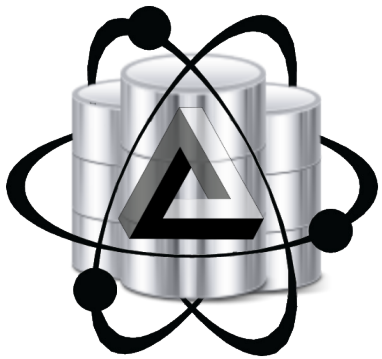
Fondements théorétiques en science des données

Machines à vecteurs de support

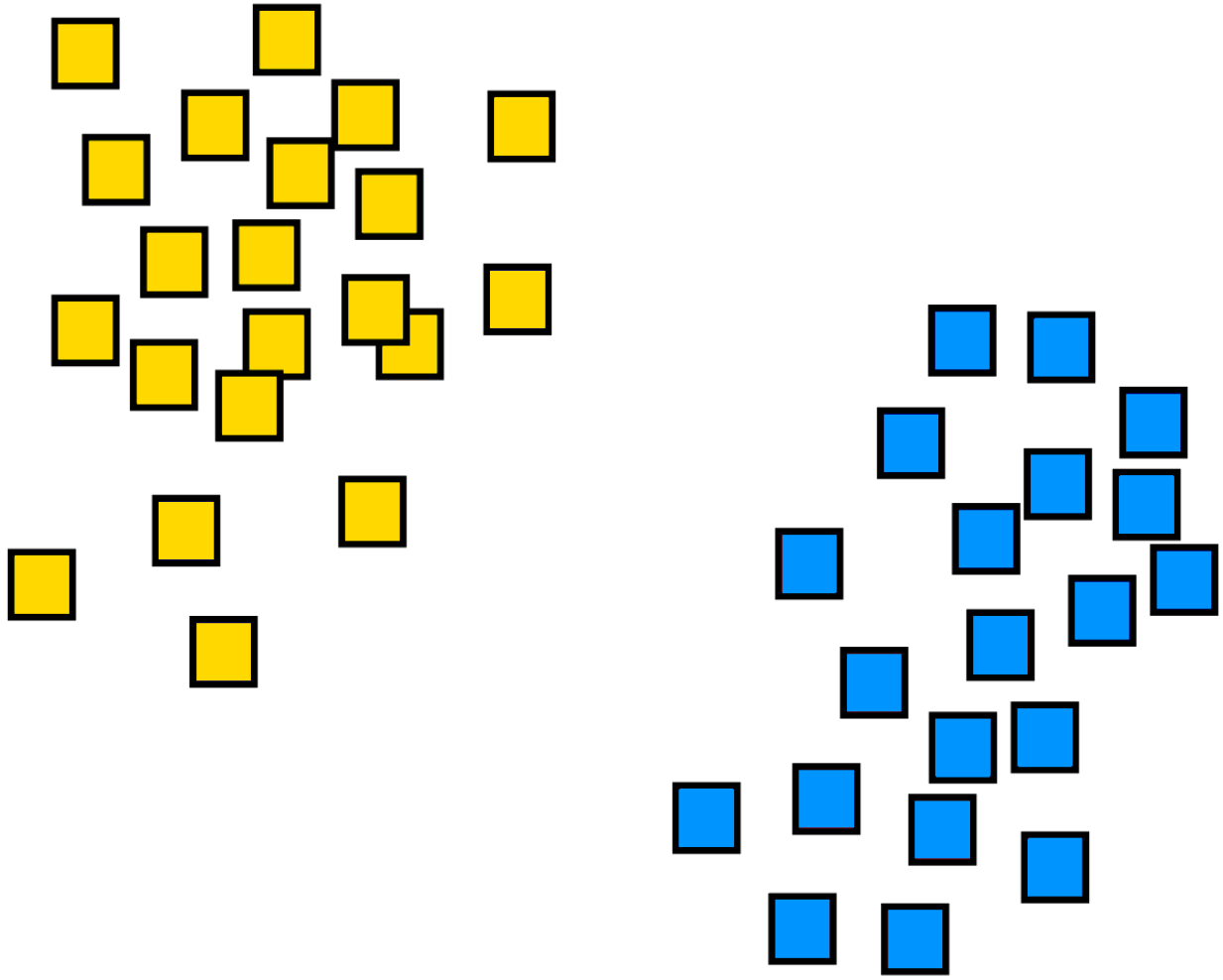
STT 3795

Guy Wolf
guy.wolf@umontreal.ca

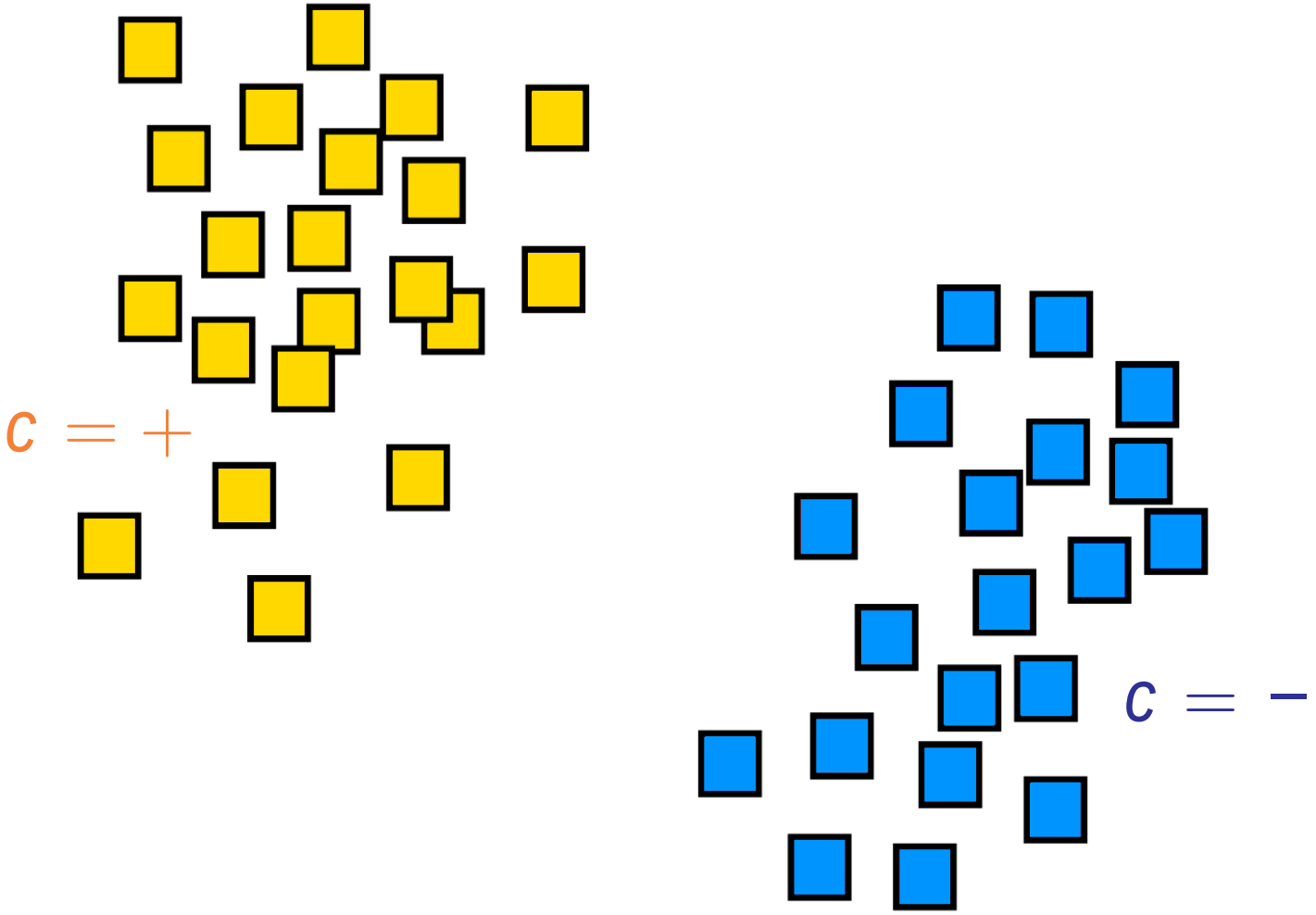
Université de Montréal
Hiver 2020



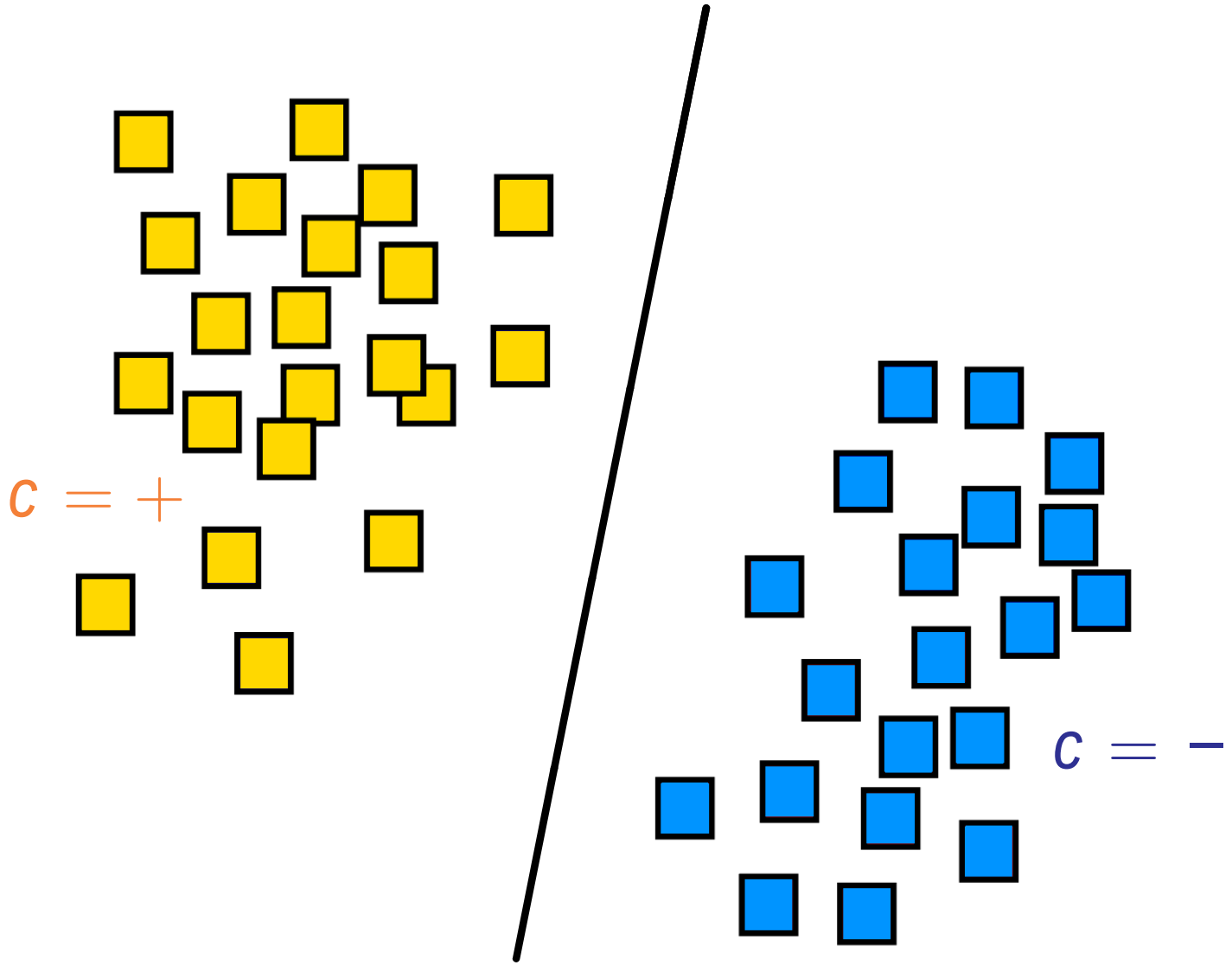
Machines à vecteurs de support



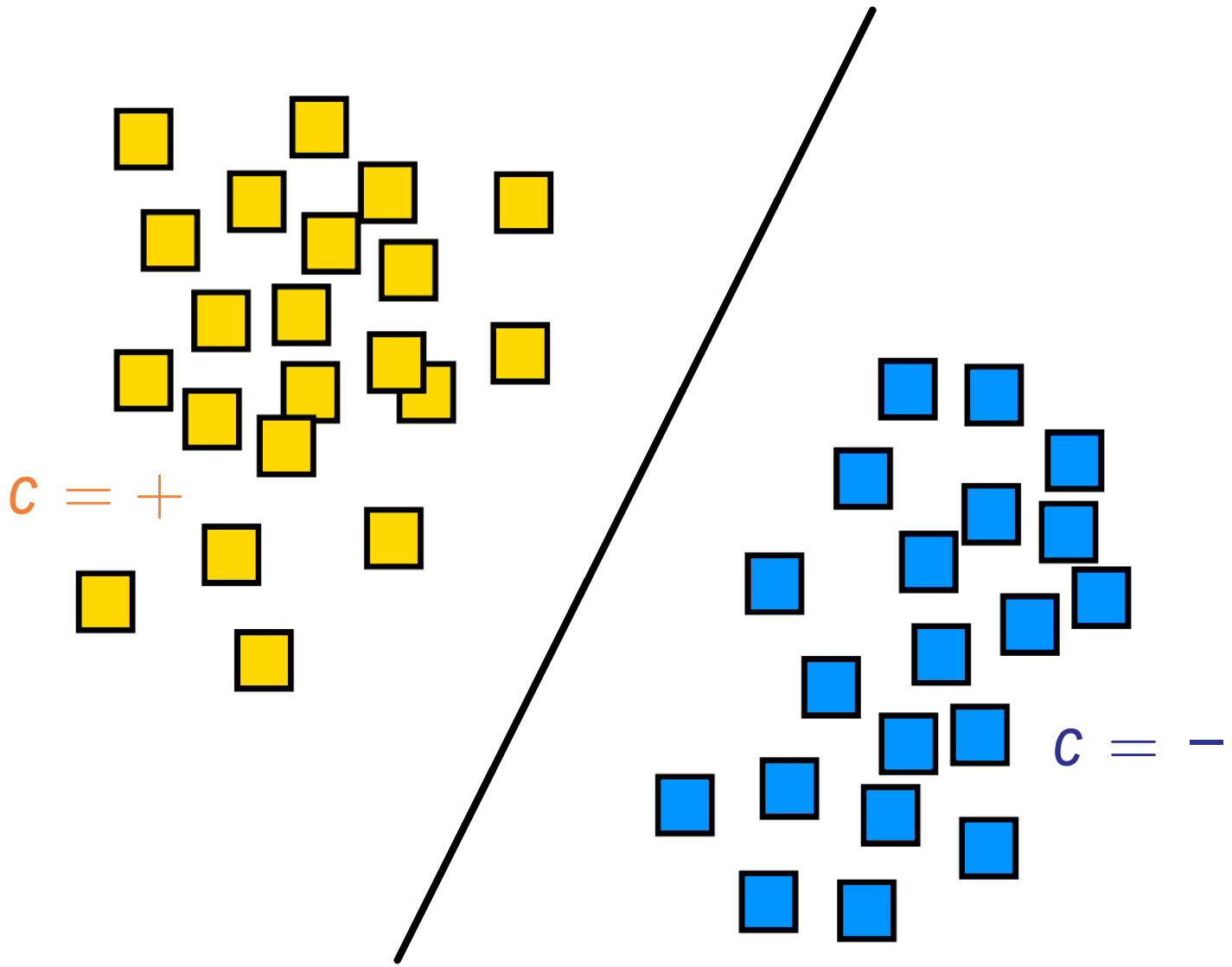
Machines à vecteurs de support



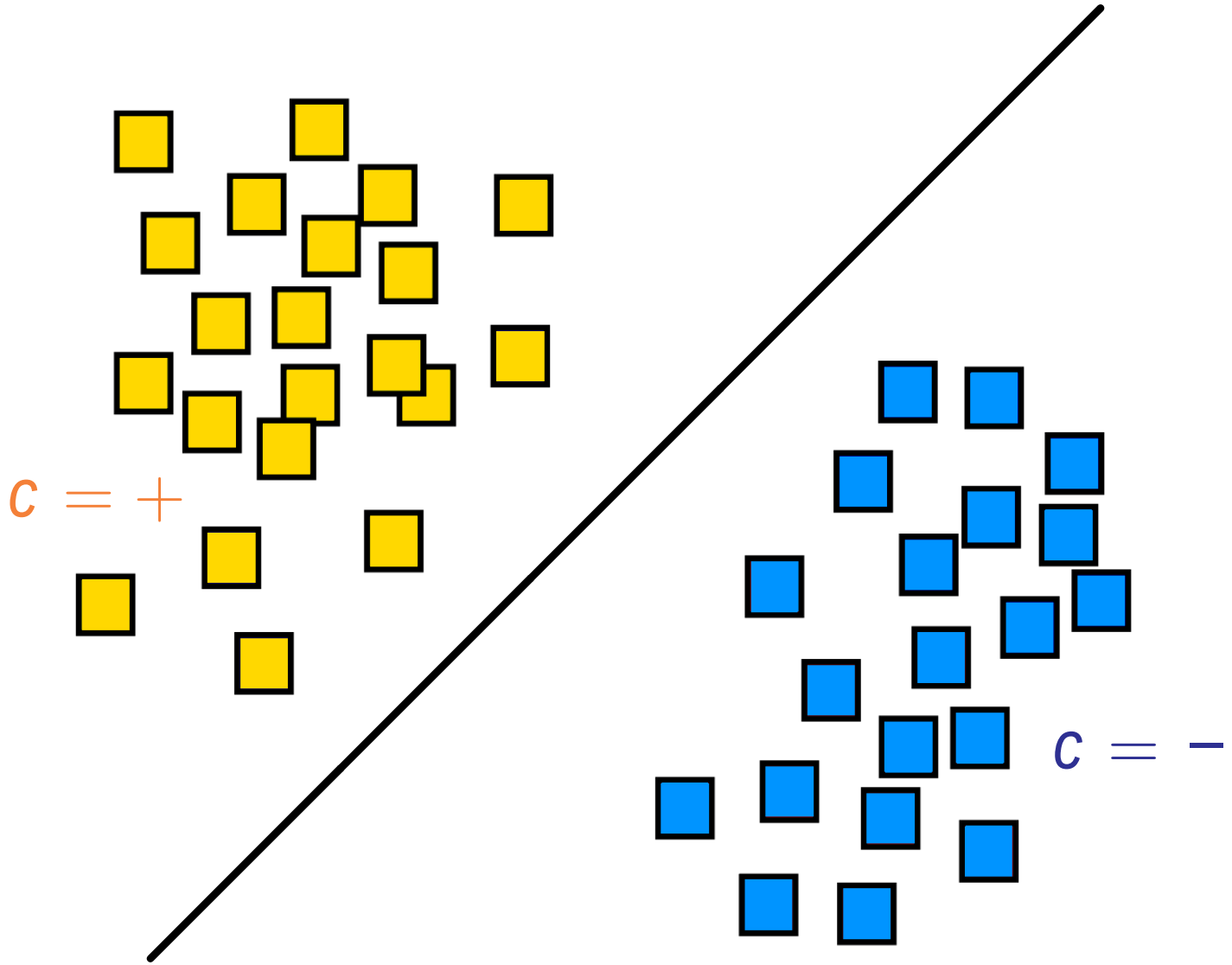
Machines à vecteurs de support



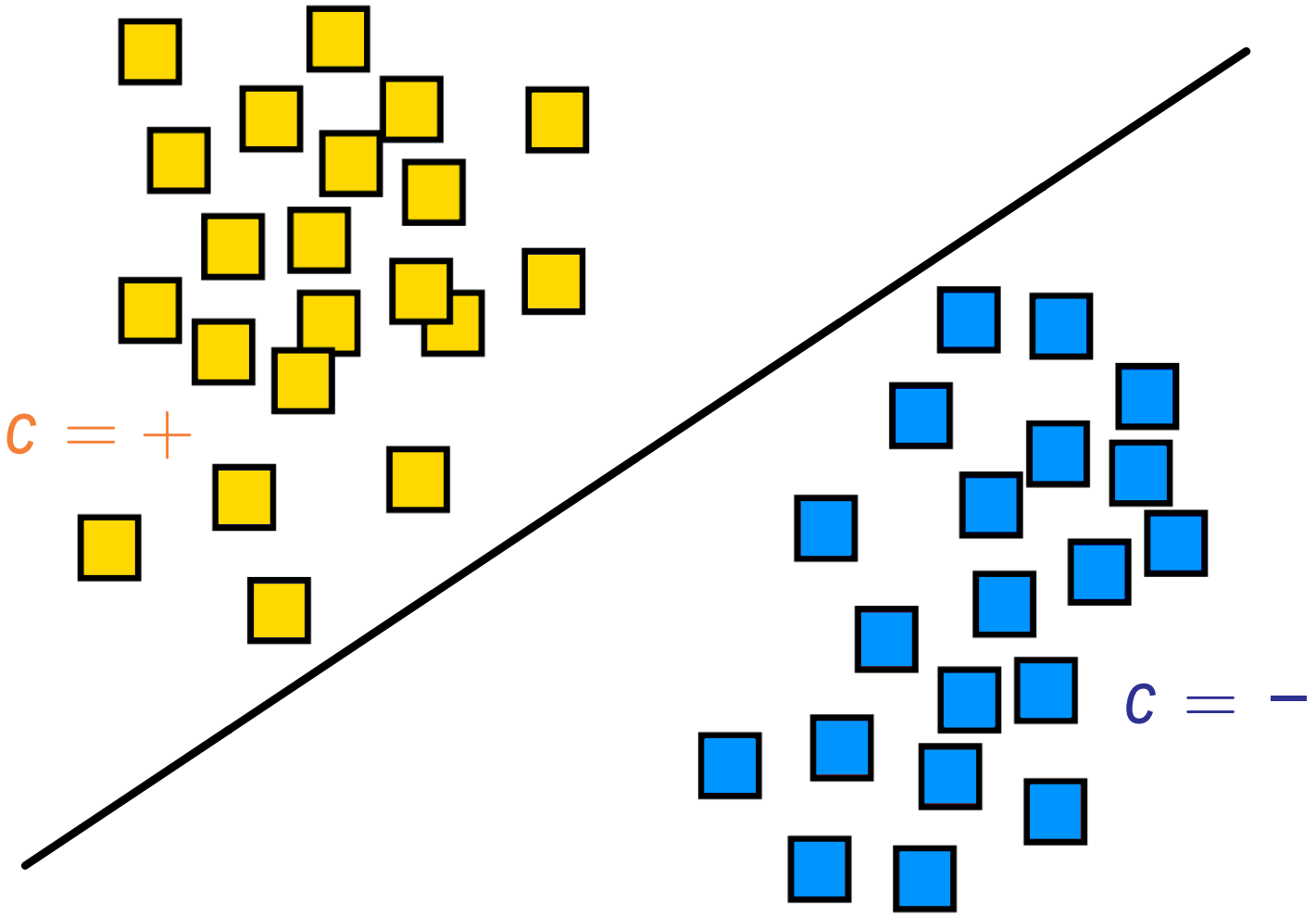
Machines à vecteurs de support



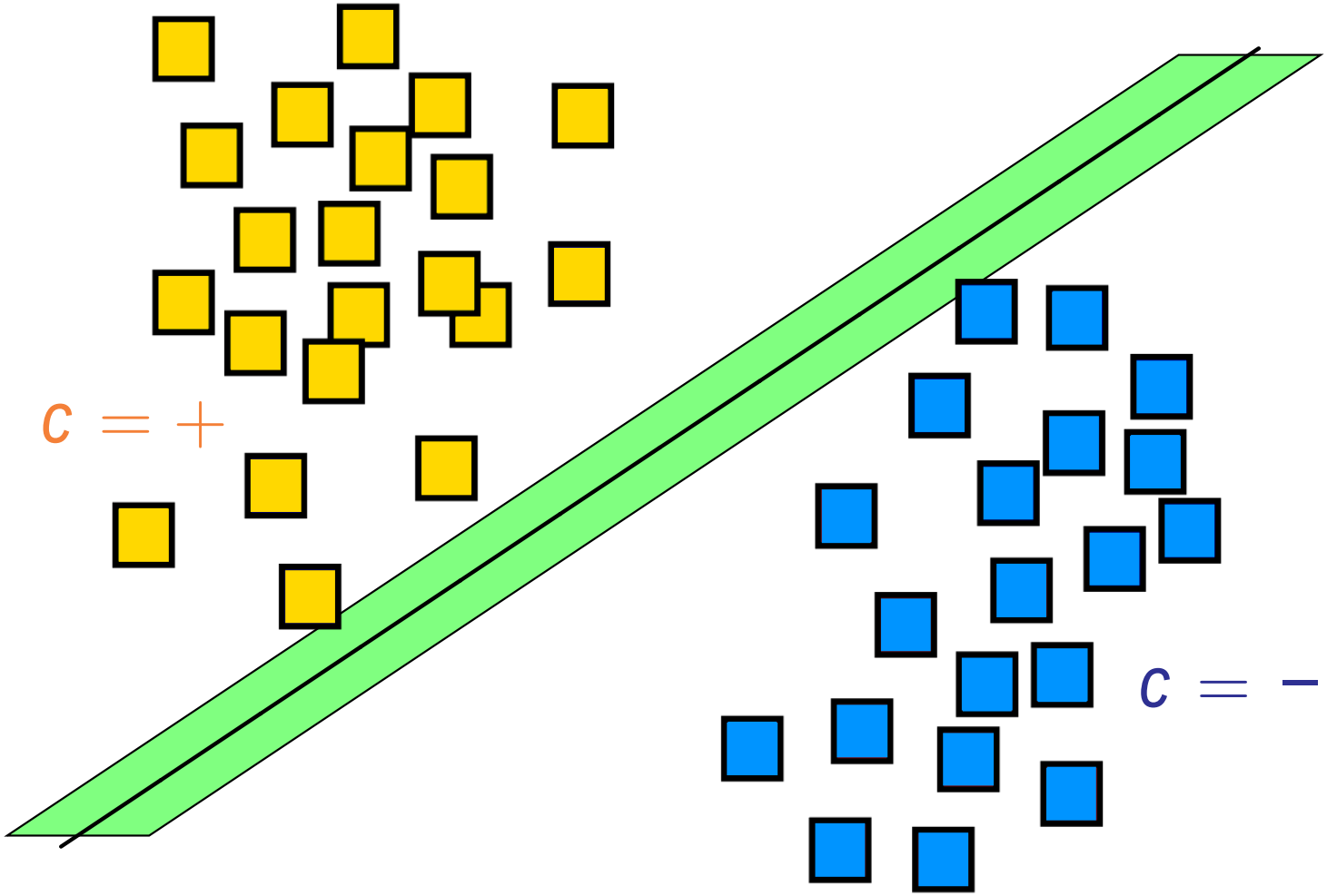
Machines à vecteurs de support



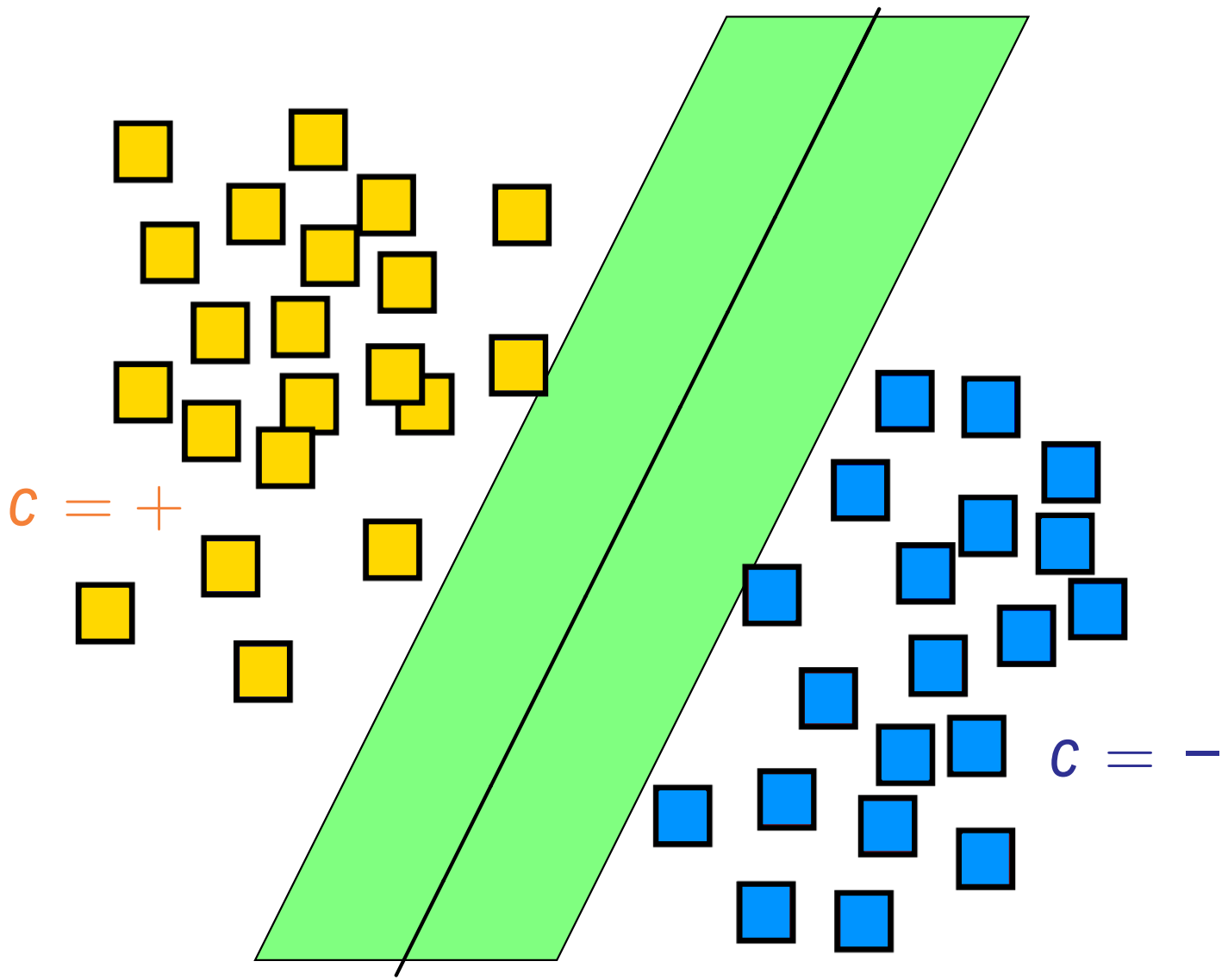
Machines à vecteurs de support



Machines à vecteurs de support



Machines à vecteurs de support



Hyperplan

Un hyperplan de $\mathbb{R}^n$ est un sous-espace de dimension $n-1$ de cet espace.

Hyperplan affine

Un hyperplan affine est l'ensemble des vecteurs qui satisfont $\sum w_j \vec{x}[j] = b$ où $\sum w_j^2 = 1$.

- Le scalaire b détermine la distance de l'origine.
- En utilisant les notations vectorielles, on écrit $\langle \vec{w}, \vec{x} \rangle = b$ où $\vec{w}$ est le vecteur normal qui détermine l'orientation de l'hyperplan.
- Les hyperplans affines divisent $\mathbb{R}^n$ en deux parties: $> b$ et $< b$.

Machines à vecteurs de support

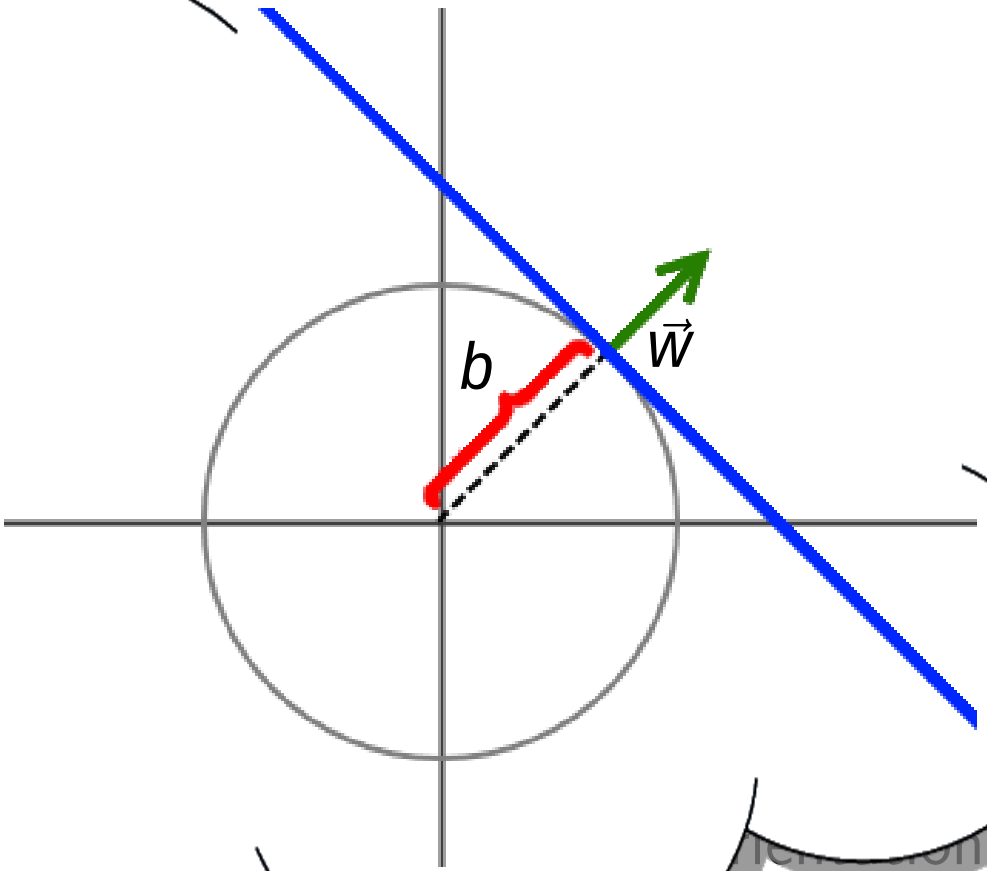
Hyperplan affine

Hyperplan

Un hyperpl de cet espace.

Hyp

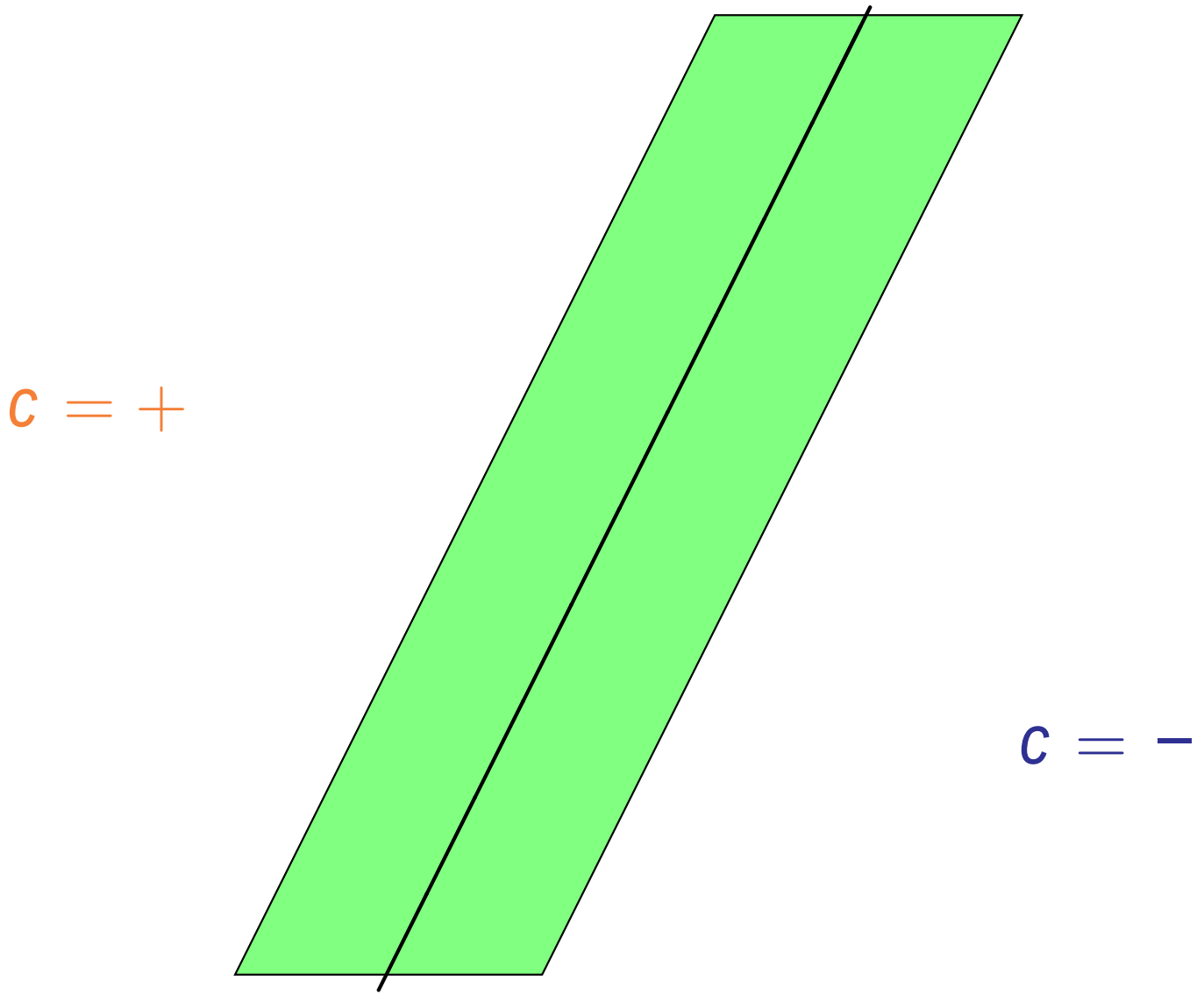
Un
$\sum w$



- En $= b$ où $\vec{w}$ est le ve de l'hyperplan.
- Les hyperplans affines deux parties: $> b$ et $< b$.

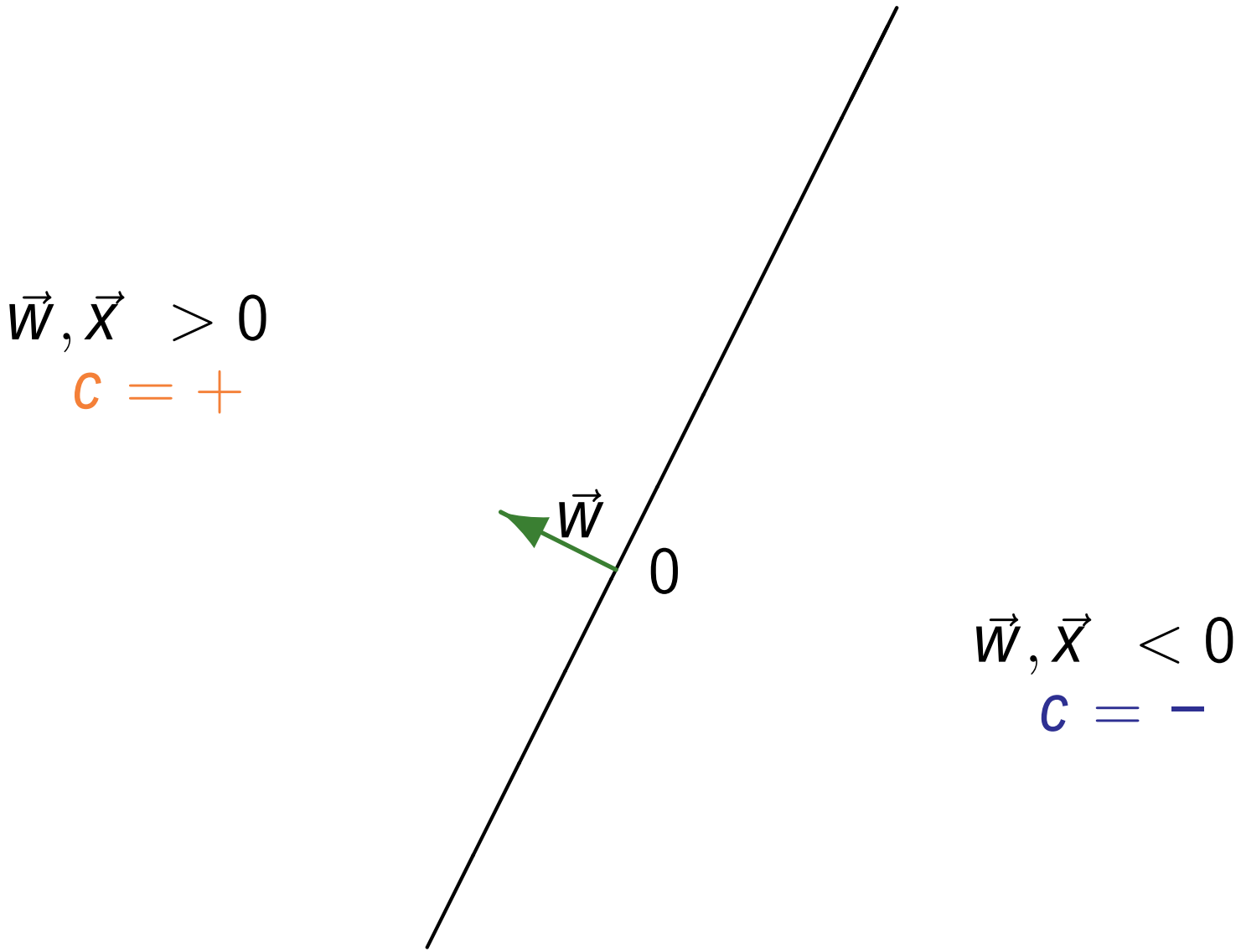
Machines à vecteurs de support

Hyperplan & formulation de la marge



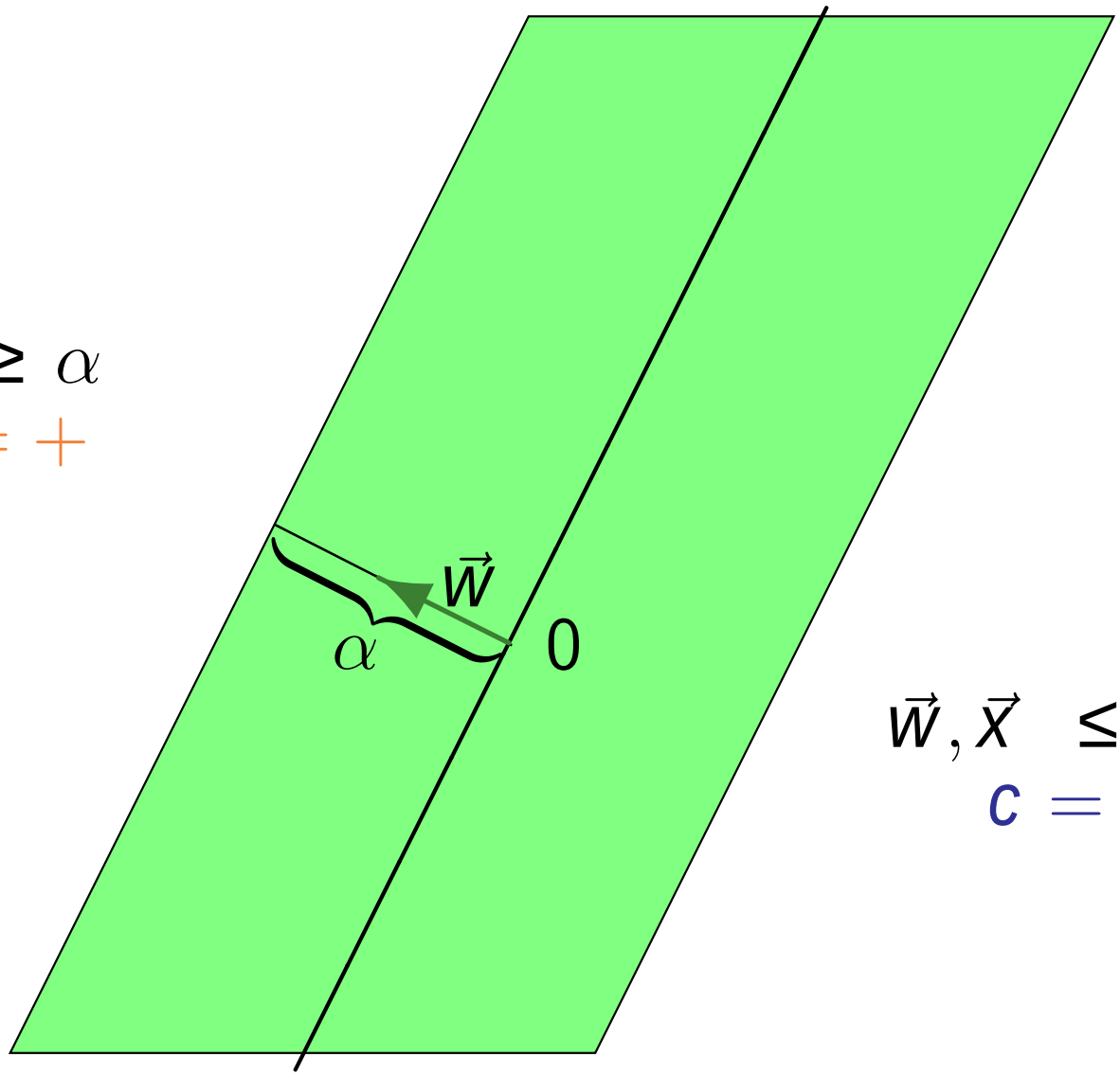
Machines à vecteurs de support

Hyperplan & formulation de la marge



Machines à vecteurs de support

Hyperplan & formulation de la marge



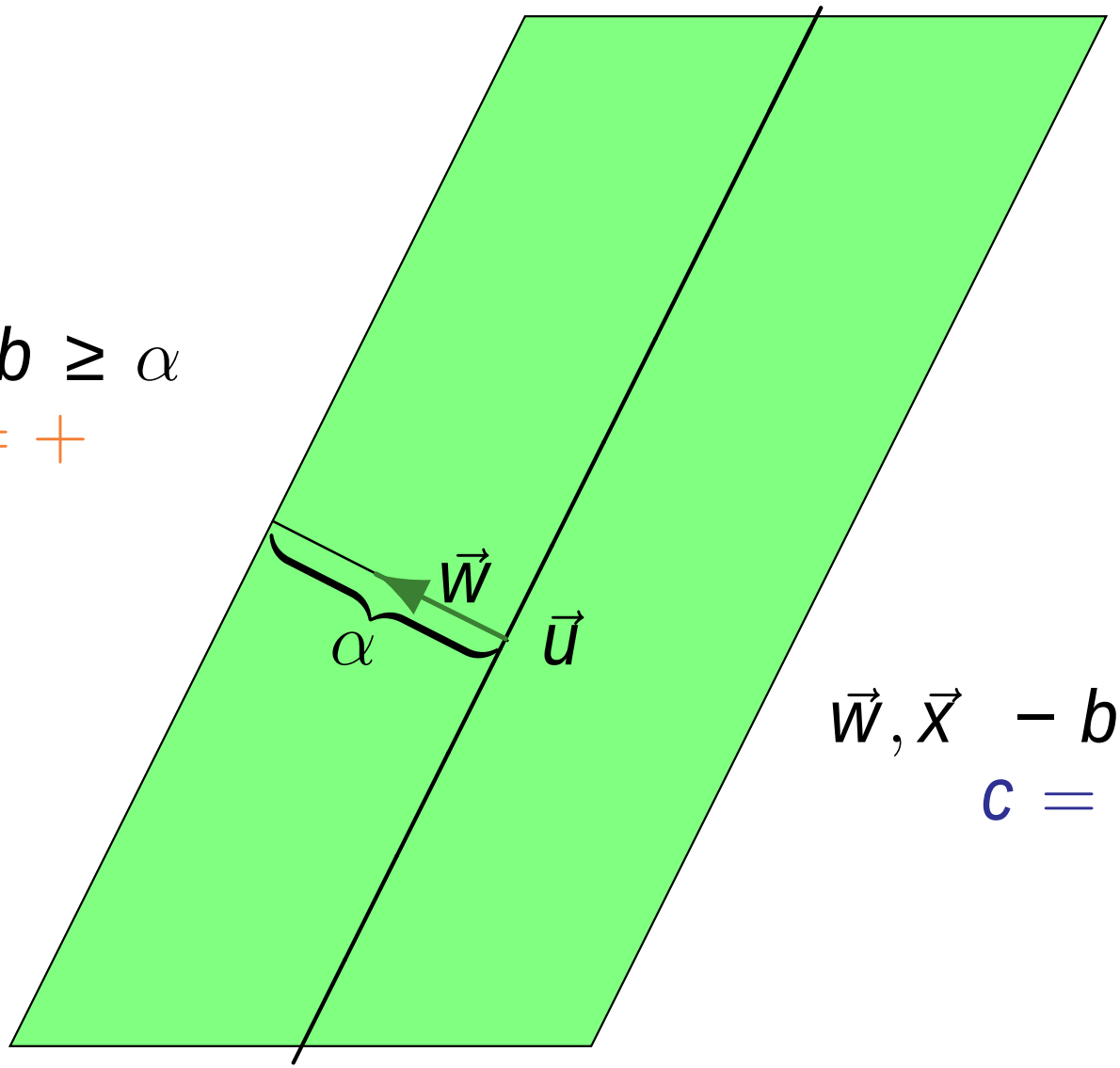
Machines à vecteurs de support

Hyperplan & formulation de la marge

$$b = \vec{w}, \vec{u}$$

$$\vec{w}, \vec{x} - b \geq \alpha$$

$$c = +$$



$$\vec{w}, \vec{x} - b \leq -\alpha$$

$$c = -$$

Entraînement de SVM

Entrées:

- Points de données $\vec{x}_1, \ldots, \vec{x}_N \quad \mathbb{R}^n$
- Cibles de classification $c_1, \ldots, c_N \quad \{-1, 1\}$

Résoudre le problème de la maximisation des marges :

$$\begin{array}{ll} \text{Find} & \max \alpha > 0 \\ \text{s.t.} & c_i(\ \vec{w}, \vec{x}_i\ - b) \geq \alpha \\ & \vec{w}\ = 1 \end{array}$$

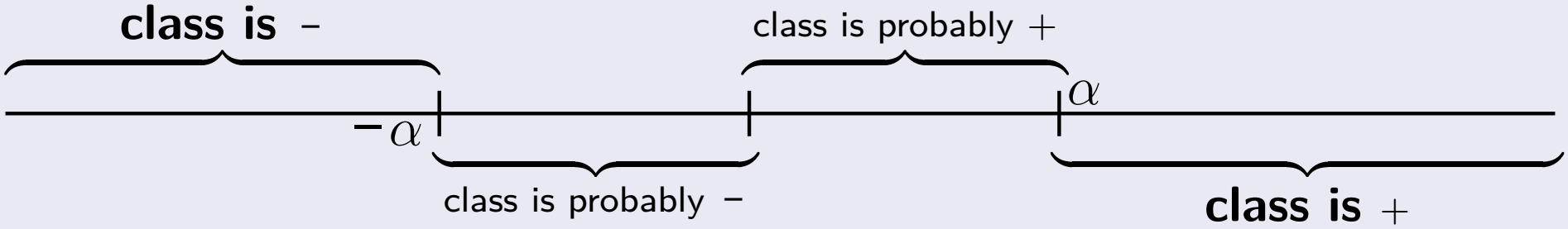
Sortie: la solution $(\vec{w}, b, \alpha)$

Classificateur SVM

Entrées:

- Nouveau point de données $\vec{x}$
- La solution $(\vec{w}, b, \alpha)$ de l'entraînement de SVM

Classer selon la valeur de $\langle \vec{w}, \vec{x} \rangle - b$:



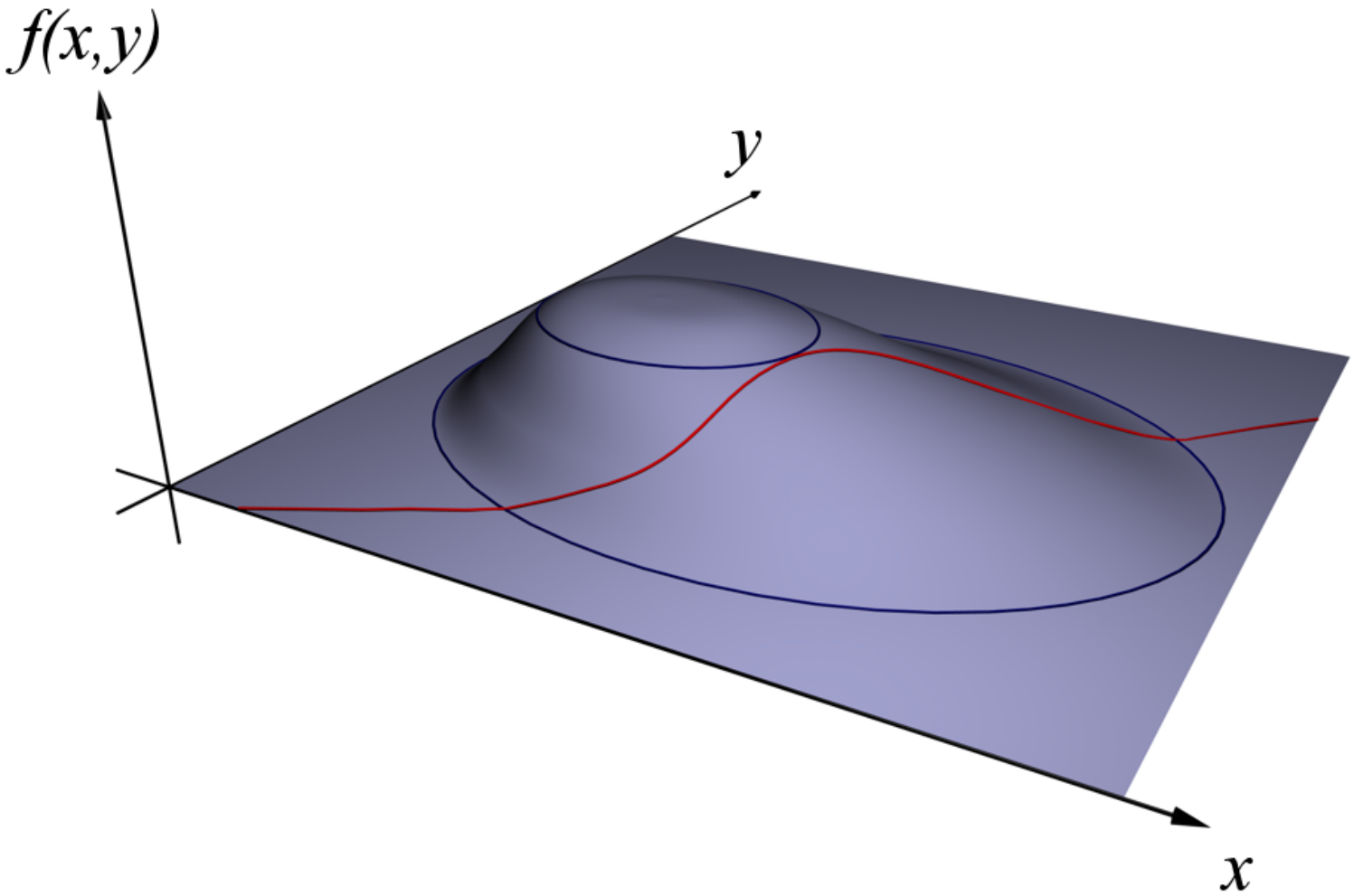
Considérons $\vec{w}_\alpha = \vec{w}/\alpha$ et $b_\alpha = \langle \vec{w}_\alpha, \vec{u} \rangle = b/\alpha$.
Alors, la contrainte de SVM peut être reformulée comme

$$c_i(\langle \vec{w}, \vec{x}_i \rangle - b) \geq \alpha \Leftrightarrow c_i(\langle \vec{w}_\alpha, \vec{x}_i \rangle - b_\alpha) \geq 1$$

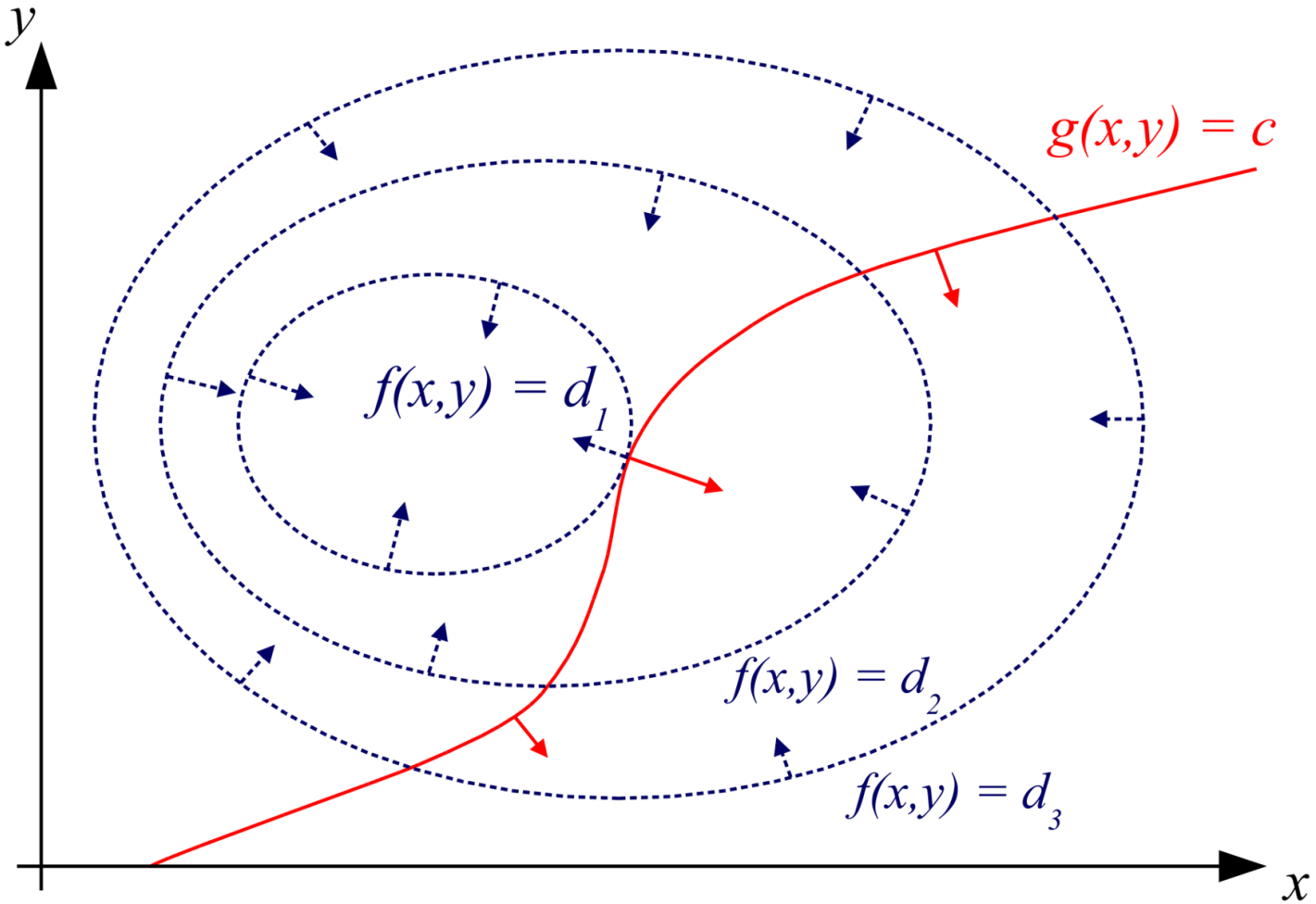
et la taille de la marge devient $2\alpha = 2/\|w_\alpha\|$. Donc on peut reformuler le problème d'optimisation de SVM comme:

$$\begin{array}{ll} \arg\min_{\vec{w}_\alpha, b_\alpha} & \|\vec{w}_\alpha\|^2/2 \\ \text{s.t.} \quad {\scriptstyle 1 \leq i \leq N} & c_i(\langle \vec{w}_\alpha, \vec{x}_i \rangle - b_\alpha) \geq 1 \end{array}$$

Évidemment, minimiser $\|\vec{w}_\alpha\|^2/2$ est équivalent à maximiser α. Avec cette formulation, on a un critère quadratique avec des contraintes linéaires. Telles optimisations sont convexes et donnent une solution unique.



Tirée de Wikipedia



Tirée de Wikipedia

Lagrangien

Compte tenu d'un problème d'optimisation $\arg\min_{\vec{z}} f(\vec{z})$ s.t. $g_j(\vec{z}) = 0, j = 1, \ldots, k$, sa solution optimale doit être un point stationnaire de la fonction Lagrangienne $f_P(\vec{z}, \lambda_1, \ldots, \lambda_k) = f(\vec{z}) - \sum_{j=1}^{k} \lambda_j g_j(\vec{z})$, i.e., un point où $f_P = 0$.

Remarquons que:

- Les points stationnaires peuvent être des points cols/selles, pas seulement des points extrêmes.
- Comme $\frac{\partial f_P}{\partial \lambda_j} = g_j$, il suffit de considérer les contraintes initiales et $\frac{\partial f_P}{\partial \vec{z}} = 0$.
- C'est une condition nécessaire, mais pas suffisante.

Application des multiplicateurs de Lagrange à:

$$\begin{array}{ll} \arg\min_{\vec{w},b} & \|\vec{w}\|^2/2 \\ \text{s.t.} \quad {\scriptstyle 1\le i\le N} & c_i(\langle \vec{w},\vec{x}_i\rangle - b) \ge 1 \end{array}$$

obtient le Lagrangien (primal)

$$f_P(\vec{w},b,\lambda_1,\ldots,\lambda_N) = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{N}\lambda_i\left(c_i(\langle \vec{w},\vec{x}_i\rangle - b) - 1\right)$$

et ensuite en mettant son gradient p/r à $\vec{w}$ et b à zéro on obtient:

$$\frac{\partial}{\partial\vec{w}}f_P = 0 \qquad \vec{w} = \sum_{i=1}^{N}\lambda_i c_i \vec{x}_i$$

$$\frac{\partial}{\partial b}f_P = 0 \qquad \sum_{i=1}^{N}\lambda_i c_i = 0$$

On peut maintenant substituer $\vec{w} = \sum_{i=1}^{N} \lambda_i c_i \vec{x}_i$ dans les contraintes linéaires pour obtenir N d'inégalités:

$$\sum_{j=1}^{N} c_i c_j \ \vec{x}_i, \vec{x}_j \ \lambda_j - c_i b \geq 1 \qquad i = 1, \ldots, N$$

S'il s'agissait de contraintes d'égalité, nous pourrions résoudre l'ensemble des équations $N + 1$ (avec $\sum_{i=1}^{N} \lambda_i c_i = 0$) dans les variables $N + 1$ $(b, \lambda_1, \ldots, \lambda_N)$. Cependant, comme il s'agit d'inégalités, cette approche directe ne s'applique pas ici.

Pour résoudre ce problème, on applique deux techniques d'optimisation avancées.

Transformation primal-dual: En substituant $\vec{w} = \sum_{i=1}^{N} \lambda_i c_i \vec{x}_i$ dans le lagrangien primal on obtient:

$$f_P = \frac{1}{2} \sum_{i,j=1}^{N} c_i c_j \lambda_i \lambda_j \ \vec{x}_i, \vec{x}_j \ - \sum_{i,j=1}^{N} c_i c_j \lambda_i \lambda_j \ \vec{x}_i, \vec{x}_j \ + b \sum_{i=1}^{N} c_i \lambda_i + \sum_{i=1}^{N} \lambda_i$$

Ensuite, en utilisant $\sum_{i=1}^{N} \lambda_i c_i = 0$ et en simplifiant on obtient le lagrangien dual:

$$f_D(\lambda_1, \ldots, \lambda_N) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{N} c_i c_j \lambda_i \lambda_j \ \vec{x}_i, \vec{x}_j$$

Transformation primal-dual: Trouvez le maximum du dual:

$$f_D(\lambda_1, \ldots, \lambda_N) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{N} c_i c_j \lambda_i \lambda_j \ \vec{x}_i, \vec{x}_j$$

Remarquons que:

- Comme le primal, le dual est aussi une forme quadratique
- Par contre, il ne dépend pas de $\vec{w}, b$ – seulement des multiplicateur $\lambda_1, \ldots, \lambda_N$
- Alors que le point stationnaire du primal est un minimum, pour le dual c'est un maximum

Transformation primal-dual: Trouvez le maximum du dual:

$$f_D(\lambda_1, \ldots, \lambda_N) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{N} c_i c_j \lambda_i \lambda_j \ \vec{x}_i, \vec{x}_j$$

Conditions KKT: On peut montrer que la solution de la minimisation originale satisfait pour chaque $i = 1, \ldots, N$:

$$\lambda_i \geq 0 \qquad \text{and} \qquad \lambda_i \left(c_i (\ \vec{w}, x_i \ - b) - 1 \right) = 0$$

Par conséquent, les $\vec{w}$ et b peuvent être calculés à partir de $\lambda_1, \ldots, \lambda_N \geq 0$ qui maximisent le dual, qui peut lui-même être estimé numériquement en utilisant la programmation quadratique.

Les frontières des marges sont données par $\vec{w}, \vec{x}_i \; - b = c_i = \pm 1$, donc tout $\vec{x}_i$ qui ne repose pas dessus doit avoir $\lambda_i = 0$, puisque $\lambda_i \left(c_i(\; \vec{w}, \vec{x}_i \; - b) - 1\right) = 0$. Par ailleurs, comme $\sum_{i=1}^{N} \lambda_i c_i \vec{x}_i = \vec{w} = 0$ (sinon les contraintes deviennent $_i - c_i b \geq 1$ et sont violées par une des classes), certains λ_i doivent être non nuls.

Vecteurs de support

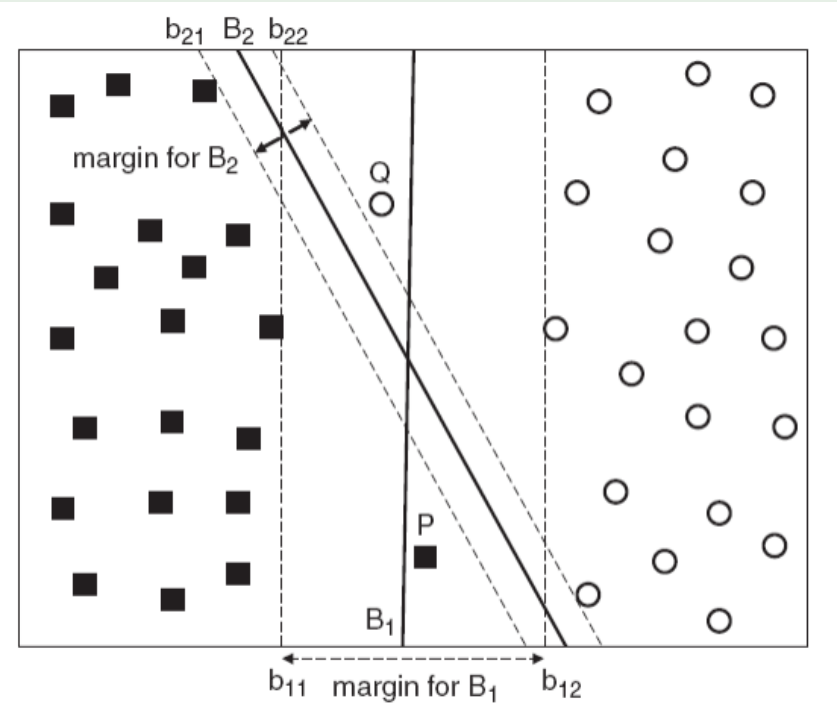
Les vecteurs de soutien sont les points dans $\{\vec{x}_i | \lambda_i = 0, i = 1, \ldots, N\} =$, qui se situent tous exactement sur les frontières de la marge.

Le $\vec{w}$ et le b ne dépendent que des vecteurs de support identifiés. Toutefois, en raison d'erreurs numériques, ces vecteurs peuvent donner plusieurs estimations de b, de sorte qu'en pratique, c'est l'estimation moyenne qui est utilisée.

Question: le SVM choisit-il toujours la meilleure marge de sép.?

Question: le SVM choisit-il toujours la meilleure marge de sép.?

Exemple

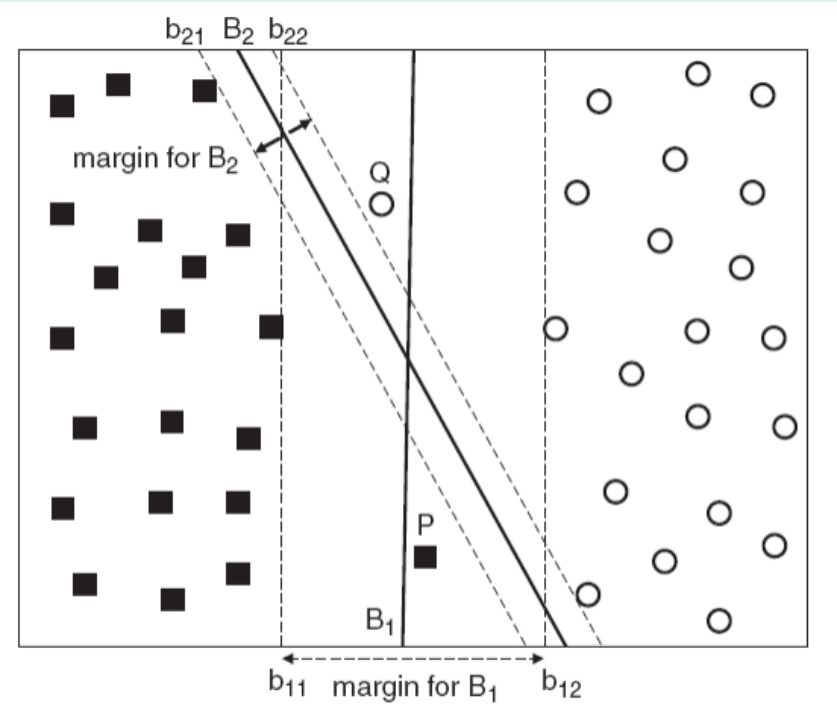


Machines à vecteurs de support

Marge souple

Question: le SVM choisit-il toujours la meilleure marge de sép.?

Exemple



Conclusion: des contraintes rigides rendent le SVM décrit sensible au bruit et aux anomalies!

Machines à vecteurs de support

Marge souple

On peut assouplir le modèle SVM en ajoutant des variables ressort $\xi_1, \ldots, \xi_N \geq 0$ et reformuler les contraintes comme:

$$_{1 \leq i \leq N} c_i(\ \vec{w}, \vec{x}_i\ \ - b) \geq 1 - \xi_i$$

Ainsi, dans la classe positive ($c_i = 1$), on obtient $\vec{w}, \vec{x}_i\ \ - b \geq 1 - \xi_i$ et dans la classe négative ($c_i = -1$) on obtient $\vec{w}, \vec{x}_i\ \ - b \leq -1 + \xi_i$.

Les nouvelles contraintes permettent aux points de données de se situer dans la marge, ou même du « mauvais » côté de la limite de décision, tandis que le « slack » $\xi_i \geq 0$ quantifie la distance entre x_i et la satisfaction des contraintes initiales.

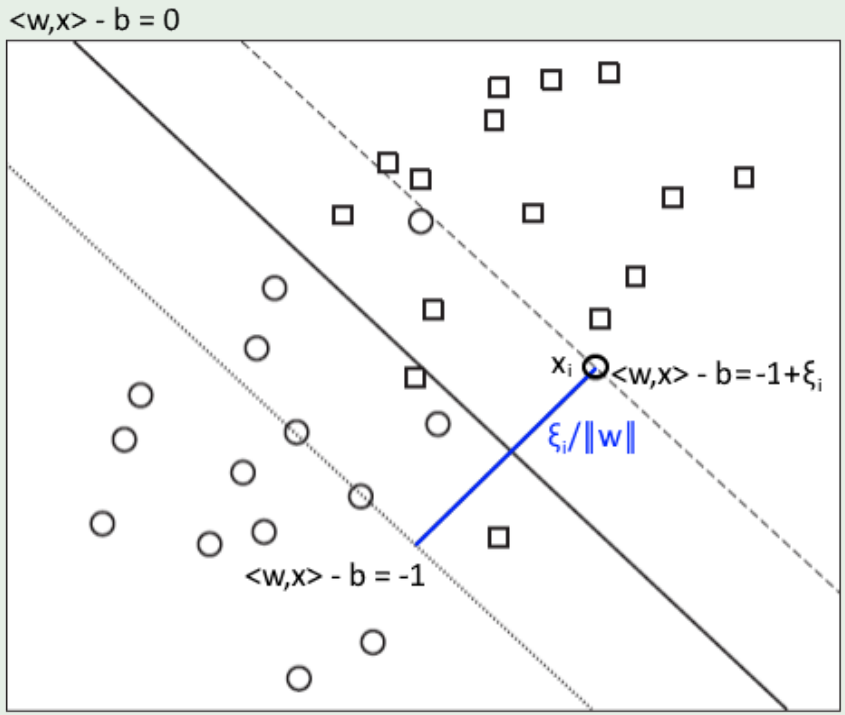
Semblable au poids de Lagrange $\lambda_i \geq 0$, lorsque $\xi_i = 0$ le point x_i est bien classé loin de la limite de la marge.

Machines à vecteurs de support

Marge souple

Les variables de «slack» non seulement rendent SVM plus robuste, mais elles permettent aussi de classer les données non séparables (ou non linéairement séparables) :

Exemple

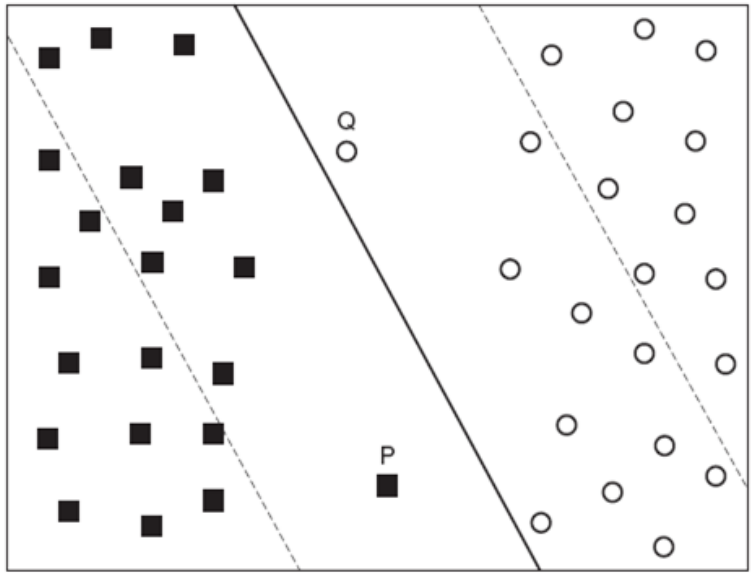


Machines à vecteurs de support

Marge souple

Avec les nouvelles contraintes, plusieurs choix d'hyperplans et de marges peuvent être adaptés pour les satisfaire avec des valeurs de relâchement appropriées.

Problème: l'objectif initial d'optimisation de la marge maximale est insuffisant avec ces contraintes, car il ne contrôle pas la quantité ou l'ampleur des erreurs de classification dans le modèle. Nous pouvons facilement obtenir une marge large où la plupart des données sont mal classées ou se trouvent dans la marge.



Solution: adapter la fonction cible pour minimiser également les valeurs de relâchement.

Machines à vecteurs de support

Marge souple

La formation au SVM à marge souple utilise le problème d'optimisation suivant, où β est une constante configurable par l'utilisateur:

$$\begin{array}{ll} \arg\min_{\vec{w},b,\xi_1,\ldots,\xi_N} & \|\vec{w}\|^2/2 + \beta\sum_{i=1}^{N}\xi_i \\ \text{s.t.} \quad {\scriptstyle 1\le i\le N} & c_i(\langle \vec{w},\vec{x}_i\rangle - b) \ge 1-\xi_i \\ & \xi_i \ge 0 \end{array}$$

L'application des multiplicateurs de Lagrange donne le primal:

$$L_P = \|\vec{w}\|^2/2 + \beta\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\lambda_i(c_i(\langle \vec{w},\vec{x}_i\rangle - b) - 1 + \xi_i) - \sum_{i=1}^{N}\mu_i\xi_i$$

L'application des multiplicateurs de Lagrange donne le primal:

$$L_P = \|\vec{w}\|^2/2 + \beta \sum_{i=1}^{N} \xi_i$$

$$- \sum_{i=1}^{N} \lambda_i (c_i(\langle \vec{w}, \vec{x}_i \rangle - b) - 1 + \xi_i) - \sum_{i=1}^{N} \mu_i \xi_i$$

En fixant $L_P = 0$, on obtient:

$$\frac{\partial L_P}{\partial \vec{w}[j]} = 0 \qquad \vec{w}[j] - \sum_{i=1}^{N} \lambda_i c_i x_i[j] = 0 \qquad \vec{w} = \sum_{i=1}^{N} \lambda_i c_i \vec{x}_i$$

$$\frac{\partial L_P}{\partial b} = 0 \qquad - \sum_{i=1}^{N} \lambda_i c_i \qquad \sum_{i=1}^{N} \lambda_i c_i = 0$$

$$\frac{\partial L_P}{\partial \xi_i} = 0 \qquad \beta - \lambda_i - \mu_i = 0 \qquad \lambda_i + \mu_i = \beta$$

Ainsi, on obtient le dual:

$$
\begin{aligned}
L_D &= \frac{1}{2}\sum_{i,j=1}^{N} \lambda_i\lambda_j c_i c_j \ \langle x_i, x_j \rangle + \beta\sum_{i=1}^{N}\xi_i - \sum_{i,j=1}^{N} \lambda_i\lambda_j c_i c_j \ \langle x_i, x_j \rangle \\
&\quad + b\sum_{i=1}^{N} c_i\lambda_i + \sum_{i=1}^{N}\lambda_i - \sum_{i=1}^{N}\lambda_i\xi_i - \sum_{i=1}^{N}(\beta - \lambda_i)\xi_i \\
&= \sum_{i=1}^{N}\lambda_i - \frac{1}{2}\sum_{i,j=1}^{N} \lambda_i\lambda_j c_i c_j \ \langle x_i, x_j \rangle
\end{aligned}
$$

Ce dual est le même que celui linéairement séparable. Cependant, dans ce cas, les conditions KKT exigent $\lambda_i \geq 0$ et $\mu_i \geq 0$, $i = 1, \ldots, N$, donc avec $\lambda_i + \mu_i = \beta$ on obtient des contraintes $0 \leq \lambda_i \leq \beta$.

Finalement, on applique la programmation quadratique pour résoudre:

$$\begin{aligned} &\arg\max_{\lambda_1,\ldots,\lambda_N} && \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{N} \lambda_i \lambda_j c_i c_j \ \langle x_i, x_j \rangle \\ &\text{s.t.} \quad {\scriptstyle 1 \le i \le N} && 0 \le \lambda_i \le \beta \\ & && \sum_{i=1}^{N} c_i \lambda_i = 0 \end{aligned}$$

Ensuite, on définit $\mu_i = \beta - \lambda_i$, $\vec{w} = \sum_{i=1}^{N} c_i \lambda_i x_i$ et récupère $b, \xi_1, \ldots, \xi_n$ des conditions KKT qui fournissent un ensemble suffisant d'équations linéaires:

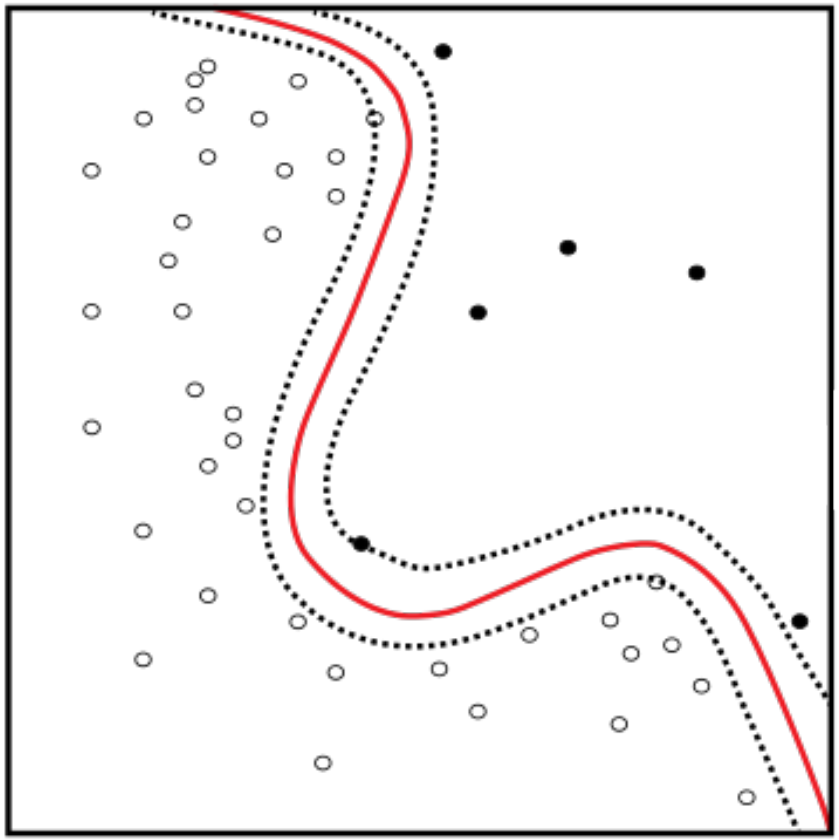
$$\mu_i \xi_i = 0 \quad \text{and} \quad \lambda_i \left(c_i (\langle \vec{w}, x_i \rangle - b) - 1 + \xi_i \right) = 0 \quad \text{and} \quad \xi_i \ge 0$$

Les vecs. de support ici sont situés sur les frontières de marge ou des points avec $\xi_i > 0$ (c-à-d mal classés & à l'intérieur d.l. marge).
Notez que le classificateur résultant n'a pas besoin des variables «slack» car il n'utilise que $\vec{w}$ et b comme toujours avec le SVM linéaire.

SVM non linéaire

Que faire si les données ne sont pas séparables de manière linéaire ?

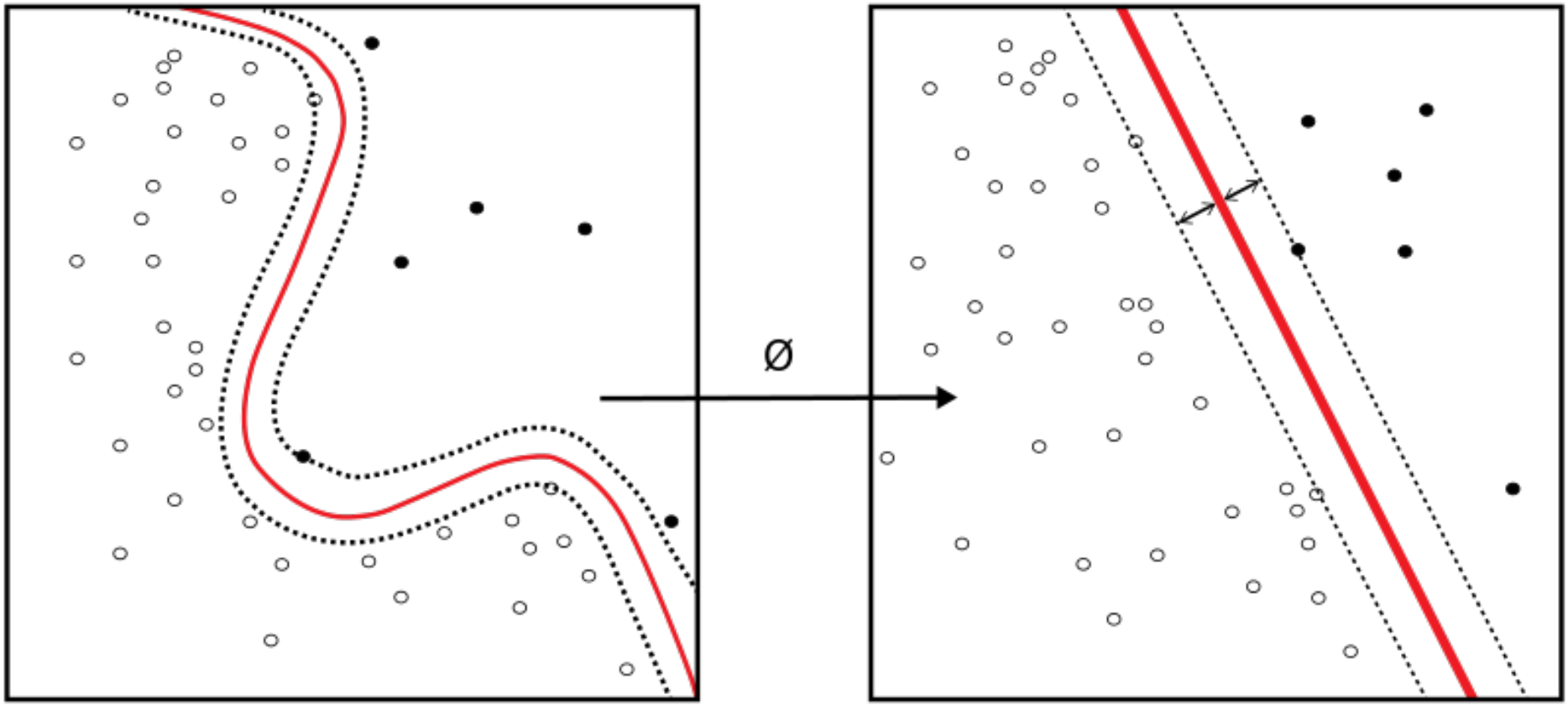
Question: Peut-on trouver une frontière de décision non linéaire?



SVM non linéaire

Que faire si les données ne sont pas séparables de manière linéaire ?

Question: Peut-on trouver une frontière de décision non linéaire?

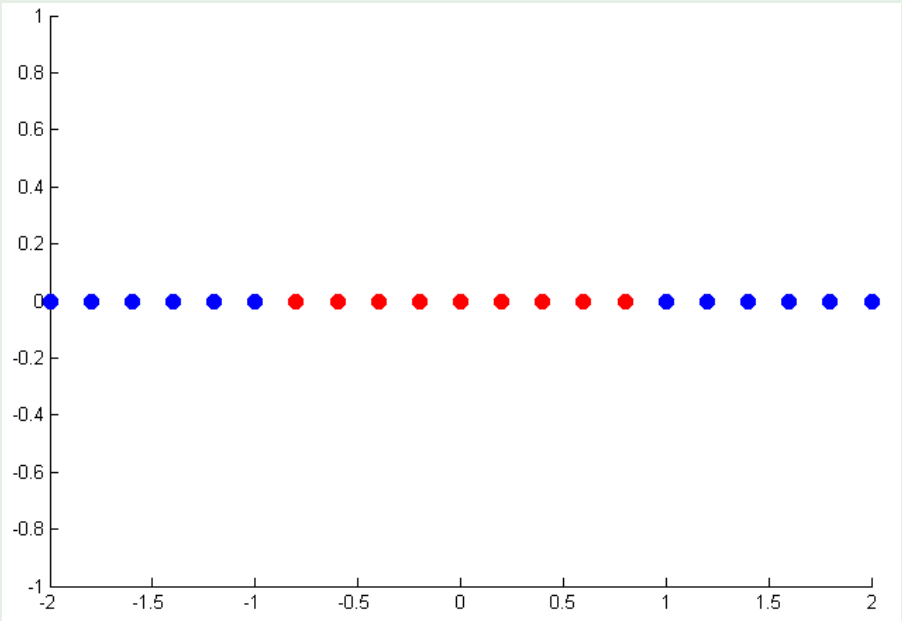


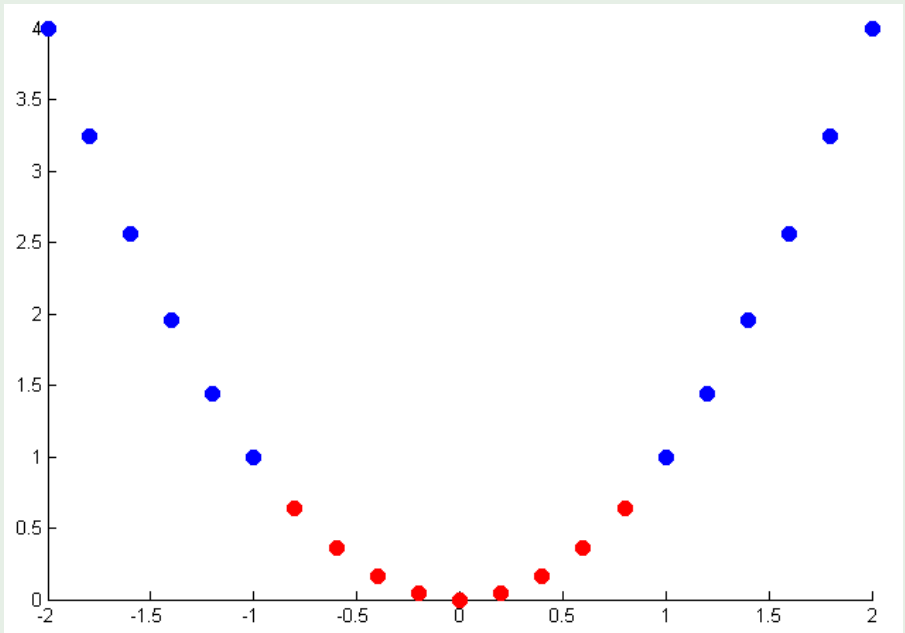
Solution: utiliser une transformation $x \rightarrow \Phi(x)$ des données en une nouvel espace de représentation où elles sont linéairement séparables.

SVM non linéaire

Typiquement, l'augmentation de la dimensionnalité des données non linéairement séparables peut les transformer en une représentation linéairement séparable.

Exemple

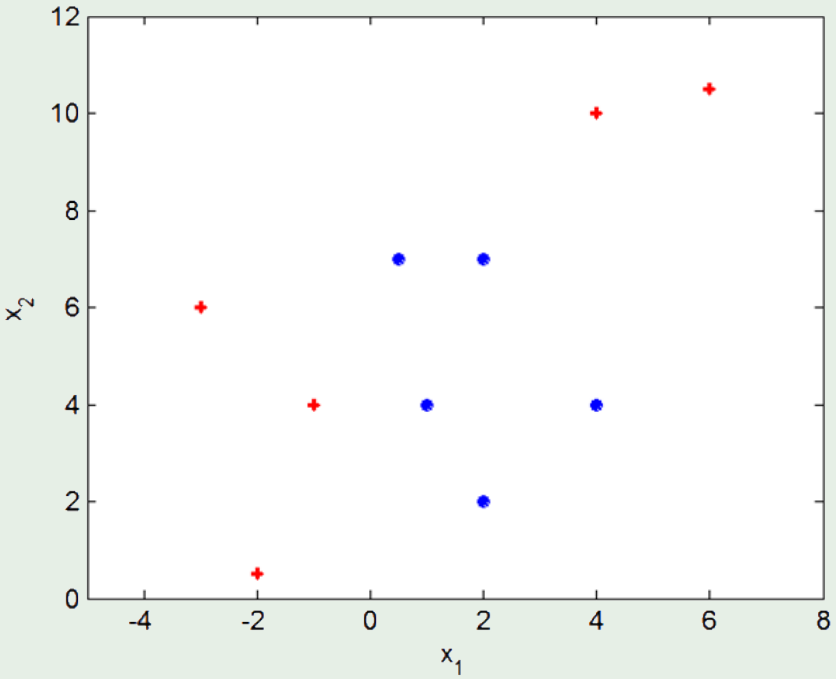




$$x \quad \mathbb{R} \rightarrow (x, x^2) \quad \mathbb{R}^2$$

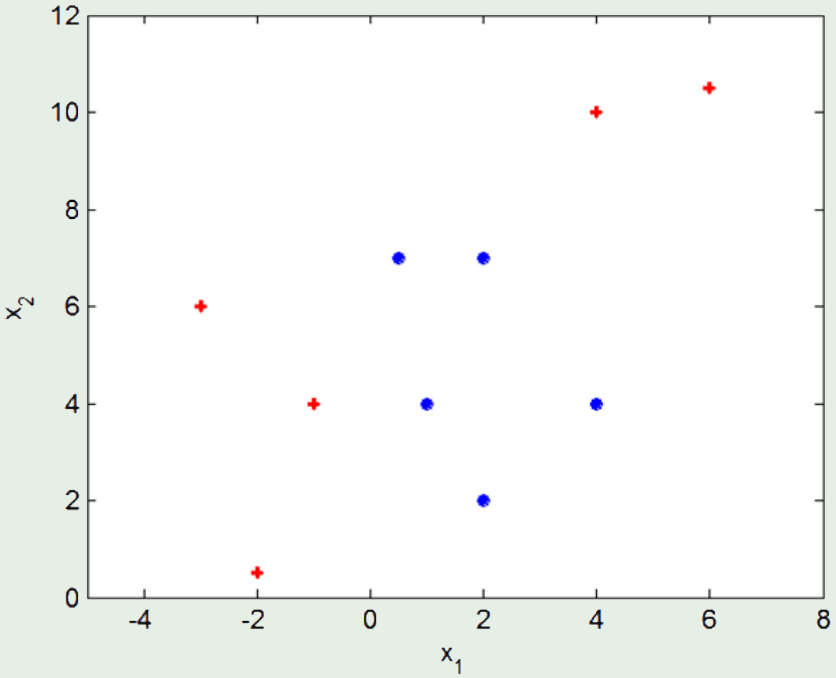
SVM non linéaire

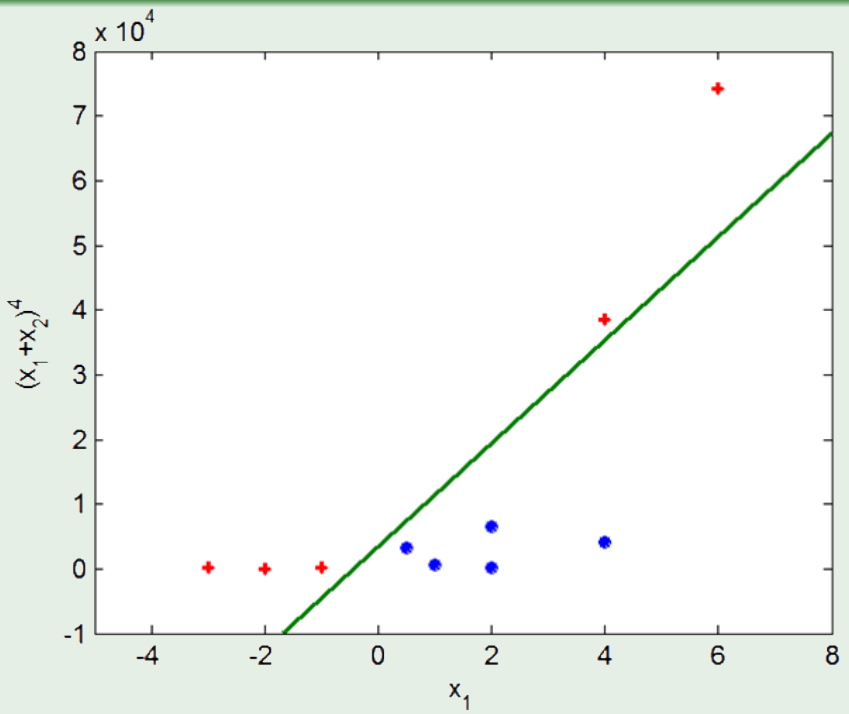
Exemple



SVM non linéaire

Exemple

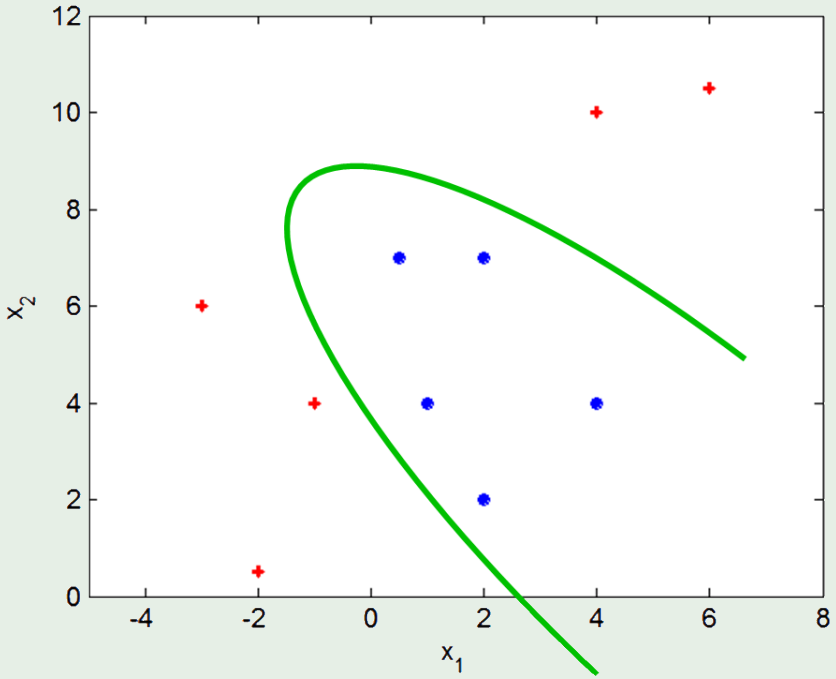


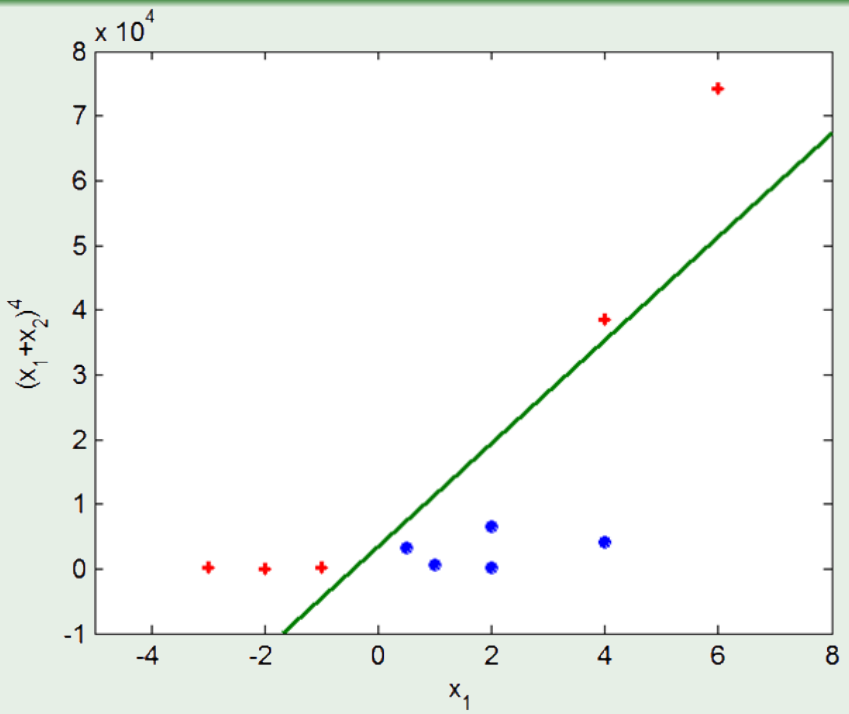


$$\Phi(x) = (x[1], (x[1] + x[2])^4)$$

SVM non linéaire

Exemple

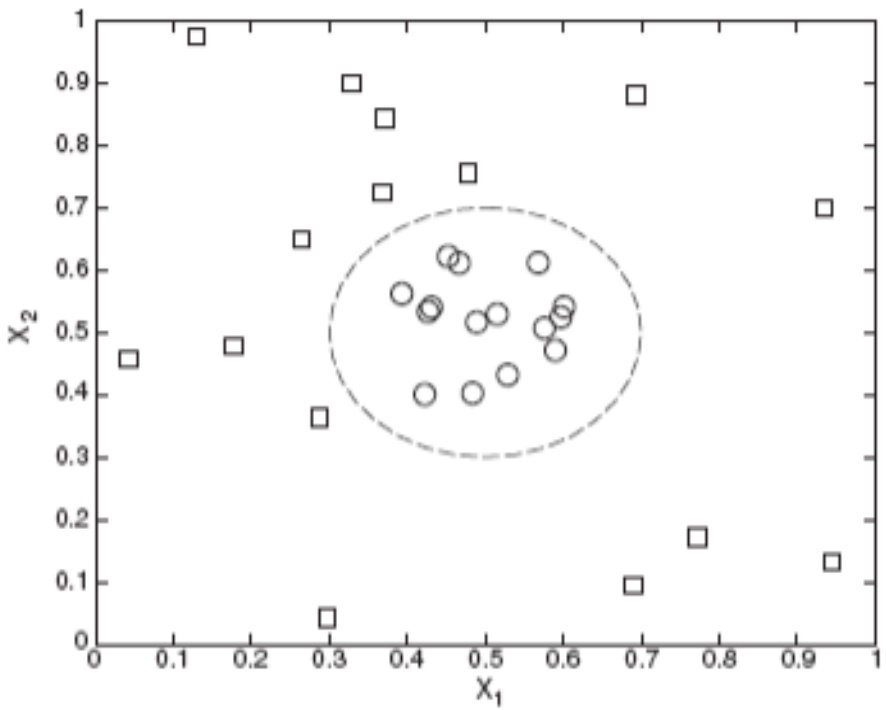


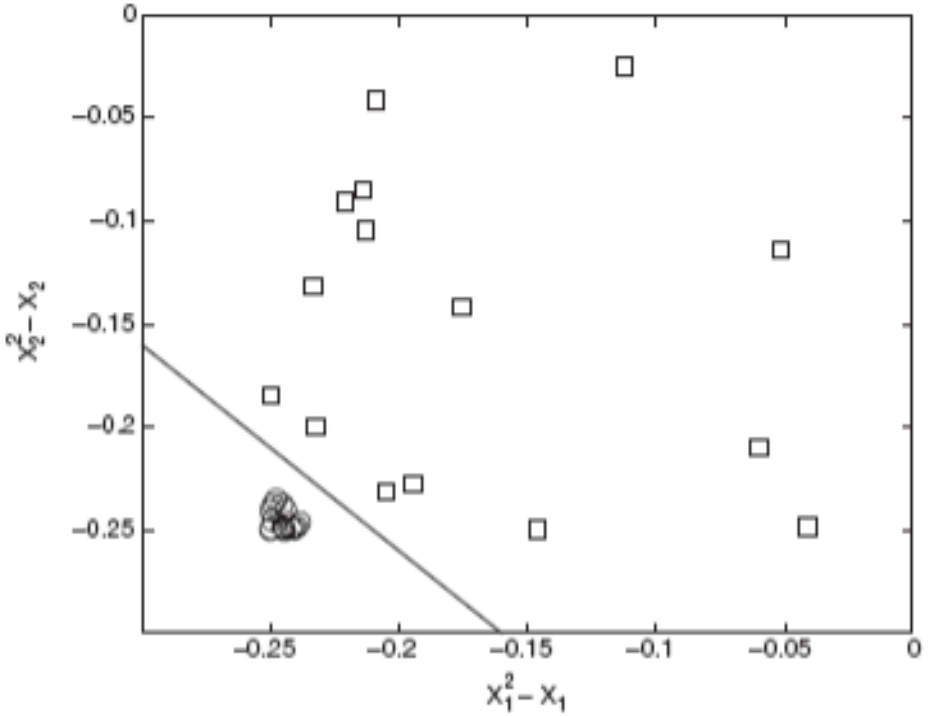


$$\Phi(x) = (x[1], (x[1] + x[2])^4)$$

SVM non linéaire

Exemple





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

SVM non linéaire

Extraction de représentation

En utilisant Φ, le SVM peut être reformulé pour utiliser ces caractéristiques plutôt que les données originales, qui ne doivent plus être en $\mathbb{R}^n$:

$$\begin{aligned} &\arg\min_{\vec{w},b} && \|\vec{w}\|^2/2 \\ &\text{s.t.} \quad _{1\le i\le N} && c_i(\langle \vec{w}, \Phi(x_i)\rangle - b) \ge 1 \end{aligned}$$

SVM non linéaire

Extraction de représentation

En utilisant Φ, le SVM peut être reformulé pour utiliser ces caractéristiques plutôt que les données originales, qui ne doivent plus être en $\mathbb{R}^n$:

$$\begin{aligned} &\arg\min_{\vec{w},b} && \|\vec{w}\|^2/2 \\ &\text{s.t.} \quad _{1\le i\le N} && c_i(\langle \vec{w}, \Phi(x_i)\rangle - b) \ge 1 \end{aligned}$$

L'optimisation suit les mêmes étapes qu'auparavant, et la classification tient compte $\langle \vec{w}, \Phi(y)\rangle - b$ pour un nouveau point y.

SVM non linéaire

Extraction de représentation

En utilisant Φ, le SVM peut être reformulé pour utiliser ces caractéristiques plutôt que les données originales, qui ne doivent plus être en $\mathbb{R}^n$:

$$\begin{aligned} &\arg\min_{\vec{w},b} && \|\vec{w}\|^2/2 \\ &\text{s.t.} \quad {\scriptstyle 1\le i\le N} && c_i(\langle \vec{w}, \Phi(x_i)\rangle - b) \ge 1 \end{aligned}$$

L'optimisation suit les mêmes étapes qu'auparavant, et la classification tient compte $\langle \vec{w}, \Phi(y)\rangle - b$ pour un nouveau point y.

Les transformations de représentation peuvent être **conçus** (p.ex., «filter banks» et «scattering») ou **apprises** des données (p.ex., à l'apprentissage profond). Cependant, l'extraction des caractéristiques appropriées n'est pas toujours claire ou pratique.

SVM non linéaire

Extraction de représentation

En utilisant Φ, le SVM peut être reformulé pour utiliser ces caractéristiques plutôt que les données originales, qui ne doivent plus être en $\mathbb{R}^n$:

$$\begin{aligned} &\arg\min_{\vec{w},b} && \|\vec{w}\|^2/2 \\ &\text{s.t.} \quad {\scriptstyle 1\le i\le N} && c_i(\langle \vec{w}, \Phi(x_i)\rangle - b) \ge 1 \end{aligned}$$

L'optimisation suit les mêmes étapes qu'auparavant, et la classification tient compte $\langle \vec{w}, \Phi(y)\rangle - b$ pour un nouveau point y.

Les transformations de représentation peuvent être **conçus** (p.ex., «filter banks» et «scattering») ou **apprises** des données (p.ex., à l'apprentissage profond). Cependant, l'extraction des caractéristiques appropriées n'est pas toujours claire ou pratique.

Question: peut-on utiliser les relations entre les points de données (p.ex., des distances / similarités / dissimilitudes) au lieu d'utiliser directement les caractéristiques d'entrée ou d'extraction?

SVM non linéaire

L'astuce du noyau

On commence par traiter les produits scalaires comme des relations entre des points de données, et on considère l'optimisation de u dual de SVM:

$$\begin{array}{ll} \arg\max_{\lambda_1,\ldots,\lambda_N} & \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{N} c_i c_j \lambda_i \lambda_j \ \Phi(x_i), \Phi(x_j) \\ \text{s.t.} \quad 1 \le i \le N & \lambda_i \ge 0 \\ & \sum_{i=1}^{N} c_i \lambda_i = 0 \end{array}$$

Notez que cette optimisation ne nécessite pas l'accès à des caractéristiques explicites, mais seulement à des produits scalaires $\Phi(x_i), \Phi(x_j)$

Ainsi, les multiplicateurs de Lagrange $\lambda_1, \ldots, \lambda_N$ peuvent être récupérés en utilisant uniquement les produits scalaires des caractéristiques.

SVM non linéaire

L'astuce du noyau

Rappelons que $\vec{w} = \sum_{i=1}^{N} c_i \lambda_i \Phi(x_i)$, et par conséquent on peut reformuler l'extraction de b comme:

$$\lambda_j \left(c_j\left(\ \vec{w}, \Phi(x_j)\ \ - b\right) - 1\right) = 0 \qquad \lambda_j = 0$$

$$c_j\left(\ \vec{w}, \Phi(x_j)\ \ - b\right) = 1 \overset{c_j = \pm 1}{=\!=\!=} \ \vec{w}, \Phi(x_j)\ \ - b = c_j$$

$$b = \ \vec{w}, \Phi(x_j)\ \ - c_j = \sum_{i=1}^{N} c_i \lambda_i \ \Phi(x_i), \Phi(x_j)\ \ - c_j$$

$$b = \text{mean}\left\{\sum_{i=1}^{N} c_i \lambda_i \ \Phi(x_i), \Phi(x_j)\ \ - c_j | \lambda_j > 0, j = 1, \ldots, N\right\}$$

Donc, b peut également être récupéré en utilisant uniquement les produits scalaires, sans accès aux caractéristiques elles-mêmes.

SVM non linéaire

L'astuce du noyau

La dernière étape de l'algorithme SVM consiste à classer un nouveau point de données y en testant $\vec{w}, \Phi(y) \ - b$, mais en utilisant encore $\vec{w} = \sum_{i=1}^{N} c_i \lambda_i \Phi(x_i)$ on obtient la règle de classification:

$$\sum_{i=1}^{N} c_i \lambda_i \ \Phi(x_i), \Phi(y) \ - b \quad \begin{cases} (-\infty, -1] & \text{class is } - \\ (-1, 0) & \text{class is probably } - \\ [0, 1) & \text{class is probably } + \\ [1, \infty) & \text{class is } + \end{cases}$$

Cette règle ne repose pas sur la récupération de $\vec{w}$, donc une fois que l'on a b et $\lambda_1, \ldots, \lambda_N$ la classification peut être faite selon les produits scalaires, sans exiger de caractéristiques explicites.

SVM non linéaire

L'astuce du noyau

Comme le SVM ne repose que sur des produits scalaire, il peut être reformulé à l'aide d'une **fonction de noyau** $k : X \times X \to \mathbb{R}$, t.q. $k(x, y) = \langle \Phi(x), \Phi(y) \rangle$ au lieu d'utiliser explicitement Φ.

Il est souvent **plus simple de formuler des noyaux** que des caractéristiques, et ils **ne nécessitent pas de connaissance de la dimensionnalité** de l'espace de caractéristiques.

Exemple

Pour $\Phi : \mathbb{R}^2 \to \mathbb{R}^6$ qui transforme $\Phi(u) = (u[1]^2, u[2]^2, \sqrt{2}u[1], \sqrt{2}u[2], \sqrt{2}u[1]u[2], 1)$, on peut utiliser le noyau quadratique $k(x, y) = (\langle x, y \rangle + 1)^2 = x[1]^2 y[1]^2 + x[2]^2 y[2]^2 + 2x[1]y[1] + 2x[2]y[2] + 2x[1]x[2]y[1]y[2] + 1 = \langle \Phi(x), \Phi(y) \rangle$ sans calculer directement Φ.

Question: comment savoir quelles fonctions peuvent être utilisées comme noyau sans tenir compte des transformation de caractéristiques?

Question: comment savoir quelles fonctions peuvent être utilisées comme noyau sans tenir compte des transformation de caractéristiques?

Le théorème de Mercer

Si $k(x,y)$ est symétrique, continu et semi-défini positif (c-à-d $_{x_1,\ldots,x_n \in X; g \in L^2} \sum_{i=1}^{n} \sum_{j=1}^{n} k(x_i,x_j) g(x_i) g(x_j) \geq 0$, où X est l'espace de données et L^2 est l'espace des fonctions carrées intégrables sur les données), alors il existe une fonction Φ telle que $k(x,y) = \langle \Phi(x), \Phi(y) \rangle$.

Question: comment savoir quelles fonctions peuvent être utilisées comme noyau sans tenir compte des transformation de caractéristiques?

Le théorème de Mercer

Si $k(x,y)$ est symétrique, continu et semi-défini positif (c-à-d $_{x_1,\ldots,x_n \square X; g \square L^2} \sum_{i=1}^{n} \sum_{j=1}^{n} k(x_i,x_j)g(x_i)g(x_j) \geq 0$, où X est l'espace de données et L^2 est l'espace des fonctions carrées intégrables sur les données), alors il existe une fonction Φ telle que $k(x,y) = \Phi(x), \Phi(y)$.

Un **noyau Mercer**, avec l'espace ambiant des données, définit un espace préhilbertien. Dans des contextes continus, ces espaces sont connus sous le nom de «**reproducing kernel Hilbert spaces (RKHS)**» et sont largement étudiées dans l'analyse fonctionnelle.

Entraînement du «Kernel SVM»

Entrée:

- Matrice d'un noyau Mercer $K \quad \mathbb{R}^{N\times N}$ t.q. $K_{ij} = k(x_i, x_j)$
- Cibles de classification $c_1, \ldots, c_N \quad \{-1, 1\}$

Résolvez le problème de maximisation :

$$\begin{array}{ll} \arg\max_{\lambda_1,\ldots,\lambda_N} & \sum_{i=1}^N \lambda_i - \frac{1}{2}\sum_{i,j=1}^N c_i c_j \lambda_i \lambda_j K_{ij} \\ \text{s.t.} \quad {\scriptstyle 1\le i\le N} & \lambda_i \ge 0 \\ & \sum_{i=1}^N c_i \lambda_i = 0 \end{array}$$

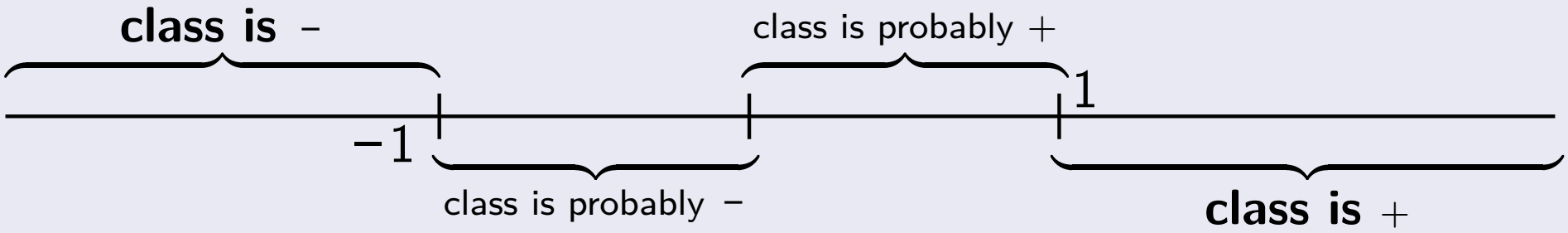
Calculez: $b = \text{mean}\{\sum_{i=1}^N c_i \lambda_i K_{ij} - c_j | \lambda_j > 0, j = 1, \ldots, N\}$
Sortie: les vecteurs de biais b and des poids $\vec{v} \quad \mathbb{R}^N$ t.q. $\vec{v}[i] = c_i \lambda_i$
$i = 1, \ldots, N$

classificateur « kernel SVM »

Entrée:

- Nouvelle rangée du noyau $\vec{k}_y \quad \mathbb{R}^N$ t.q. $\vec{k}_y[i] = k(y, x_i)$
- Les paramètres du modèle $b, \vec{v}$ entraîné

Classez par la valeur de $\sum_{i=1}^{N} \vec{v}[i]\vec{k}_y[i] - b$:



Remarquons que cette méthode peut être optimisée pour ne considérer que les vecteurs de support en éliminant tous les $\{i|\lambda_i = 0\}$, donc les $\vec{v}$ et les $\vec{k}_y$ peuvent être très clairsemés.

De nombreux noyaux dépendant de tâche ou de données peuvent être définis, mais le plupart des applications utilisent des noyaux standard populaires, tels que

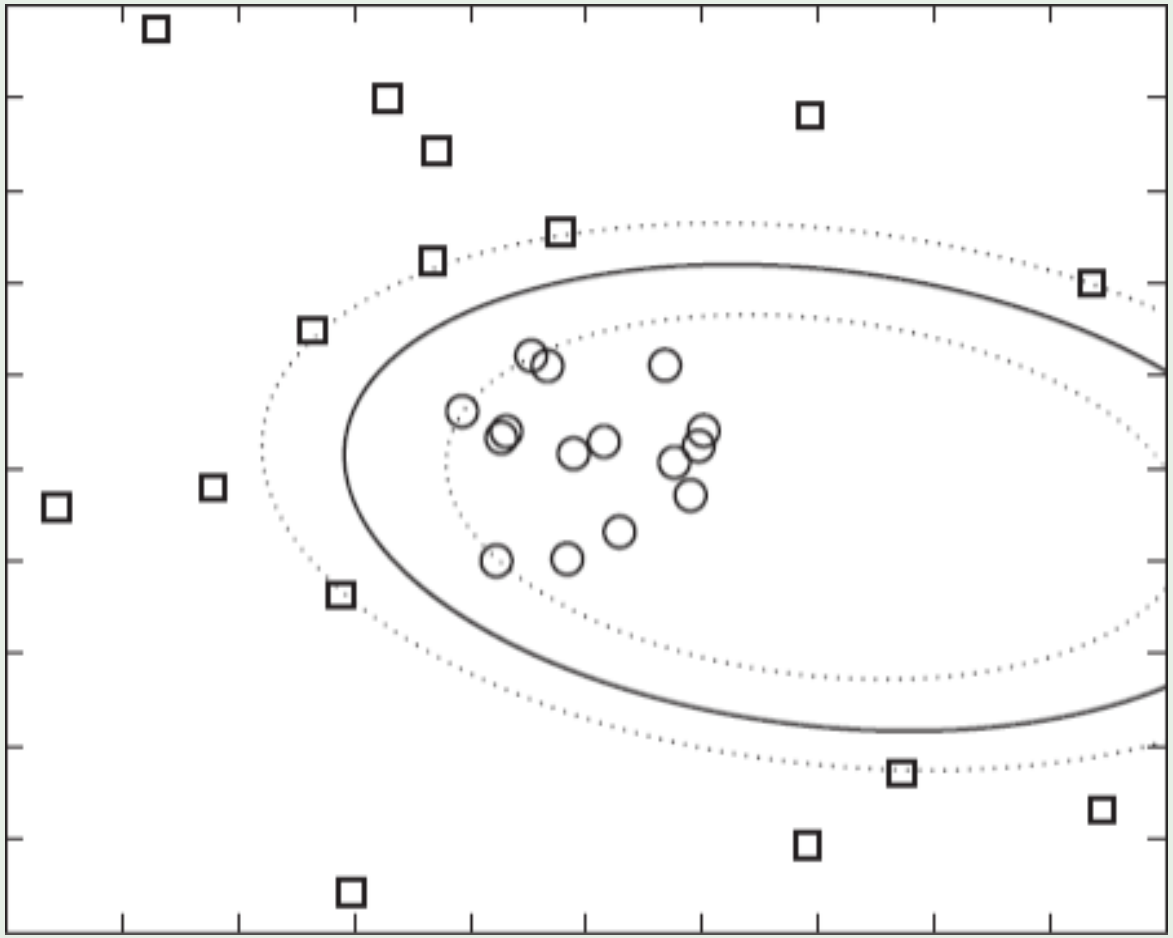
Noyau polynomial

Le noyau polynomial est défini par $k(x,y) = (\ x,y\ \ + 1)^p$, pour certain degré p. Les degrés particuliers comprennent $p = 1$, qui donne le SVM linéaire classique, et $p = 2$, qui donne le noyau quadratique.

Fonction de base radiale (RBF - Radial Basis Function)

Un noyau RBF est défini comme $k(x,y) = \exp\left(-\frac{x-y\ ^2}{\sigma^2}\right)$, pour certains $\sigma > 0$.

Exemple (Noyau polynomial)



Récapitulation

Le SVM linéaire définit un modèle de classification basé sur un hyperplan et la marge qui l'entoure.

- L'entraînement se fait par l'optimisation pour maximiser la largeur de la marge.
- Le classificateur formé ne dépend que d'un petit nombre de vecteurs de support.

Le kernel SVM étend ce modèle aux frontières de décision non linéaires.

- La formation au SVM ne repose que sur les produits internes aux données, qui sont remplacés par un noyau Mercer.
- Il est également possible de concevoir et d'apprendre des «feature maps» non linéaires (p.ex., via l'apprentissage profond).
- Les noyaux sont choisis et réglés par validation croisée de plusieurs alternatives.

Enfin, des marges souples peuvent être appliquées à la fois dans les cas linéaires et non linéaires pour permettre la robustesse au bruit et aux valeurs aberrantes.