Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 31 (2014) 639 – 647

Procedia Computer Science

2nd International Conference on Information Technology and Quantitative Management, ITQM 2014

Weighted linear loss support vector machine for large scale problems

Yuan-Hai Shao[a], Zhen Wang[b], Zhi-Min Yang[a,*], Nai-Yang Deng[c,**]

[a]*Zhijiang College, Zhejiang University of Technology, Hangzhou, 310024, P.R.China*
[b]*College of Mathematics, Jilin University, Changchun, 130012, P.R.China*
[c]*College of Science, China Agricultural University, Beijing, 100083, P.R.China*

Abstract

In this paper, instead of using the Hinge loss in standard support vector machine, we introduce a weighted linear loss function and propose a weighted linear loss support vector machine (WLSVM) for large scale problems. The main characteristics of our WLSVM are: (1) by adding the weights on linear loss, the training points in the different positions are proposed to give different penalties, avoiding over-fitting to a certain extent and yielding better generalization ability than linear loss. (2) by only computing very simple mathematical expressions to obtain the separating hyperplane, the large scale problems can be easy dealt. All experiments on synthetic and real data sets show that our WLSVM is comparable to SVM and LS-SVM in classification accuracy but with needs computation time, especially for large scale problems.

Keywords: Pattern recognition, support vector machines, linear loss, weighed coefficient, large scale problems.

1. Main Text

Support vector machine (SVM) is an excellent kernel-based tool for pattern recognition[1,2]. It was introduced by Vapnik and co-workers[1] based on the statistical learning theory. The SVM has outperformed most other systems in a wide variety of applications, including a wide spectrum of research areas ranging from text categorization[3], biomedicine[4,5], and financial regression[6] etc. The standard linear support vector classification is the earliest SVM model. Its primal problem can be understood in the following way: construct two parallel hyperplanes such that, on the one hand, the band between the two parallel hyperplanes separates the two classes (the positive and negative data points) well; on the other hand, the width between the two hyperplanes is maximized. This leads to introduce the regularization term and implement the structural risk minimization principle. The final separating hyperlane is selected to be the "middle one" between the above two hyperplanes.

Though SVM owns better generalization performance compared with many other machine learning methods, the training stage involves the solution of a quadratic programming problem (QPP). In fact its computational complexity is

* Corresponding author.
** Corresponding author. Tel.: +086-010-62736265; fax: +086-010-62736265
E-mail addresses: yzm9966@126.com (Zhi-Min Yang)., dengnaiyang@cau.edu.cn (Nai-Yang Deng).

$O(m^3)$, where m is the total size of training data points. This drawback restricts the application to large scale problems. To address this problem, so far, many improved algorithms have been proposed, e.g. Chunking[7], SMO[8], SVMLight[9], Libsvm[10] and Liblinear[11]. They aim at fast resolving the optimization problem by optimizing a small subset of the variables in the dual during the iteration procedure. On the other hand, a number of new SVM models[12,13] have been presented in recent years. For example, least squares SVM (LS-SVM)[12] replaces the QPP in SVM with a linear system by using a squared loss function instead of the Hinge one, resulting in a fast training speed.

As we know, the linear loss may be fail for constructing the classifier because it's value will lower than 0. In this paper, we improve the linear loss function to weighted linear loss function, which has the similar properties of Hinge loss function. Then, a weighted linear loss support vector machine (WLSVM) for large scale problems is proposed. The main contributions of this paper are: (1) by using the merit of the linear loss, which makes the training points of each class are far from separating hyperplane. (2) by adding the weights, the training points in the different positions are proposed to give different penalties, avoiding over-fitting to a certain extent and yielding better generalization ability. (3) by only computing very simple mathematical expressions to obtain the separating hyperplane, the large scale problems can be easy dealt. The experiments on synthetic and real data sets show that our WLSVM is comparable to SVM and LS-SVM in classification accuracy but needs less computation time, especially for large scale problems.

This paper is organized as follows. Section 2 introduces the related works. Section 3 proposes our WLSVM and depicts its properties and analyzes its generalization performance. Experiment results are shown in Section 4. Finally we conclude in Section 5.

2. Background

For binary classification problems, we consider the training set $T = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$, where x_i is a training input in space R^n and $y_i \in \{-1, +1\}$ is its corresponding class label, $i = 1, ..., m$.

2.1. Support Vector Machines

Linear support vector machine (SVM)[1,2,14] for classification problem searches for a separating hyperplane

$$f(x) = w^\top x + b = 0, \tag{1}$$

where $w \in R^n$ and $b \in R$. To measure the empirical risk, the soft margin loss function $\sum_{i=1}^m \max(0, 1 - y_i(w^\top x_i + b))$ is used. Figure 1 (a) shows this loss function as an example. By introducing the regularization term $\frac{1}{2}||w||^2$ and the slack variable $\xi = (\xi_1, ..., \xi_m)$, the primal problem of SVM can be expressed as

$$\begin{aligned}
&\min_{w,b,\xi} \frac{1}{2}||w||^2 + C\sum_{i=1}^m \xi_i,\\
&\text{s.t. } y_i(w^\top x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, ..., m,
\end{aligned} \tag{2}$$

where $C > 0$ is a parameter. Note that the minimization of the regularization term $\frac{1}{2}||w||^2$ is equivalent to the maximization of the margin between two parallel supporting hyperplanes $w^\top x + b = 1$ and $w^\top x + b = -1$. And the structural risk minimization principle is implemented in this problem. When we obtain the optimal solution of (2), a new data point is classified as $+1$ or -1 according to whether the decision function, Class $i = \text{sgn}(w^\top x + b)$, yields 1 or 0 respectively.

In practice, rather than solving (2), we solve its dual problem to get the appropriate soft (or hard) margin classifier. The case of nonlinear kernels is handled on lines similar to linear kernels.

2.2. Least squares SVM

Similar to SVM, linear least squares support vector machine (LS-SVM)[12] also searches for a separating hyperplane (1). To measure the empirical risk, the quadratic loss function $\sum_{i=1}^m (1 - y_i(w^\top x_i + b))^2$ is used. Figure 1 (b) shows this

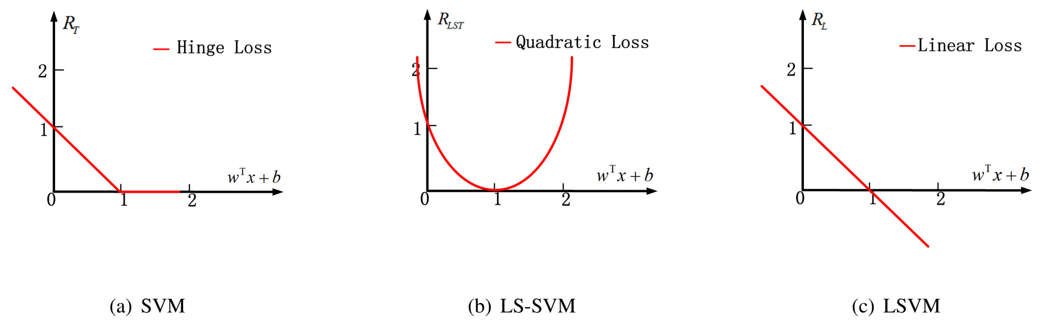

Fig. 1. (a): Hinge loss. (b): Quadratic loss. (b): Linear loss.

loss function as an example. Also, by introducing the regularization term $\frac{1}{2}||w||^2$ and the slack variable ξ_i, the primal problem of LS-SVM can be expressed as

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}||w||^2 + \frac{C}{2}\sum_{i=1}^{m}\xi_i^2, \\ &\text{s.t. } y_i(w^{\top}x_i + b) = 1 - \xi_i, i = 1, ..., m, \end{aligned} \tag{3}$$

where $C > 0$ is a parameter. Similar to SVM, the minimization of the regularization term $\frac{1}{2}||w||^2$ is equivalent to the maximization of the margin between two parallel proximal hyperplanes $w^{\top}x+b = 1$ and $w^{\top}x+b = -1$. The solutions of (3) can be obtained by solving a linear equations. When we obtain the optimal solution of (3), a new data point is classified the same with SVM.

3. Weighted linear loss support vector machine

3.1. Linear WLSVM

In this section, to classifier the training set efficiency, we replace the loss function $\sum_{i=1}^{m}\max(0, 1 - y_i(w^{\top}x_i + b))$ in SVM by the linear loss $\sum_{i=1}^{m}(1 - y_i(w^{\top}x_i + b))$. Figure 1 (c) shows this loss function as an example. Then we propose the linear loss support vector machine (LSVM). By introducing the regularization term $\frac{1}{2}(||w||^2 + b^2)$ similar to[13,15,16] and the slack variable ξ_i, the primal problem of our LSVM can be expressed as

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}(||w||^2 + b^2) + C\sum_{i=1}^{m}\xi_i, \\ &\text{s.t. } y_i(w^{\top}x_i + b) = 1 - \xi_i, i = 1, ..., m, \end{aligned} \tag{4}$$

where $C > 0$ is a regular parameter.

Let us compare the empirical risks of SVM, LS-SVM and LSVM. Obviously, they are different due to the different loss functions although they have the same aim to put the decision hyperplane farther from the each class. In fact, SVM uses soft margin loss function, trying to make the classes has at least one distance from the hyperplane, while LS-SVM uses the least squares loss function, trying to make the classes has one distance from the hyperplane. And LSVM uses the linear loss function, trying to make the classes far away from the hyperplane. However, compare with the SVM, both LS-SVM and LSVM loss the spares, and more vulnerable by outliers.

To balance the influence of each point to the hyperplane, following the notion of rough sets[17], we introduce the weighted factor v_i and propose the weighted linear loss support vector machine (WLSVM) by reformulating the (4)

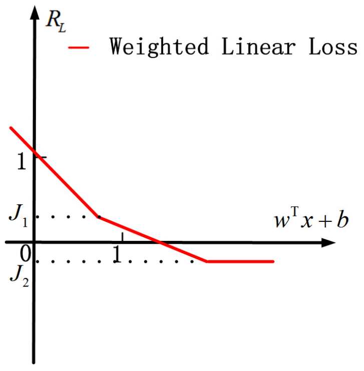

Fig. 2. The loss of WLSVM.

as

$$\min_{w,b,\xi} \frac{1}{2}(\|w\|^2 + b^2) + C\sum_{i=1}^{m} \nu_i \xi_i, \quad \text{s.t. } y_i(w^\top x_i + b) = 1 - \xi_i, i = 1, ..., m, \tag{5}$$

where ν_i is the weight parameter and determined by the following formula:

$$\nu_i = \begin{cases} 1 & if\ \xi_i \neq J_1, \\ \frac{J_1 - \xi_i}{J_1 - J_2} & if\ J_2 \leq \xi_i \leq J_1, \\ 10^{-4} & otherwise, \end{cases} \tag{6}$$

where $J_1 \geq 0$ and $J_2 \leq 0$ are parameters. When ν_i is fixed, to solve the problem (5), on substituting the equality constraints into the objective function, QPP (5) becomes:

$$L(w, b) = \tfrac{1}{2}(\|w\|^2 + b^2) + C\textstyle\sum_{i=1}^{m} \nu_i(1 - y_i(w^\top x_i + b)). \tag{7}$$

Setting the gradient of (7) with respect to w and b to zero, gives:

$$w - C\textstyle\sum_{i=1}^{m} \nu_i y_i x_i = 0, \implies w = C\sum_{i=1}^{m} \nu_i y_i x_i, \tag{8}$$

$$b - C\textstyle\sum_{i=1}^{m} \nu_i y_i = 0, \implies b = C\sum_{i=1}^{m} \nu_i y_i. \tag{9}$$

When we obtain the optimal solution of (5) from (8) and (9), similar to SVM, a new data point is classified as +1 or −1 according to whether the decision function, $y = \text{sgn}(w^\top x + b)$, yields 1 or 0 respectively.

In order to determine the weights, we consider the weight-setting method using in WLS-SVM[18] which similar to the formula (6). To verify the effectiveness of the proposed linear WLSVM and to give the proper $J_1 \geq 0$ and $J_2 \leq 0$, then we have the following algorithm

Training algorithm for linear WLSVM :

step 1: Given the training data $T = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$, set $\nu_i^1 = 1$, $i = 1, ..., m$. Compute weight vectors w^1 and b^1 from (8) and (9) with proper penalty parameters C, which are usually selected based on validation.

step 2: Calculate the slack values ξ_i^1 by w^1 and b^1 in (5), then obtain ν_i^* from (6), where $J_1 \geq 0$, $J_1 = |\xi_{+mean}/N_+ - \xi_{-mean}/N_-|$ and $J_2 \leq 0, J_2 = -|\xi_{-mean}/N_- - \xi_{+mean}/N_+|$.

step 3: Compute weight vectors w^* and b^* from (8) and (9) with the ν_i^* and the proper penalty parameters C.

step 4: Classify the new point x by $y = \text{sgn}(w^{*\top} x + b^*)$.

3.2. Nonlinear WLSVM

In order to extend the above results to the nonlinear case, similar to[2,19,20,21], we consider the following kernel-generated surfaces instead of hyperplanes

$$w^\top \varphi(x) + b = 0, \tag{10}$$

where function $\varphi(x)$ project the data point x into the feature space.

Similar to the linear cases, the above surface is obtained through the following QPP:

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}(\|w\|^2 + b^2) + C\sum_{i=1}^{m} \nu_i \xi_i, \\ &\text{s.t. } y_i(w^\top \varphi(x_i) + b) = 1 - \xi_i, i = 1, ..., m, \end{aligned} \tag{11}$$

where $C > 0$ is a regular parameter.

To solve the problem (11), on substituting the equality constraints into the objective function, QPP (11) becomes:

$$L(w, b) = \tfrac{1}{2}(\|w\|^2 + b^2) + C\textstyle\sum_{i=1}^{m} \nu_i(1 - y_i(w^\top \varphi(x_i) + b)). \tag{12}$$

Setting the gradient of (12) with respect to w and b to zero, gives:

$$w - C\textstyle\sum_{i=1}^{m} \nu_i y_i \varphi(x_i) = 0, \implies w = C\sum_{i=1}^{m} \nu_i y_i \varphi(x_i), \tag{13}$$

$$b - C\textstyle\sum_{i=1}^{m} \nu_i y_i = 0, \implies b = C\sum_{i=1}^{m} \nu_i y_i. \tag{14}$$

When we obtain the optimal solution of (11) from (13) and (14), a new data point is classified as +1 or −1 according to whether the decision function, $y = \text{sgn}(w^\top \varphi(x) + b)$, yields 1 or 0 respectively. To verify the effectiveness of the proposed nonlinear WLSVM, we have the following algorithm

Training algorithm for nonlinear WLSVM :

step 1: Given the training data $T = \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$, set $v_i^1 = 1$, $i = 1, ..., m$. Compute weight vectors w^1 and b^1 from (13) and (14) with proper penalty parameters C and kernel function, which are usually selected based on validation.

step 2: Calculate the slack values ξ_i^1 by w^1 and b^1 in (11), then obtain v_i^* from (6), where $J_1 \geq 0$, $J_1 = |\xi_{+mean}/N_+ - \xi_{-mean}/N_-|$ and $J_2 \leq 0, J_2 = -|\xi_{-mean}/N_- - \xi_{+mean}/N_+|$.

step 3: Compute weight vectors w^* and b^* from (13) and (14) with the v_i^* and the proper penalty parameters C and kernel function.

step 4: Classify the new point x by $y = \text{sgn}(w^{*\top} \varphi(x) + b^*)$.

4. Experimental results

In order to evaluate our WLSVM, we investigate its classification accuracies and computational efficiencies on eight UCI benchmark data sets[22] and David Musicant's NDC Data Generator[23] data sets. In experiments, we focus on the comparison between our WLSVM and two state-of-the-art classifiers, including SVM and LS-SVM. All the classifiers are implemented in MATLAB 7.0 environment on a PC with Intel P4 processor (2.9 GHz) with 1 GB RAM. In order to give as fast as training speed, the SVM is implemented by Libsvm and Liblinear, the LS-SVM is implemented by LibLSSVM[24]. LSVM and WLSVM is realized by simple MATLAB operations. The "Accuracy" is used to evaluate methods, which is defined as: $Accuracy = (TP + TN)/(TP + FP + TN + FN)$, where TP, TN, FP and FN are the number of true positive, true negative, false positive and false negative, respectively. As for the problem of selecting hyper-parameters, we employ standard 10-fold cross-validation technique. Furthermore, the parameters for the all methods are selected from the set $\{2^{-8}, ..., 2^7\}$.

Firstly, we experimented with the UCI benchmark datasets, and report the results of the five methods on the selected datasets in Table 1. We computed the means and standard errors of the 10-folds and CPU time for training for each

Table 1. Ten-fold testing percentage test set accuracy (%) on UCI data sets for linear classifiers.

Datasets	SVM(Libsvm) Acc % Time (s)	LS-SVM Acc % Time (s)	SVM(Liblinear) Acc % Time (s)	LSVM Acc % Time (s)	WLSVM Acc % Time (s)
Hepatitis	**84.27±1.77**	82.72±1.46	84.45±0.98	81.24±1.37	82.32±1.30
155×19	0.0012	0.0222	0.0007	0.0001	0.0002
Heartstat-log	84.05±0.61	**84.88±0.47**	83.07±0.30	63.79±0.63	83.44±0.74
270×13	0.0031	0.0624	0.0005	0.0002	0.0005
Sonar	77.91±1.25	**78.68±1.64**	72.07±1.47	55.14±1.38	57.55±2.03
208×60	0.0119	0.0415	0.0086	0.0008	0.0009
WPBC	79.39±1.51	77.23±1.36	79.64±1.19	73.11±1.97	75.76±0.01
569×30	0.0228	0.1002	0.0023	0.0004	0.0007
Australian	**86.46±0.34**	85.87± 0.22	86.18±0.24	68.42±0.40	76.91±0.90
690×14	0.0422	0.1754	0.0022	0.0009	0.0013
German	**75.21±0.37**	74.52±0.33	75.13±0.33	70.00±0.00	70.10±0.01
1000×20	0.0923	0.4937	0.0031	0.0011	0.0020
Diabetics	77.22±0.17	76.25±0.51	**77.34±0.25**	66.13±0.32	70.10±0.53
768×8	0.0499	0.2621	0.0010	0.0006	0.0014
CMC	73.42±0.10	71.39±0.40	76.09±0.13	70.26±0.350	**76.71±0.04**
1473×9	0.0112	1.0360	0.0028	0.0013	0.0021

classifier. Table 1 shows that the generalization capability of our WLSVM are better than LSVM, but weak lower than SVM and LS-SVM on many of the datasets considered. It also reveals that our WLSVM, whose solution is obtained by computing very simple mathematical expressions, performs comparable to LS-SVM and SVM. We note that, on the many data sets, the performance difference between our WLSVM and the other classifiers is insignificant, where our WLSVM is significantly better than LSVM. It can be clearly seen that the WLSVM obtains the comparable classification performance than the LSVM for the most cases. From Table 1, we also can see that our WLSVM and LSVM are fastest in all of others, while WLSVM is bit slower than LSVM in terms of training time.

Table 2. Ten-fold testing percentage test set accuracy (%) on UCI data sets for nonlinear classifiers.

Datasets	SVM(Libsvm) Acc % Time (s)	LS-SVM Acc % Time (s)	LSVM Acc % Time (s)	WLSVM Acc % Time (s)
Heartstat-log	84.05±0.61	**84.08±0.32**	67.26±1.52	79.64±1.52
270×13	0.0031	0.0015	0.0019	0.0038
Sonar	**88.93±1.59**	86.57±1.05	65.91±1.49	75.57±1.33
208×60	0.0662	0.0038	0.0059	0.0072
WPBC	**79.75±1.10**	80.76±1.35	72.22±1.72	77.96±1.55
569×30	0.0471	0.0024	0.0026	0.0172
German	81.71±4.52	76.62±0.30	70.00±0.00	**81.84±1.05**
1000×20	0.3765	0.4937	0.0136	0.0198
Diabetics	**80.95±8.38**	76.82±0.37	65.57±1.07	75.26±0.59
768×8	0.2623	0.0776	0.0064	0.0146

Table 2 is concerned with kernel SVM, LS-SVM, LSVM and WLSVM. The Gaussian kernel $K(x, x') = e^{-\mu\|x-x'\|^2}$ is used. The kernel parameter μ is obtained through searching from the range 2^{-10} to 2^5. The training CPU time is also listed. The results in Table 2 are similar with that appeared in Table 1, and therefore confirm the above conclusion further.

4.1. NDC data sets

We also conducted experiments on large datasets, generated using David Musicants NDC Data Generator[23] to get a clear picture of how the computing time of all these methods scale with respect to number of data points. The NDC datasets are divided into a training set and prediction set. We report the training accuracy and prediction accuracy, respectively. For experiments with NDC datasets, we fixed penalty parameters of all algorithms to be the same (i.e., $C = 1$,$c_1 = 1$, $c_2 = 1$, $\nu_1 = 0.5$, $\nu_2 = 0.5$).

Table 3 shows the comparison of computing time and accuracy for linear SVM, LS-SVM, LSVM and our WLSVM on NDC data sets. For almost same accuracy, WLSVM and LSVM performed several orders of magnitude faster than SVM and LS-SVM on all datasets. It is also worth mentioning that our WLSVM and LSVM does not require any special optimizers. For large datasets (more than ten thousand training points), we only report the results on LS-SVM, Liblinear, LSVM and WLSVM, it is because the Libsvm may ran out of memory. From Table 3, we can see that WLSVM train the classifier with 1 million data in 1.6018s which is faster than the training time 6.85s as required by LS-SVM. In addition, LSVM and our WLSVM also can training 2 million data in PC. The results undoubtedly prove the boost of computational efficiency of our WLSVM over SVM and LS-SVM.

Table 3. Comparison on NDC data sets for linear classifiers.

Datasets	SVM(Libsvm)
	Train (%)
	Test (%)
	Time(s)
		LS-SVM
		Train (%)
		Test (%)
		Time(s)
			SVM(Liblinear)
			Train (%)
			Test (%)
			Time(s)
				LSVM
				Train (%)
				Test (%)
				Time(s)
					WLSVM
					Train (%)
					Test (%)
					Time(s)

NDC-3k	83.73 81.42 0.4267	84.86 83.42 0.0037	83.48 82.04 0.0878	77.34 73.78 0.0025	82.22 80.32 0.0025
NDC-5k	81.87 79.93 1.2534	82.82 81.37 0.0072	81.69 80.34 0.0087	72.46 72.07 0.0041	80.79 79.62 0.1461
NDC-10k	84.29 83.57 7.8161	86.85 83.69 0.0132	84.52 84.18 0.3288	84.15 80.19 0.0177	84.63 82.45 0.0084
NDC-50k	* * *	85.48 80.97 0.06973	86.54 80.35 1.3835	75.12 74.32 0.0425	82.06 78.65 0.0929
NDC-100k	* * *	80.95 79.46 0.1277	81.21 80.79 2.4775	78.45 76.48 0.0826	79.42 79.70 0.1944
NDC-1m	* * *	84.71 79.86 1.0716	* * *	80.32 77.42 0.8395	82.45 80.15 1.6018
NDC-2m	* * *	* * *	* * *	77.55 74.69 1.6018	81.02 80.82 1.7002

* Experiments ran out of memory.

5. Conclusions

In this paper, we have proposed a weighted linear loss support vector machine (WLSVM) by introducing the weighted linear loss function. By using the weighted linear loss, the separating hyperplane is obtained by only computing simple mathematical expressions, and the large scale problems can be dealt easily. In addition, as the weighted loss has the similar properties of Hinge loss function, which improved the generalization ability of linear loss SVM. The experiments on synthetic and real data sets shown that our WLSVM is suitable for handling large scale problems. However, the weights setting is important for WLSVM, our future works including to construct better weighting function and improve the generalization ability further. It is also interesting to extend our WLSVM to other pattern recognition problems, such as[25,26,27].

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Nos.11201426, 10971223 and 11371365), the Zhejiang Provincial Natural Science Foundation of China (Nos.LQ12A01020 and LQ13F030010) and the Ministry of Education, Humanities and Social Sciences Research Project of China (No.13YJC910011).

References

1. Boser, B., Guyon, I., Vapnik, V.. A training algorithm for optimal margin classifiers. In: *Proceedings of the fifth annual workshop on Computational learning theory*. ACM; 1992, p. 144–152.
2. Deng, N., Tian, Y., Zhang, C.. *Support Vector Machines: Theory, Algorithms, and Extensions*. CRC Press; 2012.
3. Hao, P., Chiang, J., Tu, Y.. Hierarchically svm classification based on support vector clustering method and its application to document categorization. *Expert Systems with Applications* 2007;**33**(3):627–635.
4. Li, Y., Shao, Y., Jing, L., Deng, N.. An efficient support vector machine approach for identifying protein s-nitrosylation sites. *Protein and Peptide Letters* 2011;**18**(6):573–587.
5. Li, Y., Shao, Y., Deng, N.. Improved prediction of palmitoylation sitesusing pwms and svm. *Protein and Peptide Letters* 2011;**18**(2):186–193.
6. Ince, H., Trafalis, T.B.. Support vector machine for regression and applications to financial forecasting. In: *International Joint Conference on Neural Networks*. Italy; 2002, .
7. Vapnik, V.. *The nature of statistical learning theory*. Springer-Verlag New York Inc; 2000.
8. Platt, J.. Fast training of support vector machines using sequential minimal optimization. In: *Advances in kernel methods-support vector learning*. Cambridge, MA: MIT Press; 1999, p. 185–2008.
9. Joachims, T.. Making large-scale svm learning practical. *Advances in Kernel Methods Support Vector Learning* 1999;:169–184.
10. Chang, C.C., Lin, C.J.. LIBSVM*: A library for support vector machines*. `http://www.csie.ntu.edu.tw/ cjlin`; 2001.
11. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.. LIBLINEAR: a library for large linear classification. *Journal of Machine Learning Research* 2008;**9**:1871–1874.
12. Suykens, J.A.K., Vandewalle, J.. Least squares support vector machine classifiers. *Neural Process Letter* 1999;**9**(3):293–300.
13. Fung, G., Mangasarian, O.L.. Proximal support vector machine classifiers. In: *Proceedings of seventh international conference on knowledge and data discovery*. San Francisco; 2001, p. 77–86.
14. Tian, Y., Shi, Y., Liu, X.. Recent advances on support vector machines research. *Technological and Economic Development of Economy* 2012;**18**(1):5–33.
15. Shao, Y., Zhang, C., Wang, X., Deng, N.. Improvements on twin support vector machines. *IEEE Transactions on Neural Networks,* 2011; **22**(6):962–968.
16. Shao, Y.H., Chen, W.J., Deng, N.Y.. Nonparallel hyperplane support vector machine for binary classification problems. *Information Sciences* 2014;**263**(0):22–35.
17. Pawlak, Z.. Rough sets. *International Journal of Parallel Programming* 1982;**11**:341–356.
18. J.A.K.Suykens, , Brabanter, J., L.Lukas, , J.Vandewalle, . Weighted least squares support vector machines:robustness and sparse approximation. *Neurocomputing* 2002;**48**:85–105.
19. Shao, Y., Deng, N.. A coordinate descent margin based-twin support vector machine for classification. *Neural networks* 2012;**25**:114–121.
20. Shao, Y.H., Wang, Z., Chen, W.J., Deng, N.Y.. A regularization for the projection twin support vector machine. *Knowledge-Based Systems* 2013;**37**:203–210.
21. Qi, Z., Tian, Y., Shi, Y.. Structural twin support vector machine for classification. *Knowledge-Based Systems* 2013;**43**:74–81.
22. Blake, C.L., Merz, C.J.. *UCI Repository for Machine Learning Databases*. `http://www.ics.uci.edu/ mlearn/MLRepository.html`; 1998.
23. Musicant, D.R.. *NDC: Normally Distributed Clustered Datasets*. `http://www.cs.wisc.edu/dmi/svm/ndc/`; 1998.
24. Pelckmans, K., Van Gestel, T., De Brabanter, J., Lukas, L., Hamers, B., De Moor, B.. *LS-SVMlab: a MATLAB/C toolbox for Least Squares Support Vector Machines*. `http://www.esat.kuleuven.ac.be/sista/lssvmlab`; 2008.
25. Qi, Z., Tian, Y., Shi, Y.. Twin support vector machine with universum data. *Neural Networks* 2012;**36**:112–119.

26. Qi, Z., Tian, Y., Shi, Y.. Robust twin support vector machine for pattern classification. *Pattern Recognition* 2013;**46**(1):305–316.
27. Qi, Z., Tian, Y., Shi, Y.. Laplacian twin support vector machine for semi-supervised classification. *Neural Networks* 2012;**35**:46–53.