RoS-LSSVM: Robust and Sparse 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

Least squares support vector machines (LS-SVM) are popular methods to solve classification problems. The 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 system instead of a quadratic programming problem. However, compared to normal SVM's, support vector sparseness is lost and the classifiers efficiency is highly influenced by outliers. In this paper, we discuss a method which can overcome these two drawbacks. Robustness is achieved by the kernel concentration steps, which allow us to estimate a robust estimator of multivariate location and scatter in feature space. Afterwards, the robust LS-SVM is pruned by sampling diverse points out of a region of interest, close to the decision boundary. Experiments show that the method is highly robust against trainingset contamination, where extreme spareness simultaneous achieved without diminishing the performance.

Keywords: Robust support vector machines, Non-linear outlier detection, Support vector pruning, Sparse LS-SVM

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

1. Introduction

Least-squares support vector machines (LS-SVM) are powerful methods for solving pattern recognition and regression problems [20]. The LS-SVM maps data to a higher dimensional space in which one constructs an optimal separating hyperplane. It has been applied to many real-world problems such as optimal control [21], financial time series prediction [23], system identification [6], electrical load forecasting [4] and many others. However, the classifier model has two major disadvantages. First, it is (very) sensitive to outliers, as the underlying system of equations lacks robustness. This results in the orientation of the separating hyperplane in the direction of anomalies which, in turn, might cause a large amount of misclassifications. The second disadvantage is that its solution lacks sparseness, which hinders the application of LS-SVM models for real-time prediction.

To reduce the influence of outliers, Suykens et al. [18] proposed the weighted LS-SVM. By iteratively assigning small weights to outliers and retraining, the method diminishes the effect of extreme values. Another solution was proposed by Yang et al. [25], where a truncated loss function is used in the objective that is solved by a concave-convex procedure and the newton algorithm. A third solution is suggested by Debruyne et al. [3], which proposes a weighted LS-SVM classification where weights are equal to the spatial rank with respect to the other feature vectors in the class.

In comparison to the standard support vector machines (SVM), the LS-SVM only requires solving a linear system, but it lacks sparseness in the number of solution terms. To solve this problem, Suykens et al. [19] propose a method that iteratively prunes the datapoints with lowest support values and retrains. Yang et al. [24] propose a one-step compressive pruning strategy to reduce the number of support vectors. Fixed-size LS-SVM sparsifies the LS-SVM by selecting important point or landmark points based on the quadratic Renyi Entropy criterion [20]. However the landmark points are fixed in advance and don't take into account information of the classification itself, which could lead to sub-optimal results. This is in contrast to the sparse LS-SVM [19], which chooses observations based on the impact on the classification boundary. A comparison of different pruning methods can be found in [7].

In this paper, we focus on tackling both problems at once. Other methods

are found in [1], where a primal formulation of the LS-SVM with a truncated loss is introduced, sparseness is obtained by the Nyström approximation. A second method is the weighted LS-SVM of Suykens et al. [18]. We propose a robust LS-SVM based on the minimum covariance determinant (MCD) [8], which is known to be a highly robust estimator of multivariate location and scatter. However, in contrast to the original formulation, the location and scatter is determined in feature space using the kernel trick, which enables the detection of non-linear outliers. Afterwards extreme sparsity is obtained by determinantal point processes (DPP) [13] (or maximizing the quadratic Renyi entropy [20]), which are capable of selecting diverse samples. To summarize, our main contributions consist off:

1. The development of a robust algorithm that is able to detect non-linear correlated outliers.
2. The development of a new LS-SVM support vector selection strategy, based on entropy and determinantal point processes.

The aggregation of both method's intermediate results yields our robust LS-SVM pruning methodology.

The remainder of this paper is organized as follows. A brief overview of LS-SVM is given in section 2. In section 3 we propose our classifier pruning strategy. The extensive simulation results of both theoretical and real, industrial datasets listed in section 4 confirm 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. LS-SVM for classification

The classifier goal is to learn a prediction model that assigns the correct label $y \in \{+1, -1\}$ to an unseen test sample. Restated more formally, we seek a classifier that fits an optimal hyperplane between the two classes, where the hyperplane maximizes the distance to the closest point(s) of either class. 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 $i = 1, .., n$, the following constrained optimization must be solved to obtain the LS-SVM representation of the hyperplane margin $||w||^{-1}$ maximization:

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

$$\text{such that } y_i[w^T\varphi(x_i)+b] = 1-e_i,\ \ i=1,..,n \tag{2}$$

Here $w^T\varphi(x_i)+b$ is the hyperplane based decision function of the classifier with corresponding parameters w and b, where the regularization parameter γ is used to control the over/under-fitting trade-off. Finally, $\varphi(.)$ is the transformation from input space $\mathbb{R}^p$ to the kernel feature space $\mathcal{H}$.

The specified constraint states that every given multivariate sample should 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 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 α as follows:

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

As a direct consequence of the equality sign in the given constraint, the specified Lagrangian is now solvable trough a linear system of equations:

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

$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{i=1}^{n}\alpha_i y_i = 0 \tag{5}$$

$$\frac{\partial L}{\partial e_i} = 0 \rightarrow \alpha_i = \gamma e_i,\ \ i=1,..,n \tag{6}$$

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

Combining the first and last equation and defining $\Omega(i,j)$ using the kernel trick for two observations with index i and j as:

$$\Omega(i,j) = y_i y_j \varphi(x_i)^T\varphi(x_j) \tag{8}$$

$$= y_i y_j K(x_i,x_j). \tag{9}$$

Common kernels are:

$$\text{Linear kernel}: K(x,y) = x^{\mathrm{T}}y \tag{10}$$

$$\text{Polynomial kernel}: K(x,y) = (\tau + x^{\mathrm{T}}y)^d \tag{11}$$

$$\text{RBF kernel}: K(x,y) = \exp\Big(-\frac{||x-y||_2^2}{\sigma^2}\Big) \tag{12}$$

This results in the following set of linear equations, where the kernel matrix K transforms observations to the kernel feature space $\mathcal{H}$.

$$\begin{bmatrix} 0 & y^T \\ y & \Omega + \gamma^{-1}\mathrm{I} \end{bmatrix} \begin{bmatrix} b \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{13}$$

where $y = [y_1; ...; y_n]$, $1 = [1; ...; 1]$ and I the identity matrix. 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 used for test set prediction is defined as:

$$\hat{y}(x) = \text{sign}\Big(\sum_{i=1}^{n} \alpha_i y_i K(x, x_j) + b\Big). \tag{14}$$

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 where the sign is omitted in the prediction function.

3. Proposed method

We first start by laying out the main topology of our proposed method for a 2-class classification problem, where our classifier pre-and post-processing operations are shown in gray, on Figure 1. The trainingset is first standardized (Step 1), followed by the outlier detection routine. Outliers are detected per class label $y \in \{+1, -1\}$ subset, which results in a parallel algorithm topology (Step 2-3, Step 4-5). The detected regular observations (the support vectors candidates) are then used to train a standard, LS-SVM classifier (step 6)). Because the solution lacks sparsity, a second algorithm is used (section 3.2) to extract the support vectors based on an a priori chosen sampling methodology. Finally, the information of all regular observations is transferred to the support vectors to improve its classification efficiency (Step 11).

Before continuing with the explanation of the methodology, it is important to first make some remarks. The first step of outlier detection is essential before continuing with the entropy procedure. Large contributions to the entropy will come from elements that have little or no structure [5]. If the first step is omitted, there is a large chance of sampling outliers as landmark points. The same reasoning is present for the k-DPP. In order to promote diversity, points that have a high similarity have a low chance of being sampled together. Consequently outliers, that have a low similarity to any other point in the dataset, have a high chance of being sampled. Secondly the method is elaborated for a 2-class problem, however the outlier detection and landmark selection is done on each class separately. The proposed method is thus easily applicable in a multi-class classification setting.

3.1. Outlier detection in kernel feature space

Modern multivariate robust statistical methods are frequently based on the Minimum Covariance Determinant (MCD) method, first introduced in [15]. This Mahalanobis distance based estimator can withstand a substantial amount of trainingset contamination (up to 50%), and is nowadays (also) employed as refinement step. Robust two-step mechanisms (initial estimation $\rightarrow$ refinement) inherit the robustness of the MCD, but offers an improved statistical efficiency [see e.g. [9]]. Deterministic variants and well know PCA-based applications are described in [11] and [10] respectively.

Given a trainingset X of n observations in p dimensions, the MCD-objective is to find the h observations whose sample covariance matrix has the lowest possible determinant, where the amount of regular observations $h < n$ is specified before the algorithm starts. The MCD-estimate of location c_h is then the average of these h points (the h-subset), whereas the scatter estimate is a multiple of their covariance matrix $\hat{\Sigma}_h$.

In order to find the MCD-estimate, the FastMCD-algorithm [16] uses so-called concentration steps (C-steps). Based on its robustness properties, simplicity and proven convergence, we propose to modify the C-step algorithm so that it can work in kernel feature space. This, in turn, allows the detection of *non-linear* outliers: a noteworthy difference compared to off-the-shelve robust statistical algorithms as normality is frequently assumed.

To enable the C-steps methodology in feature space $\mathcal{H}$, we integrate the

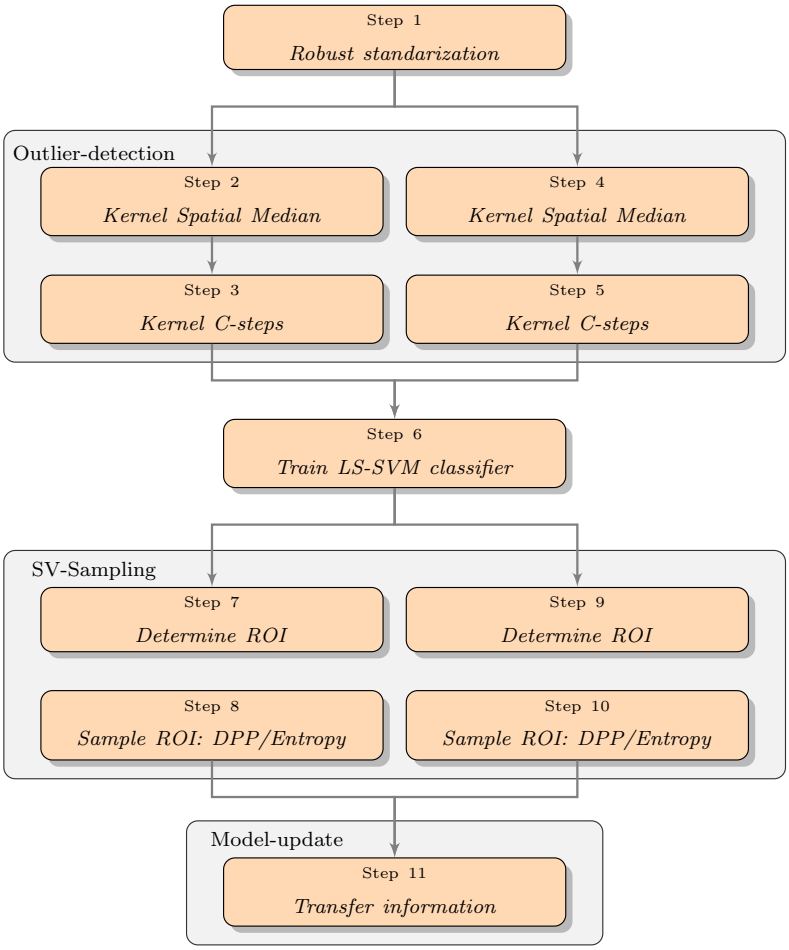

Figure 1: Topological order of the LS-SVM pruning method. First, an outlier detection algorithm is used to extract regular observations; These samples are then used to build a classic LS-SVM model. As the classifier solution lacks sparsity, it is subjected to a sampling algorithm to extract support vectors. Finally, the trained model is converted to a sparse LS-SVM model by an information transfer step. The outlier detection and landmark selection is done on each class separately which results in a parallel pipeline topology.

mapping function $\phi(.)$ in the required algorithm formula's:

$$c_h = \frac{1}{h} \sum_{i=1}^{h} \phi(x_i). \tag{15}$$

Likewise, the Mahalanobis distance in $\mathcal{H}$ is defined as:

$$||\phi(x) - c_h||^2_{\Sigma_h} = (\phi(x) - c_h)\Sigma_h^{-1}(\phi(x) - c_h). \tag{16}$$

Where the biased covariance matrix of the h-subset is calculated as:

$$\Sigma_h = \frac{1}{h}\sum_{i=1}^{h}(\phi(x_i) - c_h)(\phi(x_i) - c_h)^T. \tag{17}$$

As we do not explicit know the mapping function $\phi(.)$, equation (16) cannot be calculated, a problem circumvented by the eigendecomposition of the covariance matrix:

$$\Sigma_h = V^T D V. \tag{18}$$

Where V is the matrix of eigenvectors v^k and D resembles the diagonal matrix of eigenvalues λ^k for $k = 1, 2, \ldots, h$. The relation between eigenvalues and eigenvectors follows the identity:

$$\Sigma_h v^k = \lambda^k v^k. \tag{19}$$

From the definition of equation (17), it can be seen that each eigenvector is a linear combination of the observations $\phi(x_i)$ in kernel feature space:

$$v^k = \sum_{1}^{n} \alpha_i^k(\phi(x_i) - c_n). \tag{20}$$

If we substitute the above in equation (19), it follows that:

$$n\lambda^k \alpha^k = \tilde{K}\alpha^k. \tag{21}$$

Here, $\tilde{K}$ denotes the symmetric kernel matrix of h rows, centered in $\mathcal{H}$. The weights α are determined by solving the eigendecomposition problem.
By refactoring Σ^{-1} as $V^T D^{-1/2} D^{-1/2} V$ and exploiting reductant operations in the Mahalanobis distance formula, Nader et al. [14] show that equation (16) can be calculated in $\mathcal{H}$ as:

$$||\phi(x) - c_n||^2_{\Sigma} = \sum_{k=1}^{n}(\lambda^k)^{-1}\Big(\sum_{k=1}^{n}\alpha^k \tilde{k}(x_i, x)\Big)^2. \tag{22}$$

Finally, the centered kernel matrix $\tilde{K}$ is calculated as [17]:

$$\tilde{K}_{ij} = \left(\phi(x_i) - \frac{1}{n}\sum_{m=1}^{n}\phi(x_m)\right) \cdot \left(\phi(x_j) - \frac{1}{n}\sum_{k=1}^{n}\phi(x_k)\right) \tag{23}$$

$$= K_{ij} - \frac{1}{n}\sum_{m=1}^{n}K_{mj} - \frac{1}{n}\sum_{k=1}^{n}K_{ik} + \frac{1}{n^2}\sum_{m=1}^{n}\sum_{k=1}^{n}K_{mk} \tag{24}$$

$$= \left(K - 1_n K - K 1_n + 1_n K 1_n\right)_{ij}, \tag{25}$$

where $\tilde{K} \in \mathbb{R}^{n\times n}$ and $1_n \in \mathbb{R}^{n\times n}$ with all entries set to $1/n$. The centering of the kernel matrix of t observations, given the original trainingset follows a similar derivation:

$$\tilde{K}_t = K_t - 1_t K - K_t 1_n + 1_t K 1_n, \tag{26}$$

with kernel matrix $\tilde{K}_t \in \mathbb{R}^{t\times n}$ and the matrix $1_t \in \mathbb{R}^{t\times n}$ with all entries set to $1/n$. Using the methods we have discussed so far, our outlier detection algorithm is as follows.

Start with the robust standardization of each observation:

$$z_i = \frac{(x_i - \hat{\mu}_{mcd})}{\hat{\sigma}_{mcd}}, \tag{27}$$

where $\hat{\mu}_{mcd}$ and $\hat{\sigma}_{mcd}$ are the univariate location and scale estimations of the univariate MCD [16], estimated on the entire dataset. this procedure reduces the impact of different feature scales. Next, the standardized dataset is split in two subsets according to its class label $y \in \{+1, -1\}$ and our proposed procedure is applied (per subset) to derive at an outlier free support vector candidate set, as follows:

1. Calculate the spatial median in $\mathcal{H}$, as introduced in [2]. The spatial median can be written as a linear combination of the feature vectors. Denote $\gamma = (\gamma_1, \ldots, \gamma_n)^T$ the vector of coefficients, where $\sum_{i=1}^{n}\phi(x_i)\gamma_i$ equals the spatial median in feature space. Initialise $\gamma = (1/n, 1/n, \ldots)$ and recursively compute the normalized coefficients $\gamma = w/\sum_{i=1}^{n} w_i$, where w_i for the observation with index i is calculated as:

$$w_i = \frac{1}{\sqrt{K_{i,i} - 2\gamma^T K_{.,i} + \gamma^T K \gamma}}. \tag{28}$$

Ten iterations are usually enough to obtain a good approximation.

2. Calculate the Euclidean distance for each observation to the spatial median:

$$d_i = ||\phi(x_i) - \sum_{j=1}^{N} \gamma_j \phi(x_j)||^2 \tag{29}$$

$$= ||\phi(x_i)||^2 + ||\sum_{j=1}^{N} \gamma_j \phi(x_j)||^2 - 2 < \phi(x_i), \sum_{j=1}^{N} \gamma_j \phi(x_j) > \tag{30}$$

$$= K(x_i, x_i) + \sum_{j=1}^{N} \sum_{k=1}^{N} \gamma_j \gamma_k K(x_j, x_k) - 2 \sum_{j=1}^{N} \gamma_j K(x_i, x_j). \tag{31}$$

3. Sort these distances in ascending order. Define the initial h-subset by the h_{init} observations with the lowest distance.
4. Apply kernel C-step for a fixed amount of iterations as follows:
   (a) Calculate the kernel matrix K_h for the given h-subset of observations.
   (b) Center K_h in $\mathcal{H}$ by equation 23, obtaining $\tilde{K}_h$
   (c) Perform the singular value decomposition on $\tilde{K}_h$ to obtain the eigenvectors α and eigenvalue vector L_d.
   (d) Normalize each eigenvector α^k by its corresponding eigenvalue λ_k:

$$||\alpha^k|| = \frac{1}{n\lambda_k}, \text{for all components } k. \tag{32}$$

   (e) Calculate the kernel matrix for the given h-subset of observations in n features.
   (f) Center the kernel matrix by equation (26) and obtain the Mahalanobis distances with equation (22) or, expressed in vector notation:

$$d^2 = (\tilde{K}_t \alpha)^{\circ 2} \lambda^{-1}, \tag{33}$$

   where $d^2 \in \mathbb{R}^{n \times 1}$, $\tilde{K}_t \in \mathbb{R}^{t \times n}$, $\lambda^{-1} = [\lambda_1^{-1}, ..., \lambda_k^{-1}]^{\mathrm{T}} \in \mathbb{R}^{k \times 1}$, $\alpha \in \mathbb{R}^{n \times k}$ and $^{\circ 2}$ the element-wise square.
   (g) Sort the distance vector d^2 in ascending order and redefine the h-subset as the h observations with smallest distance.
   (h) If the new h-subset is equal to the previous one, exit the C-step loop.
5. If the h-subset size is lower then h_{end}, increase the h-subset size by (say) 5% and go back to step 4.

6. Determine the final support vector candidate mask. Assign a binary weight to each observation i:

$$w(d^2) = \begin{cases} 1 & \text{if } d^2 \leq c \\ 0 & \text{if } d^2 > c \end{cases} \tag{34}$$

where c is a threshold which we choose here as $c = max(\chi^2_p(0.975), max(d^2_h))$, with d^2_h the distance vector of the observations in the h-subset.

Finally, the weights w are then used to train a weighted LS-SVM, where only the points with a weight equal to 1 are used in training the model. The algorithm workings are visualised in Figure 2, where it can be seen that the initial estimate converges to the (linear and non-linear) outlier free observation subset.

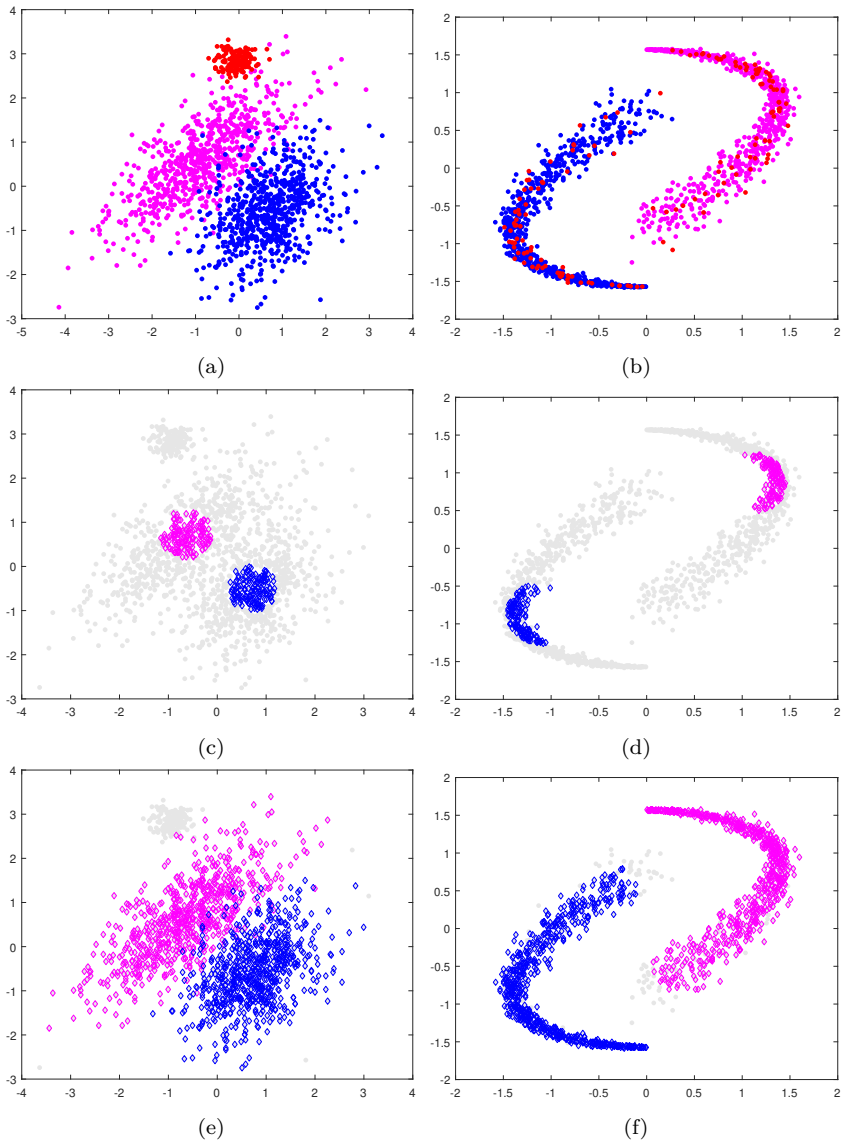

Figure 2: Figure a depicts a normal distributed, binary classification problem. A linear kernel will be used in the outlier detection of this dataset. Figure c shows a non-linear classification problem, where an RBF kernel with bandwidth $\sigma = 0.5$ is used. Both classes (blue, green) contain 10% outliers from, shown in red. After splitting the dataset by their class labels, the h_{init} closest points from the spatial median are used as initial subsets for the C-step algorithm [STEP 3], depicted in green for the first class of both datamodels (Figures b, d). After iterative kernel C-steps, we obtain the re-weighted results [STEP 6], shown in Figures e and f. Support vectors are then determined from the detected regular observation list by a landmark selection algorithm (section 3.2).

3.2. Imposing sparseness: support vector section methodology

The next step in the algorithm is introducing sparseness by pruning the previously trained robust LS-SVM. A first LS-SVM pruning approach was suggest by Suykens et al. [19], where sparseness is imposed by pruning support values from the sorted support value spectrum resulting from the solution to the linear system. The motivation comes from the fact that LS-SVM support values are proportional to the errors at datapoints. Thus omitting the points that contribute least to the training error. 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 landmarks. The first problem is addressed by running a kernelized minimum covariance determinant, which detects and omits potential outliers.

The second problem is solved by introducing a subset for each class $y \in \{+1, -1\}$ called the "region of interest" (ROI), which consist of points close to the decision boundary. Determining the ROI is done for each class and consists out of two steps:

1. Determine the subset which consists of all points belonging to the class and are not considered outliers by the kernel MCD procedure.
2. Reduce the subset by selecting the points where the sign of α_i is equal to sign of the currently analyzing class.
3. The subset is further reduced by selecting the points that satisfy:

$$|\alpha_i| \geq \text{median}(|\alpha|), \text{for } \; i = 1, ..., j \tag{35}$$

   where $\alpha = [\,|\alpha_1|, ..., |\alpha_j|\,]^{\mathrm{T}}$ and α_i the support value of points in the subset of step 2 with cardinality j.

In contrast to the proposed pruning strategy by Suykens et al. [19], the sign of alphas is taken in consideration when determining the ROI (step 2). Potential landmark points are now points with large importance and that predict the right class. This is easily seen from the prediction equation (14), where the contribution of a training point in the prediction a test point x_t towards class $y \in \{+1, -1\}$ is proportional to $y\,\mu_i$, where $\mu_i = \alpha_i\, y_i$.

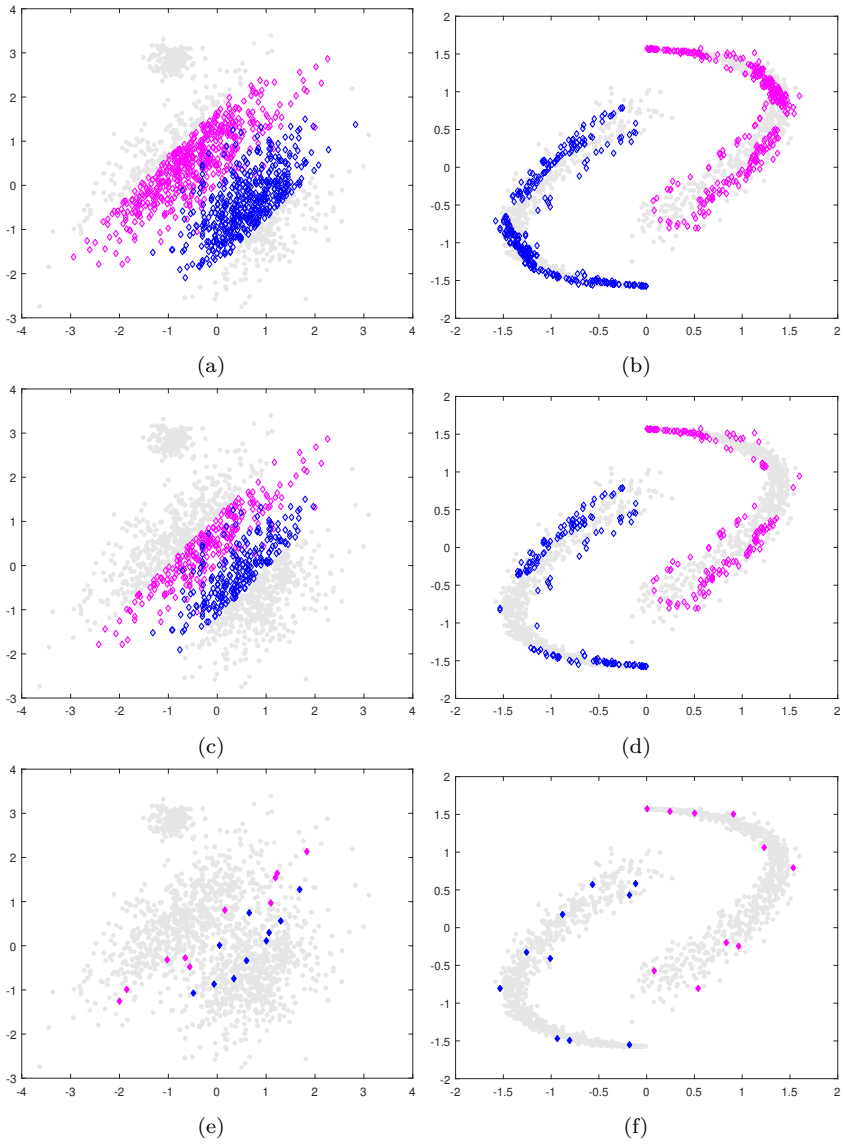

Figure 3: An example of the sparsification procedure where the same datasets as in Figure 2 are analysed. Figures a and b show step 2 of the ROI algorithm, where only points that have an equal sign to the corresponding class are selected. The subset if reduced even further by selecting points with the largest contribution to the decision boundary. This corresponds to step 3 of the ROI algorithm and is visible on Figures c and d. The final step is sampling diverse points out of the ROI. Figures e and f show the result of this procedure, where the ROI is nicely approximated and no clumping effect is visible.

Figure 2 confirms this behavior, where the ROI represents a subset with points important for the decision boundary for every class. Using a sampling algorithm that promotes diversity, l landmark points are selected for every class to represent the ROI. Due to the diversity, no clumping is possible.

3.2.1. Entropy

Selection of the landmark points is based on quadratic Renyi entropy [5] and the fixed size LS-SVM [20]. The landmarks points are chosen to maximize the quadratic Renyi Entropy:

$$H_R = -\log \int p(x)^2 dx. \tag{36}$$

The quadratic Renyi Entropy is approximated by the following equation[5]:

$$\int \hat{p}(x)^2 dx = \frac{1}{N^2} 1_v^{\mathrm{T}} \Omega 1_v, \tag{37}$$

where $1_v = [1; 1; ...; 1]$ and a normalized kernel is assumed with respect to density estimation. Entropy quantifies the diversity or randomness of a system, which is essential to select landmark points such that no clumping is possible . 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 [20]. We propose to first determine to ROI, 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 l landmark points. This ensures that the ROI is well approximated and there is no clumping effect present. The downsides of the entropy criteria are that a relatively large number of iterations are necessary to get a diversified selection, secondly only the Gaussian kernel can be used [5].

Given the ROI of candidate support vectors for a class $y \in \{+1, -1\}$, the entropy based selection algorithm is repeated for every class [20]:

1. Choose a working set of size l by randomly selection support vector out of the ROI.

2. Randomly select a point out of the ROI and replace it by a random point in the working set.
3. Calculate the quadratic Renyi entropy for the new working set using equations (36) and (37).
4. If the entropy increases, keep the swapped point, otherwise return to the original working set.
5. Stop if the change in entropy is small or a maximum number of iterations is reached. Otherwise go back to step 2.

3.2.2. Determinantal point process

Selection of landmark points is based on determinantal point processes (DPPs) [13]. 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:

$$\mathcal{P}(C \subseteq \mathcal{Y}) = \det(K_{\mathcal{C}}), \tag{38}$$

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 (38) follows:

$$\mathcal{P}(i \in \mathcal{Y}) = K_{i,i} \tag{39}$$

$$\mathcal{P}(i, j \in \mathcal{Y}) = K_{i,i}K_{j,j} - K_{i,j}K_{j,i} \tag{40}$$

$$= \mathcal{P}(i \in \mathcal{Y})\mathcal{P}(j \in \mathcal{Y}) - K_{i,j}^2. \tag{41}$$

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. DPP's are probabilistic models with global, negative correlations with respect to a similarity measure, this ensures diversity.

In our case, we would like to build a DPP based on the kernel matrix K, which is done using L-ensembles [13]. The probability of observing a subset $C \subseteq \mathcal{Y}$ is now equal to:

$$\Pr(\mathcal{C}) = \frac{\det(K_{\mathcal{C}})}{\det(K + I)}, \tag{42}$$

where I is the identity matrix, and K a positive semidefinite matrix indexed by the elements of $\mathcal{Y}$. In contrast to equation (38), K only has to be positive semidefinite, while the eigenvalues previously where bounded above. Sampling a DPP is done in two phases [13]. In the first phase, a subset of the eigenvectors V of the kernel matrix K is selected at random, where the probability of selecting each eigenvector depends on its associated eigenvalue. In the second phase, a sample Y is produced based on the selected vectors. Note that on each iteration of the second loop, the cardinality of Y increases by one and the dimension of V is reduced by one. When conditioning on a fixed cardinality $k = |\mathcal{C}|$, one gets the k-DPP [12].

The landmark selection algorithm consists of the following steps: We propose to first determine the ROI, which includes the points that have the highest contribution to the robust LS-SVM model. On this reduced dataset, a k-DPP [12], where $k = l$, is used to sample the l landmark points. This ensures that the ROI is well approximated and there is no clumping effect present.

Given the ROI of candidate support vectors for a class $y \in \{+1, -1\}$, the k-DPP selection algorithm is repeated for every class to select l landmark points [12]:

1. Calculate the eigenvector/value pairs $\{(v_r, \lambda_r)\}$ from the L-ensemble matrix $L = K_R(I - K_R)^{-1}$, where $K_R \in \mathbb{R}^{N_R \times N_R}$ is the kernel matrix containing the points in the ROI.
2. Iterate over all components of the eigenvector/value pairs and include the component r if:
$$u \sim U[0,1] < \lambda_r \frac{e_{k-1}^{r-1}}{e_k^r}, \tag{43}$$
where k is number of components selected so far, iterator $r = N_R, ..., 1$ and k elementary polynomial $e_k^r = \sum_{\substack{J \subseteq \{1,...,r\}\} \\ |J|=k}} \prod_{n \in J} \lambda_n$. Continue to the next step if you have selected l components. The matrix V is now equal to the eigenvectors of included components.
3. Select a landmark point x_i from ROI with probability:
$$\Pr(x_i) = \frac{1}{|V|} \sum_{v \in V} (v^{\mathrm{T}} e_i)^2. \tag{44}$$
The vector e_i is the ith standard basis, which is all zeros except for a one in the ith position.

4. Recalculate $V \leftarrow V_\perp$, an orthonormal basis for the subspace of V orthogonal to e_i. Afterwards go back to step 3 until l landmark points are selected.

3.3. Information transfer

The information contained in the not selected support vectors can then be transferred to the selected support vectors, an idea originally introduced in [22]. Starting from equation (14) for the outlierfree dataset, we get in matrix notation:

$$\hat{y} = \text{sign}(K\beta + b1), \tag{45}$$

with $\hat{y} = [\hat{y}_1; ...; \hat{y}_h]$ the prediction vector and $\beta = [\alpha_i y_i, ..., \alpha_i y_h]^{\text{T}}$, where h is the number of points in the outlierfree dataset. We can partition this expression in the selected landmark part and non-selected support vectors:

$$\hat{y} = \text{sign}(K_l\beta_l + K_{nl}\beta_{nl} + b1), \tag{46}$$

where K_l is the matrix whose columns are selected from K using the landmark selection while the other columns of K form matrix K_{nl}, and β_l is a column vector whose elements are the multipliers corresponding to columns in K_l while the other multipliers compose the vector β_{nl}. Next, one transfers the information of $\beta_{nl}K_{nl}$ using the update $\Delta\beta$:

$$\Delta\beta K_l = K_{nl}\beta_{nl} \tag{47}$$

$$\Delta\beta = K_l^\dagger K_{nl}\beta_{nl}. \tag{48}$$

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

$$\hat{\beta}_l = \beta_l + \Delta\beta. \tag{49}$$

The classier prediction equation finally boils down to:

$$\hat{y}(x) = \text{sign}\Big(\sum_{i=1}^{l} K(x, x_i)\hat{\beta}_i + b\Big). \tag{50}$$

3.4. Summary RoS-LSSVM

The following procedure is applied on a contaminated dataset:

[INPUT high level algorithm here]

The final solution is visible on Figure 4, where extreme sparsity and a robust classification is achieved.

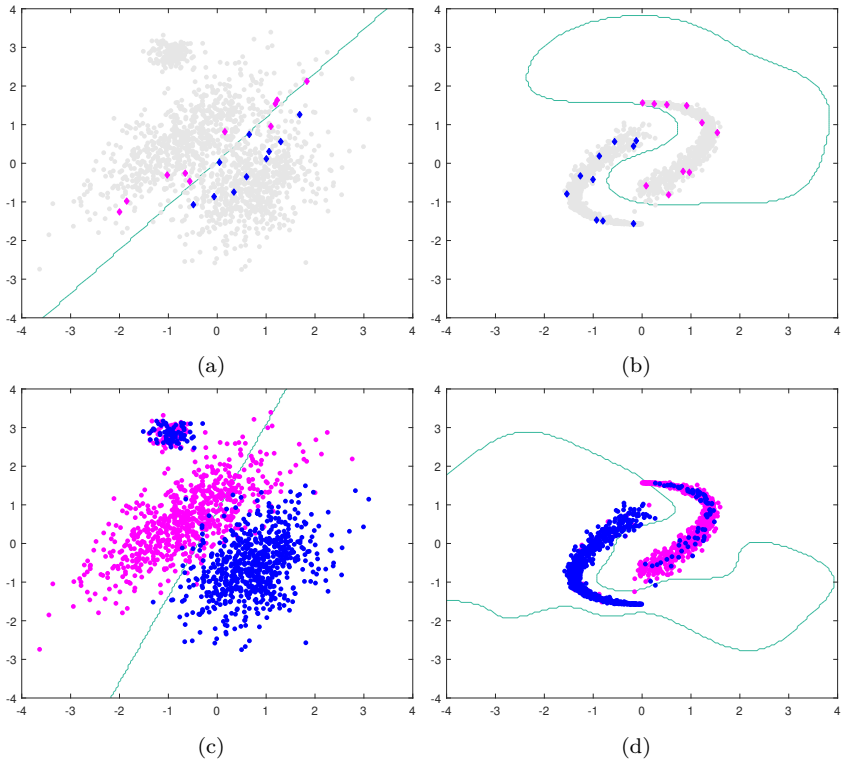

Figure 4: The final solution after applying RoS-LSSVM on the toy example is visible on Figures a and b. A robust classification is achieved, this while reducing the number of support vector to a bare minimum. The opposite effect is visible for the normal LS-SVM classification on Figure c and d.

4. Experiments

4.1. Simulation results

4.1.1. Linear example

Two Gaussians close, the reverse labels behind at large distance. See Robustified LS-SVM [3]

4.1.2. Non-Linear example

Yin-Yang, Two spiral dataset or Cross Dataset [25]

4.1.3. UCI

Robust: Banana, Celveland heart, Glass, Heartstatlog, liver disorder, monk PIMA, ripley, Transfusion, Vehicle [25]

Robust + sparse: Splice, Pendigits (choose two digits 3 vs 4), Satimage (1 vs 6), USPS (1 vs 2), Mushrooms, Protein (1 vs 2).[1] Outliers = 30 % samples that where far away from decision hyperplane, then randomly sample 1/3 of them and flip labels. datasets are in https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

Robust: Wine dataset + Linear, Glass, Biscuit Dough, Alon colon cancer, Hepatocellular carcinoma dataset [3]. However mostly linear

4.2. Industrial data results

5. Conclusions and future work

We proposed a new, improved LS-SVM model, coined RoS-LSSVM, inspired by the [WHAT, kernel C-steps, DPP, sampling, info transfer]-principle. We compared the performance of our prototype with other models by means of simulations for small, moderate and large UCI data sets. We found that our classifier model performed better than other models in terms of outlier robustness and classification accuracy.

Further research should go to the extension of sparsity in the classifier training phase, as well as [???]. It would also be worthwhile to integrate an optimization routine to estimate the optimal h-subset size for a given problem, as no theoretical results can predict the contamination degree before the actual processing yet.

Acknowledgements

We thank Johan Speybrouck for providing us the industrial datasets, and Tim Wynants, Doug Reid for their 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] Li Chen and Shuisheng Zhou. Sparse algorithm for robust lssvm in primal space. *Neurocomputing*, 275:2880–2891, 2018.

[2] Michiel Debruyne, Mia Hubert, and Johan Van Horebeek. Detecting influential observations in kernel pca. *Computational Statistics & Data Analysis*, 54(12):3007–3019, 2010.

[3] Michiel Debruyne, Sven Serneels, and Tim Verdonck. Robustified least squares support vector classification. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 23(9):479–486, 2009.

[4] Marcelo Espinoza, Johan AK Suykens, and Bart De Moor. Fixed-size least squares support vector machines: A large scale application in electrical load forecasting. *Computational Management Science*, 3(2):113–129, 2006.

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

[6] Ivan Goethals, Kristiaan Pelckmans, Johan AK Suykens, and Bart De Moor. Identification of mimo hammerstein models using least squares support vector machines. *Automatica*, 41(7):1263–1272, 2005.

[7] 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.

[8] Mia Hubert and Michiel Debruyne. Minimum covariance determinant. *Wiley interdisciplinary reviews: Computational statistics*, 2(1):36–43, 2010.

[9] Mia Hubert, Peter Rousseeuw, Dina Vanpaemel, and Tim Verdonck. The dets and detmm estimators for multivariate location and scatter. *Computational Statistics & Data Analysis*, 81:64–75, 2015.

[10] Mia Hubert, Peter J Rousseeuw, and Karlien Vanden Branden. Robpca: a new approach to robust principal component analysis. *Technometrics*, 47(1):64–79, 2005.

[11] Mia Hubert, Peter J Rousseeuw, and Tim Verdonck. A deterministic algorithm for robust location and scatter. *Journal of Computational and Graphical Statistics*, 21(3):618–637, 2012.

[12] 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.

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

[14] Patric Nader, Paul Honeine, and Pierre Beauseroy. Mahalanobis-based one-class classification. In *2014 IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2014.

[15] Peter J Rousseeuw. Multivariate estimation with high breakdown point. *Mathematical statistics and applications*, 8(283-297):37, 1985.

[16] Peter J Rousseeuw and Katrien Van Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223, 1999.

[17] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[18] Johan AK Suykens, Jos De Brabanter, Lukas Lukas, and Joos Vandewalle. Weighted least squares support vector machines: robustness and sparse approximation. *Neurocomputing*, 48(1-4):85–105, 2002.

[19] 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.

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

[21] Johan AK Suykens, Joos Vandewalle, and Bart De Moor. Optimal control by least squares support vector machines. *Neural networks*, 14(1):23–35, 2001.

[22] Shaohui Tao, Dezhao Chen, and Weixiang Zhao. Fast pruning algorithm for multi-output ls-svm and its application in chemical pattern classification. *Chemometrics and intelligent Laboratory systems*, 96(1):63–69, 2009.

[23] Tony Van Gestel, Johan AK Suykens, D-E Baestaens, Annemie Lambrechts, Gert Lanckriet, Bruno Vandaele, Bart De Moor, and Joos Vandewalle. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Transactions on neural networks*, 12(4):809–821, 2001.

[24] Lixia Yang, Shuyuan Yang, Rui Zhang, and Honghong Jin. Sparse least square support vector machine via coupled compressive pruning. *Neurocomputing*, 131:77–86, 2014.

[25] Xiaowei Yang, Liangjun Tan, and Lifang He. A robust least squares support vector machine for regression and classification with noise. *Neurocomputing*, 140:41–52, 2014.