Linear Simplex Support Vector Regression

Quentin KLOPFENSTEIN[1], Samuel VAITER[2]

[1]Institut Mathématiques de Bourgogne
9 avenue Alain Savary, 21078, DIJON Cedex, France

[2]CNRS, Institut Mathématiques de Bourgogne
9 avenue Alain Savary, 21078, DIJON Cedex, France
quentin.klopfenstein@u-bourgogne.fr, samuel.vaiter@u-bourgogne.fr

Résumé – L'algorithme de Support Vector Regression est utilisé notamment en biostatistique par la méthode appelée Cibersort, afin d'estimer les proportions de cellules présentes au sein d'une tumeur. De par la nature de ce qui est estimé, l'estimateur obtenu doit avoir des coefficients positifs dont la somme est égale à un. La méthode la plus utilisée aujourd'hui dans ce domaine ne gère ces contraintes que dans un second temps, dans une étape de post-normalisation. Nous proposons donc un nouvel estimateur appelé Linear Simplex Support Vector Regression (LSSVR) qui prend en compte les contraintes liées à l'estimation de proportions directement dans le problème d'optimisation. Nous proposons un algorithme de résolution généralisant les techniques de type Sequential Minimization Optimization.

Abstract – The Support Vector Regression algorithm is used in biostatistics by the Cibersort method, to estimate the proportions of cells present inside the tumor. Since we are estimating proportions, the estimator has to have positive coefficients and their sum should be equal to one. The gold standard method used today only manage these constraints in a second step, in a post-normalization step. We propose a new estimator called Linear Simplex Support Vector Regression (LSSVR) which takes into account these constraints directly. We propose an algorithm that solve the LSSVR problem generalizing the Sequential Optimizing Optimization technics.

1 Introduction

Les algorithmes de Machine à Vecteurs de Support (SVM), proposés par Vapnik [12], sont très utilisés pour effectuer de la classification ou de la régression. On les retrouve pour la classification dans des tâches de reconnaissance faciale [6], classification d'images [2], classification des types de cancer [5] et pour la partie régression dans la prédiction de série temporelle en finance [11] et la prédiction de quantité de cellules au sein d'une tumeur [8] entre autres. C'est cette dernière application qui est le contexte de notre travail. Le but est de retrouver à partir de l'expression des gènes de la tumeur (signal brouillé), la quantité de cellules de différents types présentes au sein de celle-ci. On dispose également de l'expression des gènes de cellules seules (signal pur).

Soit $X \in \mathbb{R}^{l \times n}$ la matrice qui contient les signaux purifiés des populations cellulaires. Cette matrice est appelée matrice de signature. Le nombre de lignes l représente le nombre de gènes et n le nombre de populations cellulaires dont la quantitée est à estimer. Soit également $y \in \mathbb{R}^l$ un signal brouillé obtenu à partir d'un échantillon tumoral dont on veut connaitre la composition cellulaire. On peut alors modéliser la relation entre X et y par un modèle linéaire :

$$y = X\beta + \epsilon \tag{1}$$

où $\beta \in \mathbb{R}^n$ est le vecteur des quantités que l'on estimera à partir des données contenues dans X et y et $\epsilon \in \mathbb{R}^n$ est le terme d'erreur.

Cibersort [8] est le nom de la méthode de référence utilisée aujourd'hui pour effectuer l'estimation de ces proportions de cellules. Elle est basée sur l'algorithme de Support Vector Regression (SVR) [4]. Cette méthode ne prend pas en compte les contraintes liées à l'estimation de proportions directement dans le problème d'optimisation mais va en tenir compte seulement dans une étape de projections après l'estimation. Nous proposons un estimateur basé sur la SVR mais qui inclut les contraintes de positivités et de somme des coefficients égale à 1 dans le problème d'optimisation. L'estimateur obtenu peut donc directement être interprété comme des proportions de cellules sans avoir à le normaliser. Cet estimateur, Linear Simplex Support Vector Regression (LSSVR), ajoute ces deux contraintes linéaires au problème quadratique classique de la SVR. Cela contraint l'estimateur à appartenir au simplexe de probabilités de l'espace auquel il appartient. Cette idée s'inscrit dans la même ligne que les travaux effectués dans [7] et [3] sur respectivement l'incorporation de connaissance préalable dans la SVR et la régression ordinale. Nous proposons un algorithme de résolution du problème LSSVR généralisant les techniques de type Sequential Minimization Optimization (SMO). Nous comparons les performances de cet algorithme avec l'algorithme de la SVR classique et l'algorithme du solver CVXopt.

2 Support Vector Regression

La version du problème la plus utilisée est la ϵ-SVR mais la version utilisée en biostatisque est la version ν-SVR de Schölkopf [10]. Les différences entre ces deux algorithmes viennent notamment de leurs hyperparamètres. La version ϵ-SVR a deux hyperparamètres : ϵ qui contrôle le seuil à partir duquel une penalité sera appliquée à une observation et C qui contrôle l'erreur globale. La ν-SVR a deux hyperparamètres ν, qui permet de contrôler le nombres de vecteurs de support utilisés pour l'estimation et C qui contrôle l'erreur globale. ϵ est une variable dans le problème d'optimisation.

2.1 Le problème d'optimisation

Le but de la SVR est de trouver les coefficients d'une fonction linéaire qui caractérise la relation entre X et y. Cette fonction linéaire que l'on note f a la forme suivante :

$$f(x) = \beta^T x + b, \text{avec } \beta \in \mathbb{R}^n, b \in \mathbb{R} \tag{2}$$

L'estimation des coefficients β est le résultat du problème d'optimisation primal :

$$\begin{aligned} \min_{\beta,b,\xi_i,\xi_i^*,\epsilon} \quad & \frac{1}{2}||\beta||^2 + C(\nu\epsilon + \frac{1}{l}\sum_{i=1}^{l}(\xi_i + \xi_i^*)) \\ \text{sujet à} \quad & y_i - \beta^T x_i - b \leq \epsilon + \xi_i \\ & \beta^T x_i + b - y_i \leq \epsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0, \epsilon \geq 0. \end{aligned} \tag{3}$$

Le choix des hyperparamètres $C \in \mathbb{R}$ et $\nu \in [0, 1]$ est souvent réalisé par validation croisée en choisissant le couple de paramètres qui minimisent l'erreur quadratique moyenne par exemple. Les variables ξ et ξ^* sont des variables de ressort qui permettent de relacher les contraintes. Sans ces variables le problème d'optimisation (3) n'aurait de solution que si toutes les observations y_i seraient au plus éloignées de ϵ de $\beta^T x_i$. Ce qui dans la pratique n'est pas souvent le cas. Cela revient à regarder la fonction de coût suivante :

$$|\xi|_\epsilon = \begin{cases} 0 & \text{si } |\xi| \leq \epsilon \\ |\xi| - \epsilon & \text{sinon.} \end{cases} \tag{4}$$

Dans la plupart des cas la solution de (3) est plus facile à obtenir en utilisant sa formulation du problème dual. En écrivant le lagrangien du problème (3) et les conditions d'optimalités KKT, on arrive à la formulation suivante du problème dual :

$$\begin{aligned} \min_{\alpha,\alpha^*} \quad & \frac{1}{2}(\alpha - \alpha^*)Q(\alpha - \alpha^*) + y^T(\alpha - \alpha^*) \\ \text{sujet à} \quad & 0 \leq \alpha_i, \alpha_i^* \leq \frac{C}{l} \\ & \mathbf{e}^T(\alpha - \alpha^*) = C\nu \\ & \mathbf{e}^T(\alpha - \alpha^*) = 0, \end{aligned} \tag{5}$$

où $Q_{ij} = \langle x_i, x_j \rangle$ et $\mathbf{e} \in \mathbb{R}^l$ est le vecteur avec seulement des 1.

On remarque que (5) ne dépend que du produit scalaire entre x_i et x_j. Ce problème quadratique avec contraintes linéaires peut être résolu de différentes manières mais nous allons présenter rapidement la méthode de résolution avec l'algorithme de minimisation minimale séquentielle (Sequential Minimization Optimization, SMO) proposé par Platt [9].

2.2 L'algorithme de résolution SMO

L'algorithme implémenté dans la librairie libSVM [1] pour résoudre le problème (5) utilise la SMO. A chaque itération seulement deux variables sont choisies et optimisées. Le reste des variables est considéré comme constant au cours de cette étape. L'avantage principale de cette méthode est qu'il existe une écriture explicite de la solution pour deux variables. Il n'y a donc pas besoin d'utiliser un solver. Nous allons maintenant rapidement présenté cet algorithme.

On note $f(\alpha, \alpha^*) = \frac{1}{2}(\alpha - \alpha^*)Q(\alpha - \alpha^*) + y^T(\alpha - \alpha^*)$ et $\nabla f(\alpha, \alpha^*)$ son gradient. On peut montrer que les conditions d'optimalités pour le problème (5) se résume à :

Pour les variables α_i

— **Case 1**- $\alpha_i = 0$ implique que $\nabla f(\alpha, \alpha_*)_i - \mu_1 - \mu_2 \geq 0$

— **Case 2**- $\alpha_i = \frac{C}{l}$ implique que $\nabla f(\alpha, \alpha_*)_i - \mu_1 - \mu_2 \leq 0$

— **Case 3**- $0 < \alpha_i < \frac{C}{l}$ implique que $\nabla f(\alpha, \alpha_*)_i - \mu_1 - \mu_2 = 0$

On considère alors deux ensembles d'indices :

$$I_{up} = \{i \in \{1, \ldots, l\} : \alpha_i < \frac{C}{l}\} \tag{6}$$

$$I_{low} = \{i \in \{1, \ldots, l\} : \alpha_i > 0\} \tag{7}$$

La condition d'optimalité pour ces variables sont satisfaites si et seulement si $\min_{i \in I_{up}} \nabla f(\theta)_i \geq \max_{j \in I_{low}} \nabla f(\theta)_j$

Pour les variables α_i^*

— **Case 1**- $\alpha_i^* = 0$ implique que $\nabla f(\alpha, \alpha_*)_i - \mu_1 + \mu_2 \geq 0$

— **Case 2**- $\alpha_i^* = \frac{C}{l}$ implique que $\nabla f(\alpha, \alpha_*)_i - \mu_1 + \mu_2 \leq 0$

— **Case 3**- $0 < \alpha_i^* < \frac{C}{l}$ implique que $\nabla f(\alpha, \alpha_*)_i - \mu_1 + \mu_2 = 0$

On considère alors deux ensembles d'indices :

$$I_{up}^* = \{i \in \{1, \ldots, l\} : \alpha_i^* < \frac{C}{l}\} \tag{8}$$

$$I_{low}^* = \{i \in \{1, \ldots, l\} : \alpha_i^* > 0\} \tag{9}$$

La condition d'optimalité pour ces variables sont satisfaites si et seulement si $\min_{i \in I_{up}^*} \nabla f(\theta)_i \geq \max_{j \in I_{low}^*} \nabla f(\theta)_j$

L'algorithme SMO va donc à chaque itération choisir deux variables à optimiser en prenant les variables qui s'éloignent le plus de la condition d'optimalité. C'est à dire que si la condi-

tion n'est pas respecté on va choisir les indices :

$$i = \underset{i \in I_{up}}{\operatorname{argmin}} \nabla f(\theta)_i \tag{10}$$

$$j = \underset{i \in I_{low}}{\operatorname{argmax}} \nabla f(\theta)_i \tag{11}$$

$$i^* = \underset{i \in I^*_{up}}{\operatorname{argmin}} \nabla f(\theta)_i \tag{12}$$

$$j^* = \underset{i \in I^*_{low}}{\operatorname{argmax}} \nabla f(\theta)_i \tag{13}$$

Les variables à optimiser seront donc α_i et α_j si $\nabla f(\theta)_j - \nabla f(\theta)_i \geq \nabla f(\theta)_{j^*} - \nabla f(\theta)_{i^*}$ et α^*_i, α^*_j sinon.

3 Linear Simplex Support Vector Regression (LSSVR)

A partir de ces travaux déjà bien connus sur la SVR et les applications dont nous avons parlé en introduction, nous avons étudié ce que l'ajout de contraintes linéaires sur β pourrait changer dans la résolution du problème d'optimisation. Nous avons travaillé sur les contraintes liées au simplexe de probabilités. Ces contraintes apparaissent naturellement dans l'application de la SVR en biostatistiques lorsqu'on cherche à estimer des proportions de cellules présent dans une tumeur. Le problème d'optimisation primal (3) avec l'ajout de ces contraintes devient donc :

$$\begin{aligned} \min_{\beta,\epsilon,\xi_i,\xi_i^*,b} \quad & \frac{1}{2}||\beta||^2 + C(\nu\epsilon + \frac{1}{l}\sum_{i=1}^{l}(\xi_i + \xi_i^*)) \\ \text{sujet à} \quad & (\beta^T x_i + b) - y_i \leq \epsilon + \xi_i \\ & y_i - (\beta^T x_i + b) \leq \epsilon + \xi_i^* \\ & \sum_{j=1}^{n} \beta_j = 1 \\ & \beta_j \geq 0 \\ & \xi_i, \xi_i^* \geq 0, \epsilon \geq 0 \end{aligned} \tag{14}$$

On s'est intéressé au problème dual afin de voir s'il était possible de trouver des similitudes avec (5) et éventuellement d'appliquer un algorithme de type SMO modifié. Le dual du problème d'optimisation (14) peut s'écrire sous la forme :

$$\begin{aligned} \min_{\theta} \quad & \frac{1}{2}\theta^T \bar{Q}\theta + p^T\theta \\ \text{sujet à} \quad & 0 \leq \theta_i \leq \frac{C}{l}, \text{ pour } i = 1, \ldots, 2l \\ & \sum_{i=1}^{2l} \theta_i = C\nu \\ & \sum_{i=1}^{l} \theta_i - \sum_{i=l+1}^{2l} \theta_i = 0 \\ & \theta_i \geq 0 \text{ pour } i = 2l+1, \ldots, 2l+n, \end{aligned} \tag{15}$$

où $\bar{Q} \in \mathbb{R}^{N \times N}$ avec $N = 2l + n + 1$, θ et $p \in \mathbb{R}^N$.

Les conditions KKT vont donner différentes conditions d'optimalités pour les différents blocs de variables correspondant aux différentes contraintes sur les coefficients de θ. Elles vont avoir de grandes similarités avec les conditions du problème (5) mais on va avoir quatre blocs de variables au lieu de deux.

On définit $f(\theta) = \frac{1}{2}\theta^T\mathbb{Q}\theta + p^T\theta$ et $\nabla f(\theta)$ son gradient et également 4 ensembles d'indices :

$$I_{up} = \{i \in \{1, \ldots, l\} : \theta_i < \frac{C}{l}\} \tag{16}$$

$$I_{low} = \{i \in \{1, \ldots, l\} : \theta_i > 0\} \tag{17}$$

$$I^*_{up} = \{i \in \{l+1, \ldots, 2l\} : \theta_i < \frac{C}{l}\} \tag{18}$$

$$I^*_{low} = \{i \in \{1+1, \ldots, 2l\} : \theta_i > 0\} \tag{19}$$

Les conditions d'optimalités sont les suivantes :

— Si $i \in \{1, \ldots, l\}$: les conditions d'optimalités sont satisfaites si et seulement si

$$\min_{i \in I_{up}} \nabla f(\theta)_i \geq \max_{j \in I_{low}} \nabla f(\theta)_j \tag{20}$$

— Si $i \in \{l+1, \ldots, 2l\}$: les conditions d'optimalités sont satisfaites si et seulement si

$$\min_{i \in I^*_{up}} \nabla f(\theta)_i \geq \max_{j \in I^*_{low}} \nabla f(\theta)_j \tag{21}$$

— Si $i \in \{2l+1, \ldots, 2l+n\}$: les conditions d'optimalités sont satisfaites si et seulement si

$$\nabla f(\theta)_i \geq 0 \tag{22}$$

— Si $i = 2l + n + 1$: les conditions d'optimalités sont satifaites si et seulement si

$$\nabla f(\theta)_i = 0 \tag{23}$$

Les conditions (20) et (21) sont les mêmes que pour la SVR classique. Les conditions (22) et (23) viennent de l'ajout des contraintes liées au simplexe de probabilités. A partir de ces conditions d'optimalités nous proposons un algorithme adapté de la SMO pour la SVR classique mais qui va alterner des étapes de SMO classique et des étapes d'optimisation des variables liées aux contraintes du simplexe de probabilités.

L'algorithme que nous proposons part d'un vecteur dans le domaine de faisabilité et à chaque itération choisie une ou deux variables à optimiser. On considère :

$$\delta_1 = \nabla f(\theta)_j - \nabla f(\theta)_i \tag{24}$$

$$\delta_2 = \nabla f(\theta)_{j^*} - \nabla f(\theta)_{i^*} \tag{25}$$

$$\delta_3 = -\min_{k \in \{2l+1, \ldots, 2l+n\}} \nabla f(\theta)_k \tag{26}$$

$$\delta_4 = \nabla f(\theta)_{2l+n+1} \tag{27}$$

Si $\delta_1 = \max(\delta_1, \delta_2, \delta_3, \delta_4)$ alors on applique une étape de type SMO classique. De même si $\delta_2 = \max(\delta_1, \delta_2, \delta_3, \delta_4)$. Si

$\delta_3 = \max(\delta_1, \delta_2, \delta_3, \delta_4)$, on selectionne la variable qui s'éloigne le plus la condition d'optimalité (22) et cela revient à résoudre à problème d'optimisation à une variable avec une contrainte de positivité. Enfin si $\delta_4 = \max(\delta_1, \delta_2, \delta_3, \delta_4)$, il suffit simplement de trouver le minimum d'une fonction quadratique à une variable sans contraintes. Notre algorithme passe de bloc en bloc en optimisant la/les variable(s) qui sont le plus loin de la condition d'optimalité.

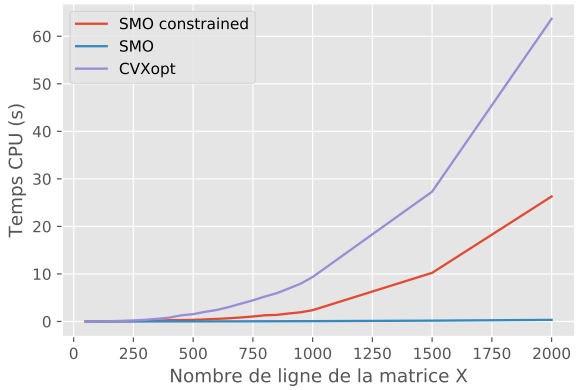

FIGURE 1 – Comparaison des temps de calcul pour la résolution de la SVR et LLSVR

Nous avons généré des données synthétiques pour tester la vitesse de résolution de notre algorithme par rapport au solveur CVXopt et la SMO pour la SVR classique. Ces données ont été obtenues via la fonction *make_regression* de scikit-learn. Le résultat de cette génération est donc une matrice X, un vecteur y et un vecteur de coefficient β_0 qui donne la relation linéaire entre X et y. Afin que β_0 appartienne au simplexe de probabilités, nous avons simplement appliqué une projection de β_0 sur cet espace noté β_0^Δet modifié le y en $y^\Delta = X\beta_0^\Delta$. Pour cette simulation, nous avons fixé le nombre de colonnes de X, $n = 5$ et les paramètres de la LLSVR $C = 1$ et $\nu = 0.5$. Les temps de calculs ont été obtenus sur une machine avec un processeur Intel Core i5 de 2.3 GHz et possédant 8Go de RAM.

La figure 1 présente les temps de calcul pour les différentes méthodes de résolution des problèmes (3) et (14). On remarque que l'algorithme SMO pour la LSSVR est plus rapide que la résolution avec le solver CVXopt. Cependant, l'ajout des contraintes du simplexe dans le problème d'optimisation augmente de manière importante le temps de résolution du problème. L'algorithme SMO pour résoudre la SVR classique reste plus rapide est reste de l'ordre de la seconde.

4 Conclusion

L'estimateur LLSVR intègre les contraintes liées à l'estimation de proportions à l'intérieur du modèle d'optimisation de la SVR. Nous proposons un algorithme de résolution basé sur la SMO qui est plus efficace que les solvers classiques. Une des questions sous-jacentes à la modélisation est de savoir si le modèle linéaire est le modèle le plus adapté pour effectuer la quantification des populations cellulaires. Les méthodes de type SVM permettent facilement d'adapter la méthode à d'autres types de relation, grâce notamment au "kernel trick". Il pourra être intéressant de voir s'il est possible d'adapter la LSSVR à d'autres noyaux (gaussien, RBF, sigmoid, ...)

Références

[1] C. Chang and C. Lin. Training v-support vector regression : Theory and algorithms. *Neural Comput.*, 14(8) :1959–1977, August 2002.

[2] O. Chapelle, P. Haffner, and V. N. Vapnik. Support vector machines for histogram-based image classification. *IEEE Transactions on Neural Networks*, 10(5) :1055–1064, Sep. 1999.

[3] W. Chu and S. S. Keerthi. Support vector ordinal regression. *Neural Comput.*, 19(3) :792–815, March 2007.

[4] H. Drucker, C. J. C. Burges, L. Kaufman, A. J. Smola, and V. Vapnik. Support vector regression machines. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 155–161. MIT Press, 1997.

[5] David Haussler, David W. Bednarski, Michèl Schummer, Nello Cristianini, Nigel Duffy, and Terrence S. Furey. Support vector machine classification and validation of cancer tissue samples using microarray expression data . *Bioinformatics*, 16(10) :906–914, 10 2000.

[6] H. Jia and A. M. Martinez. Support vector machines in face recognition with occlusions. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 136–141, June 2009.

[7] F. Lauer and G. Bloch. Incorporating prior knowledge in support vector regression. *Machine Learning*, 70, 01 2008.

[8] A. M. Newman, C.L. Liu, M. R. Green, A. J. Gentles, W. Feng, Y. Xu, C. D. Hoang, M. Diehn, and A. A. Alizadeh. Robust enumeration of cell subsets from tissue expression profiles. *Nature methods*, 12(5) :453–457, May 2015.

[9] J. Platt. Sequential minimal optimization : A fast algorithm for training support vector machines. page 21, April 1998.

[10] B. Schölkopf, P. Bartlett, A. Smola, and R. Williamson. Shrinking the tube : A new support vector regression algorithm. In *Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II*, pages 330–336, Cambridge, MA, USA, 1999. MIT Press.

[11] T. Van Gestel, J. A. K. Suykens, D. . Baestaens, A. Lambrechts, G. Lanckriet, B. Vandaele, B. De Moor, and J. Vandewalle. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Transactions on Neural Networks*, 12(4) :809–821, July 2001.

[12] V Vapnik and A Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24, 1963.