Classification multi-classes au prix d'un classifieur binaire

Zineb Noumir, Paul Honeine, Cédric Richard

Classification multi-classe au prix d'un classifieur binaire

Zineb NOUMIR[1], Paul HONEINE[1], Cédric RICHARD[2+]

[1] Institut Charles Delaunay (UMR STMR CNRS 6279) - LM2S, Université de technologie de Troyes, France

[2] Laboratoire H. Fizeau (UMR CNRS 6525, OCA), Université de Nice Sophia-Antipolis, France

[+] Institut Universitaire de France

zineb.noumir@utt.fr, paul.honeine@utt.fr, cedric.richard@unice.fr

Résumé – Cet article traite du problème de classification multi-classe en reconnaissance des formes. La résolution de ce type de problèmes nécessite des algorithmes au coût calculatoire souvent beaucoup plus élevé que les méthodes d'apprentissage dédiées à la classification binaire. On propose dans cet article une nouvelle formulation pour la conception de classifieurs multi-classes, nécessitant essentiellement la même complexité calculatoire que l'apprentissage d'un classifieur binaire. On montre que ce socle commun offre un cadre pour élaborer des algorithmes multi-classes en utilisant les mêmes routines d'optimisation que celles utilisées pour les problèmes de classification binaire. On illustre ce résultat avec les algorithmes SVM, LS-SVM et RLSC.

Abstract – This paper deals with the problem of multi-class classification in machine learning. Various techniques have been successfully proposed to solve such problems, with a computation cost often much higher than techniques dedicated to binary classification. To address this problem, we propose a novel formulation for designing multi-class classifiers, with essentially the same computational complexity as binary classifiers. The proposed approach provides a framework to develop multi-class algorithms using the same optimization routines as those already available for binary classification tasks. The effectiveness of our approach is illustrated with Support Vector Machines (SVM), Least-Squares SVM (LS-SVM), and Regularized Least Squares Classification (RLSC).

1 Introduction

En apprentissage statistique, le problème de classification multi-classe est rencontré dans de nombreuses applications du traitement du signal et des images, avec un nombre de classes à considérer de plus en plus important. C'est le cas par exemple en reconnaissance optique de caractères, reconnaissance de la parole [1] et identification faciale [2]. D'autres applications incluent la surveillance, comme la classification des structures urbaines dans une image [3] ou encore la surveillance de la qualité de l'eau dans un réseau de distribution [4] pour n'en nommer que quelques-unes.

Etant donné un ensemble d'apprentissage comprenant des observations de différentes classes, l'objectif est d'en déduire une règle de décision qui attribue correctement à toute nouvelle observation, sa classe d'appartenance. Si les méthodes de classification binaire ont été largement étudiées dans la littérature, on demeure en quête d'une généralisation de ces principes aux problèmes multi-classes. Outre des méthodes de résolution globale qui nécessitent des ressources importantes [5, 6, 7], plusieurs stratégies ont été élaborées sur la base d'une décomposition du problème en une collection de sous-problèmes binaires, dont il convient ensuite de combiner les résultats pour déterminer la solution multi-classe finale [5]. En particulier, deux stratégies ont été largement employées dans la littérature. La stratégie du *un-contre-tous* consiste à chercher à discriminer chacune des classes par rapport à toutes les autres, la règle de décision finale associée étant la loi du plus fort [8]. La stratégie du *un-contre-un* vise à élaborer un classifieur pour chaque paire de classes possibles, un vote majoritaire déterminant la règle de décision finale [9]. Ces deux stratégies offrent souvent des performances similaires, au moins aussi bonnes que les approches globales [8, 10]. Ceci ne rend que plus approprié celle qui nécessite le plus faible coût calculatoire, en l'occurrence la stratégie du *un-contre-tous*. Toutefois, la complexité algorithmique de celle-ci croît de manière linéaire avec le nombre de classes.

L'article présenté propose un cadre pour la conception de classifieurs multi-classes, avec essentiellement la même complexité calculatoire que celle requise pour l'apprentissage d'un classifieur binaire. Ceci est rendu possible en s'inspirant de la stratégie du un-contre-tous, où chaque classifieur binaire est optimisé à partir des mêmes données d'apprentissage que ses congénères, la différence que nous y apportons résidant dans les étiquettes binaires attribuées, et surtout un faisceau de relations entre les classifieurs individuels qui permet d'en limiter le nombre de degrés de liberté et de les optimiser collectivement en une passe.

En s'inspirant des récents travaux en apprentissage multi-tâche [11, 12, 13], nous revisitons ici des méthodes de classification binaires classiques afin d'en proposer une version multi-classe, sur la base des mêmes routines d'optimisation et sans en augmenter véritablement le coût calculatoire. Plus précisément, nous nous attardons sur trois méthodes largement décrites dans

Remerciements — Ce travail est soutenu par l'ANR dans le cadre du Projet ANR-08-SECU-013-02 "VigiRes'Eau".

la littérature :

- *Support Vecteurs Machines* (SVM) : En reposant sur l'idée de maximisation de marge à l'aide d'une fonction coût charnière, seule une petite fraction des données d'apprentissage contribue à la solution. Le prix à payer est un problème d'optimisation de type programmation quadratique [14].
- *Least-squares SVM* (LS-SVM) : En considérant une fonction coût de moindres carrés avec des contraintes d'égalité pour la maximisation de marge, le problème d'optimisation correspond à un système d'équations linéaires, résolu par une procédure d'inversion matricielle [15].
- *Regularized least-squares classification* (RLSC) : Dans [16], une fonction coût de moindres carrés est considérée avec une régularisation de type Tikhonov, qui conduit à une solution après inversion matricielle.

2 Approche proposée

Pour un problème de classification binaire, les méthodes d'apprentissage statistique classiques [14, 15, 16] visent à déterminer une fonction de décision de la forme

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i \, y_i \, \boldsymbol{x}_i^\top \boldsymbol{x}.$$

On a pour cela recours à un ensemble d'apprentissage de la forme $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$ où chaque élément est étiqueté selon $y_i = \pm 1$. Une fonction de décision non-linéaire est aisément obtenue, si nécessaire, en remplaçant le produit scalaire $\boldsymbol{x}_i^\top \boldsymbol{x}$ par une fonction noyau $\kappa(\boldsymbol{x}_i, \boldsymbol{x})$.

Un problème de classification multi-classe peut être résolu en traitant préalablement plusieurs sous-problèmes de classification binaire, puis en combinant le résultat des différentes fonctions de décision. Chaque sous-problème définit alors une fonction de décision de la forme

$$f_k(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_{i,k} \, y_{i,k} \, \boldsymbol{x}_i^\top \boldsymbol{x}. \tag{1}$$

Bien qu'utilisant les mêmes données $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ pour l'apprentissage, chaque sous-problème de classification binaire définit ses propres étiquettes $y_{1,k}, y_{2,k}, \ldots, y_{n,k} \in \{-1; +1\}$ pour $f_k(\cdot)$.

Dans une stratégie de type un-contre-tous, [1] on fait le choix de regrouper les m fonctions de la façon suivante : $\boldsymbol{f}(\cdot) = [f_1(\cdot)\ f_2(\cdot)\ \cdots\ f_m(\cdot)]^\top$. On définit par ailleurs la classe d'appartenance de $\boldsymbol{x}_i$ par un vecteur étiquette $\boldsymbol{y}_i$ de taille $1 \times m$, dont le ℓ-ème élément est donné par

$$[\boldsymbol{y}_i]_\ell = \begin{cases} \sqrt{\frac{m-1}{m}} & \text{si l'observation } \boldsymbol{x}_i \text{ appartient à la classe } \ell; \\ \frac{-1}{\sqrt{m(m-1)}} & \text{sinon.} \end{cases}$$

1. Il est entendu que l'approche proposée se prête également à la stratégie du un-contre-un, avec $y_{1,k}, y_{2,k}, \ldots, y_{n,k} \in \{-1; 0; +1\}$, mise de côté ici faute de place. Pour la même raison, on ne traite pas la question du codage optimal où m fonctions de sortie peuvent coder jusqu'à 2^m classes différentes.

Cette représentation vectorielle est une simple généralisation au cas multi-classe de l'étiquette ± 1, choisie de sorte que la norme de $\boldsymbol{y}_i$ soit égale à 1, et que la moyenne de ses composantes soit nulle. Chacune des fonctions $f_1(\cdot), f_2(\cdot), \ldots, f_m(\cdot)$ est de la forme (1), ce qui nécessite l'estimation de nm coefficients $\alpha_{i,k}$. Nous proposons ici de réduire significativement le nombre d'inconnues, en imposant une relation entre ces fonctions, c'est-à-dire entre les coefficients qui les caractérisent, selon

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{W}^\top \boldsymbol{x} + \boldsymbol{b}, \tag{2}$$

avec $\boldsymbol{W}$ la matrice de taille $d \times m$ définie par

$$\boldsymbol{W}^\top = \sum_{i=1}^{n} \alpha_i \, \boldsymbol{y}_i \, \boldsymbol{x}_i^\top.$$

Ceci réduit le problème à l'identification de n coefficients à partir de l'ensemble d'apprentissage. Pour chaque nouvelle observation, la régle de décision consiste à comparer la sortie $\boldsymbol{f}(\boldsymbol{x})$ à l'ensemble des m vecteurs étiquettes $\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots \boldsymbol{y}_m$ possibles [2]

$$\begin{aligned} d(\boldsymbol{x}) &= \arg\max_k \ \boldsymbol{y}_k^\top \boldsymbol{f}(\boldsymbol{x}) \\ &= \arg\max_k \sum_{i=1}^{n} \alpha_i \, \boldsymbol{y}_k^\top \boldsymbol{y}_i \, \boldsymbol{x}_i^\top \boldsymbol{x} + \boldsymbol{y}_k^\top \boldsymbol{b}. \end{aligned}$$

3 Algorithmes multi-classes

Algorithme SVM multi-classe

L'algorithme SVM est basé sur une fonction coût charnière qui confère un caractère parcimonieux à la solution, au prix d'une optimisation par programmation quadratique. Le problème d'optimisation ici revisité est donné par

$$\begin{gathered} \min_{\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{\xi}} \ \frac{1}{2} \|\boldsymbol{W}\|_F^2 + \gamma \sum_{i=1}^{n} \xi_i, \\ \text{sous contraintes : } \boldsymbol{y}_i^\top (\boldsymbol{W}^\top \boldsymbol{x}_i + \boldsymbol{b}) \geq 1 - \xi_i, \\ \xi_i \geq 0, \quad \forall i \end{gathered}$$

pour $i = 1, 2, \cdots, n$, où γ est le paramètre de régularisation et ξ_i est une variable d'écart autorisant à un petit nombre de données d'apprentissage à violer la règle de grande marge. Dans cette expression, $\|\cdot\|_F$ désigne la norme de Frobenius. La solution à ce problème d'optimisation sous contrainte est donnée par le point-selle du Lagrangien, à savoir

$$\begin{aligned} \min_{\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{\xi}} \max_{\boldsymbol{\alpha}, \boldsymbol{\beta}} \ \frac{1}{2} \|\boldsymbol{W}\|_F^2 + \gamma \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \beta_i \, \xi_i \\ - \sum_{i=1}^{N} \boldsymbol{\alpha}_i (\boldsymbol{y}_i (\boldsymbol{W}^\top \boldsymbol{x}_i + \boldsymbol{b}) - 1 + \xi_i) \end{aligned}$$

2. D'autres régles de décision auraient pu être adoptées, comme par exemple la distance de Hamming.

où $\boldsymbol{\alpha}$ et $\boldsymbol{\beta}$ sont les multiplicateurs de Lagrange. Les conditions d'optimalité sont alors données par

$$\begin{cases} \frac{\partial L}{\partial \boldsymbol{W}} = 0 & \Rightarrow \quad \boldsymbol{W}^\top = \sum_{i=1}^n \alpha_i \boldsymbol{y}_i \boldsymbol{x}_i^\top \\ \frac{\partial L}{\partial \boldsymbol{b}} = 0 & \Rightarrow \quad \sum_{i=1}^n \alpha_i \boldsymbol{y}_i = \boldsymbol{0}_m \\ \frac{\partial L}{\partial \xi_i} = 0 & \Rightarrow \quad \alpha_i + \delta_i = \gamma \end{cases}$$

Ceci conduit au problème dual suivant :

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j \, \boldsymbol{y}_i^\top \boldsymbol{y}_j \, \boldsymbol{x}_i^\top \boldsymbol{x}_j$$
$$\text{sous contraintes : } \sum_{i=1}^n \alpha_i \, \boldsymbol{y}_i = \boldsymbol{0}_m,$$
$$0 \leq \alpha_i \leq \gamma, \quad \forall i$$

où $\boldsymbol{0}_m$ (resp. $\boldsymbol{1}_m$) désigne le vecteur nul (resp. unité) à m éléments. Sous forme matricielle avec $\boldsymbol{K}$ la matrice de Gram d'éléments $\boldsymbol{y}_i^\top \boldsymbol{y}_j \, \boldsymbol{x}_i^\top \boldsymbol{x}_j$, on obtient $\max_{\boldsymbol{\alpha}} \boldsymbol{1}_n^\top \boldsymbol{\alpha} - \frac{1}{2} \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha}$, sous contraintes $\boldsymbol{Y} \boldsymbol{\alpha} = \boldsymbol{0}_m$, et $\boldsymbol{0}_n \leq \boldsymbol{\alpha} \leq \gamma \boldsymbol{1}_n$. Il s'agit donc essentiellement du même problème de programmation quadratique que celui des SVM binaires, la seule différence résidant dans les contraintes d'égalité. Néanmoins, les mêmes routines d'optimisation peuvent être utilisées pour les deux types de tâches de classification, binaire et multi-classe.

Algorithme LS-SVM multi-classe

En considérant à présent une fonction coût quadratique plutôt que charnière, on aboutit au problème d'optimisation avec contraintes d'égalité suivant

$$\min_{\boldsymbol{W}, \boldsymbol{b}, \boldsymbol{\xi}} \frac{1}{2} \|\boldsymbol{W}\|_F^2 + \frac{\gamma}{2} \sum_{i=1}^n \xi_i^2,$$
$$\text{sous contraintes : } \boldsymbol{y}_i^\top (\boldsymbol{W}^\top \boldsymbol{x}_i + \boldsymbol{b}) = 1 - \xi_i, \quad \forall i.$$

Le Lagrangien associé est alors

$$\frac{1}{2} \|\boldsymbol{W}\|_F^2 + \frac{\gamma}{2} \sum_{i=1}^n \xi_i^2 - \sum_{i=1}^n \boldsymbol{\alpha}_i (\boldsymbol{y}_i (\boldsymbol{W}^\top \boldsymbol{x}_i + \boldsymbol{b}) - 1 + \xi_i)$$

où $\boldsymbol{\alpha}$ est le multiplicateur de Lagrange, avec les conditions d'optimalité

$$\begin{cases} \frac{\partial L}{\partial \boldsymbol{W}} = 0 & \Rightarrow \quad \boldsymbol{W}^\top = \sum_{i=1}^n \alpha_i \boldsymbol{y}_i \boldsymbol{x}_i^\top \\ \frac{\partial L}{\partial \boldsymbol{b}} = 0 & \Rightarrow \quad \sum_{i=1}^n \alpha_i \boldsymbol{y}_i = 0 \\ \frac{\partial L}{\partial \xi_i} = 0 & \Rightarrow \quad \alpha_i = \gamma \xi_i \end{cases}$$

Cela conduit au système linéaire suivant

$$\left[\begin{array}{c|c} \boldsymbol{0}_{m,m} & \boldsymbol{Y} \\ \hline \boldsymbol{Y}^\top & \boldsymbol{K} + \gamma^{-1} \boldsymbol{I}_{n,n} \end{array} \right] \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_m \\ \boldsymbol{1}_n \end{bmatrix}, \qquad (3)$$

où $\boldsymbol{0}_{m,m}$ (resp. $\boldsymbol{I}_{m,m}$) est la matrice nulle (resp. identité) de taille $m \times m$. La résolution de ce système nécessite l'inversion d'une matrice de taille $(n+m) \times (n+m)$. Puisque le nombre de classes est généralement très inférieur à la taille de la base d'apprentissage, le coût calculatoire requis reste comparable à celui du problème bi-classe correspondant. Ces grandeurs sont à mettre en perspective avec celles de l'algorithme LS-SVM multiclasses, qui nécessite l'inversion d'une matrice de taille $(mn+m) \times (mn+m)$ [5], et le recours à la stratégie du un-contre-tous qui conduirait ici à la résolution de m problèmes du type (3).

Algorithme RLSC multi-classe

En considérant toujours la fonction coût quadratique, d'autres formes sont possibles. En s'inspirant de [16] pour la classification binaire où le biais n'est pas utilisé, voir ci-dessous pour une discussion sur le biais, on considère le problème d'optimisation

$$\min_{\boldsymbol{W}} \sum_{j=1}^n \|\boldsymbol{W}^\top \boldsymbol{x}_i - \boldsymbol{y}_i\|^2 + \gamma \, \mathcal{R}(\boldsymbol{W}),$$

où le terme de régularisation est donné par

$$\mathcal{R}(\boldsymbol{W}) = \sum_{i,j=1}^n \alpha_i \alpha_j \, \boldsymbol{y}_i^\top \boldsymbol{y}_j \, \boldsymbol{x}_i^\top \boldsymbol{x}_j.$$

Sous forme matricielle, on aboutit au problème d'optimisation

$$\min_{\boldsymbol{\alpha}} \boldsymbol{Y}^\top \boldsymbol{Y} - 2 \, \boldsymbol{d} \, \boldsymbol{\alpha} + \boldsymbol{\alpha}^\top \boldsymbol{G} \boldsymbol{\alpha} + \gamma \, \boldsymbol{\alpha}^\top \boldsymbol{K} \boldsymbol{\alpha},$$

avec $\boldsymbol{d}$ le vecteur d'éléments $\sum_i \boldsymbol{y}_i^\top \boldsymbol{y}_j \, \boldsymbol{x}_i^\top \boldsymbol{x}_j$, et $\boldsymbol{G}$ la matrice d'éléments $\boldsymbol{y}_i^\top \boldsymbol{y}_j (\boldsymbol{x}_i^\top \boldsymbol{x}_j)^2$, pour $i, j = 1, 2, , \ldots, n$. En annulant le gradient de la fonction coût ci-dessus par rapport à $\boldsymbol{\alpha}$, on obtient la solution finale

$$(\boldsymbol{G} + \gamma \, \boldsymbol{K}) \, \boldsymbol{\alpha} = \boldsymbol{d}.$$

Il est clair que le problème de classification multi-classe est donné par un système linéaire de n équations à n inconnues. La complexité calculatoire est alors cubique en n, le nombre de données d'apprentissage, mais demeure toutefois indépendante du nombre de classes.

Discussion sur le biais b

L'utilisation d'une version biaisée ou non biaisée est encore une question ouverte en apprentissage statistique, dans le cas de classification binaire [17] comme dans le cas multi-classe [18]. Par exemple, pour les SVM, et par analogie avec le cas binaire, on peut estimer la valeur du biais $\boldsymbol{b}$ en faisant une moyenne sur tous les vecteurs de supports. Dans [19, page 203], il est conseillé de ne pas utiliser une telle estimation, mais plutôt de l'ajuster selon le taux de faux positifs et faux négatifs. De nombreux auteurs écartent complètement le terme de biais du modèle, comme étudié dans [20, 17]. Il est à noter que dans les implémentations de SVM qui écartent le terme de biais, la contrainte linéaire $\sum_{i=1}^n \alpha_i \, \boldsymbol{y}_i = \boldsymbol{0}_m$ est alors relachée. Pour l'approche LS-SVM, de nombreuses études motivent l'utilisation de la version non biaisée, voir par exemple [16]. Dans la pratique, il s'avère que la version non biaisée du LS-SVM fournit essentiellement les mêmes performances que celle biaisée. Pour toutes ces raisons, on considère le cas sans-biais dans les expérimentations.

				un-contre-tous						approche proposée					
				SVM		LS-SVM		RLSC		SVM		LS-SVM		RLSC	
	n	m	d	erreur	temps	erreur	temps	erreur	temps	erreur	temps	erreur	temps	erreur	temps
iris	150	3	4	4.0	**8.2**	2.0	**1.3**	2.6	**1.2**	4.0	**0.9**	2.7	**0.5**	0.0	**0.7**
wine	178	3	13	1.1	**16.9**	0.0	**5.2**	0.5	**4.6**	2.2	**2.6**	0.0	**1.9**	0.0	**2.1**
glass	214	6	13	22.9	**139.6**	24.3	**10.9**	19.3	**10.7**	26.6	**3.1**	19.2	**2.3**	20.8	**2.6**
vowel	528	11	10	1.1	$>$ **500**	0.9	**190.0**	0.6	**194.4**	1.1	**21.4**	0.9	**22.7**	0.5	**30.5**

TABLE 1 – Taux d'erreur (%) et temps moyen (en seconde) pour la stratégie un-contre-tous et pour l'approche proposée. Quatre bases de données sont étudiées, avec n échantillons disponibles, d attributs et m classes.

4 Expérimentations

Pour monter la pertinence de notre approche, on considère quatre bases de données du UCI Repository connues dans la littérature : iris, wine, glass, et vowel. Les données ont été normalisées afin d'appartenir à l'intervalle $[-1\ 1]$. Le noyau Gaussien a été utilisé, avec $\exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2)$. Les paramètres γ et σ ont été optimisés selon la grille $[2^{-5}; 2^{-4}; \cdots; 2^5] \times [2^{-5}; 2^{-4}; \cdots; 2^5]$. L'erreur moyenne de classification a été estimée par une validation croisée 10-fold. Dans la Table 3, nous donnons le taux d'erreur de classification et le temps de calcul pour chaque classificateur, en comparant la méthode proposée avec celle du un-contre-tous. On remarque que les performances sont semblables, et comparables aux résultats donnés dans [10], alors que le temps moyen de calcul, donné ici à titre indicatif, [3] est largement inférieur pour notre approche.

5 Conclusion et perspectives

Dans cet article, nous avons étudié le problème de classification multi-classe en apprentissage statistique. En proposant trois algorithmes de classification multi-classes, nous avons montré que la conception de la classification multi-classe est possible avec essentiellement la même complexité calculatoire qu'un classifieur binaire. La pertinence de l'approche proposée est soutenue par des expérimentations sur des bases de données connues dans la littérature.

Dans de futurs travaux, on envisage d'adapter notre approche à la stratégie du *un-contre-un*. D'autre part, on souhaite étudier l'impact de différentes règles de décision, comme par exemple la distance de Hamming.

Références

[1] A. Klautau, N. Jevtic, and A. Orlitsky, "Combined binary classifiers with applications to speech recognition," in *International Conference on Spoken Language Processing*, vol. 4, 2002.

[2] Z. Lihong, S. Ying, Z. Yushi, Z. Cheng, and Z. Yi, "Face recognition based on multi-class SVM," in *Proc. 21st annual international conference on Chinese control and decision conference*. IEEE Press, 2009.

[3] P. Honeine and C. Richard, "The angular kernel in machine learning for hyperspectral data classification," in *Proc. IEEE Workshop on Hyperspectral Image and Signal Processing : Evolution in Remote Sensing (WHISPERS)*, Reykjavík, Iceland, 2010.

3. Temps de calcul estimé avec une implémentation sous Matlab sur un portable Macbook Pro biprocesseur 2.53 GHz Intel ; 4 GB RAM.

[4] K. Klise and S. Mckenna, "Water quality change detection : multivariate algorithms," in *Proceedings- Spie The International Society For Optical Engineering*, vol. 6203. SPIE Defense and Security Symposium, 2006.

[5] J. Suykens and J. Vandewalle, "Multiclass least squares support vector machines," in *Proc. International Joint Conference on Neural Networks*. World Scientific, 1999.

[6] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *Journal of Machine Learning Research*, vol. 1, pp. 143–160, 2002.

[7] G. M. Fung and O. L. Mangasarian, "Multicategory proximal support vector machine classifiers," *Mach. Learn.*, vol. 59, pp. 77–97, May 2005.

[8] R. Rifkin and A. Klautau, "In defense of one-vs-all classification," *Journal of Machine Learning Research*, vol. 5, pp. 101–141, 2004.

[9] J. Kittler, I. C. Society, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, pp. 226–239, 1998.

[10] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multi-class support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, pp. 415–425, 2002.

[11] T. Evgeniou, C. A. Micchelli, and M. Pontil, "Learning multiple tasks with kernel methods," *Journal of Machine Learning Research*, vol. 6, pp. 615–637, 2005.

[12] S. Szedmak and J. Shawe-Taylor, "Multiclass learning at one-class complexity," 2005, technical Report, ISIS Group, Electronics and Computer Science.(Unpublished).

[13] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables." *Journal of Machine Learning Research*, vol. 6, pp. 1453–1484, 2005.

[14] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, 1998.

[15] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, pp. 293–300, 1999.

[16] R. Rifkin, "Everything old is new again : A fresh look at historical approaches in machines learning," in *PhD thesis, MIT*, 2002.

[17] T. Poggio, S. Mukherjee, R. Rifkin, A. Rakhlin, and A. Verri., "b," *Uncertainty in Geometric Computations*, pp. 131–141, 2002.

[18] L. G. Abril, C. Angulo, F. Velasco, and J. A. Ortega, "A note on the bias in svms for multiclassification," *IEEE Transactions on Neural Networks*, vol. 19, no. 4, pp. 723–725, 2008.

[19] B. Schölkopf and A. J. Smola, *Learning with Kernels : Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. The MIT Press, 2001.

[20] S. Ertekin, L. Bottou, and C. L. Giles, "Nonconvex online support vector machines," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 2, pp. 368 – 381, 2011.