Robust Least Squares Support Vector Machines for Classification

Iwein Vranckx[a,*], Joachim Schreurs[b], Bart De Ketelaere[c], Mia Hubert[a], Johan Suykens[b]

[a]*KU Leuven, Department of Mathematics and LStat, Celestijnenlaan 200B, BE-3001 Heverlee, Belgium*
[b]*KU Leuven, ESAT-STADIUS, Kasteelpark Arenberg 10, BE-3001 Heverlee, Belgium*
[c]*KU Leuven, Division of Mechatronics, Biostatistics and Sensors, Kasteelpark Arenberg 30, BE-3001 Heverlee, Belgium*

Abstract

(Abstract uit weighted LS-SVM, ter voorbeeld). Least squares support vector machines (LS-SVM) is an SVM version which involves equality instead of inequality constraints and works with a least squares cost function. In this way, the solution follows from a linear KarushKuhnTucker system instead of a quadratic programming problem. However, sparseness is lost in the LS-SVM case and the estimation of the support values is only optimal in the case of a Gaussian distribution of the error variables. In this paper, we discuss a method which can overcome these two drawbacks. We show how to obtain robust estimates for regression by applying a weighted version of LS-SVM. We also discuss a sparse approximation procedure for weighted and unweighted LS-SVM. It is basically a pruning method which is able to do pruning based upon the physical meaning of the sorted support values, while pruning procedures for classical multilayer perceptrons require the computation of a Hessian matrix or its inverse. The methods of this paper are illustrated for RBF kernels and demonstrate how to obtain robust estimates with selection of an appropriate number of hidden units, in the case of outliers or non-Gaussian error distributions with heavy tails

Keywords: Robust support vector machines, Non-linear outlier detection,

[*] Corresponding author
URL: wis.kuleuven.be/stat/robust (Iwein Vranckx), iwein.vranckx@kuleuven.be (Iwein Vranckx)

1. Introduction

Contribution

1. Weighted LS-SVM variant voor classificatie (new?).
2. Kernel Concentration steps (new)
3. Soft reweighting based on Stahel-Donoho outlyingness
4. New Pruning stategy based on information transfer and Entropy (?)

We are/should be robust against high degrees of contamination in non-linear classification problems.

The remainder of this paper is organized as follows. In section 2 we introduce our weighted least squares support vector machine (LS-SVM). In section 3 we propose our robust outlier detection routine, followed by our support vector sparseness routine (??). The extensive simulation results of both theoretical and real, industrial datasets listed in section 4 conform the robustness, prediction time speed-up and improved classifier efficiency of our proposed method. Finally, our main findings and suggestions for further research are summarized in the conclusion, section 5.

2. Weighted LS-SVM for classification

2.1. LS-SVM classification

A binary classifier's goals is to learn a prediction model that assigns the correct label to an unseen test sample. Restated more formally, we seek an SVM-based classifier that fits an optimal hyperplane between the two classes, where the optimal hyperplane maximizes the distance to the closest point of either class. Maximizing the hyperplane margin $||w||^{-1}$, one obtains a classifier with good generalisation properties. More naturally formulated, this inherent principle leads to a (much) better classification performance on (unseen) test data, for example compared with density based estimators.

Given an p-variate trainingsset $x \in \mathbb{R}^p$ and the binary class label $y_i \in [+1, -1]$ for observation x_i with index i, the following constrained optimization must be solved to obtain the WEIGHTED LS-SVM representation:

$$min\ J(w, b, e_i) = \frac{1}{2} w^T w + \frac{\gamma}{2} \sum_{i=1}^{n} e_i^2 \qquad (1)$$

...subject to the equality constraints:

$$y_i[w^T\varphi(x_i) + b] = 1 - e_i \tag{2}$$

Here $w^T\varphi(x_i)+b$ is the hyperplane based decision function of the classifier with corresponding parameters w and b. The weight vector $WEIGHT$ denotes the normalized outlier weight for each observation, γ is used to control the over/under-fitting trade-off and $\varphi(.)$ is the transformation from input space $\mathbb{R}^p$ to the kernel feature space.

The specified constraint states that every given multivariate sample should be lie on the correct side of hyperplane. Stated differently, the classifier should predict the class label of each sample correctly, where each observation x_i has an corresponding error e_i due to the constraint equality sign. This, in turn, implies that a LS-SVM loses its support vector sparseness compared to ordinary SVM's.

This constrained optimization problem is solved by the optimality conditions of its Lagrangian as follows:

$$L(w, b, e; \alpha) = J(w, b, e) - \sum_{i=1}^{n} \alpha_i(y_i[w^T\varphi(x_i) + b] - 1 + e_i) \tag{3}$$

Here, α_i is the Lagrange multiplier whose value can be either positive or negative, which is also different from the standard SVM proposed by Vapnik. As a direct consequence of the equality sign in the given constraint the specified Lagrangian is now solvable trough a linear system of equations:

$$\frac{\partial L}{\partial w} = 0 \rightarrow w = \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i) \tag{4}$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow \alpha^T y = 0 \tag{5}$$

$$\frac{\partial L}{\partial e} = 0 \rightarrow \alpha = \gamma e \tag{6}$$

$$\frac{\partial L}{\partial \alpha_i} = 0 \rightarrow y_i[w^T\varphi(x_i) + b] = 1 - e_i \tag{7}$$

Combining the first and last equation and defining

$$\begin{aligned} \Omega_{ij} &= y_i y_j \varphi(x_i)^T \varphi(x_j) && (8) \\ &= y_i y_j K(x_i, x_j) && (9) \end{aligned}$$

...yields the following set of linear equations.

$$\begin{bmatrix} 0_{(1,1)} & y^T_{(n,1)} \\ y_{(n,1)} & \Omega_{(n,n)} + \gamma^{-1} I_{(n,n)} \end{bmatrix} \begin{bmatrix} b_{(1,1)} \\ \alpha_{(n,1)} \end{bmatrix} = \begin{bmatrix} 0_{(1,1)} \\ \mathbf{1}_{(n,1)} \end{bmatrix} \tag{10}$$

The least squares solution of the aforementioned system of equations is then used to obtain the Lagrange coefficients α and offset b, where the decision function is:

$$\hat{y}(x) = \sum_{1}^{n} \alpha_i y_i K(x, x_j) + b \tag{11}$$

Or, written in a more convenient matrix notation, short-writing $\beta_i = (\alpha_i \; y_i)^T$

$$\hat{y}(x) = \beta K + b\mathbf{1} \tag{12}$$

$\hat{y}_i$ is the predicted output of the ith training data point. Note that α satisfies the linear constraint $\sum_{i=1}^{n} \alpha_i e_i = 1$, and that the derivation for least squares support vector regression follows the same reasoning

3. (Proposed method) Robust classification by Kernelized minimum covariance determinant

Uitleggen wat MCD is, waar het gebruikt wordt en dat dit het hart van ons algoritme is. This concentration step uses the Mahalanobis distance between an observation and the center of the training data in that space as a outlying measure in order to detect outliers.

The Minimum Covariance Determinant (MCD) method (Rousseeuw, 1984) is a highly robust estimator of multivariate location and scatter. Given an nxp data matrix X = (x1, . . . , xn)T with xi = (xi1, . . . , xip)T , its objective is to find h observations (with ceil(n + p + 1)/2) 6 h 6 n) whose covariance matrix has the lowest determinant. The MCD estimate of location μ is then the average of these h points, and the scatter estimate Σ is a multiple of their covariance matrix. In addition to being highly resistant to outliers, the MCD is affine equivariant, i.e. the estimates behave properly under affine transformations of the data. Although the MCD was already introduced in 1984, its practical use only became feasible since the introduction of the computationally efficient FASTMCD algorithm of Rousseeuw and Van Driessen (1999). Since then the MCD has been applied in various fields such as quality control, medicine, finance, image analysis and chemistry, see e.g. Hubert et al. (2008) and Hubert and Debruyne (2010) for references. The MCD is also being used as a basis to develop robust and computationally efficient multivariate techniques, such as e.g. principal component analysis (Croux and Haesbroeck, 2000; Hubert et al., 2005), factor analysis (Pison et al., 2003), classification (Hubert and Van Driessen, 2004), clustering (Hardin and Rocke, 2004), and multivariate regression (Rousseeuw et al., 2004). For a review see (Hubert et al., 2008). In this article we will present a deterministic algorithm for robust location and scatter, coined Kernel-MCD (kMCD)

In this section we briefly describe the FASTMCD algorithm and the spatial median estimator, as our new algorithm will use aspects of both.

3.1. Introducing the kernel minimum covariance determinant method

A major component of the FASTMCD algorithm is the concentration step (C-step), which works as follows. Given initial estimates μ_{old} for the center and Σ_{old} for the scatter matrix,

1. Compute the distances $d_{old(i)} = D(x_i, \mu_{old}, \Sigma_{old})$ for i = 1, . . . , n.

2. Sort these distances
3. compute μ_{new}, Σ_{new}
4. iterate....

In Theorem 1 of Rousseeuw and Van Driessen (1999) it was proved that the determinants converge to a steady-state solution. Therefore, if we apply C-steps iteratively, the sequence of determinants obtained in this way must converge in a finite number of steps.

The expectation of observations in kernel feature space is defined as:

$$\mathbb{E}[\phi(x)] = \int \phi(x)P(x)dx \tag{13}$$

Where $P(x)$ denotes the probability distribution of the training samples. Since the distribution $P(x)$ is usually unknown, one can estimate this expectation by the empirical center of the training dataset, defined as:

$$c_n = \frac{1}{n}\sum_{i=1}^{n}\phi(x_i) \tag{14}$$

The Mahalanobis distance between a sample $\phi(x)$ and the empirical center c_n is defined as:

$$d^2 = (\phi(x) - c_n)\Sigma^{-1}(\phi(x) - c_n) \tag{15}$$

Here, Σ is the biased covariance matrix of the observations in kernel feature space, formally defined as:

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}\Big((\phi(x_i) - c_n)(\phi(x_i) - c_n)^T\Big) \tag{16}$$

Without any explicit knowledge of the mapping function $\phi(.)$, the covariance matrix cannot be expressed in terms of the data $\phi(x)$ in the feature space. To overcome this problem, we use the singular value decomposition of the covariance matrix $\Sigma = V^TDV$. V denotes the matrix of eigenvectors v_k of Σ and D is the matrix of corresponding singular values λ_k, for $k = 1, 2, ..., n$, where each (λ_k, v_k) pair satisfies

$$\lambda_k v_k = \Sigma v_k \tag{17}$$

From the definition of the covariance matrix, equation 16, it can be seen that each eigenvector is a linear combination of the samples $\phi(x_i)$ in the feature space:

Algorithm 2— Reweighted K-MCD based LS-SVM.

4. Imposing support vector sparseness

4.1. Selection of the best support vectors

Having established the data model and optimized classification accuracy for uncontaminated datasets, let us return to the problem at hand: the development of a robust pruning strategy. This implies that prediction formula should be partitioned in a *relevant* (the support vectors, found by the pruning algorithm) - and *irrelevant* part. An overview of existing pruning methods can be found in Hoegaerts et. al [3]. A first approach was suggest by Suykens et. al [6], where sparseness is imposed by pruning support values from the sorted support value spectrum which results from the solution to the linear system. The motivation comes from the fact that the LS-SVM support values are proportional to the errors at the datapoints. Thus omitting the points that contribute least to the training error. An example of the support value spectrum can be seen on Figure **??**. The values with a high $|\alpha_k|$ reside close to the decision boundary and are thus important to classify correctly. However blindly taking the points with largest support value spectrum could lead to sub-optimal results. Firstly, when outliers are present, you want to be certain that these are not chosen. Secondly, points are chosen independently towards each other. This results in clumping of sv's. The first problem is address by running a kernelized minimum covariance determinant, which detects and omits potential outliers. The second problem is solved by introducing a "region of interest" (ROI), which consist of the points with the β percentage highest absolute value $|\alpha_k|$. From the ROI, h landmark points are determined, where h is the desired amount of support vectors.

(a) Example bad Alpha spectrum (b) Example good SV selection

4.1.1. Entropy

Selection of the landmark points is based on quadratic Renyi entropy [2] and the fixed size LS-SVM [7]. The landmarks points are chosen to maximize the quadratic Renyi Entropy:

$$H_R = -\log \int p(x)^2 dx. \tag{18}$$

The quadratic Renyi Entropy is approximated by the following equation[2]:

$$\int \hat{p}(x)^2 dx = \frac{1}{N^2} 1_v^{\mathrm{T}} \Omega 1_v, \tag{19}$$

where $1_v = [1; 1; ...; 1]$ and a normalized kernel is assumed with respect to density estimation. In the fixed-size LS-SVM approach, one chooses a fixed working set of size M which is initialized randomly. Afterwards, random points are swapped in and out. If the entropy increases, the swapped point is accepted, otherwise the original subset is kept. This process continues until the change in entropy is small or a fixed number of iterations is reached.

In contrast to the original fixed-size LS-SVM formulation, which is used to find a representative subset for the Nyström approximation [7]. We propose to first determine to region of interest, which includes the points that have the highest contribution to the robust LS-SVM model. On this reduced dataset, the fixed-size LS-SVM algorithm is used to select the h landmark points. This ensures that the ROI is well approximated and there is no clumping effect present. The improved sv selection for the toy-problem is visible on Figure **??**.

4.1.2. Determinantal point process

Selection of landmark points is based on determinantal point processes (DPPs) [5]. DPPs are particularly interesting for set selection problems where diversity is preferred. A point process $\mathcal{P}$ on a ground set $\mathcal{Y} = 1, 2, ..., N$ is a probability measure over point patterns of $\mathcal{C}$, which are finite subsets of $\mathcal{Y}$. When $\mathcal{C}$ is a random subset, drawn according to the DPP $\mathcal{P}$, we have:

$$\mathcal{P}(C \subseteq \mathcal{Y}) = \det(K_{\mathcal{C}}), \tag{20}$$

where $K \preceq I$ is a positive symmetric semidefinite matrix with all eigenvalues smaller then or equal to 1, containing the index elements of $\mathcal{Y}$. $K_{\mathcal{C}} = [K_{i,j}]_{i,j \in \mathcal{C}}$ contains the selected rows and columns of K and $\det(K_{\emptyset}) = 1$. From equation (20) follows:

$$\begin{align} \mathcal{P}(i \in \mathcal{Y}) &= K_{i,i} \tag{21} \\ \mathcal{P}(i, j \in \mathcal{Y}) &= K_{i,i}K_{j,j} - K_{i,j}K_{j,i} \tag{22} \\ &= \mathcal{P}(i \in \mathcal{Y})\mathcal{P}(j \in \mathcal{Y}) - K_{i,j}^2. \tag{23} \end{align}$$

The diagonal elements of the kernel matrix give the marginal probability of inclusion for individual elements of $\mathcal{Y}$, whereas the off-diagonal elements determine the (anti-)correlation between pairs of elements. Thus for large values of $K_{i,j}$, or a high similarity, points are unlikely the appear together.

In our case, we would like to build a DPP based on the kernel matrix K, which is done using L-ensembles [1]. The probability of observing a subset $C \subseteq \mathcal{Y}$ is now equal to:

$$\Pr(\mathcal{C}) = \frac{\det(K_{\mathcal{C}})}{\det(K + I)}, \tag{24}$$

where I is the identity matrix, and K a positive semidefinite matrix indexed by the elements of $\mathcal{Y}$. In contrast to equation (20), K only has to be positive semidefinite, while the eigenvalues previously where bounded above. When conditioning on a fixed cardinality $k = |\mathcal{C}|$, one gets the k-DPP [4].

The landmark selection algorithm consists of the following steps: We propose to first determine to region of interest, which includes the points that have the highest contribution to the robust LS-SVM model. On this reduced dataset, a k-DPP [4], where $k = h$, is used to sample the h landmark points. This ensures that the ROI is well approximated and there is no clumping effect present. The improved sv selection for the toy-problem is visible on Figure **??**.

4.2. Information transfer

The information contained in the prune candidates can then be transferred to the support vectors - an idea originally introduced in [???]. Starting from equation 12 with the appropriate matrix dimensions:

$$\hat{y}_{(1,n_2)} = \beta_{(1,n_1)} K_{(n_1,n_2)} + b_{(1,1)}\ \mathbf{1}_{(1,n2)} \tag{25}$$

Introducing sparse matrices, we could partition this expression in a (non) support vector part, denoted by the subscript S and N respectively.

$$\hat{y}_{(1,n_2)} = \beta_{S(1,n_1)} K_{S(n_1,n_2)} + \beta_{N(1,n_1)} K_{N(n_1,n_2)} + b_{(1,1)}\ \mathbf{1}_{(1,n2)} \tag{26}$$

Next, one transfers the information of $\beta_N K_N$ using the update $\Delta\beta$:

$$\Delta\beta K_S = \beta_N K_N \tag{27}$$

$$\Delta\beta = K_S^{\dagger} \beta_N K_N \tag{28}$$

We now have obtained an explicit expression for the update of our Lagrange multipliers.

$$\hat{\beta}_S = \beta_S + \Delta\beta \tag{29}$$

If we omit all zero rows in the matrices above we obtain a compressed matrix of size m_1 rows in n_2 dimensions. Here, m_1 equals the (a priori, before the training stage) defined number of support vectors. The classier prediction equation finally boils down to:

$$\hat{y}_{(1,n_2)} = \beta_{(1,m)} K_{(m,n_2)} + b_{(1,1)}\ \mathbf{1}_{(1,n2)} \tag{30}$$

Which is implemented using equation **??** given the knowledge that $n_1 = m$ - the number of a priori defined support vectors.

The only problem that remains is the section of the most relevant support vectors....

Algorithm 3— Sparse K-MCD based LS-SVM. summary all steps algorithm

5. Experiments

5.1. Simulation results

5.2. Industrial data results

6. Conclusions and future work

Acknowledgements

We thank Johan Speybrouck for providing us the industrial datasets and Tim Wynants for his feedback throughout this project. We also acknowledge the financial support of the VLAIO, grant HBC.2016.0208, for making this industrial research possible.

References

[1] Alexei Borodin. Determinantal point processes. *arXiv preprint arXiv:0911.1153*, 2009.

[2] Mark Girolami. Orthogonal series density estimation and the kernel eigenvalue problem. *Neural computation*, 14(3):669–688, 2002.

[3] Luc Hoegaerts, Johan AK Suykens, Joos Vandewalle, and Bart De Moor. A comparison of pruning algorithms for sparse least squares support vector machines. In *International Conference on Neural Information Processing*, pages 1247–1253. Springer, 2004.

[4] Alex Kulesza and Ben Taskar. k-dpps: Fixed-size determinantal point processes. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1193–1200, 2011.

[5] Alex Kulesza, Ben Taskar, et al. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

[6] Johan AK Suykens, Lukas Lukas, and Joos Vandewalle. Sparse approximation using least squares support vector machines. In *2000 IEEE International Symposium on Circuits and Systems. Emerging Technologies for the 21st Century. Proceedings (IEEE Cat No. 00CH36353)*, volume 2, pages 757–760. IEEE, 2000.

[7] Johan AK Suykens, Tony Van Gestel, and Jos De Brabanter. *Least Squares Support Vector Machines*. World Scientific, 2002.