WIKIPÉDIA

Machine à vecteurs de support

Les **machines à vecteurs de support** ou **séparateurs à vaste marge** (en anglais *support vector machine*, SVM) sont un ensemble de techniques d'apprentissage supervisé destinées à résoudre des problèmes de discrimination[note 1] et de régression. Les SVM sont une généralisation des classifieurs linéaires.

Les séparateurs à vaste marge ont été développés dans les années 1990 à partir des considérations théoriques de Vladimir Vapnik sur le développement d'une théorie statistique de l'apprentissage : la théorie de Vapnik-Chervonenkis. Ils ont rapidement été adoptés pour leur capacité à travailler avec des données de grandes dimensions, le faible nombre d'hyperparamètres, leurs garanties théoriques, et leurs bons résultats en pratique.

Les SVM ont été appliqués à de très nombreux domaines (bio-informatique, recherche d'information, vision par ordinateur, finance[1]...). Selon les données, la performance des machines à vecteurs de support est de même ordre, ou même supérieure, à celle d'un réseau de neurones ou d'un modèle de mélanges gaussiens[réf. souhaitée].

Sommaire

Historique

Les séparateurs à vastes marges reposent sur deux idées clés : la notion de marge maximale et la notion de fonction noyau. Ces deux notions existaient depuis plusieurs années avant qu'elles ne soient mises en commun pour construire les SVM.

L'idée des hyperplans à marge maximale a été explorée dès 1963 par Vladimir Vapnik et A. Lerner[2], et en 1973 par Richard Duda et Peter Hart dans leur livre *Pattern Classification*[3]. Les fondations théoriques des SVM ont été explorées par Vapnik et ses collègues dans les années 70 avec le développement de la théorie de Vapnik-Chervonenkis, et par Valiant et la théorie de l'apprentissage PAC[4].

L'idée des fonctions noyaux n'est pas non plus nouvelle : le théorème de Mercer date de 1909[5], et l'utilité des fonctions noyaux dans le contexte de l'apprentissage artificiel a été montré dès 1964 par Aizermann, Bravermann et Rozoener[6].

Ce n'est toutefois qu'en 1992 que ces idées seront bien comprises et rassemblées par Boser, Guyon et Vapnik dans un article, qui est l'article fondateur des séparateurs à vaste marge[7]. L'idée des variables ressorts, qui permet de résoudre certaines limitations pratiques importantes, ne sera introduite qu'en 1995. À partir de cette date, qui correspond à la publication du livre de Vapnik[8], les SVM gagnent en popularité et sont utilisés dans de nombreuses applications.

Un brevet américain sur les SVM est déposé en 1997 par les inventeurs originels[9].

Résumé intuitif

Les séparateurs à vastes marges sont des classificateurs qui reposent sur deux idées clés, qui permettent de traiter des problèmes de discrimination non linéaire, et de reformuler le problème de classement comme un problème d'optimisation quadratique.

La première idée clé est la notion de *marge maximale*. La marge est la distance entre la frontière de séparation et les échantillons les plus proches. Ces derniers sont appelés *vecteurs supports*. Dans les SVM, la frontière de séparation est choisie comme celle qui maximise la marge. Ce choix est justifié par la théorie de Vapnik-Chervonenkis (ou théorie statistique de l'apprentissage), qui montre que la frontière de séparation de marge maximale possède la plus petite capacité[10]. Le problème est de trouver cette frontière séparatrice optimale, à partir d'un ensemble d'apprentissage. Ceci est fait en formulant le problème comme un problème d'optimisation quadratique, pour lequel il existe des algorithmes connus.

Afin de pouvoir traiter des cas où les données ne sont pas linéairement séparables, la deuxième idée clé des SVM est de transformer l'espace de représentation des données d'entrées en un espace de plus grande dimension (possiblement de dimension infinie), dans lequel il est probable qu'il existe une séparation linéaire. Ceci est réalisé grâce à une fonction noyau, qui doit respecter les conditions du théorème de Mercer, et qui a l'avantage de ne pas nécessiter la connaissance explicite de la transformation à appliquer pour le changement d'espace. Les fonctions noyau permettent de transformer un produit scalaire dans un espace de grande dimension, ce qui est coûteux, en une simple évaluation ponctuelle d'une fonction. Cette technique est connue sous le nom de kernel trick.

Principe général

Les SVM peuvent être utilisés pour résoudre des problèmes de discrimination, c'est-à-dire décider à quelle classe appartient un échantillon, ou de régression, c'est-à-dire prédire la valeur numérique d'une variable. La résolution de ces deux problèmes passe par la construction d'une fonction h qui à un vecteur d'entrée x fait correspondre une sortie y :

$$y = h(x)$$

On se limite pour l'instant à un problème de discrimination à deux classes (discrimination binaire), c'est-à-dire $y \in \{-1, 1\}$, le vecteur d'entrée x étant dans un espace X muni d'un produit scalaire. On peut prendre par exemple $X = \mathbb{R}^N$.

Discrimination linéaire et hyperplan séparateur

Le cas simple est le cas d'une fonction discriminante linéaire, obtenue par combinaison linéaire du vecteur d'entrée $x = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^T$, avec un vecteur de poids $w = (\mathbf{w}_1, \ldots, \mathbf{w}_N)^T$:

$$h(x) = w^T x + w_0$$

Il est alors décidé que x est de classe 1 si $h(x) \geq 0$ et de classe -1 sinon. C'est un classifieur linéaire.

La frontière de décision $h(x) = 0$ est un hyperplan, appelé *hyperplan séparateur*, ou *séparatrice*. Le but d'un algorithme d'apprentissage supervisé est d'apprendre la fonction h(x) par le biais d'un ensemble d'apprentissage :

$$\{(x_1, l_1), (x_2, l_2), \ldots, (x_k, l_k), \ldots, (x_p, l_p)\} \subset \mathbb{R}^N \times \{-1, 1\}$$

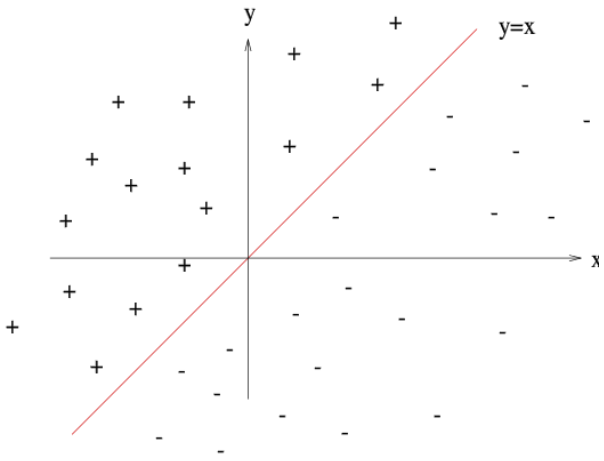
où les l_k sont les labels, p est la taille de l'ensemble d'apprentissage, N la dimension des vecteurs d'entrée. Si le problème est linéairement séparable, on doit alors avoir :

$$l_k h(x_k) \geq 0 \quad 1 \leq k \leq p, \qquad \textbf{autrement dit} \qquad l_k(w^T x_k + w_0) \geq 0 \quad 1 \leq k \leq p.$$

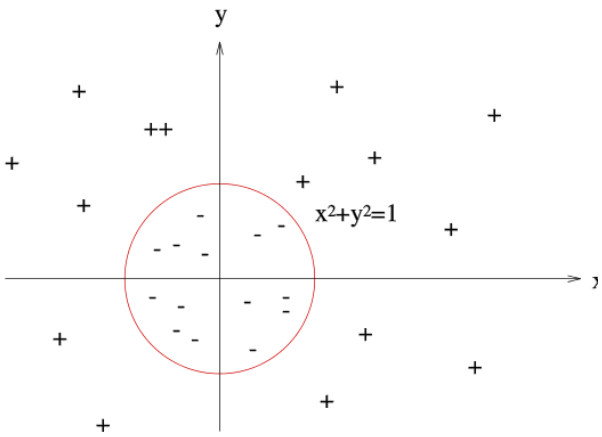
Exemple

Imaginons un plan (espace à deux dimensions) dans lequel sont répartis deux groupes de points. Ces points sont associés à un groupe : les points (+) pour $y > x$ et les points (-) pour $y < x$. On peut trouver un séparateur linéaire évident dans cet exemple, la droite d'équation $y = x$. Le problème est dit *linéairement séparable*.

Pour des problèmes plus compliqués, il n'existe en général pas de séparatrice linéaire. Imaginons par exemple un plan dans lequel les points (-) sont regroupés à l'intérieur d'un cercle, avec des points (+) tout autour : aucun séparateur linéaire ne peut correctement séparer les groupes : le problème n'est pas *linéairement séparable*. Il n'existe pas d'hyperplan séparateur.



Exemple d'un problème de discrimination à deux classes, avec une séparatrice linéaire : la droite d'équation $y = x$. Le problème est *linéairement séparable*.

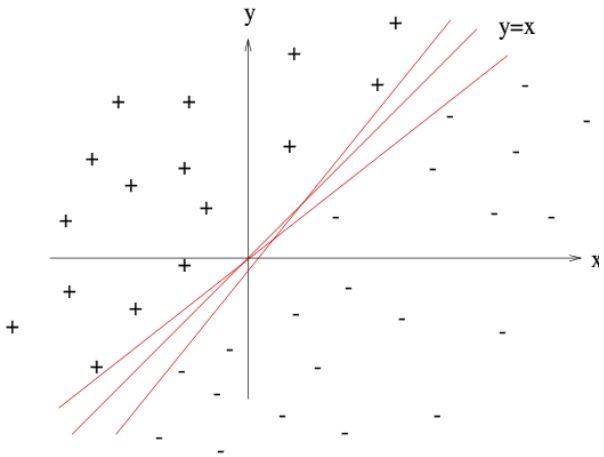


Exemple d'un problème de discrimination à deux classes, avec une séparatrice non-linéaire : le cercle unité. Le problème *n'est pas linéairement séparable*.

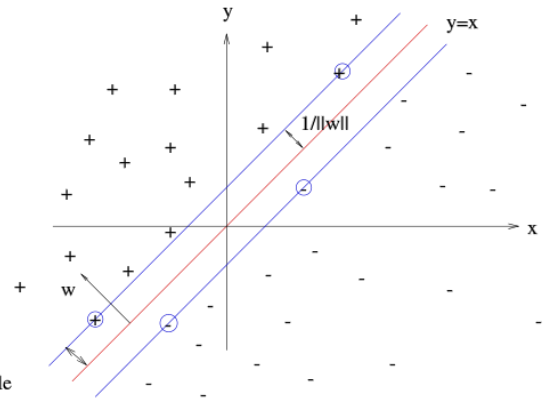
Marge maximale

On se place désormais dans le cas où le problème est linéairement séparable. Même dans ce cas simple, le choix de l'hyperplan séparateur n'est pas évident. Il existe en effet une infinité d'hyperplans séparateurs, dont les performances en apprentissage sont identiques (le risque empirique est le même), mais dont les performances en généralisation peuvent être très différentes. Pour résoudre ce problème, il a été montré[11], qu'il existe un unique hyperplan optimal, défini comme l'hyperplan qui maximise la marge entre les échantillons et l'hyperplan séparateur.

Il existe des raisons théoriques à ce choix. Vapnik a montré[11] que la capacité des classes d'hyperplans séparateurs diminue lorsque leur marge augmente.



Pour un ensemble de points linéairement séparables, il existe une infinité d'hyperplans séparateurs.



L'hyperplan optimal (en rouge) avec la marge maximale. Les échantillons entourés sont des *vecteurs supports*.

La marge est la distance entre l'hyperplan et les échantillons les plus proches. Ces derniers sont appelés **vecteurs supports**. L'hyperplan qui maximise la marge est donné par :

$$\arg\max_{w,w_0} \min_{k} \{||x - x_k|| : x \in \mathbb{R}^N, w^T x + w_0 = 0\}$$

Il s'agit donc de trouver w et w_0 remplissant ces conditions, afin de déterminer l'équation de l'hyperplan séparateur :

$$h(x) = w^T x + w_0 = 0$$

Recherche de l'hyperplan optimal

Formulation primale

La marge est la plus petite distance entre les échantillons d'apprentissage et l'hyperplan séparateur qui satisfasse la condition de séparabilité (à savoir $l_k(w^T x_k + w_0) \geq 0$ comme expliqué précédemment). La distance d'un échantillon x_k à l'hyperplan est donnée par sa projection orthogonale sur le vecteur de poids[note 2] :

$$\frac{l_k(w^T x_k + w_0)}{||w||}.$$

L'hyperplan séparateur (w,w_0) de marge maximale est donc donné par :

$$\arg\max_{w,w_0} \left\{ \frac{1}{||w||} \min_k \left[l_k(w^T x_k + w_0) \right] \right\}$$

Afin de faciliter l'optimisation, on choisit de normaliser w et w_0, de telle manière que les échantillons à la marge (x^+_{marge} pour les vecteurs supports sur la frontière positive, et x^-_{marge} pour ceux situés sur la frontière opposée) satisfassent :

$$\begin{cases} w^T x^+_{marge} + w_0 = 1 \\ w^T x^-_{marge} + w_0 = -1 \end{cases}$$

D'où pour tous les échantillons, $k = 1, \ldots, p$

$$l_k(w^T x_k + w_0) \geq 1$$

Cette normalisation est parfois appelée la forme canonique de l'hyperplan, ou *hyperplan canonique*[12].

Avec cette mise à l'échelle, la marge vaut désormais $\frac{1}{||w||}$, il s'agit donc de maximiser $||w||^{-1}$. La formulation dite *primale* des SVM s'exprime alors sous la forme suivante[note 3] :

$$\boxed{\text{Minimiser} \quad \frac{1}{2}||w||^2 \quad \text{sous les contraintes} \quad l_k(w^T x_k + w_0) \geq 1}$$

Ceci peut se résoudre par la méthode classique des multiplicateurs de Lagrange, où le lagrangien est donné par :

$$L(w, w_0, \alpha) = \frac{1}{2}||w||^2 - \sum_{k=1}^{p} \alpha_k \left\{ l_k(w^T x_k + w_0) - 1 \right\} \qquad (1)$$

Le lagrangien doit être minimisé par rapport à w et w_0, et maximisé par rapport à α.

Formulation duale

En annulant les dérivées partielles du lagrangien, selon les conditions de Kuhn-Tucker, on obtient :

$$\begin{cases} \sum_{k=1}^{p} \alpha_k l_k x_k & = w^* \\ \sum_{k=1}^{p} \alpha_k l_k & = 0 \end{cases}$$

En réinjectant ces valeurs dans l'équation (1), on obtient la *formulation duale* :

$$\boxed{\text{Maximiser } \tilde{L}(\alpha) = \sum_{k=1}^{p} \alpha_k - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j l_i l_j x_i^T x_j \qquad (2)}$$

$$\text{sous les contraintes } \alpha_k \geq 0 \text{ , et } \sum_{k=1}^{p} \alpha_k l_k = 0$$

Ce qui donne les multiplicateurs de Lagrange optimaux α_k^*.

Afin d'obtenir l'hyperplan solution, on remplace w par sa valeur optimale w*, dans l'équation de l'hyperplan $h(x)$, ce qui donne :

$$h(x) = \sum_{k=1}^{p} \alpha_k^* l_k (x \cdot x_k) + w_0$$

Conséquences

- Il y a trois remarques intéressantes à faire à propos de ce résultat. La première découle de l'une des conditions de Kuhn-Tucker, qui donne :

  $\alpha_k [l_k h(x_k) - 1] = 0 \quad 1 \leq k \leq p.$
  d'où
  $\begin{cases} \alpha_k & = 0 \\ l_k h(x_k) & = 1 \end{cases}$.
  Les seuls points pour lesquels les contraintes du lagrangien sont actives sont donc les points tels que $l_k h(x_k) = 1$, qui sont les points situés sur les hyperplans de marges maximales. En d'autres termes, seuls les *vecteurs supports* participent à la définition de l'hyperplan optimal.

- La deuxième remarque découle de la première. Seul un sous-ensemble restreint de points est nécessaire pour le calcul de la solution, les autres échantillons ne participant pas du tout à sa définition. Ceci est donc efficace au niveau de la complexité. D'autre part, le changement ou l'agrandissement de l'ensemble d'apprentissage a moins d'influence que dans un modèle de mélanges gaussiens par exemple, où tous les points participent à la solution. En particulier, le fait d'ajouter des échantillons à l'ensemble d'apprentissage qui ne sont pas des vecteurs supports n'a aucune influence sur la solution finale.

- La dernière remarque est que l'hyperplan solution ne dépend que du produit scalaire entre le vecteur d'entrée et les vecteurs supports. Cette remarque est l'origine de la deuxième innovation majeure des SVM : le passage par un espace de redescription grâce à une fonction noyau.

Cas non séparable : *kernel trick*

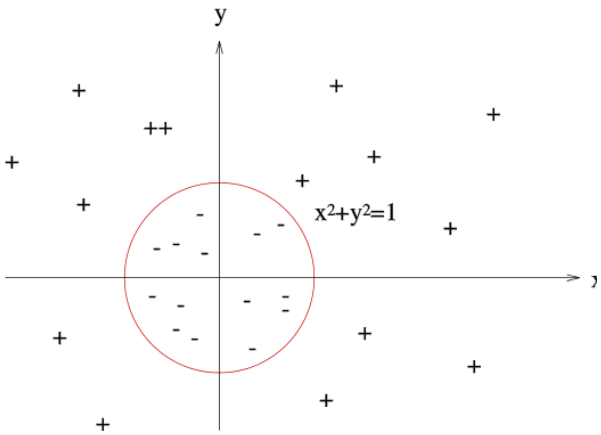
Principe

La notion de marge maximale et la procédure de recherche de l'hyperplan séparateur telles que présentées pour l'instant ne permettent de résoudre que des problèmes de discrimination linéairement séparables. C'est une limitation sévère qui condamne à ne pouvoir résoudre que des problèmes jouets, ou très particuliers. Afin de remédier au problème de l'absence de séparateur linéaire, l'idée de l'astuce du noyau (en anglais *kernel trick*) est de reconsidérer le problème dans un espace de dimension supérieure, éventuellement de dimension infinie. Dans ce nouvel espace, il est alors probable qu'il existe une séparation linéaire.

Plus formellement, on applique aux vecteurs d'entrée x une transformation non-linéaire ϕ. L'espace d'arrivée $\phi(X)$ est appelé ***espace de redescription***. Dans cet espace, on cherche alors l'hyperplan

$$h(x) = w^T \phi(x) + w_0$$

qui vérifie



Exemple simple de transformation : le problème n'est pas linéairement séparable en coordonnées cartésiennes, par contre en coordonnées polaires, le problème devient linéaire. Il s'agit ici d'un exemple très simple, l'espace de redescription étant de même dimension que l'espace d'entrée.

$l_k h(x_k) > 0$, pour tous les points x_k de l'ensemble d'apprentissage, c'est-à-dire l'hyperplan séparateur dans l'espace de redescription.

En utilisant la même procédure que dans le cas sans transformation, on aboutit au problème d'optimisation suivant :

$$\textbf{Maximiser } \tilde{L}(\alpha) = \sum_{k=1}^{p} \alpha_k - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j l_i l_j \phi(x_i)^T \phi(x_j) \qquad (3)$$

$$\textbf{Sous les contraintes } \alpha_i \geq 0 \textbf{ , et } \sum_{k=1}^{p} \alpha_k l_k = 0$$

Le problème de cette formulation est qu'elle implique un produit scalaire entre vecteurs dans l'espace de redescription, de dimension élevée, ce qui est couteux en termes de calculs. Pour résoudre ce problème, on utilise une astuce connue sous le nom de Kernel trick, qui consiste à utiliser une fonction noyau, qui vérifie :

$$K(x_i, x_j) = \phi(x_i)^T \cdot \phi(x_j)$$

d'où l'expression de l'hyperplan séparateur en fonction de la fonction noyau :

$$h(x) = \sum_{k=1}^{p} \alpha_k^* l_k K(x_k, x) + w_0$$

L'intérêt de la fonction noyau est double :

- Le calcul se fait dans l'espace d'origine, ceci est beaucoup moins coûteux qu'un produit scalaire en grande dimension.
- La transformation ϕ n'a pas besoin d'être connue explicitement, seule la fonction noyau intervient dans les calculs. On peut donc envisager des transformations complexes, et même des espaces de redescription de dimension infinie.

Choix de la fonction noyau

En pratique, on ne connaît pas la transformation ϕ, on construit plutôt directement une fonction noyau. Celle-ci doit respecter certaines conditions, elle doit correspondre à un produit scalaire dans un espace de grande dimension. Le théorème de Mercer explicite les conditions que K doit satisfaire pour être une fonction noyau : elle doit être symétrique, semi-définie positive.

L'exemple le plus simple de fonction noyau est le noyau linéaire :

$$K(x_i, x_j) = x_i^T \cdot x_j$$

On se ramène donc au cas d'un classifieur linéaire, sans changement d'espace. L'approche par Kernel trick généralise ainsi l'approche linéaire. Le noyau linéaire est parfois employé pour évaluer la difficulté d'un problème.

Des noyaux usuels employés avec les SVM sont :

- le noyau polynomial $K(x_i, x_j) = (x_i^T \cdot x_j + 1)^d$;
- le noyau gaussien $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2}\right)$.

Extensions

Marge souple

En général, il n'est pas non plus possible de trouver une séparatrice linéaire dans l'espace de redescription. Il se peut aussi que des échantillons soient mal étiquetés, et que l'hyperplan séparateur ne soit pas la meilleure solution au problème de classement.

En 1995, Corinna Cortes et Vladimir Vapnik proposent une technique dite de *marge souple*[13], qui tolère les mauvais classements. La technique cherche un hyperplan séparateur qui minimise le nombre d'erreurs grâce à l'introduction de *variables ressort* ξ_k (*slack variables* en anglais), qui permettent de relacher les contraintes sur les vecteurs d'apprentissage :

$$l_k(w^T x_k + w_0) \geq 1 - \xi_k \quad \xi_k \geq 0, \quad 1 \leq k \leq p$$

Avec les contraintes précédentes, le problème d'optimisation est modifié par un terme de pénalité, qui pénalise les variables ressort élevées :

$$\text{Minimiser } \frac{1}{2}||w||^2 + C\sum_{k=1}^{p}\xi_k \quad , \quad C > 0$$

où C est une constante qui permet de contrôler le compromis entre nombre d'erreurs de classement, et la largeur de la marge. Elle doit être choisie par l'utilisateur, en général par une recherche exhaustive dans l'espace des paramètres, en utilisant par exemple la validation croisée sur l'ensemble d'apprentissage. Le choix automatique de ce paramètre de régularisation est un problème statistique majeur.

Cas multi-classe

Plusieurs méthodes ont été proposées pour étendre le schéma ci-dessus au cas où plus de deux classes sont à séparer. Ces schémas sont applicables à tout classifieur binaire, et ne sont donc pas spécifiques aux SVM[14]. Les deux plus connues sont appelées *one versus all* et *one versus one*. Formellement, les échantillons d'apprentissage et de test peuvent ici être classés dans M classes $\{C_1, C_2, \ldots, C_M\}$.

La méthode *one-versus-all* (appelée parfois *one-versus-the-rest*) consiste à construire M classifieurs binaires en attribuant le label 1 aux échantillons de l'une des classes et le label -1 à toutes les autres. En phase de test, le classifieur donnant la valeur de confiance (e.g la marge) la plus élevée remporte le vote.

La méthode *one-versus-one* consiste à construire $M(M-1)/2$ classifieurs binaires en confrontant chacune des M classes. En phase de test, l'échantillon à classer est analysé par chaque classifieur et un vote majoritaire permet de déterminer sa classe. Si l'on note x_t l'échantillon à classer et $h_{ij}(.)$ le classifieur SVM séparant la classe C_i et la classe C_j et renvoyant le label attribué à l'échantillon à classer, alors le label attribué à x_t peut formellement se noter $Card(\{h_{i,j}(x_t)\} \cap \{k\}; i, j, k \in [1, M], i < j)$. C'est la classe qui sera le plus souvent attribuée à x_t quand il aura été analysé par tous les h_{ij}. Il peut exister une ambigüité dans le résultat du comptage, s'il n'existe pas de vote majoritaire[15]

Une généralisation de ces méthodes a été proposée en 1995[16] sous le nom d'*ECOC*, consistant à représenter les ensembles de classifieurs binaires comme des codes sur lesquels peuvent être appliqués les techniques de correction d'erreur.

Ces méthodes souffrent toutes de deux défauts. Dans la version one-versus-all, rien n'indique que les valeurs du résultat de classification des M classifieurs soient comparables (pas de normalisation, donc possibles problèmes d'échelle)[15]. De plus le problème n'est plus équilibré, par exemple avec M=10, on utilise seulement 10 % d'exemples positifs pour 90 % d'exemples négatifs.

SVM pour la régression

Vladimir Vapnik, Harris Drucker, Chris Burges, Linda Kaufman et Alex Smola ont proposé en 1996 une méthode pour utiliser des SVM afin de résoudre des problèmes de régression[17].

Notes et références

Notes

1. Le terme anglais pour *discrimination* est *classification*, qui a un sens différent en français (se rapporte au *clustering*). On utilise aussi le terme *classement* à la place de *discrimination*, plus proche du terme anglais, et plus compréhensible.
2. Pour un hyperplan d'équation $w^T x + w_0 = 0$, la distance de tout point *a* de l'espace à l'hyperplan est $\frac{|w^T a + w_0|}{||w||}$
3. Maximiser $||w||^{-1}$ est équivalent à minimiser $||w||^2$. Le carré est pris pour se débarrasser de la racine carrée incluse dans la norme. Le 1/2 n'est présent que pour faciliter la lisibilité des calculs et du résultat de l'optimisation.

Références

1. **(en)** Bernhard Schölkopf, Alexander J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization and Beyond*, 2002, MIT Press.
2. **(en)** Vladimir Vapnik et A. Lerner, *Pattern Recognition using Generalized Portrait Method*, Automation and Remote Control, 1963.

3. **(en)** Richard O. Duda, Peter E. Hart, David G. Stork, *Pattern Classification*, Wiley-interscience, 2001 (ISBN 0-471-05669-3) [détail des éditions].
4. **(en)** L.G. Valiant. *A Theory of the Learnable*, Communications of the ACM, 27(11), p. 1 134-1 142, novembre 1984.
5. **(en)** J. Mercer, *Functions of Positive and Negative Type, and their Connection with the Theory of Integral Equations*. Philos. Trans. Roy. Soc. London, A 209:415--446, 1909.
6. M. Aizerman, E. Braverman, and L. Rozonoer, *Theoretical foundations of the potential function method in pattern recognition learning*, Automation and Remote Control 25:821--837, 1964.
7. **(en)** Bernhard E. Boser, Isabelle M. Guyon, Vladimir N. Vapnik, *A Training Algorithm for Optimal Margin Classifiers (http://www.clopinet.com/isabelle/Papers/colt92.ps.Z)* In Fifth Annual Workshop on Computational Learning Theory, pages 144--152, Pittsburgh, ACM. 1992
8. V. Vapnik, *The Nature of Statistical Learning Theory*, N-Y, Springer-Verlag, 1995.
9. **(en)** *Pattern recognition system using support vectors*. B. Boser, I. Guyon, and V. Vapnik, US Patent 5,649,068 1997.
10. **(en)** Marti A. Hearst, *Support Vector Machines*, IEEE Intelligent Systems, vol. 13, no. 4, p. 18-28, Jul/Aug, 1998
11. V. Vapnik, et S. Kotz, *Estimation of Dependences Based on Empirical Data*, Springer Series in Statistics, 1982, (ISBN 978-0387907338).
12. **(en)** Bernhard Schölkopf, Alexander J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization and Beyond*, 2002, MIT Press, p. 190
13. **(en)** Corinna Cortes and V. Vapnik, « *Support-Vector Networks* », *Machine Learning*, 20, 1995. http://www.springerlink.com/content/k238jx04hm87j80g/.
14. **(en)** Christopher M. Bishop, *Pattern Recognition And Machine Learning*, Springer, 2006 (ISBN 0-387-31073-8) [détail des éditions], p. 182-184
15. **(en)** Christopher M. Bishop, *Pattern Recognition And Machine Learning*, Springer, 2006 (ISBN 0-387-31073-8) [détail des éditions], p. 338-339
16. Dietterich, T.G. and Bakiri, G., Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research. v2. 263-286.
17. **(en)** Harris Drucker, Chris J. C. Burges, Linda Kaufman, Alex Smola and Vladimir Vapnik (1997). « *Support Vector Regression Machines* ». *Advances in Neural Information Processing Systems 9, NIPS 1996*, 155-161, MIT Press.

Voir aussi

Bibliographie

- **(en)** Vapnik, V. Statistical Learning Theory. Wiley-Interscience, New York, (1998)
- **(en)** Bernhard Schölkopf, Alexander J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization and Beyond*, 2002, MIT Press.
- **(en)** John Shawe-Taylor, Nello Cristianini, *Support Vector Machines and other kernel-based learning methods*, Cambridge University Press, 2000.
- **(en)** Christopher J.C. Burges, *A Tutorial on Support Vector Machines for Pattern Recognition*, Data Mining and Knowledge Discovery, 2, 121–167 (1998)
- **(en)** Christopher M. Bishop, *Pattern Recognition And Machine Learning*, Springer, 2006 (ISBN 0-387-31073-8) [détail des éditions]

On trouve une présentation en français de la classification par les machines à vecteurs de support dans :

- Jean Beney, *Classification supervisée de documents : théorie et pratique*, Hermes Science, février 2008, 184p.
- Antoine Cornuéjols, Laurent Miclet, Yves Kodratoff, *Apprentissage Artificiel : Concepts et algorithmes*, Eyrolles, 2002 (ISBN 2-212-11020-0) [détail des éditions]

Articles connexes

- Apprentissage automatique
- Apprentissage supervisé
- Réseau de neurones
- Modèle de mélanges gaussiens
- Théorie de Vapnik-Chervonenkis
- Machine à vecteurs de pertinence **(en)**

Liens externes

- **(en)** Introduction aux SVM (http://www.support-vector.net/references.html)
- **(en)** SVM-Light - Implémentation en langage C de Joachims Thorsten (http://svmlight.joachims.org/)

La dernière modification de cette page a été faite le 8 avril 2018 à 12:50.