Théorie de l'apprentissage de Vapnik et SVM, Support Vector Machines

Yann GUERMEUR et Hélène PAUGAM-MOISY

LIP, URA CNRS 1398
Ecole Normale Supérieure de Lyon
46 allée d'Italie
F-69364 LYON cedex 07 - FRANCE

Résumé En reprenant les propos de Vapnik, on commencera par un bref historique et par la définition du problème de l'apprentissage, selon sa théorie. On énoncera les résultats fondamentaux concernant la minimisation du risque, du risque empirique et du risque structurel. On introduira ensuite la notion de "Support Vector" et l'on présentera successivement les SVM, "Support Vector Machines" linéaires (cas séparable et cas non séparable) et non linéaires. On montrera comment les SVM sont liées à l'utilisation de fonctions noyaux, aussi bien pour la résolution de problèmes de classification[1] que de régression.

Mots-Clés Théorie de l'apprentissage statistique, Classification, Régression et approximation de fonctions, Estimation de densités de probabilité, Réseaux de neurones.

Abstract Starting from Vapnik's talks, we present a brief historical review and a definition of the learning problem, according to his theory. Fundamental results about minimizing the risk functional, the principles of empirical risk minimization and structural risk minimization, are stated next. The notion of Support Vector is defined and different SVM, Support Vector Machines, are presented, for the separable linear case, for the non separable linear case and for the nonlinear case. The link between SVM and kernel functions will be emphasized, for solving both classification and regression problems.

Keywords Statistical Learning Theory, Classification, Regression, Probability Density Estimation, Neural Networks.

1 Introduction

Le problème de la "bonne" généralisation ne peut pas se réduire à celui de la recherche d'un "bon" apprentissage, contrairement à ce que l'on avait pu croire dans la première moitié de ce siècle (ex. analyse discriminante de Fisher dans les années 1920-30, principe du maximum de vraisemblance). En suivant Vapnik [58], on opposera deux philosophies : l'*analyse appliquée* et l'*analyse théorique* des processus d'apprentissage.

Tandis que l'analyse appliquée repose sur l'hypothèse, posée *a priori*, que la minimisation des erreurs sur l'ensemble d'apprentissage est un principe d'induction évident en soi, l'analyse théorique rejette cette hypothèse ou cherche si elle est justifiable, et dans quels cas. Le but de l'analyse théorique est plutôt de trouver un principe d'induction conduisant directement à une capacité de généralisation la plus grande possible, ainsi que de construire des algorithmes réalisant ce principe d'induction.

[1] Le mot "classification" satisfait ici à l'usage courant, mais le terme "classement" serait mieux approprié.

Notons que l'analyse appliquée des processus d'apprentissage avait cru trouver des justifications dans certains résultats établis à propos du Perceptron, notamment un théorème de convergence et des bornes [48, 41, 1]. Toutefois, la poursuite d'études allant dans ce sens, tout au long des années 1960-70, n'a pas fourni de résultats très convaincants, ni de portée très générale, en particulier dans le domaine des réseaux de neurones artificiels. C'est seulement en 1986 [33, 49] que cette approche a débouché sur la découverte de la méthode d'apprentissage par rétropropagation qui a donné leur essor aux réseaux multicouches et qui a, dans son élan, relancé toute la recherche dans le domaine connexionniste. À l'opposé, et pendant cette même période, ont commencé à se développer plusieurs méthodes et plusieurs théories que nous présentons succinctement.

1.1 Les bases de la théorie statistique de l'apprentissage [61, 57, 62, 58]

En se limitant tout d'abord à la reconnaissance des formes et aux fonctions indicatrices (à valeurs booléennes), Vapnik et Chervonenkis ont défini les notions de VC-dimension et de VC-entropie, notions à partir desquelles ils ont établi des conditions nécessaires et suffisantes à la convergence uniforme des fréquences vers leurs probabilités [61].

Définition 1. La dimension de Vapnik-Chervonenkis, ou **VC-dimension** en abrégé, d'un modèle d'apprentissage est la taille maximum d'un *échantillon* S (i.e. un ensemble d'exemples) qui peut être *pulvérisé* par le modèle (i.e. toutes les dichotomies sont réalisables sur S par le modèle).

Définition 2. La **VC-entropie** d'un modèle est l'espérance du logarithme de la diversité de l'ensemble des fonctions que le modèle peut réaliser (i.e. du nombre de séparations différentes possibles), sur un échantillon de taille donnée.

Puis ces résultats ont été étendus au cas des fonctions réelles : conditions nécessaires et suffisantes à la convergence uniforme des moyennes vers leurs espérances [57]. Ils ont ensuite établi des conditions assurant la consistance du principe inductif de minimisation du risque empirique, ainsi que de la méthode du maximum de vraisemblance [62]. Des résultats fondamentaux concernant la minimisation de la fonctionnelle de risque, celles du risque empirique et du risque structurel, sont développés dans [58], nous allons y revenir au cours de cet article.

1.2 La théorie de la régularisation [52, 28, 43]

Le regain d'intérêt actuel pour cette théorie est issu de la constatation suivante : la plupart des problèmes réels sont des *problèmes mal-posés*. La notion de problème mal-posé, ou plutôt celle de problème *bien-posé*, se définit, au sens d'Hadamard [25, 53], de la façon suivante :

Définition 3. Considérons qu'un problème est défini par une donnée g et une solution u. On dira que le problème est **bien-posé** si :

1. pour chaque donnée g, il existe une solution u dans un espace X de solutions potentielles,
2. la solution u est unique,
3. la solution u est stable, i.e. elle dépend de g de façon continue.

Par exemple, pour un opérateur linéaire M, la résolution de l'équation $Mu = g$, avec $u \in X$ peut être un problème mal-posé, en particulier en ce qui concerne la condition de stabilité. Cela signifie que, si l'on connaît seulement une valeur approchée g_ϵ de la donnée g, les fonctions u_ϵ qui minimisent la fonctionnelle

$$R(u) = ||Mu - g_\epsilon||^2$$

ne sont pas nécessairement des valeurs approchées de la solution u. Il s'est avéré, plusieurs dizaines d'années après la définition d'Hadamard, que les problèmes mal-posés n'étaient pas seulement des aberrations mathématiques, mais qu'il en existait de nombreux dans le monde réel. Citons par exemple la résolution des problèmes inverses (inversion des relations de causes à effets) ou l'estimation de fonctions de densité à partir de données empiriques. Des travaux développés dans les années 1960 [52, 28, 43] ont montré que certains problèmes pouvaient être ramenés à des problèmes bien-posés si l'on cherchait à minimiser, au lieu de $R(u)$, la fonctionnelle *régularisée* :

$$R^*(u) = ||Mu - g_\epsilon||^2 + \gamma(\epsilon)\Omega(f(u))$$

où $\Omega(f)$ est une certaine fonctionnelle bien choisie et $\gamma(\epsilon)$ un coefficient approprié, que l'on appelle *coefficient de régularisation*, et qui pondère l'influence du terme de régularisation par rapport à celle du terme d'erreur, dans l'expression à minimiser.

Notons que la théorie de la régularisation ne sera introduite de manière rigoureuse dans l'étude des réseaux de neurones qu'en 1990, par un article de Poggio et Girosi [44]. Certaines règles d'apprentissage, dites *régularisées*, comme celle connue sous le nom de "weight decay", consistent en l'ajout d'un terme contrôlant les variations du réseau dans l'expression de la fonction de coût.

1.3 Les méthodes statistiques non-paramétriques [47, 42, 14, 64]

La méthode du maximum de vraisemblance ne constitue pas un principe d'inférence adéquat à l'estimation de densités de probabilité, lorsque l'espace dans lequel on cherche les densités solutions est trop grand pour être déterminé par un nombre fini de paramètres. Les espaces de densités non-paramétriques nécessitent des méthodes basées sur l'utilisation de techniques de régularisation. Plusieurs types d'algorithmes non-paramétriques ont été proposés dans les années 1960 [47, 42, 14]. Une méthode plus systématique pour dériver de tels algorithmes des procédés s'appliquant aux problèmes mal-posés a été proposée à la fin des années 1970 [64]. Selon Vapnik, les techniques non-paramétriques ne sont applicables à l'estimation d'une densité qu'à partir d'un échantillon de très grande taille, taille qui n'est généralement pas disponible dans les problèmes réels (applications médicales, industrielles...). Nous ne nous étendrons pas sur le domaine, au demeurant très vaste, mais assez spécialisé, des méthodes non-paramétriques. Pour une distinction entre modèle paramétrique ou non et un lien avec les réseaux neuronaux, on consultera l'introduction de [24].

Toujours dans les années 1960-70, certaines prises de position ont rendu célèbres, puis ont discrédité, les partisans de la "ligne dure" de l'intelligence artificielle, ceux qui prétendaient que : "*Les algorithmes simples, eux, marchent, contrairement aux théories complexes.*". Il s'en est suivi une tentative, bien justifiée, d'introduire un aspect théorique dans les concepts de l'intelligence artificielle. Cette tentative s'est concrétisée par la proposition par Valiant, en 1984, du modèle d'apprentissage PAC, pour "Probablement Approximativement Correct" [55]. On notera que le modèle PAC est assez proche de la notion d'inférence non-paramétrique au sens de la théorie de l'apprentissage statistique de Vapnik[2] [57].

1.4 L'étude des algorithmes et de la complexité [51, 32, 13, 45]

Inspirée de la théorie de l'information, une notion fondamentale de complexité des objets, à partir des algorithmes pouvant les générer, est née, dans les années 1960, des tentatives de réponses aux deux questions suivantes :

- Quelle est la nature de l'inférence inductive [51] ?
- Quelle est la nature de l'aléatoire [32, 13] ?

[2] Pour en savoir plus sur l'apprentissage PAC et la VC-dimension, on pourra consulter [3, 26, 30, 2]

Ces deux questions sont étroitement liées dans la mesure où l'inférence inductive devient quasiment impossible si la règle (inconnue) qui préside à la génération d'une base d'exemples est totalement aléatoire. La notion fondamentale issue de ces axes de recherche est la suivante :

Définition 4. La **complexité de Kolmogorov** d'un objet représentable par une suite finie est la taille du plus petit programme qui produise cette suite (la dépendance du langage de programmation ou de l'ordinateur ne fait varier cette mesure, dans les cas usuels, qu'à une constante additive près).

Cette complexité peut être interprétée comme la quantité d'information contenue dans l'objet[3]. Clairement, l'idée sous-jacente est que, si la description d'une suite ne peut pas être comprimée au moyen d'un ordinateur, alors cette suite possède toutes les propriétés d'une série aléatoire, d'où l'on peut imaginer que l'objet de cette suite sera très difficilement apprenable...

Sur la base de ces idées, à la fin des années 1970, Rissanen a proposé la méthode d'inférence inductive MDL, pour "Minimum Description Length", pour résoudre des problèmes d'apprentissage. Ce principe de *longueur de description minimum* peut être vu comme une sorte de principe de maximum de vraisemblance *global*, dans la mesure où il permet de comparer n'importe quelles classes de modèles, qu'elles aient ou non le même nombre de paramètres [46].

1.5 Les orientations des années 1990

Pendant les dix années récentes allant de 1986 à 1995, malgré une explosion du nombre des modèles connexionnistes et de leurs variantes, ainsi que de très nombreuses applications du monde réel résolues par ces méthodes, le plus souvent avec succès, nous pouvons considérer avec Vapnik [58] que, sur le strict plan de la théorie de l'apprentissage, il n'y a pas eu de progrès notable. C'est seulement depuis 1995 que l'on constate un regain d'intérêt pour les approches théoriques, ainsi que des ébauches de résultats nouveaux, qui permettraient de mieux comprendre les processus d'apprentissage, et peut-être, et *surtout*, d'appréhender les capacités de généralisation des modèles. C'est dans cette optique que se situent la théorie de l'apprentissage de Vapnik et les nouveaux systèmes d'apprentissage qu'il propose, les SVM, "Support Vector Machines". Dans la suite de cet article, nous allons présenter quelques aspects techniques de cette théorie et définir ces nouveaux systèmes d'apprentissage.

2 Le problème de l'apprentissage

Commençons par préciser le formalisme dans lequel nous allons nous placer. Il s'agit d'un apprentissage supervisé, sur une base d'exemples S, de taille finie. On considère que des vecteurs x sont générés dans $\mathbb{R}^d$, indépendamment les uns des autres, selon une distribution de probabilité fixée mais inconnue $P(x)$. Un *professeur* attribue à chaque vecteur d'entrée x une valeur y, parfois appelée *sortie désirée*, selon une loi $P(y|x)$ également fixée mais inconnue. Une base d'exemples ou *échantillon* S, de taille m, est alors un ensemble de m observations

$$S = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$$

indépendantes et identiquement distribuées (i.i.d.) selon la loi jointe $P(x, y) = P(x)P(y|x)$.

[3] Pour de plus amples informations sur la complexité de Kolmogorov, on pourra consulter les livres [35, 19]

D'autre part, on dispose d'une *machine à apprendre*, ou d'un *système*[4] *d'apprentissage*, capable de réaliser une famille de fonctions $\{f(x,\alpha) \;/\; \alpha \in \Lambda\}$, où Λ est un ensemble de paramètres abstraits.

Le problème de l'apprentissage consiste en la recherche d'une fonction qui, parmi les $f(x,\alpha)$, réalise la meilleure approximation y^* de la réponse y générée par le *professeur*, meilleure au sens de la minimisation d'une certaine *mesure de désaccord* $D(y,y^*)$. Autrement dit, le but est de minimiser la **fonctionnelle de risque**, c'est-à-dire l'espérance du désaccord, définie par :

$$R(\alpha) = \int D(y, f(x,\alpha))dP(x,y)$$

La mesure de désaccord peut prendre des formes différentes, selon le type de problème auquel on est confronté. Précisons l'expression usuelle de $D(y,y^*)$ dans les trois cas les plus courants :

Classification, reconnaissance des formes
La réponse du *professeur* est binaire, et la mesure de désaccord compte le nombre d'erreurs. Il faut trouver, parmi les fonctions potentielles, une fonction qui minimise la probabilité d'erreur.

$$D(y,y^*) = \begin{cases} 0 \text{ if } y = y^* \\ 1 \text{ if } y \neq y^* \end{cases}$$

Régression, approximation de fonctions
La réponse est réelle, les fonctions sont dans L^2, et l'on calcule l'erreur quadratique. Parmi les fonctions potentielles, on cherche celle qui est la plus proche, au sens de la métrique de $\mathbb{R}^2$, de la fonction à approcher.

$$D(y,y^*) = (y - y^*)^2$$

Estimation d'une densité de probabilité
Les fonctions sont des densités de probabilité $p(x,\alpha)$ et l'on cherche à estimer la loi $P(x)$ du générateur d'exemples (le *professeur* attribue les sorties $y = x$). On peut montrer alors que, en choisissant d'appliquer le principe de maximum de vraisemblance, minimiser le risque devient équivalent à rechercher, parmi les densités potentielles, celle qui est la plus proche de la densité désirée, au sens de l'information de Kullback-Leibler.

$$D(y,y^*) = -\ \ln \quad p(x,\alpha)$$

Revenons à un formalisme uniforme, en introduisant la notion de *fonction de coût*. Posons $z_i = (x_i, y_i)$, pour tout exemple i tel que $1 \leq i \leq m$ et supposons que $z \in \mathbb{R}^n$. On considère maintenant un ensemble de fonctions de coût mesurables $\{Q(z,\alpha) \;/\; \alpha \in \Lambda\}$ et l'on cherche à minimiser le **risque**, dont la fonctionnelle s'exprime alors sous la forme :

$$R(\alpha) = \int Q(z,\alpha)dP(z) \tag{1}$$

où la mesure de probabilité $P(z)$ est inconnue. Seule une base finie $S = \{z_i\}_{1\leq i\leq m}$ d'exemples est donnée, selon un tirage i.i.d. respectant $P(z)$. L'objectif est de trouver une fonction $Q(z,\alpha_0)$ qui minimise le risque, mais la seule méthode applicable consiste à trouver une fonction $Q(z,\alpha_m)$ qui minimise le **risque empirique**, calculable sur les données $\{z_i\}_{1\leq i\leq m}$, et défini par :

$$R_{emp}(\alpha) = \frac{1}{m} \sum_{i=1}^{m} Q(z_i,\alpha) \tag{2}$$

---

[4] On appellera *système* d'apprentissage une méthode permettant de générer un *modèle* d'apprentissage, le terme "modèle" étant réservé à l'ensemble des fonctions produites par le "système".

Dans les trois cas de figure classiques, **le principe de minimisation du risque empirique** se traduit par des méthodes connues, et qui sont usuellement mises en œuvre, à savoir :

Classification, reconnaissance des formes : on minimise le nombre d'erreurs en apprentissage.

Régression, approximation de fonctions : on applique la méthode des moindres carrés.

Estimation d'une densité de probabilité : c'est le principe du maximum de vraisemblance.

On va s'intéresser à la validité, comme méthode d'apprentissage, du principe d'induction consistant à minimiser (2) le risque empirique, et à son lien avec la minimisation (1) du risque $R(\alpha)$.

3 La théorie de l'apprentissage de Vapnik

Dans sa conception de la théorie de l'apprentissage [58, 59], Vapnik distingue quatre étapes fondamentales, que l'on peut résumer comme suit :

(a) **Étudier la théorie de la consistance des processus d'apprentissage**
Trouver des conditions nécessaires et suffisantes pour qu'un processus d'apprentissage basé sur le principe d'induction de la minimisation du risque empirique soit *consistant* (à un sens qui va être précisé ci-dessous).

(b) **Établir des bornes sur la vitesse de convergence des processus d'apprentissage**
Se demander quel est le risque $R(\alpha_m)$ de la fonction qui minimise le risque empirique $R_{emp}(\alpha_m)$ sur un échantillon de m exemples, et chercher à quelle distance celui-ci se situe du plus petit risque possible $\inf_{\alpha\in\Lambda} R(\alpha)$.

(c) **Contrôler la capacité en généralisation des processus d'apprentissage**
Trouver un moyen de contrôler la capacité en généralisation d'un modèle d'apprentissage et déterminer le bon principe d'induction à utiliser, pour une taille donnée m de la base d'exemples.

(d) **Mettre au point une théorie pour construire des algorithmes d'apprentissage**
En partant des trois considérations ci-dessus, chercher à construire des algorithmes qui contrôlent la capacité en généralisation d'un modèle d'apprentissage.

3.1 Consistance du principe de minimisation du risque empirique

Précisons tout d'abord ce que l'on entend par *consistant*, lorsque l'on parle de l'application du principe de minimisation du risque empirique :

Définition 5. Le principe est **consistant**, pour l'ensemble de fonctions $\{Q(z,\alpha) \ / \ \alpha \in \Lambda\}$ et pour la distribution de probabilité $P(z)$, si la méthode de minimisation du risque empirique, appliquée à chaque taille m d'échantillon, produit une suite de fonctions $\{Q(z,\alpha_m) \ / \ m \in \mathbb{N}\}$ telle que :

$$R(\alpha_m) \underset{m\to\infty}{\overset{P}{\longrightarrow}} \inf_{\alpha\in\Lambda} R(\alpha)$$

$$R_{emp}(\alpha_m) \underset{m\to\infty}{\overset{P}{\longrightarrow}} \inf_{\alpha\in\Lambda} R(\alpha)$$

i.e. pour laquelle la suite des espérances du risque et celle des risques empiriques convergent en probabilité vers la même limite, celle-ci étant la plus petite valeur possible du risque $R(\alpha)$.

Le résultat-clé de la théorie de l'apprentissage de Vapnik tient alors dans le théorème suivant :

Théorème 6. *Soit* $\{Q(z,\alpha),\ \alpha \in \Lambda\}$ *un ensemble de fonctions qui vérifient la condition :*

$$A \ \leq \ \int Q(z,\alpha)\ dF(z) \ \leq \ B$$

Alors, une condition nécessaire et suffisante de consistance non triviale du principe de minimisation du risque empirique est que le risque empirique $R_{emp}(\alpha)$ *converge uniformément vers le vrai risque* $R(\alpha)$ *sur l'ensemble de fonctions* $Q(z,\alpha),\ \alpha \in \Lambda$*, au sens suivant :*

$$\lim_{m \to \infty} P\{\sup_{\alpha \in \Lambda} (R(\alpha) - R_{emp}(\alpha)) > \epsilon\} = 0, \quad \forall \epsilon > 0 \tag{3}$$

Il est important de remarquer que la présence de la borne supérieure dans l'expression (3) implique que la consistance du principe de minimisation du risque empirique est nécessairement déterminée par la *pire* des fonctions de l'ensemble $\{Q(z,\alpha),\ \alpha \in \Lambda\}$. Pour cette raison, on parle parfois d'*analyse au pire cas* lorsque l'on évoque cette théorie. Vapnik affirme que ce point est incontournable.

Ce théorème est un résultat théorique, mais il n'est pas directement applicable, puisque le risque $R(\alpha)$ n'est pas calculable, la distribution de probabilité $P(z)$ étant inconnue. En utilisant plusieurs notions intermédiaires, de la VC-entropie à la *fonction de croissance* (voir plus loin), Vapnik en déduit d'autres conditions suffisantes pour la consistance du principe de minimisation du risque empirique, ainsi qu'une succession de bornes qui permettront d'aboutir à un résultat exploitable.

Le premier résultat consiste à établir une condition suffisante faisant intervenir la VC-entropie, dans le cas des fonctions à valeurs booléennes (classification, reconnaissance des formes). Soit $H^{\Lambda}(m)$ la VC-entropie d'un ensemble de fonctions de coût $\{Q(z,\alpha),\ \alpha \in \Lambda\}$ sur un échantillon de taille m (cf. définition 2, donnée dans la partie 1.1 de l'introduction). La condition :

$$\lim_{m \to \infty} \frac{H^{\Lambda}(m)}{m} = 0$$

est une condition nécessaire et suffisante pour la consistance du principe de minimisation du risque empirique (convergence en probabilité, uniforme, en valeur absolue, du risque empirique vers le vrai risque). Toutefois, la VC-entropie est définie comme l'espérance d'une certaine quantité représentant la diversité de l'ensemble de fonctions sur les échantillons. Tout comme $R(\alpha)$, cette espérance ne peut pas être calculée si l'on ignore la distribution de probabilité $P(z)$ des exemples. Cette notion de VC-entropie peut cependant être étendue à des ensembles de fonctions à valeurs réelles, en faisant intervenir un paramètre ϵ, petit, strictement positif[5]. La condition :

$$\lim_{m \to \infty} \frac{H^{\Lambda}(\epsilon, m)}{m} = 0, \quad \forall \epsilon > 0$$

remplace alors celle exprimée ci-dessus. Dans les deux cas, ces conditions signifient que la diversité de l'espace de fonctions sur les exemples doit croître moins vite que la taille de la base. Toutefois, ces conditions nécessaires et suffisantes ne fournissent aucun renseignement sur la vitesse de convergence de la suite des risques $R(\alpha_m)$ vers le plus petit risque possible. C'est pour pallier cet inconvénient que Vapnik introduit ensuite les notions de *VC-entropie recuite* et de *fonction de croissance*.

[5] Si l'on s'intéresse aux définitions précises de ces quantités, on pourra consulter [58, p. 40-43]

3.2 Bornes sur la vitesse de convergence des processus d'apprentissage

Pour plus de clarté, introduisons la quantité $N^{\Lambda}(z_1, \ldots, z_m)$ qui compte le nombre de vecteurs distincts suffisant à "représenter" (à ϵ près, dans le cas des fonctions à valeurs réelles) l'ensemble de fonctions $\{Q(z, \alpha),\ \alpha \in \Lambda\}$ sur l'échantillon $S = \{z_1, \ldots, z_m\}$. Pour les fonctions à valeurs binaires, on peut donner, à partir de cette quantité, les définitions précises des quantités suivantes :

la VC-entropie (*VC entropy*)	$H^{\Lambda}(m) = \mathcal{E}\ \ln\ N^{\Lambda}(z_1, \ldots, z_m)$
la VC-entropie recuite (*Annealed VC entropy*)	$H^{\Lambda}_{ann}(m) = \ln\ \mathcal{E}\ N^{\Lambda}(z_1, \ldots, z_m)$
la fonction de croissance (*Growth function*)	$G^{\Lambda}(m) = \ln\ \sup_S N^{\Lambda}(z_1, \ldots, z_m)$

ces définitions montrant bien que les trois quantités peuvent être rangées par ordre croissant :

$$H^{\Lambda}(m) \leq H^{\Lambda}_{ann}(m) \leq G^{\Lambda}(m)$$

On dira que la convergence des risques $R(\alpha_m)$ vers le plus petit risque possible, quand m tend vers l'infini, est une **convergence rapide** s'il existe une constante c et un entier m_0 tels que, pour tout $m > m_0$, on ait la borne exponentielle suivante :

$$P\{R(\alpha_m) - \inf_{\alpha \in \Lambda} R(\alpha) > \epsilon\} < e^{-c\epsilon^2 m}$$

Il s'avère qu'une telle borne peut être obtenue dès lors que la condition suivante est vérifiée :

$$\lim_{m \to \infty} \frac{H^{\Lambda}_{ann}(m)}{m} = 0$$

Cette condition sur la VC-entropie recuite constitue donc une condition suffisante à la convergence rapide du principe de minimisation du risque empirique, mais malheureusement la quantité $H^{\Lambda}_{ann}(m)$ n'est toujours pas calculable en pratique. Notons d'autre part que la question de la nécessité de cette condition reste ouverte.

La fonction de croissance, quant à elle, ne contient plus de calcul d'espérance, et cela permet de progresser encore vers une condition de convergence rapide qui est, cette fois-ci, indépendante de la distribution de probabilité :

$$\lim_{m \to \infty} \frac{G^{\Lambda}(m)}{m} = 0$$

Toutefois, le calcul de la fonction de croissance suppose implicitement la connaissance de tous les échantillons possibles, ce qui n'est réellement envisageable que si l'espace des exemples est un ensemble fini, de petite taille, ce qui est rarement le cas en pratique.

Vapnik et Chervonenkis ont étendu les définitions des trois quantités ci-dessus à des fonctions à valeurs réelles. A partir de ces quantités, ils ont démontré un assez grand nombre de bornes supérieures, ainsi que quelques bornes inférieures, concernant d'une part la différence entre le risque $R(\alpha)$ et le risque empirique $R_{emp}(\alpha)$ et d'autre part la différence entre les risques $R(\alpha_m)$ obtenus sur des échantillons de taille m et le plus petit risque possible $\inf_{\alpha \in \Lambda} R(\alpha)$. Ces bornes sont valables dans les conditions de convergence exprimées ci-dessus, c'est-à-dire, concrètement, lorsque certaines caractéristiques de la diversité des fonctions de coût croissent moins vite que la taille de la base d'exemples. Les expressions de ces bornes présentent des variantes selon que l'on considère (a) des fonctions bornées, (b) des fonctions positives et bornées, ou (c) des fonctions positives (non nécessairement bornées)[6].

De cette théorie des bornes, nous ne retiendrons que le résultat principal, celui qui s'appuie sur la VC-dimension du modèle d'apprentissage, et que nous pouvons exprimer ainsi :

[6] Toutes ces bornes sont présentées en détail dans [58, p. 65-74]

Théorème 7. *Pour que le principe de minimisation du risque empirique soit consistant, indépendamment de la distribution de probabilité des exemples, il suffit que l'ensemble des fonctions que le système d'apprentissage est capable d'implémenter possède une* ***VC-dimension finie****.*

Avant de revenir plus en détails sur la notion de VC-dimension, nous allons préciser ces résultats dans le cas des fonctions de coût bornées : $a \leq Q(z,\alpha) \leq b$, où a et b sont des constantes réelles. On peut alors affirmer que la relation suivante est vérifiée, avec une probabilité au moins égale à $1-\eta$, pour toutes les fonctions $Q(z,\alpha)$ (en particulier pour celle qui minimise le risque empirique) :

$$|R(\alpha) - R_{emp}(\alpha)| \;\leq\; (b-a)\sqrt{\frac{h(\ln(2m/h)+1)-\ln(\eta/4)}{m}} \tag{4}$$

où h est la VC-dimension de l'ensemble des fonctions de coût. On peut également affirmer que, avec probabilité au moins $1-2\eta$, on a, pour la fonction $Q(z,\alpha_m)$ qui minimise le risque empirique :

$$R(\alpha_m) - \inf_{\alpha\in\Lambda} R(\alpha) \;\leq\; (b-a)\sqrt{\frac{-\ln(\eta)}{2m}} + (b-a)\sqrt{\frac{h(\ln(2m/h)+1)-\ln(\eta/4)}{m}} \tag{5}$$

3.3 Quelques précisions sur la notion de VC-dimension

Les bornes citées ci-dessus se déduisent directement de celles qui ont pu être établies pour la fonction de croissance car la VC-dimension est une notion directement liée à cette fonction [61]. En effet, la fonction de croissance vérifie l'égalité $G^{\Lambda}(m) = m\ln 2$ jusqu'à une certaine valeur h de la variable m à partir de laquelle la courbe de $G^{\Lambda}(m)$ présente une rupture de pente et, pour tout $m > h$, la fonction de croissance est bornée[7] par une fonction logarithmique de m :

$$G^{\Lambda}(m) \leq h\left(\ln\frac{m}{h}+1\right)$$

Cette valeur h est justement la VC-dimension de l'ensemble de fonctions $\{Q(z,\alpha),\ \alpha\in\Lambda\}$. Cette définition correspond exactement à la définition 1 dans le cas des fonctions à valeurs binaires. La définition peut être étendue au cas des fonctions à valeurs réelles [57], en considérant la VC-dimension de la famille des fonctions indicatrices, à valeurs binaires, définie par :

$$I(z,\alpha,\beta) = \mathcal{H}\left(Q(z,\alpha)-\beta\right), \quad \alpha\in\Lambda, \quad \beta\in(A,B)$$

où $\mathcal{H}$ est la fonction de Heaviside[8] et (A,B) désigne l'intervalle (éventuellement non borné) dans lequel varient les valeurs des $Q(z,\alpha)$.

Le fait d'exprimer des inégalités faisant intervenir la VC-dimension du modèle peut être considéré comme l'étape ultime de la théorie des bornes, dans la mesure où ces bornes-là, en plus de garantir une convergence rapide et indépendante de toute distribution de probabilité, présentent l'intérêt d'être des bornes constructives. En effet, on sait calculer la VC-dimension d'un certain nombre de modèles d'apprentissage. Par exemple, la VC-dimension d'un Perceptron à d entrées est égale à $d+1$. Cette valeur est également la VC-dimension de la famille des fonctions réelles linéaires[9].

---

[7] Cette borne est mieux connue sous sa forme exponentielle, sous le nom de Lemme de Sauer.

[8] La fonction de Heaviside vaut 0 lorsque son argument est négatif et 1 s'il est positif ou nul.

[9] Un Perceptron est un modèle de réseau de neurones composé d'unités à seuil, sans couche cachée. Le Perceptron ne peut réaliser qu'une séparation *linéaire* de son espace d'entrée.

Il est important de remarquer que la VC-dimension d'un modèle est une notion très différente de celle de *nombre de paramètres* du modèle. Dans le cas du Perceptron, en effet, les deux valeurs sont identiques (d poids + 1 biais), mais ceci est loin d'être vrai dans le cas général. Des situations extrêmes peuvent même se produire, comme dans les deux exemples suivants.

Exemple 1. Si l'on considère l'ensemble des *rayons*, c'est-à-dire l'ensemble des fonctions caractéristiques des intervalles $[a, +\infty[$ pour tout réel a, il est clair que l'on ne peut pas réaliser toutes les dichotomies possibles sur un ensemble de deux points avec ces fonctions-là (en effet, le point de droite ne peut pas valoir 0 si celui de gauche vaut 1). D'autre part, on peut associer indifférémment 0 ou 1 à un seul point p, selon que l'on choisit $a > p$ ou $a < p$. Donc la VC-dimension de l'ensemble des rayons vaut 1. Si l'on considère, plus généralement, l'ensemble de toutes les fonctions monotones sur $\mathbb{R}$: cet ensemble a clairement une infinité de paramètres, cependant les fonctions indicatrices associées $I(z, \alpha, \beta)$ sont des rayons, pour n'importe quelles valeurs de α et de β. La VC-dimension de l'ensemble des fonctions monotones est donc égale à 1.

Exemple 2. Considérons maintenant les fonctions $f(x, \alpha) = 2sin(\alpha x) + 0.5$ pour tout réel α. Cette famille de fonctions dépend d'un seul paramètre, cependant il est facile de montrer que sa VC-dimension est infinie. En effet, quel que soit l'entier m, on peut trouver un ensemble fini de points $\{x_1, \ldots, x_m\}$ tel que, quelle que soit la répartition des valeurs 0 et 1 sur ces points, on puisse trouver une valeur de α qui fasse osciller suffisamment la fonction f pour que la fonction indicatrice $I(\alpha, 0)$ réalise la dichotomie voulue.

année	auteurs	type de réseau	VC-dimension
1965	Cover [17]	Perceptron, fxs linéaires	$O(W)$ et $\Omega(W)$
1968	Cover [18]	multicouche, fxs à seuil	$O(WlogW)$
1989	Baum et Haussler [7]	fxs à seuil (1 couche cachée)	$\Omega(W)$
1993	Maass [36]	multicouche, fxs à seuil (1 couche cachée)	$\Omega(WlogW)$
1993	Goldberg et Jerrum [23]	multicouche, polynômes par morceaux	$O(W^2)$
1995	Karpinski et Macintyre [29]	multicouche, fxs sigmoïdes	$O(W^4)$
1995	Koiran et Sontag [31]	multicouche, fxs sigmoïdes	$\Omega(W^2)$
1996	Bartlett et Williamson [6]	fxs sigmoïdes ou RBF (1 couche cachée)	
		avec entrées dans $\{-D, \ldots, +D\}^n$	$O(Wlog(WD))$
1997	Bartlett, Maiorov, Meir [5]	n couches, polynômes par morceaux	$O(WlogW)$

Tableau 1. Bornes (O supérieures et Ω inférieures) pour les VC-dimensions des réseaux multicouches

Pour conclure sur la VC-dimension, il faut noter que, pour certains modèles d'apprentissage, on ne sait pas calculer la valeur exacte de la VC-dimension, mais on sait parfois la borner en fonction du nombre de paramètres du modèle. Par exemple, pour les réseaux de neurones multicouches, définis par W poids, on connaît toute une série de résultats que nous résumons dans le tableau 1.

3.4 Contrôle de la capacité en généralisation des processus d'apprentissage

De la relation (4) donnée plus haut comme exemple de borne, on peut déduire, pour tout m, l'inégalité suivante, qui sera vérifiée avec une probabilité au moins égale à $1 - \eta$:

$$(6) \qquad R(\alpha_m) \leq R_{emp}(\alpha_m) + (b - a)\sqrt{\frac{h(\ln(2m/h) + 1) - \ln(\eta/4)}{m}}$$

Le second membre de l'inégalité (6) est composé de deux termes : le risque empirique et une quantité, dépendant de m/h, que l'on appellera **intervalle de confiance**, puisqu'elle représente la différence entre le risque empirique $R_{emp}(\alpha_m)$ et le risque $R(\alpha_m)$, celui de la fonction $Q(z, \alpha_m)$ obtenue par la minimisation du risque empirique sur l'échantillon S de taille m. De même, pour toutes les autres bornes évoquées dans le paragraphe 3.2, l'intervalle de confiance dépend de la quantité m/h, où h est la VC-dimension du modèle. Comme on peut le constater en observant l'inégalité (6), si le rapport m/h est suffisamment grand, le terme prédominant du membre de droite de l'inégalité est le risque empirique. Ainsi, la minimisation du risque empirique suffit à contrôler la capacité de généralisation du modèle : une faible valeur du risque empirique suffira à garantir une faible valeur du risque réel. En revanche, lorsque la taille m de l'échantillon est petite (cas le plus courant dans les applications du monde réel), le quotient m/h est petit (Vapnik considère comme "petite" la taille de l'échantillon dès lors que $m/h < 20$) et l'intervalle de confiance peut prendre une valeur importante. Il s'ensuit que la simple minimisation du risque empirique ne garantit plus une faible valeur du risque effectif.

La seule façon de garantir le risque consiste à contrôler la VC-dimension h du modèle[10], puisque la taille maximum de l'échantillon est généralement fixée par les conditions de l'expérience ou du problème à traiter. Vapnik propose donc d'appliquer un nouveau principe inductif qu'il nomme **principe de minimisation du risque structurel**. Ce principe est basé sur la minimisation conjointe des deux causes d'erreur : le risque empirique et l'intervalle de confiance. La méthode qu'il propose consiste à définir une *structure* sur l'ensemble H des fonctions $\{Q(z, \alpha) \ / \ \alpha \in \Lambda\}$.

Définition 8. Une **structure** sur H est une suite de sous-ensembles $H_i = \{Q(z, \alpha) \ / \ \alpha \in \Lambda_i\}$ qui sont emboîtés :

$$H_1 \subset H_2 \subset \ldots \subset H_i \subset \ldots$$

et dont les VC-dimensions h_i sont finies (même si celle de H est infinie). Ces VC-dimensions forment alors une suite croissante :

$$h_1 \leq h_2 \leq \ldots \leq h_i \leq \ldots$$

La structure est *admissible* si chaque sous-ensemble H_i contient, par exemple, un ensemble de fonctions bornées (cf. [58, p. 90-91]).

En pratique, le risque empirique décroît lorsque la VC-dimension du modèle augmente, tandis que l'intervalle de confiance est une fonction croissante de h. La borne sur le risque, qui est la somme de ces deux quantités, admet un minimum pour une valeur optimale de h, parmi les h_i. Ainsi, pour une base d'exemples donnée $\{z_1, \ldots, z_m\}$, le *principe de minimisation du risque structurel* consiste à sélectionner une fonction $Q(z, \alpha_m^k)$ qui minimise le risque empirique sur le sous-ensemble H_k pour lequel le risque *garanti* (i.e. la somme du risque empirique et de l'intervalle de confiance) est minimal. Cela revient à chercher un compromis entre la qualité de l'approximation sur l'échantillon et la complexité de la fonction qui réalise l'approximation. Notons au passage que cette démarche s'apparente à celle mise en œuvre par les techniques de régularisation.

Donnons quelques exemples de structures, parmi les modèles de réseaux de neurones.

Exemple 3. Une structure déterminée par l'architecture du réseau :
Considérons des réseaux en couches, totalement interconnectés d'une couche à la suivante. On obtient une structure si l'on fait croître régulièrement le nombre de cellules sur l'une des couches (disons par exemple sur la première couche cachée).

[10] On cherche à donner une faible valeur à la borne afin d'avoir une bonne chance de *garantir* un faible risque ; cependant, certains modèles ayant une VC-dimension élevée ou infinie, peuvent néanmoins présenter un comportement effectif meilleur que ce que la borne laisse imaginer et fournir de bonnes performances en généralisation.

Exemple 4. Une structure déterminée par la règle d'apprentissage :
Considérons un ensemble de réseaux ayant une architecture fixée et dont les paramètres sont les poids $W = \{w\}$. On obtient une structure si l'on contraint les poids à être bornés, en norme $\|w\| \leq C_i$, par une suite strictement croissante de réels $C_1 < C_2 < \ldots < C_i < \ldots$. On peut alors montrer que la minimisation du risque empirique sur H_k revient à minimiser :

$$E(w, \gamma_k) = \frac{1}{m} \sum_{i=1}^{m} L(y_i, f(x_i, w)) + \gamma_k \|w\|^2$$

où les γ_k sont une suite strictement décroissante de multiplicateurs de Lagrange appropriés. On retrouve ainsi la méthode classique du "weight decay", ce qui confirme le lien avec les techniques de régularisation.

Exemple 5. Une structure déterminée par le prétraitement :
Considérons des réseaux à architecture fixée. On peut faire subir aux entrées du réseau un prétraitement : $u = K(x, \beta)$ par une fonction noyau (par exemple) dont le paramètre β contrôle le rayon d'action. Si l'on contraint ce paramètre $\beta \geq C_i$ par une suite de réels strictement décroissante $C_1 > C_2 > \ldots > C_i > \ldots$, on obtient également une structure.

La dernière étape de la théorie de l'apprentissage de Vapnik consiste à construire des algorithmes, ou encore des systèmes d'apprentissage, qui contrôlent la capacité en généralisation des modèles qu'ils produisent. C'est en s'appuyant sur le principe de minimisation du risque structurel que Vapnik propose de construire les *"Support Vector Machines"*, des systèmes d'apprentissage dont la description fait l'objet de la dernière partie de cet article.

4 Les "Support Vector Machines"

Afin de minimiser conjointement les deux termes de la borne (6), on peut, soit fixer l'intervalle de confiance[11] (par exemple par le choix d'une architecture de réseau de neurones) et minimiser le risque empirique (par une règle d'apprentissage), soit fixer la valeur du risque empirique (idéalement, à la valeur 0) et minimiser l'intervalle de confiance. Selon Vapnik, les réseaux de neurones appliquent la première méthode, tandis que les systèmes d'apprentissage qu'il propose sont basés sur la seconde. On se doit de modérer ces propos qui, s'ils s'appliquent en effet à l'apprentissage par rétropropagation[12] dans les réseaux multicouches, se trouvent en revanche mis en défaut dans le cas des méthodes d'apprentissage incrémentales [40, 37, 4, 54] ou des techniques d'élagage [34] puisque celles-ci modifient automatiquement le nombre de poids du réseau, donc influent sur la VC-dimension (cf. Table 1).

Les systèmes d'apprentissage appelés **"Support Vector Machines"**, ou **SVM** en abrégé, sont des algorithmes basés sur les trois principes mathématiques suivants :

Le principe de Fermat (1638) :
Les points qui minimisent ou maximisent une fonction dérivable annulent sa dérivée. Ils sont appelés *points stationnaires.*

[11] L'expression du second terme de la borne montre qu'il suffit de fixer à la fois la taille de la base d'exemples et la VC-dimension du modèle, ce qui est courant, en pratique.

[12] Notons au passage que la rétropropagation ne peut garantir, comme toute méthode basée sur la descente en gradient, qu'un minimum *local* du risque empirique.

Le principe de Lagrange (1788) :
Pour résoudre un problème d'*optimisation sous contraintes*, il suffit de rechercher un point stationnaire x_0 du *lagrangien* $\mathcal{L}$ de la fonction f à optimiser, les f_i exprimant les contraintes

$$\mathcal{L}(x, \lambda) = f(x) + \sum_{i=1}^{k} \lambda_i f_i(x)$$

où les λ_i sont des constantes appelées *coefficients de Lagrange.*

Le principe de Kuhn-Tucker (1951) :
Les relations de Kuhn-Tucker peuvent s'appliquer au cas qui nous intéresse. Avec des fonctions f et f_i convexes, il est même toujours possible de trouver un *point-selle* (x_0, λ^*) qui vérifie

$$\min_x \mathcal{L}(x, \lambda^*) = \mathcal{L}(x_0, \lambda^*) = \max_{\lambda \geq 0} \mathcal{L}(x_0, \lambda)$$

Le lecteur intéressé par de plus amples précisions pourra consulter l'un des ouvrages [15, 21, 38]. Nous allons maintenant montrer comment ces principes peuvent être appliqués à la recherche d'un *hyperplan séparateur optimal,* dans le cadre de la classification. Cette notion est claire dans le cas où les données sont linéairement séparables et elle permet alors de définir la notion de *"support vector"*. On verra ensuite comment elle peut s'étendre au cas non linéairement séparable, puis comment on peut définir des "Support Vector Machines" non linéaires. L'application des SVM aux problèmes de régression sera abordé plus loin.

4.1 Cas d'une classification séparable : hyperplan optimal et "support vectors"

Une base d'exemples $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, linéairement séparable, étant donnée, un hyperplan séparateur est l'un des hyperplans qui vérifient les conditions :

(7) $$(w.x_i) + b \geq +1, \quad \text{si} \quad y_i = +1$$

(8) $$(w.x_i) + b \leq -1, \quad \text{si} \quad y_i = -1$$

ce qui peut se résumer par une description unifiée de l'ensemble des contraintes :

(9) $$y_i\left((w.x_i) + b\right) \geq 1, \quad \forall i \in \{1, \ldots, m\}$$

Définition 9. On appelle **hyperplan optimal** l'hyperplan séparateur qui est situé à la distance maximale des vecteurs x les plus proches parmi l'ensemble des exemples. On peut dire aussi que cet hyperplan **maximise la marge**.

La valeur de la marge étant $2/\|w\|_2$ (norme L^2), maximiser la marge revient à minimiser le carré de la norme $\|w\|_2^2 = (w.w)$ du vecteur w (classiquement : les poids d'un Perceptron), sous les contraintes (9). En appliquant le principe de Kuhn-Tucker, on est amené à rechercher un point-selle w^0, b^0, λ^0 du lagrangien

$$\mathcal{L}(w, b, \lambda) = \frac{1}{2}(w.w) - \sum_{i=1}^{m} \lambda_i \left[y_i\left((w.x_i) + b\right) - 1\right]$$

Ce point-selle vérifie les conditions $\lambda_i^0 \left[y_i\left((w^0.x_i) + b^0\right) - 1\right] = 0$, pour tout $i \in \{1, \ldots, m\}$.

Définition 10. Les **"support vectors"** (terme que l'on pourrait traduire par *points de support*) sont les vecteurs x_i pour lesquels l'égalité $y_i\left((w^0.x_i) + b^0\right) = 1$ est vérifiée. Concrètement, ce sont les points les plus proches de l'hyperplan optimal. Pour tous les autres exemples, on a donc $\lambda_i^0 = 0$.

Les conditions d'annulation des dérivées partielles du lagrangien permettent d'écrire les relations que vérifie l'hyperplan optimal, avec les λ_i^0 non nuls seulement pour les "support vectors" :

$$w^0 = \sum_{i=1}^{m} \lambda_i^0 y_i x_i \qquad \text{et} \qquad \sum_{i=1}^{m} \lambda_i^0 y_i = 0$$

Ces contraintes d'égalité permettent d'exprimer la forme duale du lagrangien

$$W(\lambda) = \sum_{i=1}^{m} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{m} \lambda_i \lambda_j y_i y_j (x_i.x_j) \tag{10}$$

Pour trouver le point-selle, il suffit alors de maximiser (et non plus de minimiser) $W(\lambda)$ dans le quadrant positif $\lambda_i \geq 0$, pour tout $i \in \{1, \dots, m\}$. La résolution de ce problème d'optimisation quadratique permettra de connaître une équation de l'hyperplan optimal :

$$\sum_{i=1}^{m} \lambda_i^0 y_i (x.x_i) + b^0 = 0 \qquad \text{avec} \quad b_0 = -\frac{1}{2} \left[(w_0.sv_{class+1}) + (w_0.sv_{class-1}) \right]$$

Pour une nouvelle forme x, non apprise, présentée à la machine, il suffira de regarder le signe de l'expression du membre de gauche pour savoir dans quel demi-espace cette forme se trouve, et donc quelle classe la machine lui attribue.

4.2 Cas d'une classification non séparable. Méthodes pratiques

Il est possible de reformuler les conditions (7) et (8), lorsque les exemples ne sont pas linéairement séparables, sous la forme :

$$(w.x_i) + b \geq +1 - \xi_i, \quad \text{si} \quad y_i = +1 \tag{11}$$

$$(w.x_i) + b \leq -1 + \xi_i, \quad \text{si} \quad y_i = -1 \tag{12}$$

$$\xi_i \geq 0 \quad \forall i \in \{1, \dots, m\} \tag{13}$$

La machine attribuera une sortie fausse à un vecteur x_i si le ξ_i correspondant est supérieur à 1. La somme de tous les ξ_i représente donc une borne du nombre d'erreurs. Dans le cas non séparable, au lieu de chercher le vecteur de poids w_0 qui minimisait le carré de la norme $(w.w)$, on cherche maintenant à minimiser, sous les contraintes exprimées juste ci-dessus, la quantité :

$$\phi(w, \xi) = \frac{1}{2}(w.w) + C \sum_{i=1}^{m} \xi_i$$

La solution λ_i^0 s'obtient une fois encore en maximisant le lagrangien dual admettant la même expression que dans l'équation (10), mais sous les contraintes, un peu différentes, $0 \leq \lambda_i \leq C$, pour tout $i \in \{1, \dots, m\}$, où C est un paramètre qui peut être choisi par l'utilisateur[13]. On obtient, de la même manière que précédemment, une équation d'un hyperplan qui est optimal au sens des nouvelles contraintes tenant compte des ξ_i. Cet hyperplan reçoit parfois le nom d'**hyperplan optimal généralisé** [16]. Les "support vectors" sont toujours les vecteurs-exemples les plus proches de cet hyperplan, mais, cette fois-ci, il existe des vecteurs qui sont situés dans le *mauvais* demi-espace (celui dont la classe n'est pas celle désirée pour l'exemple).

[13] Plus C est choisi grand, plus cela revient à attribuer une forte pénalité aux erreurs.

Pour réaliser les algorithmes définis ci-dessus, on doit résoudre des problèmes d'optimisation quadratique et l'on est souvent amené, en pratique, à résoudre certaines équations par des méthodes numériques. Plusieurs techniques sont proposées dans [12], article qui aborde plusieurs aspects pratiques intéressants des SVM pour la reconnaissance des formes :

- **Le gradient conjugué avec contraintes** : c'est un gradient conjugué classique, dont les directions sont projetées dans les sous-espaces définis par les contraintes $\sum_{i=1}^{m} \lambda_i^0 y_i = 0$.
- **Des méthodes de projection**, également basées sur le gradient conjugué, qui sont décrites par exemple dans [39].
- **La décomposition de Bunch-Kaufman**, qui utilise le Hessien, tout en s'appuyant sur le fait que la plupart des λ_i sont nuls [10, 11].
- **Les méthodes de points intérieurs**, par exemple l'algorithme de Vanderbei [56], méthode qui semble particulièrement intéressante lorsque les "support vectors" sont nombreux par rapport à la taille de la base d'exemples.

4.3 Quelques réflexions sur les hyperplans optimaux

On peut se demander en quoi la recherche des hyperplans optimaux répond au problème de la "bonne" généralisation posé au tout début de cet article. Pour donner des éléments de réponse à cette question, Vapnik définit une normalisation sur les hyperplans séparateurs, à partir de l'ensemble X^* des vecteurs-exemples de $\mathbb{R}^d$ contenus par une sphère de rayon R :

$$X^* = \{x_1, \ldots, x_r\} \qquad \text{et l'on a} \qquad |x_i - a| \leq R \quad \forall x_i \in X^*$$

Définition 11. Les **hyperplans canoniques** (pour les vecteurs de X^*) sont ceux qui vérifient la condition $\min_{x_i \in X^*} |(w.x_i) + b| = 1$

Théorème 12. *La VC-dimension h d'un ensemble d'hyperplans canoniques définis sur X^* et satisfaisant la contrainte $\|w\| \leq A$ admet la borne suivante :*

$$h \leq min(d, \lfloor R^2 A^2 \rfloor) + 1$$

La recherche de l'hyperplan optimal consistant à minimiser la norme de w, cette méthode aurait donc une bonne raison de fournir un modèle dont la VC-dimension est minimale, d'après la borne énoncée dans le théorème 12. Toutefois, le raisonnement ci-dessus n'est pas rigoureux car la borne établie dans le théorème dépend de certaines caractéristiques de la base d'exemples. Une approche plus rigoureuse est développée dans [12], pour les *classifieurs à marge fixée* et mène à un théorème très semblable. La minimisation du risque structurel peut être démontrée rigoureusement pour les classifieurs à marge fixée, mais le raisonnement n'est pas directement transférable aux "Support Vector Machines". Nous reviendrons plus loin sur la VC-dimension des SVM.

Un autre argument évoqué par Vapnik pour justifier le lien entre les bonnes performances en généralisation et la recherche des hyperplans optimaux découle de deux théorèmes fournissant des bornes sur la probabilité d'erreur du modèle.

Théorème 13 [58]. *Soit $sv(m)$ le nombre de "support vectors" de l'hyperplan optimal (éventuellement* généralisé*) séparant les m données de la base d'exemples. L'espérance de la probabilité d'erreur sur un exemple test est bornée par*

$$\mathcal{E}[P(erreur)] \leq \frac{\mathcal{E}[sv(m)]}{m-1}$$

Cela signifie que, si l'hyperplan optimal peut être construit à partir d'un petit nombre de "supports vectors", par rapport à la taille de la base d'exemples, alors la capacité en généralisation du modèle sera grande, indépendamment de la taille de l'espace.

Théorème 14 [12, 60]. *Si les exemples vérifient $\|x_i\| \leq D$, pour tout $1 \leq i \leq m$, et s'ils admettent un hyperplan séparateur optimal, passant par l'origine, de marge ρ, alors l'espérance de la probabilité d'erreur admet la borne*

$$\mathcal{E}[P(erreur)] \leq \frac{\mathcal{E}[\frac{D^2}{\rho^2}]}{m}$$

Ce résultat, tout récent (son énoncé apparaîtra dans le livre [60], probablement sous presse, où Vapnik détaille les preuves des théorèmes formulés dans [58]), n'est pas encore réellement exploité. On peut cependant en déduire que, si l'on sait calculer D, le diamètre de la sphère minimale contenant toutes les données, quel qu'en soit le nombre, alors la minimisation du quotient D^2/ρ^2 pourrait devenir le bon critère pour obtenir une bonne capacité en généralisation.

4.4 Les "Support Vector Machines" non linéaires

Pour permettre leur application aux cas les plus généraux, l'idée fondamentale des SVM est que l'on peut plonger les vecteurs d'entrée dans un espace de représentation interne de dimension suffisamment grande pour que l'hyperplan optimal construit sur cet espace généralise "bien" (soit parce qu'il appartient à une famille de faible VC-dimension, soit parce qu'il est défini par un petit nombre de "support vectors"), indépendamment de la dimension de cet espace. Le fait qu'un problème puisse devenir linéairement séparable lorsqu'on augmente la taille de l'espace des représentations internes peut trouver une justification dans certains travaux théoriques menés sur les réseaux multicouches d'unités à seuil [22, 9]. La possibilité de raisonner sur cette idée de base est explicitée par Vapnik dans le cas des polynômes de degré 2 : soit $x = (x^1, \ldots, x^d)$ un vecteur de l'espace d'entrée $\mathbb{R}^d$, alors un élément de l'espace de représentation interne sera $u = \varphi(x) = (u^1, \ldots, u^{\frac{1}{2}d(d+3)})$ dont les composantes sont les x^j, les $(x^j)^2$ et les $x^j x^k$ (pour $k \neq j$).

La découverte fondamentale [8] qui rend possible l'application pratique de l'idée énoncée ci-dessus est le fait qu'il est possible de construire l'hyperplan optimal dans l'espace des représentations internes sans considérer explicitement cet espace. Il suffit de savoir calculer des produits scalaires entre les "support vectors" et les vecteurs de l'espace de représentation interne. Or, on peut s'appuyer sur le résultat mathématique suivant : dans un espace de Hilbert, un produit scalaire peut s'exprimer sous la forme

$$(u.u^*) = \sum_{i=1}^{\infty} \varphi_i(x)\varphi_i(x^*) = K(x, x^*)$$

où $K(x, x^*)$ est une fonction symétrique vérifiant **les conditions de Mercer** :

$$\int K(x, x^*)\psi_i(x)\psi_i(x^*)dxdx^* \geq 0$$

pour toute fonction ψ telle que $\int \psi^2(x)dx < \infty$.

Pour résumer, il suffit de reprendre toutes les formules énoncées dans les paragraphes précédents, pour la recherche de l'hyperplan optimal (ou optimal généralisé), et de remplacer les produits scalaires des vecteurs d'entrée $(x_i.x_j)$ par ceux des variables de l'espace de représentation interne

$(u_i.u_j)$, mais en fait tout simplement par des fonctions noyaux K des variables de l'espace d'entrée $K(x_i, x_j)$. Par exemple, il suffira de maximiser la forme quadratique suivante (lagrangien dual) :

$$W(\lambda) = \sum_{i=1}^{m} \lambda_i - \frac{1}{2} \sum_{i,j=1}^{m} \lambda_i \lambda_j y_i y_j K(x_i, x_j) \tag{14}$$

sous les contraintes $0 \leq \lambda_i \leq C$ pour tout $1 \leq i \leq m$ et l'on obtiendra la solution λ^0 qui permettra de construire la règle de décision (applicable à de nouveaux motifs, non appris) :

$$\sum_{i=1}^{m} \lambda_i^0 y_i K(x, x_i) + b^0 = 0$$

Donc les SVM utilisent la convolution du produit scalaire pour construire des fonctions de décision nonlinéaires dans l'espace d'entrée, fonctions qui peuvent s'exprimer sous la forme générale :

$$f(x) = Sgn \left(\sum_{i=1}^{N} \lambda_i y_i K(x, w_i) + b \right)$$

où *Sgn* désigne la fonction *signe*. Pour cet ensemble de fonctions, on obtient le meilleur risque garanti si l'on choisit, pour les poids w_i, des exemples de la base, les "support vectors". Cette notion est ainsi étendue aux cas non linéaires. La somme ci-dessus porte alors sur $N = sv(m)$, le nombre de "support vectors", qui est calculé par l'algorithme lui-même. Vapnik donne une représentation d'une SVM sous la forme d'un réseau à une couche cachée dont le nombre de cellules est égal au nombre de "support vectors", et non à la dimension de l'espace des représentations internes, comme on aurait pu le supposer initialement.

Dans la pratique, il n'est pas toujours facile de montrer qu'une fonction vérifie les conditions de Mercer. Donnons quelques exemples classiques de SVM en précisant les fonctions noyaux K utilisées comme convolution :

Noyaux polynomiaux :
Pour construire des règles de décision basées sur des polynômes de degré p, on met en œuvre une SVM utilisant la convolution

$$K(x, x^*) = [(x.x^*) + 1]^p$$

Noyaux RBF :
Pour construire des machines de type RBF, on utilise la convolution

$$K(x, x^*) = e^{-\frac{\|x-x^*\|^2}{2\sigma^2}}$$

Noyaux sigmoïdaux :
Pour construire un réseau de neurones à une couche cachée de fonctions sigmoïdes, on peut utiliser la convolution

$$K(x, x^*) = S\{(x.x^*) + b\} \qquad \text{où} \quad S(u) = \frac{e^{au} - e^{-au}}{e^{au} + e^{-au}}$$

le paramètre a étant choisi de telle façon que les conditions de Mercer soient satisfaites.

4.5 "Support Vector Machines" pour la régression

Les "Support Vector Machines" peuvent également être mises en œuvre pour l'approximation de fonctions [20]. Pour les problèmes de régression, Huber [27] a étudié le choix optimal du critère à minimiser en fonction des connaissances disponibles sur la nature du bruit. Vapnik s'est inspiré de ces travaux pour proposer la fonction suivante [58], nommée *fonction ϵ-insensitive* et notée $|.|_\epsilon$. Pour un $\epsilon > 0$ donné, il s'agit d'une fonction paire, continue, identiquement nulle sur l'intervalle $[0, \epsilon]$, qui croît linéairement sur $[\epsilon, +\infty[$. Dans le cas non linéaire, la forme générale des fonctions calculées par les SVM en régression est donnée par :

$$f(x.w) = \sum_{i=1}^{\infty} w_i \phi_i(x)$$

La fonction de coût est définie comme suit :

$$E(w, \gamma) = \frac{1}{m} \sum_{i=1}^{m} |y_i - f(x_i, w)|_\epsilon + \gamma ||w||^2$$

où γ apparaît comme une constante de régularisation. On démontre que les solutions du problème d'optimisation correspondant peuvent s'écrire sous la forme :

$$f(x, \lambda, \lambda^*) = \sum_{i=1}^{m} (\lambda_i^* - \lambda_i)(\Phi(x_i), \Phi(x)) + b \tag{15}$$

Les valeurs des variables duales λ_i et λ_i^* s'obtiennent comme précédemment, par maximisation sous contraintes d'une fonctionnelle quadratique. En utilisant de nouveau l'expression générale du produit scalaire dans les espaces de Hilbert, le problème de programmation quadratique revient à maximiser :

$$W(\lambda, \lambda^*) = -\epsilon \sum_{i=1}^{m} (\lambda_i^* + \lambda_i) + \sum_{i=1}^{m} y_i(\lambda_i^* - \lambda_i) - \frac{1}{2} \sum_{i,j=1}^{m} (\lambda_i^* - \lambda_i)(\lambda_j^* - \lambda_j) K(x_i, x_j)$$

sous les contraintes

$$\sum_{i=1}^{m} \lambda_i^* = \sum_{i=1}^{m} \lambda_i$$

$$0 \leq \lambda_i^*, \lambda_i \leq C, \quad \forall i \in \{1, \dots, m\}$$

De même que précédemment, les points de la base d'apprentissage associés à un coefficient non nul dans le développement (15) sont nommés "support vectors".

Une caractéristique intéressante de la mise en œuvre des SVM en approximation de fonctions est la possibilité de choisir la fonction noyau K de manière à réaliser une régression fondée sur le développement en série de Fourier ou l'interpolation par fonctions "splines" [63, 59]. Dans les deux cas, la fonction noyau correspondant au cas d'un espace de dimension n quelconque est le produit de n fonctions noyaux en dimension 1.

Noyaux engendrant un développement en série de Fourier :

Un simple calcul montre que ce noyau est défini par la fonction de Dirichlet :

$$K(x_i, x_j) = \frac{\sin(N + \frac{1}{2})(x_i - x_j)}{\sin \frac{x_i - x_j}{2}}$$

Noyaux engendrant des fonctions splines :

Pour des fonctions splines avec un nombre de nœuds infini, on utilisera :

$$K(x_i, x_j) = \sum_{r=0}^{n} x_i^r x_j^r + \int_0^1 (x_i - t)_+^n (x_j - t)_+^n \, dt$$

4.6 La VC-dimension des "Support Vector Machines"

Pour terminer, nous allons nous intéresser aux VC-dimensions des "Support Vector Machines" et nous allons voir que, contrairement à leur motivation première, celles-ci n'ont pas toujours de faibles VC-dimensions. Burges [12] a démontré les deux résultats ci-dessous, pour les SVM à noyaux polynomiaux et les SVM à noyaux RBF.

Théorème 15. *Si l'espace d'entrée des exemples est de dimension d, alors la VC-dimension des SVM ayant pour noyaux les fonctions polynômes homogènes de degré p, est* $\binom{d+p-1}{p} + 1$

Ce résultat permet de mettre en évidence la croissance extrêmement rapide de la dimension de l'espace des représentations internes.

Exemple 6. Pour traiter des images de 16×16 pixels, avec des polynômes de degré 4, la dimension de l'espace d'entrée est 256, mais celle de l'espace des représentations internes vaut déjà 183 181 376.

Théorème 16. *Les SVM à noyaux RBF Gaussiens, pour lesquels, soit on est dans le cas séparable, soit la pénalité attribuée aux erreurs est autorisée à prendre n'importe quelle valeur, ont une VC-dimension infinie.*

Ces résultats peuvent paraître surprenants et, surtout, contradictoires avec le but initial, à savoir la minimisation du risque structurel. Il faut cependant noter que la valeur du risque effectif peut être en pratique très inférieure à celle garantie par la borne de l'équation (6). Les expériences qui ont été réalisées, par différents chercheurs, montrent que les SVM obtiennent de très bonnes performances en généralisation, comparées à la plupart des autres systèmes d'apprentissage. Ceci est vrai en particulier pour les SVM à noyaux gaussiens [50]. La force des SVM semble provenir de la recherche des hyperplans optimaux et des propriétés de ces derniers, plutôt que de leur VC-dimension.

5 Conclusion

Dans cet article, nous avons tenté de retracer le cheminement du raisonnement de Vapnik, depuis les fondements de la théorie de l'apprentissage statistique jusqu'à la définition d'un nouveau concept de système d'apprentissage, les "Support Vector Machines". Nous montrons comment il apporte des réponses successives aux quatre grandes étapes qu'il s'est fixées. Les principes de minimisation du risque empirique, puis du risque structurel, sont justifiés par une approche théorique. Ils sont liés à l'existence de bornes que l'on peut garantir sur le risque d'erreur en généralisation. Ils ont été à l'origine de la notion de "Support Vector Machines", même si, dans leur définition algorithmique, ces machines s'éloignent parfois un peu de leurs motivations théoriques premières. Elles sont basées sur la recherche de l'hyperplan de marge optimale, dans un espace de représentations internes (souvent de très grande dimension) dont on peut éviter l'exploration par l'usage de fonctions noyaux. Le choix de ces fonctions caractérise la SVM utilisée. Dans l'espace d'entrée, on peut séparer les exemples, selon les cas, par des fonctions polynômes, par des fonctions RBF, à l'aide de fonctions sigmoïdes, etc... Les SVM font preuve de bonnes performances sur les applications courantes (ex. reconnaissance de caractères), mais la justification de ces bonnes performances laisse encore place aujourd'hui à de nombreuses questions ouvertes.

Remerciements

Nous tenons à remercier Bruno Durand et Pascal Koiran, pour leurs conseils sur certains points techniques, ainsi qu'André Elisseeff et Olivier Teytaud, pour leurs relectures attentives.

Bibliographie

1. M.A. Aizerman, E.M. Braverman, and L.I. Rozonoer. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.
2. M. Anthony. Probabilistic analysis of leraning in artificial neural networks: The PAC model and its variants. *Neural Computing Surveys*, 1:1–47, 1997. (electronic journal: available at Web site http://www.icsi.berkeley.edu/ jagota/NCS).
3. M. Anthony and N. Biggs. *Computational Leraning Theory*. Cambridge University Press, 1992.
4. A. Azcarraga and A. Giacometti. A prototype-based incremental network model for classification tasks. In *Proc. of Neuro-Nîmes'91*, pages 121–134. EC2, 1991.
5. P.L. Bartlett, V. Maiorov, and R. Meir. Almost linear vc dimension bounds for piecewise polynomial networks. http://wwwsyseng.anu.edu.au/ bartlett, November 1997. (preprint).
6. P.L. Bartlett and R.C. Williamson. The vc dimension and pseudodimension of two-layer neural networks with discrete inputs. *Neural Computation*, 8(3):625–628, 1996.
7. E.B. Baum and D. Haussler. What size net gives valid generalization? *Neural Computation*, 1(1):151–160, 1989.
8. B.E. Boser, I.M. Guyon, and V.N. Vapnik. A training algorithm for optimal margin classifiers. In *Proc. Fifth Annual Workshop on Computational Learning Theory*, pages 144–152. Pittsburgh, ACM, 1992.
9. G. Brightwell, C. Kenyon, and H. Paugam-Moisy. Multilayer neural networks: one or two hidden layers? In M.C. Mozer, M.I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9, Proc. NIPS*96*, pages 148–154. MIT Press, 1997.
10. J.R. Bunch and L. Kaufman. Some stable methods for calculating inertia and solving symmetric linear systems. *Mathematics of Computation*, 31(137):163–179, 1977.
11. J.R. Bunch and L. Kaufman. A computational method for the indefinite quadratic programming problem. *Linear Algebra and its Applications*, 34:341–370, 1980.
12. C.J.C. Burges. A tutorial on support vector machines for pattern recognition. http://svm.research.bell-labs.com/SVMdoc.html, December 1997. (submitted to Data Mining and Knowledge Discovery).
13. G.J. Chaitin. On the length of programs for computing finite binary sequences. *J. Assoc. Comput. Mach.*, 13:547–569, 1966.
14. N.N. Chentsov. Evaluation of an unknown distribution density from observations. *Soviet Math. Docl.*, 4:1559–1562, 1963.
15. P. Ciarlet. *Introduction à l'analyse numérique matricielle et à l'optimisation*. MASSON, 1994. (nouvelle édition).
16. C. Cortes and V.N. Vapnik. Support vector networks. *Machine Learning*, 20:1–25, 1995.
17. T.M. Cover. Geometrical and statistical properties of systems of linear inequalities with application in pattern recognition. *IEEE Trans. Electronic Computers*, 14:326–334, 1965.
18. T.M. Cover. Capacity problems for linear machines. In L. Kanal, editor, *in: Pattern Recognition*, pages 283–289. Thomson Book Co., 1988.
19. J.-P. Delahaye. *Information, complexité et hasard*. Hermès, 1994.
20. H. Drucker, C.J.C. Burges, L. Kaufman, A. Smola, and V.N. Vapnik. Support vector regression machines. In M.C. Mozer, M.I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9, Proc. NIPS*96*, pages 155–161. MIT Press, 1997.
21. R. Fletcher. *Practical Methods of Optimization*. John WILEY, 1987. (2nd edition).
22. G. J. Gibson. Exact classification with two-layer neural nets. *J. Computer and System Science*, 52(2):349–356, 1996.
23. P. Goldberg and M. Jerrum. Bounding the vapnik-chervonenkis dimension of concept classes parameterized by real numbers. *Machine Learning*, 18:131–148, 1995.

24. Y. Grandvalet. *Injection de bruit dans les perceptrons multicouches.* PhD thesis, Université de Technologie de Compiègne, 1995.
25. J. Hadamard. *Le problème de Cauchy et les équations aux dérivées partielles linéaires hyperboliques.* Hermann, Paris, 1932.
26. D. Haussler. Decision theoretic generalization of the pac model for neural net and other applications. *Inform. Comp.*, 100(1):78–150, 1992.
27. P. Huber. Robust estimation of local parameter. *Annals of Mathematical Statistics*, 35(1), 1964.
28. V.V. Ivanov. On linear problems which are not well-posed. *Soviet Math. Docl.*, 3(4):981–983, 1962.
29. M. Karpinski and A. Macintyre. Polynomial bounds for vc dimension of sigmoidal and general Pfaffian neural networks. *J. Computer System Sciences*, 54(1):169–176, 1997.
30. M.J. Kearns and U.V. Vazirani. *An Introduction to Computational Learning Theory.* MIT Press, 1994.
31. P. Koiran and E.D. Sontag. Neural networks with quadratic vc dimension. *J. Computer System Sciences*, 54(1):190–198, 1997.
32. A.N. Kolmogorov. Three approaches of the quantitative definitions of information. *Problem of Inform. Transmission*, 1(1):1–7, 1965.
33. Y. LeCun. Learning processes in an asymmetric threshold network. In *Disordered systems and biological organizations*, pages 233–240. Springer, 1986.
34. Y. LeCun, J.S. Denker, and S.A. Solla. Optimal brain damage. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, pages 598–605. Morgan Kaufmann, 1989.
35. M. Li and P.M.B. Vitanyi. *Introduction to Kolmogorov Complexity and its Applications.* Springer-Verlag, 1993.
36. W. Maass. Neural nets with superlinear vc-dimension. *Neural Computation*, 6(5):877–884, 1994.
37. D. Martinez and D. Esteve. The offset algorithm: Building and learning method for multilayer neural networks. *Europhysics Letters*, 18:95–100, 1992.
38. M. Minoux. *Programmation mathématique : théorie des algorithmes (tome 1).* Dunod, 1983.
39. J.J. Moré and G. Toraldo. On the solution of large quadratic programming problems with bound constraints. *SIAM J. Optimization*, 1(1):93–113, 1991.
40. M. Mézard and J.-P. Nadal. Learning in feedforward layered neural networks: The tiling algorithm. *J. Phys. A*, 22:2191–2203, 1989.
41. A.B.J. Novikoff. On convergence proofs of perceptrons. In *Symposium on the Mathematical Theory of Automata*, volume XII, pages 615–622. Polytechnic Institute of Brooklyn, 1962.
42. E. Parzen. On estimation of probability function and mode. *Annals of Mathematical Statistics*, 33(3), 1962.
43. D.Z. Phillips. A technique for numerical solution of certain integral equation of the first kind. *J. Assoc. Comput. Mach.*, 9:84–96, 1962.
44. T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.
45. J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.
46. J. Rissanen. Stochastic complexity in learning. In P. Vitanyi, editor, *Computational Learning Theory, Proc. of EuroCOLT'95*, volume 904 of *LNAI*, pages 196–210. Springer, 1995.
47. F. Rosenblatt. Remarks on some nonparametric estimation of density function. *Annals of Mathematical Statistics*, 27:642–669, 1956.
48. F. Rosenblatt. *Principles of neurodynamics: Perceptron and theory of brain mechanisms.* Spartan Books, 1962.
49. D.E. Rumelhart, G.E. Hinton, and R.J. Williams. *Parallel Distributed Processing: Explorations in the microstructure of cognition*, volume I, chapter Learning internal representations by error propagation, pages 318–362. MIT Press, 1986.
50. B. Scholkopf, K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V.N. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Trans. Signal Processing*, 45:2758–2765, 1997.
51. R.J. Solomonoff. A formal theory of inductive inference. *Inform. Contr.*, 7:1–22 and 224–254, 1964.
52. A.N. Tikhonov. On solving ill-posed problem and method of regularization. *Doklady Akademii Nauk USSR*, 153:501–504, 1963.

53. A.N. Tikhonov and V.Y. Arsenin. *Solution of ill-posed problems.* W.H. Winston, Washington, DC, 1977.
54. J.-M. Torres-Moreno, P. Peretto, and M.B. Gordon. An evolutive architecture coupled with optimal perceptron learning for classification. In M. Verleysen, editor, *Proc. of ESANN'95*, pages 365–370, 1995.
55. L.G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984.
56. R.J. Vanderbei. Interior point methods: Algorithms and formulations. *ORSA J. Computing*, 6(1):32–34, 1994.
57. V.N. Vapnik. *Estimation of Dependencies Based on Empirical Data.* Springer, 1982.
58. V.N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.
59. V.N. Vapnik. Support vector learning machines. Tutorial at NIPS*97, Denver (CO), December 1997.
60. V.N. Vapnik. *Statistical Learning Theory.* John WILEY, 1998. (in preparation).
61. V.N. Vapnik and A.J. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory Probab. Apl.*, 16:264–280, 1971.
62. V.N. Vapnik and A.J. Chervonenkis. The necessary and sufficient conditions for consistency in the empirical risk minimization method. *Pattern Recognition and Image Analysis*, 1(3):283–305, 1991.
63. V.N. Vapnik, S.E. Golowich, and A. Smola. Support vector method for function approximation, regression estimation and signal processing. In M.C. Mozer, M.I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9, Proc. NIPS*96*, pages 281–287. MIT Press, 1997.
64. V.N. Vapnik and A.R. Stefanyuk. Nonparametric methods for estimating probability densities. *Automation and Remote Control*, 8, 1978.

Cet article a été créé en utilisant le style LaTeX READ