$$

Et le problème dual :

$$\max_{\alpha \in \mathbb{R}_+^n} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i y_i \underbrace{\left\langle \Phi\left(X_{i,\bullet}\right) ; \Phi\left(X_{j,\bullet}\right) \right\rangle}_{=K\left(X_{i,\bullet} ; X_{j,\bullet}\right)} y_j \alpha_j \quad \text{s.c. : } 0 \leq \alpha_i \leq \Gamma \text{ et } \sum_{i=1}^{n} \alpha_i y_i = 0$$

Soit $\tilde{\alpha}$ solution du dual, alors :

$$\tilde{\beta}=\sum_{i\in S}\tilde{\alpha}_i y_i \Phi\left(X_{i,\bullet}^t\right) \text{ et } \tilde{\beta}_0 = y_k - \left\langle \Phi\left(X_{k,\bullet}^t\right); \tilde{\beta} \right\rangle \ (k\in S_{\mathrm{I}})$$

$$\tilde{\beta}_0 = y_k - \left\langle \Phi\left(X_{k,\bullet}^t\right); \sum_{i\in S}\tilde{\alpha}_i y_i \Phi\left(X_{i,\bullet}^t\right) \right\rangle = y_k - \sum_{i\in S}\tilde{\alpha}_i y_i \left\langle \Phi\left(X_{k,\bullet}^t\right); \Phi\left(X_{i,\bullet}^t\right) \right\rangle$$

$$\tilde{\beta}_0 = y_k - \sum_{i\in S}\tilde{\alpha}_i y_i K\left(X_{k,\bullet}, X_{i,\bullet}^t\right) \ (k\in S_{\mathrm{I}})$$

Pour $\tilde{\beta}_0$, on dispose donc d'une formule ne faisant intervenir que le noyau. En revanche, le calcul de $\tilde{\beta}$, fait intervenir la transformation Φ. Mais $\tilde{\beta}$ sert essentiellement à déterminer la fonction score f qui s'exprime ici pour $Z\in\mathbb{R}^p : f(Z) = \left\langle \Phi(Z); \tilde{\beta} \right\rangle + \tilde{\beta}_0$. En réinjectant l'expression de $\tilde{\beta}$ dans cette formule, on obtient :

$$f(Z) = \left\langle \Phi(Z); \sum_{i\in S}\tilde{\alpha}_i y_i \Phi\left(X_{i,\bullet}^t\right) \right\rangle + \tilde{\beta}_0 = \sum_{i\in S}\tilde{\alpha}_i y_i \left\langle \Phi(Z); \Phi\left(X_{i,\bullet}^t\right) \right\rangle + \tilde{\beta}_0 = \sum_{i\in S}\tilde{\alpha}_i y_i K\left(Z, X_{i,\bullet}^t\right) + \tilde{\beta}_0$$

Résumé dual avec marge souple et noyau

$$\max_{\alpha\in\mathbb{R}_+^n} L_D(\alpha) \text{ s.c. : } 0\le\alpha_i\le\Gamma \text{ et } \sum_{i=1}^n \alpha_i y_i = 0$$

$$L_D(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i=1}^n\sum_{i=1}^n \alpha_i y_i K\left(X_{i,\bullet}, X_{j,\bullet}\right) y_j \alpha_j$$

$$L_D(\alpha) = \alpha^t \vec{\mathbf{1}} - \frac{1}{2}\alpha^t Diag(y) G\left(X_{1,\bullet},\cdots,X_{n,\bullet}\right) Diag(y)\alpha$$

S ensemble des $i\in 1\cdots n \ \tilde{\alpha}_{\mathrm{i}} \neq 0$

Si $i\in S$, $X_{i,\bullet}$ est un vecteur support

de type I si $\tilde{\xi}_i = 0$, de type II sinon

$$\tilde{\beta}_0 = y_k - \sum_{i\in S}\tilde{\alpha}_i y_i K\left(X_{k,\bullet}^{\mathrm{t}}, X_{i,\bullet}^t\right) \ (k\in S_{\mathrm{I}})$$

Fonction score $f(Z) = \sum_{i\in S}\tilde{\alpha}_i y_i K\left(Z, X_{i,\bullet}\right) + \tilde{\beta}_0$

On a alors : $\xi_{\mathrm{i}} = \max\left(0, 1 - y_i f\left(X_{i,\bullet}\right)\right)$

Quelles sont les propriétés requises pour un noyau ?

On dit qu'une application symétrique $\mathbb{R}^p\times\mathbb{R}^p \xrightarrow{K} \mathbb{R}$ représente un produit scalaire dans un espace pré-hilbertien réel H de redescription si et seulement si, il existe une application $\Phi : \mathbb{R}^p \xrightarrow{\Phi} H$ telle que :

$$\forall U,V\in\mathbb{R}^p : K(U,V) = \left\langle \Phi(U); \Phi(V) \right\rangle_H .$$

La réponse à la question à la question introductive tient dans la proposition suivante :

Proposition

Une application symétrique $\mathbb{R}^p \times \mathbb{R}^p \xrightarrow{K} \mathbb{R}$ *représente un produit scalaire si et seulement si :* $\forall (X_1,\cdots,X_n) \in \mathbb{R}^p \times \cdots \times \mathbb{R}^p$: *la matrice symétrique :*

$$G(X_1,\cdots,X_n) = \left(K(X_i\,;X_j)\right)_{i=1\cdots n,\ j=1\cdots n}$$

est semi-définie positive.

REMARQUE PREALABLE

Dans un texte sur le sujet, j'ai lu comme condition $G(X_1,\cdots,X_n)$ définie positive. C'est faux et cela même dans le cas d'un noyau linéaire. Si $p<n$, la matrice X est de rang $<n$ et la matrice $G(X_1,\cdots,X_n)=XX^t$ aussi, donc non inversible.

DEMONSTRATION

$1 \Rightarrow 2$

Soit K représentant un produit scalaire dans H.

$G(X_1,\cdots,X_n)$ semi-définie positive $\Leftrightarrow \forall U = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} : U^t G U \geq 0$. Or :

$$U^t G U = \sum_{i=1}^{n}\sum_{j=1}^{n} u_i K(X_{i,\bullet}, X_{j,\bullet}) u_j = \sum_{i=1}^{n}\sum_{j=1}^{n} u_i \left\langle \Phi(X_{i,\bullet}); \Phi(X_{j,\bullet}) \right\rangle_H u_j$$

(puisque K représente un produit scalaire)

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} \left\langle u_i \Phi(X_{i,\bullet}); u_j \Phi(X_{j,\bullet}) \right\rangle_H = \left\langle \sum_{i=1}^{n} u_i \Phi(X_{i,\bullet}) ; \sum_{j=1}^{n} u_j \Phi(X_{j,\bullet}) \right\rangle_H = \left\| \sum_{i=1}^{n} u_i \Phi(X_{i,\bullet}) \right\|^2 \geq 0$$

$2 \Rightarrow 1$

Soit K un noyau vérifiant $G(X_1,\cdots,X_n)$ semi-définie positive $\forall (X_1,\cdots,X_n) \in \mathbb{R}^p \times \cdots \times \mathbb{R}^p$.

Pour $U \in \mathbb{R}^p$, on note K_U l'application $\begin{array}{l} \mathbb{R}^p \xrightarrow{K_U} \mathbb{R} \\ Z \mid \longrightarrow K_U(Z) = K(U,Z) \end{array}$. Elle appartient à l'espace vectoriel des applications de $\mathbb{R}^p \to \mathbb{R}$. Soit $H = Vec\left(K_U / U \in \mathbb{R}^p\right)$. Autrement dit, H (de dimension infinie) est l'ensemble de toutes combinaisons linéaires finies des K_U.

Soit $\alpha = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} \in \mathbb{R}^n$, $\beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_m \end{pmatrix} \in \mathbb{R}^m$ et $J_\alpha = \sum_{i=1}^{n} \alpha_i K_{X_i}$, $J_\beta = \sum_{j=1}^{m} \beta_j K_{Y_j} \in H$, on définit :

$$\left\langle J_\alpha ; J_\beta \right\rangle_H \overset{def}{=} \sum_{i=1}^{n}\sum_{j=1}^{m} \alpha_i K(X_i, Y_j) \beta_j$$

Il est facile de vérifier qu'il s'agit d'une forme bilinéaire symétrique sur H.

$\left\langle J_\alpha ; J_\alpha \right\rangle_H = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i K(X_i, X_j) \alpha_j = \alpha^t G(X_1,\cdots,X_n) \alpha \geq 0$ puisque $G(X_1,\cdots,X_n)$ semi-définie positive.

$\left\langle\ ;\ \right\rangle_H$ est donc une forme semi-définie positive sur H.

Pour que $\langle\ ;\ \rangle_H$ soit définie positive, autrement dit soit un produit scalaire, il reste à prouver :

$$\langle J_\alpha ; J_\alpha \rangle = 0 \Rightarrow J_\alpha = 0$$

$$\forall J_\alpha \in H, U \in R^p : \langle J_\alpha ; K_U \rangle_H = \sum_{i=1}^{n} \alpha_i K(U, X_i) = \sum_{i=1}^{n} \alpha_i K_{X_i}(U) = J_\alpha(U) \quad (1)$$

Par ailleurs, l'inégalité de Cauchy-Schwarz, est vraie pour les formes semi-définies positives. D'où :

$$\forall\, J_\alpha \in H, U \in R^p : \langle J_\alpha ; K_U \rangle_H^2 \le \langle J_\alpha ; J_\alpha \rangle_H \langle K_U ; K_U \rangle_H .$$

Donc si $\langle J_\alpha ; J_\alpha \rangle_H = 0$, alors $\langle J_\alpha ; K(U,\bullet) \rangle_H = 0$ et alors d'après (1) $J_\alpha(U) = 0$ et ceci quel que soit $U \in \mathbb{R}^p$.

Pour terminer, en posant $\forall\, U \in \mathbb{R}^p : \Phi(U) = K_U$, on a d'après (1) :

$$\langle \Phi(U); \Phi(V) \rangle_H = \langle K_U ; K_V \rangle_H \overset{(1)}{=} J_\alpha(V) = K(U,V)$$

COMMENTAIRE

Cette fonction Φ qui envoie dans un espace de dimension infinie, n'a guère d'intérêt pour les calculs pratiques pour lesquels l'usage du noyau qui opère en dimension fini, s'avère beaucoup plus efficace. En revanche, la proposition fournit un critère pour qu'une fonction symétrique $\mathbb{R}^p \times \mathbb{R}^p \xrightarrow{K} \mathbb{R}$ soit un noyau convenable.

Comment fabriquer des noyaux ?

En préalable, deux ou trois choses à savoir sur les matrices semi-définies positives.

La première (aisée à vérifier) est qu'une combinaison linéaire à coefficients positifs de matrices semi-définies positives est semi -définie positive.

La deuxième concerne le produit terme à terme deux matrices, dit produit de Hadamard et noté $A \odot B$

Lemme du produit de Schur

Soit $A, B \in M_{n\times n}(\mathbb{R})$ semi-définies positives, alors $A \odot B$ l'est aussi.

DEMONSTRATION

Il est clair que A, B étant symétrique, $A \odot B$ l'est aussi.

A semi-définie positive est diagonalisable avec une matrice de passage orthogonale et toutes ses valeurs propres sont positives ou nulles :

$A = Q\,Diag(\lambda_1, \cdots, \lambda_n) Q^t \quad (0 \le \lambda_i)$ et on a : $a_{i,j} = \sum_{k=1}^{n} \lambda_k q_{i,k} q_{j,k}$.

Soit $U = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \in \mathbb{R}^n, U^t A \odot B U = \sum_{i,j=1}^{n} u_i a_{i,j} b_{i,j} u_j = \sum_{i,j=1}^{n} u_i \left(\underbrace{\sum_{k=1}^{n} \lambda_k q_{i,k} q_{j,k}}_{=a_{i,j}} \right) b_{i,j} u_j = \sum_{k=1}^{n} \lambda_k \sum_{i,j=1}^{n} u_i q_{i,k} b_{i,j} u_j q_{j,k}$

En posant : $U_k = \begin{pmatrix} u_1 q_{1,k} \\ \vdots \\ u_n q_{n,k} \end{pmatrix}$, $\sum_{i,j=1}^{n} u_i q_{i,k} b_{i,j} u_j q_{j,k}$ peut s'écrire : $U_k^t B U_k$. D'où :

$$U^t A \odot B U = \sum_{k=1}^{n} \lambda_k U_k^t B U_k$$

Or B étant semi-définie positive, $0 \le U_k^t B U_k$ et λ_k aussi.

La troisième, conséquence immédiate du lemme précédent est que si $A = \left(a_{i,j}\right)_{\substack{1\le i\le n \\ 1\le j\le n}}$ est semi-définie positive, la matrice $A^{\hat{q}} = \left(a_{i,j}^{q}\right)_{\substack{1\le i\le n \\ 1\le j\le n}}$ $q \in \mathbb{N}$ l'est aussi.

On sait que $K(U,V) = \langle U ; V \rangle$ est un noyau. Pour en fabriquer d'autres, on peut s'appuyer sur la proposition suivante :

Proposition

Si $K_1(U,V), K_2(U,V)$ *sont des noyaux, alors :*

1) $aK_1(U,V), +bK_2(U,V) \quad \left(a,b \in \mathbb{R}_+^*\right)$

2) $aK_1(U,V)+b \quad \left(a,b \in \mathbb{R}_+^*\right)$

3) $K_1(U,V) \times K_2(U,V)$

4) $K_1(U,V)^k \quad \left(k \in \mathbb{N}_+^*\right)$

5) $g(U)K_1(U,V)\,g(V)$ *où g est une fonction de* $\mathbb{R}^p \xrightarrow{\ g\ } \mathbb{R}$

6) $\exp\left(K_1(U,V)\right)$

sont des noyaux.

DEMONSTRATION :

Remarque : ces nouveaux noyaux et donc les matrices correspondantes sont évidemment symétriques.

On notera $G_K\left(X_1,\cdots,X_n\right)$, la matrice correspondant au noyau *K*.

1) $G_{aK_1+bK_2}\left(X_1,\cdots,X_n\right) = aG_{K_1}\left(X_1,\cdots,X_n\right) + bG_{K_2}\left(X_1,\cdots,X_n\right)$. Comme $G_{K_1}\left(X_1,\cdots,X_n\right)$ et $G_{K_2}\left(X_1,\cdots,X_n\right)$ sont semi-définies positives, $G_{aK_1+bK_2}\left(X_1,\cdots,X_n\right)$ l'est aussi et $aK_1(U,V), +bK_2(U,V)$ est un noyau.

2) C'est le cas précédent avec $\forall U,V : K_2(U,V) = 1$.

Dans ce cas $G_{K_2}\left(X_1,\cdots,X_n\right) = \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix} = \vec{\mathbf{1}}\vec{\mathbf{1}}^t$ où $\vec{\mathbf{1}} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$. Cette matrice est semi définie positive car $\forall Z \in \mathbb{R}^n : \underbrace{Z^t \vec{\mathbf{1}}}_{\in \mathbb{R}} \underbrace{\vec{\mathbf{1}}^t Z}_{\in \mathbb{R}} = \left(Z^t \vec{\mathbf{1}}\right)^2$.

3) $G_{K_1 \times K_2}\left(X_1,\cdots,X_n\right) = G_{K_1}\left(X_1,\cdots,X_n\right) \odot G_{K_2}\left(X_1,\cdots,X_n\right)$. Il suffit d'appliquer le lemme de Schur.

4) Conséquence immédiate de 3).

5) Soit $K_g(U,V)=g(U)K_1(U,V)g(V)$. Alors :

$$G_{K_g}(X_1,\cdots,X_n)=\left(g(X_i)K_1(X_i,Y_j)g(Y_j)\right)_{\substack{1\le i\le n\\1\le j\le n}}$$

$$=DG_{K_1}(X_1,\cdots,X_n)D \text{ en notant } D=Diag\left(g(X_1),\cdots,g(X_n)\right)$$

$$U^tG_{K_g}(X_1,\cdots,X_n)U=U^tDG_{K_1}(X_1,\cdots,X_n)DU$$

Comme D diagonale $D^t=D$ et $U^tG_{K_g}(X_1,\cdots,X_n)U=(DU)^tG_{K_1}(X_1,\cdots,X_n)DU\ge 0$.

6) On pose : $K_1(U,V)=A=\left(a_{i,j}\right)_{\substack{1\le i\le n\\1\le j\le n}}$. A est semi-définie positive.

On définit : $A_k=\left(\sum_{q=0}^{k}\frac{1}{q!}a_{i,j}^q\right)_{\substack{1\le i\le n\\1\le j\le n}}=\sum_{q=0}^{k}\frac{1}{q!}\left(a_{i,j}^q\right)_{\substack{1\le i\le n\\1\le j\le n}}=\sum_{q=0}^{k}\frac{1}{q!}A^{\hat{q}}$

D'après les remarques préalables $A^{\hat{q}}$ est semi-définie positive et $A_k=\sum_{q=0}^{k}\frac{1}{q!}A^{\hat{q}}$ aussi.

$\forall U\in\mathbb{R}^n : U^tA_kU\ge 0$. En passant à la limite $\underset{k\to+\infty}{Lim}\left(U^tA_kU\right)=U^t\underset{k\to+\infty}{Lim}\left(A_k\right)U\ge 0$.

Or $\underset{k\to+\infty}{Lim}A_k=\left(\underset{k\to+\infty}{Lim}\left(\sum_{q=0}^{k}\frac{1}{q!}a_{i,j}^q\right)\right)_{\substack{1\le i\le n\\1\le j\le n}}=\left(\exp\left(a_{i,j}\right)\right)_{\substack{1\le i\le n\\1\le j\le n}}$

Noyaux usuels

Parmi les noyaux utilisés en pratique, on trouve :

Noyau polynomial	$K(U,V)=\left(coef0+\langle U\,;V\rangle\right)^{degree}$	Si *coef0* = 0 et *degree* = 1, noyau linéaire
Noyau radial	$K(U,V)=\exp\left(-\gamma\lVert U-V\rVert^2\right)$	$\gamma\in\mathbb{R}_+^*$ par défaut $\gamma=\frac{1}{p}$
Noyau sigmoïde	$K(U,V)=\tanh\left(\gamma\langle U\,;V\rangle+coef0\right)$	

2) et 4) justifient l'appellation de noyau polynomial pour : $K(U,V)=\left(coef0+\langle U\,;V\rangle\right)^{degree}$.

En ce qui concerne $K(U,V)=\exp\left(-\gamma\lVert U-V\rVert^2\right)$:

$$\exp\left(-\gamma\lVert U-V\rVert^2\right)=\exp\left(-\gamma(U-V)^t(U-V)\right)=\exp\left(-\gamma(U^tU-2U^tV+V^tV)\right)$$

$$=\exp\left(-\gamma U^tU\right)\exp\left(2\gamma U^tV\right)\exp\left(-\gamma V^tV\right)$$

Or, comme $\gamma\in\mathbb{R}_+^*$, $K(U,V)=2\gamma U^tV$ est un noyau. Donc, d'après 6) $\exp\left(2\gamma U^tV\right)$ aussi. Et 5) Permet de conclure avec $g(U)=\exp\left(-\gamma U^tU\right)$.

Le noyau sigmoïde, bien qu'utilisé en pratique[4] ne répond pas au critère $G(X_1,\cdots,X_n)$ semi-définie positive pour tout $(X_1,\cdots,X_n)\in\mathbb{R}^p\times\cdots\times\mathbb{R}^p$.

Soit, par exemple, $X_1=\begin{pmatrix}\sqrt{2}\\\sqrt{2}\end{pmatrix}$, $X_2=\begin{pmatrix}1/\sqrt{2}\\1/\sqrt{2}\end{pmatrix}$. Avec un « noyau » sigmoïde, on a :

$G(X_1,X_2)=\begin{pmatrix}\tanh(4) & \tanh(2)\\ \tanh(2) & \tanh(1)\end{pmatrix}$ et cette matrice ayant une valeur propre ($\simeq$`-0.0908665765`) strictement négative n'est pas semi-définie positive.

[4] Il est proposé parmi les options dans le package e1071 de R consacré à SVM.

Des scores aux probabilités

D'autres méthodes que SVM peuvent être utilisées devant ce genre de problème (variables explicatives quantitatives, variable à expliquer binaire) comme, par exemple la régression logistique. À l'encontre de SVM qui fournit des scores dont le signe permet la discrimination, la régression logistique fournit des probabilités d'appartenance. La méthode suivante permet la conversion des scores en probabilités.

Méthode de Platt

Elle consiste à transformer le score $f(Z)=\sum_{i\in S}\tilde{\alpha}_i y_i K(Z,X_i)+\tilde{\beta}_0$ en probabilité par une fonction sigmoïde :

$$P[y=1/Z]=\frac{1}{1+\exp(af(Z)+b)}$$

a, b étant deux constantes à déterminer par le maximum de vraisemblance

Probabilité de l'échantillon *X*

Pour $i\in 1\cdots n$, on note $p_i=P[y=1/X_{i,\bullet}]$.

Pour la ligne *i*, la probabilité est donc p_i si $y_i=1, 1-p_i$ si $y_i=-1$, ce qui peut se synthétiser en :

$$p_i^{\frac{y_i+1}{2}}(1-p_i)^{\frac{-y_i+1}{2}} \text{ ou encore } p_i^{[y_i=1]}(1-p_i)^{[y_i=-1]}\ ^{5}$$

La vraisemblance de l'échantillon *X* est donc $\prod_{i=1}^{n} p_i^{\frac{y_i+1}{2}}(1-p_i)^{\frac{-y_i+1}{2}}$ et en passant au logarithme, le maximum de vraisemblance est :

$$\max_{a,b}\left(\sum_{i=1}^{n}([y_i=1]\ln p_i+[y_i=-1]\ln(1-p_i))\right)$$

Avec l'exemple marge souple

				a =	-0,194
				b =	0,046
individus i	x1	x2	y	f(Xi)	P[y=1/x1 ,x2]
1	1	1	-1	-0,667	45,6%
2	4	5	-1	-1,000	44,0%
3	6	9	-1	-1,667	40,9%
4	8	7	-1	-0,333	47,2%
5	7	1	-1	1,333	55,3%
6	1	3	1	-1,333	42,5%
7	5	1	1	0,667	52,1%
8	13	6	1	1,667	56,9%
9	9	4	1	1,000	53,7%
10	12	7	1	1,000	53,7%

Ces probabilités sont peu significatives du fait d'une base d'apprentissage très étriquée avec deux grosses « erreurs » 5 et 6…

[5] Si *P* est une proposition logique, $[P]=1$ si *P* vraie 0 sinon.

Un exemple avec des vraies données

spam or not

La base de données Spambase est due à :

> Hopkins, M., Reeber, E., Forman, G., & Suermondt, J. (1999).
> Spambase [Dataset]. UCI Machine Learning Repository.
> https://doi.org/10.24432/C53G6X .

Elle est téléchargeable sur la page[6] : https://archive.ics.uci.edu/dataset/94/spambase qui contient de plus nombre d'informations à son sujet.

Cette base décrit 4601 méls à l'aide de 57 variables quantitatives en précisant le statut du mél Spam (o/n).

Parmi ces 57 variables explicatives, on trouve :

- ✓ 48 représentent la fréquence d'apparition de certains mots dans le texte du mél. Par exemple la variable `wf_money` indique la fréquence du mot anglais « *money* » (tous les méls analysés sont en anglais).
- ✓ 6 représentent la fréquence d'apparition de certains caractères spéciaux dans le texte comme #, $...
- ✓ Les 3 dernières sont en lien avec des séquences de plusieurs lettres écrits en majuscules.

La variable `Spam (o/n)` est la variable à expliquer.

Après un tri aléatoire (initialement les 1873 spams figuraient en premier, les 2788 non spams en second), la base a été importée dans R sous le nom de *spam* pour y être traitée à l'aide du package e1071. Puis, la base a été scindée en 2 :

1. Les 3601 premières lignes pour constituer la base d'apprentissage (*train*).
2. Les 1000 lignes suivantes pour constituer une base de test (*test*).

```
> train=spam[1:3601,]
> test=spam[3602:4601,]
> dim(train)
[1] 3601   58
> dim(test)
[1] 1000   58
```

Remarques préalables

Dans l'exposé mathématique que nous avons fait sur SVM, les vecteurs $X_{i,\bullet}$ sont étiquetés par $y_i \in \{-1,1\}$. Mais il s'agit d'une pure convention, convention justifiée car elle permet d'exprimer simplement les conditions de séparation : $\forall i \in 1\cdots n:\ 1 \le y_i\left(X_{i,\bullet}\beta+\beta_0\right)$. Mais il s'agit fondamentalement d'une discrimination binaire. La variable à expliquer `Spam` dans l'exemple

[6] Elle se trouve aussi dans le classeur à télécharger : `spamdata.xls`

prend comme valeur `o` ou `n`. Avec le package e1071 (même chose avec Tanagra), la variable à expliquer, même si elle est codée numériquement, doit être de type `factor`.

Le classement s'opère sur le signe de la fonction de score $f(Z)$ et il faut reconnaitre quelle convention a adopté le logiciel. Dans notre exemple, par anticipation, $0 \leq f(Z)$ correspond à `Spam = n`.

La fonction `svm()` du package e1071 réalise le travail d'apprentissage :

```
> m1=svm(formula=Spam~.,data=train,cost=2,kernel="radial",probability=TRUE,
scale=TRUE)
> print(m1)
Call: svm(formula = Spam ~ ., data = train, cost = 2, kernel = "radial",
probability = TRUE, scale = TRUE)
Parameters:  SVM-Type:  C-classification
SVM-Kernel:  radial
       cost:  2
Number of Support Vectors:  959
```

Quelques explications

Arguments de la fonction

- ✓ `formula=Spam~.` précise la variable à expliquer.
- ✓ `data=train` indique la *data-frame* à utiliser.
- ✓ `cost = 2` fixe la valeur de la constante C (Γ dans notre texte) qui règle la tolérance aux erreurs.
- ✓ `kernel = "radial"` indique le noyau à utiliser à choisir parmi : `"linear","polynomial","radial", "sigmoid".`
- ✓ `probability = TRUE` calcul des coefficients a et b intervenant dans la formule de conversion score → probabilité.
- ✓ `scale=TRUE` pour standardiser les variables.

Résultats

Les résultats sont englobés dans l'objet `m1`. La commande `print(m1)` n'en dévoile qu'une petite partie. La commande `attributes(m1)` révèle d'autres attributs :

```
names
"call"       "type"       "kernel"     "cost"       "degree"           "gamma"
"coef0"      "nu"         "epsilon"    "sparse"     "scaled"           "x.scale"
"y.scale"    "nclasses"   levels"      "tot.nSV"    "nSV"              "labels"
"SV"         "index"      "rho"        "compprob"   "probA"            "probB"
"sigma"      "coefs"      "na.action"  "fitted"     "decision.values"  "terms"
```

Parmi ces attributs :

- ✓ `m1$SV` fournit la liste des vecteurs supports avec leurs coordonnées.
- ✓ `m1$coef` donne pour chaque vecteur support $\tilde{\alpha}_i \; y_i$.

  Avec un noyau `"linear"`, $\tilde{\beta}$ peut être obtenue par : `beta =t(m1$coefs) %*% m1$SV`.
- ✓ `m1$rho` donne la valeur de $\tilde{\beta}_0$.
- ✓ `m1$decision.values` donne la liste pour $i \in 1 \cdots n$ des $f\left(X_{i,\bullet}\right)$.
- ✓ `m1$probA`, `m1$probB` les valeurs des coefficients a et b évoqués en amont.

Prédiction

Après l'apprentissage vient la phase de prédiction. Elle se fait par la fonction `predict()`. Dans sa version courte, elle possède deux arguments :

1. Un objet retourné par la fonction `SVM()`
2. Une data-frame de même structure que celle qui a servi à l'apprentissage.

Avant de la mettre en œuvre sur la base *test*, on peut commencer par le faire sur la base *train* :

```
> p1=predict(m1, train)
> p1[1:10]
 1  2  3  4  5  6  7  8  9 10
 n  n  o  o  o  n  n  n  o  o
Levels: n o
```

On peut facilement créer la matrice de confusion qui confronte valeur réelle et valeur prédite :

```
> table(train$Spam,p1)
         p1
        n    o
  n 2119   59
  o  101 1322
```

Pour la base test, on obtient :

```
> p2=predict(m1, test)
> p2[1:10]
3602 3603 3604 3605 3606 3607 3608 3609 3610 3611
 n    n    n    n    n    n    o    n    o    o
Levels: n o
> table(test$Spam,p2)
       p2
      n   o
  n 585  25
  o  50 340
```

Résumé

Matrice de confusion train

		Prédiction o	Prédiction n	Σ		
Valeur réelle	o	2119	59	*2178*	TVP (sensibilité)	97,3%
	n	101	1322	*1423*	TVN (spécificité)	92,9%
	Σ	*2220*	*1381*	*3601*	Taux d'erreur	4,4%

Matrice de confusion test

		Prédiction o	Prédiction n	Σ		
Valeur réelle	o	585	25	*610*	TVP (sensibilité)	95,9%
	n	50	340	*390*	TVN (spécificité)	87,2%
	Σ	*635*	*365*	*1000*	Taux d'erreur	7,5%

On observe un taux d'erreurs plus élevé sur la base *test*. C'est un constat courant et assez logique, puisque avec la base *train* ce sont les mêmes données qui servent à l'apprentissage et que la fonction score f est élaboré pour « coller » au mieux à ces données.

La sensibilité dans les deux cas est assez bonne : ~3-4% de spams non détectés.

En revanche la spécificité est mauvaise sur la base test ~13% de méls sains sont classés en spam.

La fonction `predict()` dans sa version longue s'écrit en se limitant pour des raisons évidentes aux 5 derniers méls :

```
> predict(m1,test[996:1000,],probability=TRUE,decision.values=TRUE)
4597 4598 4599 4600 4601
   n    n    n    o    o
attr(,"decision.values")
            n/o
4597  0.9998383
4598  0.7109559
4599  0.1634921
4600 -1.5943902
4601 -1.3818685
attr(,"probabilities")
              n          o
4597 0.92451892 0.07548108
4598 0.85506814 0.14493186
4599 0.59642450 0.40357550
4600 0.01704965 0.98295035
4601 0.02883121 0.97116879
Levels: n o
```

On observe que les valeurs positives de l'attribut `decision.values` correspondent à des méls « sains » (`Spam=n`).

La pratique de SVM est un art subtil. Deux choix cruciaux sont à effectuer :

1. Le choix du noyau.
2. Le choix du paramètre C (`cost`).

Le package e1071 propose la fonction `tune()` pour guider l'utilisateur dans son choix :

```
>tune(svm, Spam~., data= train,
 ranges =list(kernel = c('linear','polynomial','radial'),
 cost =c(0.5,1,2,5,10)), tunecontrol = tune.control(sampling ="cross"))

Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
 kernel cost
 radial    5
- best performance: 0.06414589
```

La méthode employée se fonde sur une validation croisée. Les données sont scindées en 10 morceaux (`folds`) et tour à tour chacun des morceaux sert de test après apprentissage sur les 9 autres. Et il faut itérer la procédure pour les 4 `costs` et les 3 `kernels` proposés. De ce fait le temps d'exécution est assez long... (plus de 4 minutes sur mon PC).

Nous avions fait le bon choix pour le noyau (`radial`), mais pas pour `cost`. Si on recommence les calculs avec `cost=5`, on obtient les matrices de confusion suivantes :

Matrice de confusion train		Prédiction o	Prédiction n	Σ
Valeur réelle	o	2135	43	*2178*
	n	89	1334	*1423*
	Σ	*2224*	*1377*	*3601*

TVP (sensibilité)	98,0%
TVN (spécificité)	93,7%
Taux d'erreur	3,7%

Matrice de confusion test		Prédiction o	Prédiction n	Σ
Valeur réelle	o	585	25	*610*
	n	46	344	*390*
	Σ	*631*	*369*	*1000*

TVP (sensibilité)	95,9%
TVN (spécificité)	88,2%
Taux d'erreur	7,1%

où on peut constater une diminution des taux d'erreur.

Notations

Notations générales

$\lvert E\rvert$	Nombre d'éléments d'un ensemble fini E.
$\mathcal{P}(E)$	L'ensemble des parties de l'ensemble E.
$k\cdots n$	Ensemble des entiers $k\le i\le n$
$[P]$	P étant une proposition $[P]=\begin{cases}1 \text{ si P est vraie}\\ 0 \text{ sinon}\end{cases}$

Algèbre linéaire

$\mathcal{M}_{n\times p}(\mathbb{R})$	L'espace des matrices réelles de n lignes et p colonnes
$\mathcal{M}_n(\mathbb{R})$	L'espace des matrices réelles carrées de n lignes et n colonnes
$\left(a_{i,j}\right)_{\substack{i=1\cdots n\\ j=1\cdots p}}$	Matrice de taille $n\times p$ ayant pour terme $a_{i,j}$ en ligne i, colonne j.
$a_{i,j}\quad A_{i,j}$	Le terme de la matrice A situé en ligne i, colonne j
$A_{i,\bullet}$	La ligne i de la matrice A.
$A_{\bullet,j}$	La colonne j de la matrice A.
A^t	Matrice transposée de la matrice A.
$\vec{\mathbf{1}}_n, \vec{\mathbf{1}}$	Le vecteur de $\mathbb{R}^n$ dont toutes les composantes sont égales à 1.
E_i	Le $i^{ème}$ vecteur de la base canonique de $\mathbb{R}^n$.
E^*	Le dual de l'espace vectoriel E.
$\langle U\,;V\rangle$	Produit scalaire de 2 vecteurs d'un espace euclidien.
$\lVert U\rVert$	Norme euclidienne.
$A\le B\,;A,B\in\mathcal{M}_{n\times p}(\mathbb{R})$	$\forall i\in 1\cdots n, \forall j\in 1\cdots p : a_{i,j}\le b_{i,j}$
$A^{\hat{q}}$	$A^{\hat{q}}=\left(a_{i,j}^q\right)_{\substack{1\le i\le n\\ 1\le j\le n}}\ q\in\mathbb{N}$

Analyse : pour $\mathbb{R}^p \xrightarrow{h} \mathbb{R}$

$\nabla(h)$	Gradient : $\nabla(h)=\begin{pmatrix}\dfrac{\partial h}{\partial x_1}\\ \vdots\\ \dfrac{\partial h}{\partial x_p}\end{pmatrix}$
$\nabla^2(h)$	Matrice symétrique hessienne $\left(\dfrac{\partial^2 h}{\partial x_i \partial x_j}\right)_{\substack{i=1\cdots p\\ j=1\cdots p}}$

Notations pour SVM

(On s'est efforcé d'adopter les notations courantes dans ce domaine)

$p \in \mathbb{N}^*$	Nombre de variables explicatives numériques.
$n \in \mathbb{N}^*$	Nombre de cas.
$X \in M_{n\times p}(\mathbb{R})$	Matrice des données.
$y \in \{-1,1\}^n$	La variable binaire à expliquer.
$\Gamma \in \mathbb{R}_+$	La constante `cost`
$\tilde{\beta}_0 \in \mathbb{R}, \tilde{\beta} \in \mathbb{R}^p, \tilde{\xi} \in \mathbb{R}_+^n$	Solution du primal. $\tilde{\xi}$ variables d'écart (*slack variables*)
$\tilde{\alpha} \in \mathbb{R}_+^n$	Solution du dual.
S vecteurs supports	$S = \{i \in \mathbb{N}^* / 0 < \tilde{\alpha}_i\}$
S_I type *I*	$S_I = \{i \in \mathbb{N}^* / 0 < \tilde{\alpha}_i < \Gamma\}$
S_{II} type *II*	$S_{II} = \{i \in \mathbb{N}^* / \tilde{\alpha}_i = \Gamma\}$
	La fonction score :
$(\mathbb{R}^p)^* \xrightarrow{f} \mathbb{R}$	$f(Z) = \begin{cases} Z\tilde{\beta} + \beta_0 \text{ si noyau linéaire} \\ \sum_{i\in S} \tilde{\alpha}_i y_i K(Z, X_{i,\bullet}) + \tilde{\beta}_0 \text{ sinon} \end{cases}$

Classeurs Excel à télécharger

- ✓ `ex1_dual.xlsx`
- ✓ `ex2_effet_cost.xlsx`
- ✓ `spamdata.xls`

Quelques références

1 Chang C.C., Lin C.J., *LIBSVM: a library for support vector machines*
https://www.csie.ntu.edu.tw/%7Ecjlin/libsvm/

2 Frédéric Meunier *Introduction à l'optimisation*
https://cermics.enpc.fr/~meuniefr/IntroOptim.pdf

3 Freie Universität Berlin *Support Vector Machines Algorithm*
https://www.geo.fu-berlin.de/en/v/soga-r/Machine-Learning/SVM/index.html

4 Jacques Cellier *Algèbre Linéaires des bases aux applications* Presses Universitaires de Rennes 2008

5 Martin Haugh *Support Vector Machines (and the Kernel Trick)*
http://www.columbia.edu/~mh2078/MachineLearningORFE/SVMs_MasterSlides.pdf

6 P.G. Ciarlet *Introduction à l'analyse numérique matricielle et à l'optimisation* MASSON 1994

7 Ricco Rakotomalala *Normalisation des scores*
https://eric.univ-lyon2.fr/ricco/cours/slides/calibration.pdf

8 Ricco Rakotomalala *SVM Support Vector Machine*
https://eric.univ-lyon2.fr/ricco/cours/slides/svm.pdf

9 Shai Shalev-Shwartz *IFT-7002 Fondements de l'apprentissage machine SVM et méthodes à noyaux Traduit et adapté par Mario Marchand Université Laval*
http://www2.ift.ulaval.ca/~mmarchand/IFT7002/SVMetNoyaux.pdf

10 Vladimir N. Vapnik *The Nature of Statistical Learning Theory*
https://statisticalsupportandresearch.wordpress.com/wp-content/uploads/2017/05/vladimir-vapnik-the-nature-of-statistical-learning-springer-2010.pdf
Vladimir N. Vapnik est un des inventeurs de SVM

Table des matières