






# Feature Selection Using Sparse Twin Support Vector Machine with Correntropy-Induced Loss

Xiaohan Zheng<sup>1</sup> , Li Zhang<sup>1,2</sup>  , and Leilei Yan<sup>1</sup> 

<sup>1</sup> School of Computer Science and Technology, Joint International Research Laboratory of Machine Learning and Neuromorphic Computing, Soochow University, Jiangsu 215006, Suzhou, China

{20184227056,20184227032}@stu.suda.edu.cn, zhangliml@suda.edu.cn

<sup>2</sup> Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Jiangsu 215006, Suzhou, China

**Abstract.** Twin support vector machine (TSVM) has been widely applied to classification problems. But TSVM is sensitive to outliers and is not efficient enough to realize feature selection. To overcome the shortcomings of TSVM, we propose a novel sparse twin support vector machine with the correntropy-induced loss (C-ST SVM), which is inspired by the robustness of the correntropy-induced loss and the sparsity of the  $\ell_1$ -norm regularization. The objective function of C-ST SVM includes the correntropy-induced loss that replaces the hinge loss, and the  $\ell_1$ -norm regularization that can make the decision model sparse to realize feature selection. Experiments on real-world datasets with label noise and noise features demonstrate the effectiveness of C-ST SVM in classification accuracy and confirm the above conclusion further.

**Keywords:** Twin support vector machine · Feature selection · Sparsity · Correntropy-induced loss

## 1 Introduction

Recently, feature selection has been a hot area of research to address the curse of dimensionality. Feature selection refers to select an optimal subset of original features, which retains valuable features and eliminates redundant features [14]. This procedure can reduce the complexity of processing data and improve the prediction performance [2]. Many methods about feature selection have been proposed, which can be broadly classified into three types: filter, wrapper, and embedded methods [6, 17, 29]. In this paper, we focus on feature selection using support vector machine (SVM) that is the most representative one of embedded methods.

---

This work was supported in part by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China under Grant No. 19KJA550002, by the Six Talent Peak Project of Jiangsu Province of China under Grant No. XYDXX-054, by the Priority Academic Program Development of Jiangsu Higher Education Institutions, and by the Collaborative Innovation Center of Novel Software Technology and Industrialization.

© Springer Nature Switzerland AG 2020

G. Li et al. (Eds.): KSEM 2020, LNAI 12274, pp. 434–445, 2020.

[https://doi.org/10.1007/978-3-030-55130-8\\_38](https://doi.org/10.1007/978-3-030-55130-8_38)

SVM, based on the principle of structural risk minimization and the theory of VC dimension, has been used to solve classification and regression problems and has a broad variety of application in real-world tasks [1, 3, 13, 16, 19, 24, 27]. For a binary classification problems, SVM aims at seeking a separating hyperplane to maximize the margin between positive and negative samples, which has excellent generalization performance [10]. Unfortunately, SVM has a high-computational complexity because it needs to solve an entire quadratic programming problem (QPP).

Lately, twin support vector machine (TSVM) has been proposed inspired by the idea of SVM [15]. Compared to SVM, TSVM attempts to find two hyperplanes by solving a pair of smaller QPPs for a binary classification task. Thus, TSVM works faster than SVM in theory and becomes a popular classifier. Many variants of TSVM have been proposed, such as twin bounded support vector machine (TBSVM) [20], least squares twin support vector machine (LSTSVMS) [18],  $v$ -projection twin support vector machine ( $v$ -PTSVM) [9], locality preserving projection least squares twin support vector machine (LPPLSTSVMS) [8], and new fuzzy twin support vector machine (NFTSVMS) [7]. However, these methods do not have sparse representative models or cannot implement feature selection.

In order to improve the feature selection ability or the sparseness performance of TSVM-like methods, many scholars have proposed same improved methods. For example,  $\ell_p$ -norm least square twin support vector machine ( $\ell_p$ -LSTSVMS) was proposed by Zhang et al. [30], which can realize feature selection by introducing an adaptive learning procedure with the  $\ell_p$ -norm ( $0 < p < 1$ ). Sparse non-parallel support vector machine (SNSVM) was proposed in [23]. By replacing the hinge loss with both the  $\epsilon$ -insensitive quadratic loss and the soft margin quadratic loss, SNSVM has better sparseness performance than TSVM. Tanveer [22] proposed a new linear programming twin support vector machines (NLPTSVM) that uses the  $\ell_1$ -norm regularization, distance and loss term, which causes the robustness and sparsity of NLPTSVM.

In real applications, data often contains noises or outliers, which would have a negative effect on the generalization performance of the learned model. To remedy it, Xu et al. [26] proposed a novel twin support vector machine (Pin-TSVM) inspired by the pinball loss. Pin-TSVM has favorable noise insensitivity, but it does not consider the sparseness of model. That is to say, the feature selection performance of Pin-TSVM is poor.

In this paper, we propose a novel sparse twin support vector machine with the correntropy-induced loss, called C-STSVMS. The correntropy-induced loss function is a smooth, nonconvex and bounded loss, which was designed for both classification and regression tasks in [21]. For a classification task, the correntropy-induced loss measures the similarity between the prediction value and the true value from the perspective of correntropy. In addition, we know that the  $\ell_1$ -norm regularization can induce a sparse model from [28] and [31]. C-STSVMS is to minimize three terms: the distance term, the correntropy-induced loss term and the  $\ell_1$ -norm regularization term. In doing so, we expect that C-STSVMS have the ability to perform feature selection and the robustness to outliers. To find the

solution to C-STSV in the primal space, we design an alternating iterative method with the help of the half-quadratic (HQ) optimization. Experimental results verify the validity of these theories.

The paper is organized as follows. Section 2 dwells on the proposed C-STSV, including the introduction of the correntropy-induced loss and the description of the optimization problems and the solutions of C-STSV. Section 3 discusses experimental results. The last section contains the conclusions.

## 2 Sparse Twin Support Vector Machine with Correntropy-Induced Loss

In this paper, we consider a binary classification problem and define a normal training set  $X = \{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_{n_1}, y_1), (\mathbf{x}_{n_1+1}, y_2), \dots, (\mathbf{x}_{n_1+n_2}, y_2)\}$  where  $y_1 = 1$  and  $y_2 = -1$  are the positive and negative labels, respectively. Let  $\mathbf{X}_1 = [\mathbf{x}_1, \dots, \mathbf{x}_{n_1}]^T \in \mathbb{R}^{n_1 \times m}$  be the positive sample matrix,  $\mathbf{X}_2 = [\mathbf{x}_{n_1+1}, \dots, \mathbf{x}_{n_1+n_2}] \in \mathbb{R}^{n_2 \times m}$  be the negative sample matrix,  $\mathbf{e}_1$  and  $\mathbf{e}_2$  be vectors of all ones with appropriate size.

### 2.1 Correntropy-Induced Loss

For any sample  $(\mathbf{x}_i, y_i)$  in the given training set  $X$ , the predicted value of  $\mathbf{x}_i$  is defined as  $f(\mathbf{x}_i)$ . A loss function can be used to measure the difference between the predicted value  $f(\mathbf{x}_i)$  and the true value  $y_i$ . Different loss functions would result in various learners.

Here, we introduce the correntropy-induced loss function. Correntropy is a nonlinear measure of the similarity between two random variables. Inspired by that, Singh [21] presented the correntropy-induced loss function that is to maximize the similarity between the predicted values and the true values for classification tasks. The correntropy-induced loss function has the form:

$$L_C(y_i f(\mathbf{x}_i)) = \beta \left[ 1 - \exp\left(-\frac{(1 - y_i f(\mathbf{x}_i))^2}{2\sigma^2}\right) \right] \quad (1)$$

where  $\beta = [1 - \exp(-\frac{1}{2\sigma^2})]^{-1}$  and  $\sigma > 0$  is the parameter. Note that  $L_C(0) = 1$  when  $y_i f(\mathbf{x}_i) = 0$ .

As an mean square error in reproducing kernel Hilbert space, the correntropy-induced loss can further approximate a transition from like the  $\ell_2$ -norm to like the  $\ell_0$ -norm. The curve of the correntropy-induced loss is shown in Fig. 1. We can see that the correntropy-induced loss is smooth, non-convex and bounded. For its boundedness, the correntropy-induced loss is robust to outliers.

### 2.2 Optimization Problems

Similar to TSVM [15], the aim of C-STSBM is to seek two optimal hyperplanes: positive and negative ones, where the positive hyperplane is closer to the positive

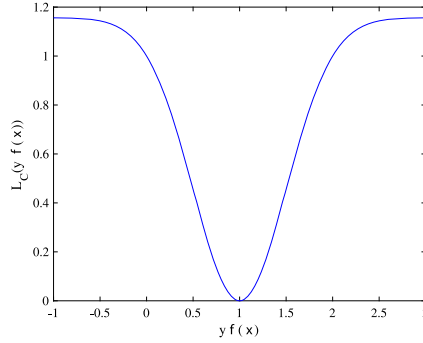


Fig. 1. Curve of correntropy-induced loss with  $\sigma = 0.5$ .

samples and as far as possible from the negative samples, and the same goes for the negative hyperplane. The two optimal hyperplanes are defined by the following discrimination functions:

$$\begin{cases} f_1(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_1 + b_1 \\ f_2(\mathbf{x}) = \mathbf{x}^T \mathbf{w}_2 + b_2 \end{cases} \quad (2)$$

where  $\mathbf{w}_1$  and  $\mathbf{w}_2$  are the weight vectors for positive and negative classes, respectively,  $b_1$  and  $b_2$  are the thresholds for two classes, respectively. To represent these weight vectors and thresholds, we construct four non-negative vectors and four non-negative variables, which is  $\beta_+^*, \beta_+, \beta_-^*, \beta_-, \gamma_+^*, \gamma_+, \gamma_-^*$  and  $\gamma_-$ , and let

$$\begin{cases} \mathbf{w}_1 = \beta_+^* - \beta_+ \quad \text{with } \beta_+^* \geq 0 \text{ and } \beta_+ \geq 0 \\ \mathbf{w}_2 = \beta_-^* - \beta_- \quad \text{with } \beta_-^* \geq 0 \text{ and } \beta_- \geq 0 \\ b_1 = \gamma_+^* - \gamma_+ \quad \text{with } \gamma_+^* \geq 0 \text{ and } \gamma_+ \geq 0 \\ b_2 = \gamma_-^* - \gamma_- \quad \text{with } \gamma_-^* \geq 0 \text{ and } \gamma_- \geq 0 \end{cases} \quad (3)$$

Then, (2) can be rewritten as

$$\begin{cases} f_1(\mathbf{x}) = \mathbf{x}^T (\beta_+^* - \beta_+) + (\gamma_+^* - \gamma_+) \\ f_2(\mathbf{x}) = \mathbf{x}^T (\beta_-^* - \beta_-) + (\gamma_-^* - \gamma_-) \end{cases} \quad (4)$$

C-STSVMS can be described as the following pair of QPPs:

$$\begin{cases} \min_{\beta_+^*, \beta_+, \gamma_+^*, \gamma_+} & \frac{1}{2} \|\mathbf{X}_1 (\beta_+^* - \beta_+) + (\gamma_+^* - \gamma_+)\|_2^2 \\ & + C_1 (\|\beta_+^*\|_1 + \|\beta_+\|_1 + \gamma_+^* + \gamma_+) + \frac{C_2}{n_2} \sum_{i=1}^{n_2} L_C(-f_1(\mathbf{x}_{2i})) \\ \min_{\beta_-^*, \beta_-, \gamma_-^*, \gamma_-} & \frac{1}{2} \|\mathbf{X}_2 (\beta_-^* - \beta_-) + (\gamma_-^* - \gamma_-)\|_2^2 \\ & + C_3 (\|\beta_-^*\|_1 + \|\beta_-\|_1 + \gamma_-^* + \gamma_-) + \frac{C_4}{n_1} \sum_{i=1}^{n_1} L_C(f_2(\mathbf{x}_{1i})) \end{cases} \quad (5)$$

where  $C_i > 0, i = 1, 2, 3, 4$  are parameters chosen a priori and  $\|\cdot\|_i, i = 1, 2$  is the  $\ell_i$ -norm. The first term of the first QPP in (5) is to minimize the distance

between the outputs of positive samples and the hyperplane  $f_1(\mathbf{x}) = 0$ . The second term is the  $\ell_1$ -norm regularization term that can reduce a sparseness solution of C-STSV. The third term is to minimize the sum of correntropy-induced loss function, which makes the negative samples as far as from the positive hyperplane. Note that QPPs in (5) are similar to each other in form. Thus, for the second QPP in (5), we have a similar explanation.

We denote the first two terms of QPPs in (5) by

$$\begin{cases} H_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+) \\ = \frac{1}{2} \|\mathbf{X}_1(\beta_+^* - \beta_+) + \mathbf{e}_1(\gamma_+^* - \gamma_+)\|_2^2 + C_1 (\|\beta_+^*\|_1 + \|\beta_+\|_1 + \gamma_+^* + \gamma_+) \\ H_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-) \\ = \frac{1}{2} \|\mathbf{X}_2(\beta_-^{*T} - \beta_-^T) + (\gamma_-^* - \gamma_-)\|_2^2 + C_3 (\|\beta_-^*\|_1 + \|\beta_-\|_1 + \gamma_-^* + \gamma_-) \end{cases} \quad (6)$$

The third term in the first QPP of (5) can be expressed as:

$$\frac{C_2}{n_2} \sum_{i=1}^{n_2} L_c(-f_1(\mathbf{x}_{2i})) = \frac{C_2}{n_2} \sum_{i=1}^{n_2} \left[ 1 - \exp\left(-\frac{(1 + (\beta_+^* - \beta_+)^T \mathbf{x}_{2i} + (\gamma_+^* - \gamma_+))^2}{2\sigma^2}\right) \right] \quad (7)$$

Let

$$L_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+) = \frac{C_2}{n_2} \sum_{i=1}^{n_2} \exp\left(-\frac{(1 + (\beta_+^* - \beta_+)^T \mathbf{x}_{2i} + (\gamma_+^* - \gamma_+))^2}{2\sigma^2}\right)$$

Then minimizing (7) is identical to maximizing  $L_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+)$ .

Thus, we can represent the first QPP in (5) as

$$\max_{\beta_+^*, \beta_+, \gamma_+^*, \gamma_+} -H_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+) + L_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+) \quad (8)$$

Similarly, the second QPP in (5) can be rewritten as follows:

$$\max_{\beta_-^*, \beta_-, \gamma_-^*, \gamma_-} -H_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-) + L_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-) \quad (9)$$

where

$$L_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-) = \frac{C_4}{n_1} \sum_{i=1}^{n_1} \exp\left(-\frac{(1 - (\beta_-^* - \beta_-)^T \mathbf{x}_{1i} - (\gamma_-^* - \gamma_-))^2}{2\sigma^2}\right).$$

### 2.3 Solutions

In this subsection, we turn the optimization problems (8) and (9) into two HQ optimization ones and use the alternating iterative method to find their solutions.

We first define two auxiliary variables  $\mathbf{v} = [v_1, \dots, v_{n_2}]^T$  and  $\mathbf{v}' = [v'_1, \dots, v'_{n_1}]^T$ , where  $v_i < 0$  and  $v'_i < 0$ , and then construct two convex functions

$$\begin{cases} G_1(\mathbf{v}) = \frac{C_2}{n_2} \sum_{i=1}^{n_2} g(v_i) = \frac{C_2}{n_2} \sum_{i=1}^{n_2} (-v_i \log(-v_i) + v_i) \\ G_2(\mathbf{v}') = \frac{C_4}{n_1} \sum_{i=1}^{n_1} g(v'_i) = \frac{C_4}{n_1} \sum_{i=1}^{n_1} (-v'_i \log(-v'_i) + v'_i) \end{cases} \quad (10)$$

Because each  $g(v_i), i = 1, \dots, n_2$  and  $g(v'_i), i = 1, \dots, n_1$  is independent of others, we analyze each  $g(v_i)$  and  $g(v'_i)$  separately.

Based on [4,5], we can derive the conjugate function  $g^*(u_i)$  and  $g^*(u'_i)$  of  $g(v_i)$  and  $g(v'_i)$  respectively:

$$\begin{cases} g^*(u_i) = \sup_{v_i < 0} \{u_i v_i - g(v_i)\} \\ g^*(u'_i) = \sup_{v'_i < 0} \{u'_i v'_i - g(v'_i)\} \end{cases} \tag{11}$$

where  $v_i$  and  $v'_i$  are the optimization variables of the right hand of (11) respectively, and the supremums can be achieved at  $v_i = -\exp(-u_i)$  and  $v'_i = -\exp(-u'_i)$  respectively. Substituting  $v_i = -\exp(-u_i)$  and  $v'_i = -\exp(-u'_i)$  into (11), we have

$$\begin{cases} g^*(u_i) = \exp(-u_i) \\ g^*(u'_i) = \exp(-u'_i) \end{cases} \tag{12}$$

Hence, we have the conjugate function  $G_1^*(\mathbf{u})$  and  $G_2^*(\mathbf{u}')$  of  $G_1(\mathbf{v})$  and  $G_2(\mathbf{v}')$  respectively, their forms as

$$\begin{cases} G_1^*(\mathbf{u}) = \frac{C_2}{n_2} \sum_{i=1}^{n_2} \exp(-u_i) = \frac{C_2}{n_2} \sum_{i=1}^{n_2} \sup_{v_i < 0} \{u_i v_i - g(v_i)\} \\ G_2^*(\mathbf{u}') = \frac{C_4}{n_1} \sum_{i=1}^{n_1} \exp(-u'_i) = \frac{C_4}{n_1} \sum_{i=1}^{n_1} \sup_{v'_i < 0} \{u'_i v'_i - g(v'_i)\} \end{cases} \tag{13}$$

where  $\sup\{\cdot\}$  represents the upper bounded of a variable.

Let  $u_i = \frac{(1+(\beta_+^* - \beta_+)^T \mathbf{x}_{2i} + (\gamma_+^* - \gamma_+))^2}{2\sigma^2}$ ,  $u'_i = \frac{(1-(\beta_-^* - \beta_-)^T \mathbf{x}_{1i} - (\gamma_-^* - \gamma_-))^2}{2\sigma^2}$ , (13) can be described as

$$\begin{cases} G_1^*(\mathbf{u}) = \frac{C_2}{n_2} \sum_{i=1}^{n_2} \exp\left(-\frac{(1+(\beta_+^* - \beta_+)^T \mathbf{x}_{2i} + (\gamma_+^* - \gamma_+))^2}{2\sigma^2}\right) = \sup_{\mathbf{v} < 0} \left\{ \frac{C_2}{n_2} \sum_{i=1}^{n_2} h(v_i | \beta_+^*, \beta_+, \gamma_+^*, \gamma_+) \right\} \\ G_2^*(\mathbf{u}') = \frac{C_4}{n_1} \sum_{i=1}^{n_1} \exp\left(-\frac{(1-(\beta_-^* - \beta_-)^T \mathbf{x}_{1i} - (\gamma_-^* - \gamma_-))^2}{2\sigma^2}\right) = \sup_{\mathbf{v}' < 0} \left\{ \frac{C_4}{n_1} \sum_{i=1}^{n_1} h(v'_i | \beta_-^*, \beta_-, \gamma_-^*, \gamma_-) \right\} \end{cases} \tag{14}$$

where  $h(v_i | \beta_+^*, \beta_+, \gamma_+^*, \gamma_+) = \left( v_i \frac{(1+(\beta_+^* - \beta_+)^T \mathbf{x}_{2i} + (\gamma_+^* - \gamma_+))^2}{2\sigma^2} - g(v_i) \right)$ ,  $i = 1, \dots, n_2$  and  $h(v'_i | \beta_-^*, \beta_-, \gamma_-^*, \gamma_-) = \left( v'_i \frac{(1-(\beta_-^* - \beta_-)^T \mathbf{x}_{1i} - (\gamma_-^* - \gamma_-))^2}{2\sigma^2} - g(v'_i) \right)$ ,  $i = 1, \dots, n_1$ .

Obviously, we have

$$\begin{cases} L_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+) = \sup_{\mathbf{v} < 0} \left\{ \frac{C_2}{n_2} \sum_{i=1}^{n_2} h(v_i | \beta_+^*, \beta_+, \gamma_+^*, \gamma_+) \right\} \\ L_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-) = \sup_{\mathbf{v}' < 0} \left\{ \frac{C_4}{n_1} \sum_{i=1}^{n_1} h(v'_i | \beta_-^*, \beta_-, \gamma_-^*, \gamma_-) \right\} \end{cases} \tag{15}$$

where the supremums of  $L_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+)$  and  $L_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-)$  are achieved at

$$\begin{cases} v_i = -\exp\left(-\frac{(1+(\beta_+^* - \beta_+)^T \mathbf{x}_{2i} + (\gamma_+^* - \gamma_+))^2}{2\sigma^2}\right), i = 1, \dots, n_2 \\ v'_i = -\exp\left(-\frac{(1-(\beta_-^* - \beta_-)^T \mathbf{x}_{1i} - (\gamma_-^* - \gamma_-))^2}{2\sigma^2}\right), i = 1, \dots, n_1 \end{cases} \tag{16}$$

respectively. Thus, the optimization problems (8) and (9) with four variables can be turned to a HQ optimization problem with five variables:

$$\begin{cases} \max_{\beta_+^*, \beta_+, \gamma_+^*, \gamma_+, \mathbf{v} < 0} & -H_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+) + \frac{C_2}{n_2} \sum_{i=1}^{n_2} h(v_i | \beta_+^*, \beta_+, \gamma_+^*, \gamma_+) \\ \max_{\beta_-^*, \beta_-, \gamma_-^*, \gamma_-, \mathbf{v}' < 0} & -H_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-) + \frac{C_4}{n_1} \sum_{i=1}^{n_1} h(v'_i | \beta_-^*, \beta_-, \gamma_-^*, \gamma_-) \end{cases} \quad (17)$$

From (17), we use the alternating iterative method to solve the optimization problem (17).

First, given  $(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+)$  and  $(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-)$  to optimize  $\mathbf{v}$  and  $\mathbf{v}'$  respectively. So that, (17) can be reduced to two independent functions with only respect to  $v_i$  or  $v'_i$ :

$$\begin{cases} \max_{\mathbf{v} < 0} & \frac{C_2}{n_2} \sum_{i=1}^{n_2} h(v_i | \beta_+^*, \beta_+, \gamma_+^*, \gamma_+) \\ \max_{\mathbf{v}' < 0} & \frac{C_4}{n_1} \sum_{i=1}^{n_1} h(v'_i | \beta_-^*, \beta_-, \gamma_-^*, \gamma_-) \end{cases} \quad (18)$$

Second, given  $\mathbf{v}$  and  $\mathbf{v}'$  to optimize  $(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+)$  and  $(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-)$  respectively. The optimization problems (17) can be rewritten as:

$$\begin{cases} \max_{\beta_+^*, \beta_+, \gamma_+^*, \gamma_+} & -H_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+) + \frac{C_2}{n_2} \sum_{i=1}^{n_2} \frac{v_i(1+(\beta_+^* - \beta_+)^T \mathbf{x}_{2i} + (\gamma_+^* - \gamma_+))^2}{2\sigma^2} \\ \max_{\beta_-^*, \beta_-, \gamma_-^*, \gamma_-} & -H_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-) + \frac{C_4}{n_1} \sum_{i=1}^{n_1} \frac{v'_i(1-(\beta_-^* - \beta_-)^T \mathbf{x}_{1i} + (\gamma_-^* - \gamma_-))^2}{2\sigma^2} \end{cases} \quad (19)$$

To solve the optimization problem (19) easily, we rewrite it as follows:

$$\begin{cases} \min_{\beta_+^*, \beta_+, \gamma_+^*, \gamma_+, \xi} & H_1(\beta_+^*, \beta_+, \gamma_+^*, \gamma_+) + C'_2 \xi^T \Omega \xi \\ \text{s.t.} & -(\mathbf{X}_2(\beta_+^* - \beta_+) + (\gamma_+^* - \gamma_+)) = 1 - \xi \\ \min_{\beta_-^*, \beta_-, \gamma_-^*, \gamma_-, \xi'} & H_2(\beta_-^*, \beta_-, \gamma_-^*, \gamma_-) + C'_4 \xi'^T \Omega' \xi' \\ \text{s.t.} & (\mathbf{X}_1(\beta_-^* - \beta_-) + (\gamma_-^* - \gamma_-)) = 1 - \xi' \end{cases} \quad (20)$$

where  $\xi = [\xi_1, \dots, \xi_{n_2}]^T$  and  $\xi' = [\xi'_1, \dots, \xi'_{n_1}]^T$  are the slack variables,  $\xi_i = 1 + (\beta_+^* - \beta_+)^T \mathbf{x}_{2i} + (\gamma_+^* - \gamma_+)$ ,  $i = 1, \dots, n_2$ ,  $\xi'_i = 1 - (\beta_-^* - \beta_-)^T \mathbf{x}_{1i} - (\gamma_-^* - \gamma_-)$ ,  $i = 1, \dots, n_1$ ,  $C'_2 = C_2 / (2n_2\sigma^2)$ ,  $C'_4 = C_4 / (2n_1\sigma^2)$  and  $\Omega = \frac{1}{n_2} \text{diag}(-\mathbf{v}) \in \mathbb{R}^{n_2 \times n_2}$ ,  $\Omega' = \frac{1}{n_1} \text{diag}(-\mathbf{v}') \in \mathbb{R}^{n_1 \times n_1}$ .

Further, let  $\alpha_1 = [\beta_+^{*T}, \beta_+^T, \gamma_+^*, \gamma_+]^T$ ,  $\alpha_2 = [\beta_-^{*T}, \beta_-^T, \gamma_-^*, \gamma_-]^T$ ,  $\zeta_1 = [C_1 \mathbf{1}_m^T, C_1 \mathbf{1}_m^T, C_1, C_1]^T$ ,  $\zeta_2 = [C_3 \mathbf{1}_m^T, C_3 \mathbf{1}_m^T, C_3, C_3]^T$ ,  $\mathbf{M}_1 = [\mathbf{X}_2, -\mathbf{X}_2, 1, -1]$ ,  $\mathbf{M}_2 = [-\mathbf{X}_1, \mathbf{X}_1, -1, 1]$ , and

$$\mathbf{Q}_1 = \begin{bmatrix} \mathbf{X}_1^T \mathbf{X}_1 & -\mathbf{X}_1^T \mathbf{X}_1 & 0.5 \mathbf{X}_1^T \mathbf{e}_1 & -0.5 \mathbf{X}_1^T \mathbf{e}_1 \\ -\mathbf{X}_1^T \mathbf{X}_1 & \mathbf{X}_1^T \mathbf{X}_1 & -0.5 \mathbf{X}_1^T \mathbf{e}_1 & 0.5 \mathbf{X}_1^T \mathbf{e}_1 \\ 0.5 \mathbf{e}_1^T \mathbf{X}_1 & -0.5 \mathbf{e}_1^T \mathbf{X}_1 & \mathbf{e}_1^T \mathbf{e}_1 & -\mathbf{e}_1^T \mathbf{e}_1 \\ -0.5 \mathbf{e}_1^T \mathbf{X}_1 & 0.5 \mathbf{e}_1^T \mathbf{X}_1 & -\mathbf{e}_1^T \mathbf{e}_1 & \mathbf{e}_1^T \mathbf{e}_1 \end{bmatrix}$$

$$\mathbf{Q}_2 = \begin{bmatrix} \mathbf{X}_2^T \mathbf{X}_2 & -\mathbf{X}_2^T \mathbf{X}_2 & 0.5 \mathbf{X}_2^T \mathbf{e}_2 & -0.5 \mathbf{X}_2^T \mathbf{e}_2 \\ -\mathbf{X}_2^T \mathbf{X}_2 & \mathbf{X}_2^T \mathbf{X}_2 & -0.5 \mathbf{X}_2^T \mathbf{e}_2 & 0.5 \mathbf{X}_2^T \mathbf{e}_2 \\ 0.5 \mathbf{e}_2^T \mathbf{X}_2 & -0.5 \mathbf{e}_2^T \mathbf{X}_2 & \mathbf{e}_2^T \mathbf{e}_2 & -\mathbf{e}_2^T \mathbf{e}_2 \\ -0.5 \mathbf{e}_2^T \mathbf{X}_2 & 0.5 \mathbf{e}_2^T \mathbf{X}_2 & -\mathbf{e}_2^T \mathbf{e}_2 & \mathbf{e}_2^T \mathbf{e}_2 \end{bmatrix}$$

Then substitute the equality constraints into the objective function in (20), we can derive that

$$\begin{cases} \min_{\alpha_1} \frac{1}{2} \alpha_1^T \mathbf{H}_1 \alpha_1 + \mathbf{f}_1^T \alpha_1 \\ \min_{\alpha_2} \frac{1}{2} \alpha_2^T \mathbf{H}_2 \alpha_2 + \mathbf{f}_2^T \alpha_2 \end{cases} \quad (21)$$

where  $\mathbf{H}_1 = 2\lambda'_1 \mathbf{M}_1 \boldsymbol{\Omega} \mathbf{M}_1 + \mathbf{Q}_1$ ,  $\mathbf{H}_2 = (2\lambda'_2 \mathbf{M}_2^T \boldsymbol{\Omega}' \mathbf{M}_2 + \mathbf{G})$ ,  $\mathbf{f}_1^T = 2C'_2 \mathbf{1}^T \boldsymbol{\Omega} \mathbf{M}_1 + \boldsymbol{\zeta}_1^T$ , and  $\mathbf{f}_2^T = 2C'_4 \mathbf{1}^T \boldsymbol{\Omega}' \mathbf{M}_2 + \boldsymbol{\zeta}_2^T$ .

Once we have the values of  $\alpha_1$  and  $\alpha_2$ , the hyperplanes  $f_1(\mathbf{x}) = 0$  and  $f_2(\mathbf{x}) = 0$  can be obtained. Hence, given a new sample  $\mathbf{x}$ , its class  $i(i = 1, 2)$  is

$$\text{class } i = \begin{cases} +1, & |f_2(\mathbf{x})| - |f_1(\mathbf{x})| > 0 \\ -1, & \text{otherwise} \end{cases} \quad (22)$$

### 3 Numerical Experiments

In this section, we carry out experiments to testify the validity of the novel algorithm C-STSVMS. We compare C-STSVMS with the state-of-the-art methods, including SVM [10], TSVM [15], TBSVM [20], KCC [25], Pin-TSVM [26], LSTSVMS [18],  $\ell_p$ LSTSVMS [30], NLPTSVM [22] and SNSVM [23], where SVM, TSVM, and TBSVM are three traditional algorithms, KCC is a linear classifier with the correntropy-induced loss, Pin-TSVM and LSTSVMS introduce the pinball loss and the square loss into TSVM, respectively, and the others were proposed for obtaining sparse models.

For all SVM-like methods, the linear kernel or the linear version is adopted. All experiments are implemented in MATLAB R2016a on Windows 10 running on a PC with a 3.0 GHz Intel Core and 8 GB of memory.

#### 3.1 Data Description and Experimental Setting

We carry out experiments on nine UCI datasets [11]: Australian (690 samples and 14 features), Breast (288 samples and 9 features), German (1000 samples and 24 features), Heart (270 samples and 13 features), Pima (768 samples and 8 features), Sonar (208 samples and 60 features), Tic\_tac\_toe (958 samples and 9 features), Vote (435 samples and 16 features) and Wdbc (569 samples and 30 features).

The repeated double cross validation [12] was used to select parameters and give the final average result. The five-fold cross validation method is used in twice. First, each dataset is randomly divided into five parts, where one part is taken as the test set at a time and the remaining four parts are used as the calibration set. Next, each calibration set is randomly split into five parts, where we take four parts as training set and the rest part as the validation set. In this process, we train models on the training set using the regularization parameters in the range of  $\{2^{-3}, \dots, 2^3\}$  one by one, and apply the trained-model to the validation set to select optimal parameters. Then we train a model on the calibration set using the optimal parameters and apply the trained-model to the test set to obtain the result.

This process is repeated five times, and the average results of five trials are reported. In addition, the parameter  $\sigma$  in both KCC and C-STSVMS is fixed at  $\sigma = 0.5$  empirically.



### 3.2 Robustness to Outliers

In order to demonstrate the robustness to outlier of C-STSVm clearly, we test our method C-STSVm and the other comparison methods on the nine UCI datasets with 0% and 10% label noise.

**Table 1.** Accuracy obtained on original UCI datasets

Datasets	SVM	TSVM	TBSVM	KCC	Pin-TSVM	LSTSVM	$\ell_p$ LSTSVM	NLPTSVM	SNSVM	C-STSVm
Australian	86.96±2.98	87.54±2.32	87.54±3.10	86.96±3.22	78.93±19.39	86.81±2.24	87.68±3.10	87.10±2.51	87.69±3.91	<b>87.82±4.22</b>
Breast	68.23±3.02	72.48±4.71	68.20±3.78	70.04±5.22	63.48±19.49	67.93±5.67	72.58±4.89	67.90±6.71	72.58±2.72	<b>73.28±0.82</b>
German	77.30±1.30	76.90±1.29	77.40±1.47	77.80±2.02	75.10±1.52	77.30±1.89	77.30±1.64	76.30±0.97	77.40±2.07	<b>78.10±0.82</b>
Heart	<b>84.44±2.81</b>	83.33±2.93	83.70±2.75	81.85±4.01	81.85±7.22	82.59±1.01	82.59±2.11	83.70±2.03	82.59±2.11	82.59±3.36
Pima	76.56±1.95	76.43±1.43	76.83±3.05	76.31±2.59	72.66±5.89	75.52±1.92	76.56±2.18	75.29±2.61	76.82±2.35	<b>76.96±2.28</b>
Sonar	77.96±5.85	73.13±7.46	75.02±7.55	75.60±7.85	<b>80.41±10.65</b>	75.04±6.28	76.01±7.77	77.01±5.80	77.93±4.97	74.67±10.44
Tic_tac_toe	65.34±0.15	68.89±1.40	68.47±1.23	69.52±0.99	65.34±0.15	69.52±1.47	<b>69.62±1.17</b>	65.34±0.15	68.47±2.27	66.39±1.01
Vote	92.20±3.83	91.74±4.13	93.80±2.05	93.35±3.35	77.74±24.65	93.12±3.00	93.80±1.91	94.02±1.71	92.43±3.26	<b>94.26±1.78</b>
Wdbc	<b>98.25±1.07</b>	98.07±1.14	96.31±1.91	96.48±2.18	70.69±30.55	97.19±1.32	97.72±0.99	97.89±0.78	96.66±2.12	97.54±1.32

Table 1 shows the results for the original UCI datasets and the best classification accuracy is highlighted in bold. It can be seen that C-TSVM has satisfying results. The C-TSVM method obtains the highest classification accuracy on five out of nine datasets, while SVM obtain the best classification accuracy on Heart and Wdbc dataset, and  $\ell_p$ LSTSVM and Pin-TSVM has the best performance on Tic\_tac\_toe and Sonar dataset respectively. Then, we corrupt the label of each calibration set. For each calibration set, the ratio of label noise is 10%. In this case, the average accuracy and standard deviation are presented on Table 2. As shown in Table 2, C-TSVM has the best performance on six out of nine datasets in terms of classification accuracy. TSVM and Pin-TSVM has the best accuracy on Australian and Sonar dataset respectively.

From the comparison between Table 1 and 2, we can state the following points: (1) The accuracy of C-STSVm has a small change by increasing the number of label noise and has the best classification accuracy on the most of datasets; (2) The accuracy of Pin-TSVM on German and Pima dataset have a substantial reduction by increasing the number of label noise; (3) SVM has the best accuracy on two out of original nine datasets. However, as the number of label noise increases, its classification advantage does not seem to be maintained. (4) The performances of TSVM, TBSVM, KCC, LSTSVM,  $\ell_p$ LSTSVM, NLPTSVM and SNSVM method are average. In summary, C-STSVm has the better robustness and classification accuracy to outlier.

**Table 2.** Accuracy obtained on UCI datasets with 10% label noise

Datasets	SVM	TSVM	TBSVM	KCC	Pin-TSVM	LSTSVM	$\ell_p$ LSTSVM	NLPTSVM	SNSVM	C-STSVm
Australian	86.96±2.98	<b>88.12±2.48</b>	87.25±3.17	87.39±2.90	86.52±2.79	87.53±2.16	87.68±2.46	86.81±2.87	87.83±2.86	87.68±3.16
Breast	69.68±6.00	66.45±6.43	67.54±8.43	66.79±2.36	68.70±9.97	67.89±3.41	68.27±3.71	67.54±3.40	65.74±5.59	<b>73.27±1.77</b>
German	74.40±1.67	75.60±2.38	75.50±1.70	75.90±1.78	45.60±19.43	75.30±1.52	75.80±1.89	74.90±0.74	75.30±1.30	<b>76.60±1.95</b>
Heart	81.48±6.55	82.59±4.26	81.11±3.31	83.33±5.56	81.85±5.91	82.59±5.00	83.33±4.34	82.22±5.34	82.96±4.42	<b>84.81±4.79</b>
Pima	77.08±3.10	75.78±1.99	75.52±2.63	76.95±3.01	58.19±21.37	76.04±2.09	76.43±2.99	76.43±1.43	76.69±1.77	<b>77.21±2.16</b>
Sonar	74.67±9.86	75.01±5.47	70.74±8.34	74.07±7.81	<b>78.92±5.30</b>	74.08±4.93	75.00±5.00	77.43±6.15	74.11±7.97	76.95±8.42
Tic_tac_toe	63.99±1.46	63.26±1.61	62.74±1.39	64.20±1.54	59.29±4.54	63.16±1.23	63.78±0.81	64.99±1.97	64.09±0.77	<b>65.03±0.73</b>
Vote	93.12±3.00	93.57±2.74	92.88±3.15	93.12±3.00	93.12±0.30	91.50±4.77	93.34±3.35	93.12±0.30	92.89±3.35	<b>94.48±2.07</b>
Wdbc	92.08±1.90	94.91±2.65	91.74±2.71	91.39±2.90	92.45±1.76	93.31±1.85	93.85±2.90	95.26±2.91	91.75±3.13	<b>96.13±1.61</b>

### 3.3 The Ability of Feature Selection

We add 50 noise features to each UCI dataset to validate the ability of feature selection of these methods, where the last 50 features are noise features and the others are valid features. These noise features are drawn from the Gaussian distribution with 0 mean and 0.01 variance. Table 3 summarizes the results of numerical experiments. It is easy to see that the accuracy of C-STSVMS is significantly better than other methods on most of datasets. Pin-TSVM and NLPTSVM algorithm has the best on Sonar and Wdbc dataset respectively, followed by C-STSVMS. Compare Table 1 and Table 3, we can find that the increase in noise features does not negatively affect classification accuracy of C-STSVMS, and it still maintains better classification performance than other methods.

In order to clearly show the feature selection performance, we calculated the ratio of sum of the absolute values of the  $w_1$  ( $w_2$ ) corresponding to the valid features to the sum of the absolute values of total  $w_1$  ( $w_2$ ) and Table 4 shows the results of some TSVM-like methods. In theory, the greater the contribution of a feature to the classification result, the greater the weight value corresponding to this feature. Hence, for a weight value obtained by training some algorithm, the greater the proportion of weight values corresponding to the valid features, the better the feature selection performance of this algorithm. In Table 4, the highest proportion is shown in bold figures. Table 4 depicts NLPTSVM and C-STSVMS has the best feature selection performance on the most of datasets, followed by Pin-TSVM. But the accuracy of C-STSVMS is higher than NLPTSVM and Pin-TSVM. The feature selection performance of TSVM, TBSVM, LSTSVMS,  $\ell_p$ LSTSVMS, and SNSVMS are unsatisfactory. The results in Table 3 and Table 4

**Table 3.** Accuracy obtained on UCI datasets with 50 noise features

Datasets	SVM	TSVM	TBSVM	KCC	Pin-TSVM	LSTSVMS	$\ell_p$ LSTSVMS	NLPTSVM	SNSVMS	C-STSVMS
Australian	86.81±3.36	86.95±3.55	87.39±4.16	87.83±3.52	87.40±4.36	87.39±4.42	87.54±4.03	87.39±1.51	87.24±3.50	<b>88.12±3.17</b>
Breast	70.41±5.57	70.05±5.59	71.16±5.81	69.68±4.23	70.04±6.10	69.67±2.70	70.75±2.77	67.50±6.19	70.04±3.75	<b>72.58±0.56</b>
German	77.10±1.34	74.90±1.47	75.10±1.52	75.40±1.52	75.20±1.52	75.60±1.98	75.60±1.75	74.60±2.33	76.00±2.22	<b>77.30±0.97</b>
Heart	84.81±3.04	82.59±3.61	81.48±5.07	81.48±6.14	82.96±4.01	80.74±3.84	80.37±3.10	81.85±4.22	83.70±3.04	<b>85.19±2.27</b>
Pima	76.31±2.40	72.92±1.70	73.57±1.31	75.52±2.79	73.18±1.73	72.91±2.32	73.05±1.57	75.39±2.16	75.78±2.55	<b>76.94±1.64</b>
Sonar	78.86±5.41	57.21±5.71	60.48±8.76	74.06±6.17	<b>78.47±7.15</b>	70.04±10.69	70.10±10.50	74.57±7.56	76.01±5.89	77.01±8.28
Tic_tac_toe	65.66±0.38	66.08±0.78	66.81±0.95	67.02±1.07	59.68±14.09	65.87±0.54	65.97±0.44	65.34±0.15	66.50±1.84	<b>68.06±1.90</b>
Vote	91.51±4.36	93.35±3.03	92.65±3.28	92.42±3.30	92.89±3.06	93.35±0.34	92.66±3.47	92.66±3.47	92.66±3.48	<b>94.95±1.29</b>
Wdbc	97.19±1.31	94.73±1.62	95.26±1.57	95.25±1.85	92.98±2.51	95.08±1.74	95.78±2.02	<b>97.54±1.32</b>	95.77±2.47	97.36±2.16

**Table 4.** The proportions of  $w_1$  and  $w_2$  corresponding to valid features

Datasets	TSVM		TBSVM		Pin-TSVM		LSTSVMS		$\ell_p$ LSTSVMS		NLPTSVM		SNSVMS		C-STSVMS	
	$w_1$	$w_2$	$w_1$	$w_2$	$w_1$	$w_2$	$w_1$	$w_2$	$w_1$	$w_2$	$w_1$	$w_2$	$w_1$	$w_2$	$w_1$	$w_2$
Australian	16.66%	41.36%	16.83%	29.49%	91.47%	91.28%	18.11%	18.11%	15.16%	33.62%	<b>100.00%</b>	<b>100.00%</b>	29.84%	25.55%	<b>100.00%</b>	<b>100.00%</b>
Breast	3.48%	5.44%	3.76%	5.32%	76.39%	76.34%	4.88%	4.61%	5.21%	5.53%	<b>100.00%</b>	<b>100.00%</b>	19.79%	18.89%	87.70%	90.21%
German	15.66%	18.02%	21.07%	18.21%	67.45%	52.51%	17.80%	19.04%	18.26%	19.21%	48.83%	43.28%	18.96%	20.72%	<b>87.70%</b>	<b>90.21%</b>
Heart	9.88%	8.82%	8.42%	8.84%	82.23%	<b>75.68%</b>	9.15%	10.63%	12.86%	10.45%	12.07%	19.52%	32.33%	38.53%	<b>100.00%</b>	46.16%
Pima	23.07%	22.24%	24.33%	22.20%	19.35%	20.78%	23.55%	23.81%	24.10%	23.10%	31.91%	<b>36.61%</b>	25.69%	24.83%	<b>80.90%</b>	20.98%
Sonar	57.03%	49.20%	60.23%	49.99%	65.50%	78.27%	53.75%	45.44%	48.97%	48.24%	95.71%	52.30%	55.65%	58.61%	<b>98.80%</b>	<b>88.32%</b>
Tic_tac_toe	4.30%	4.05%	4.32%	3.80%	74.52%	72.58%	4.25%	4.29%	4.22%	0.00%	0.00%	4.43%	3.89%	<b>90.93%</b>	<b>87.23%</b>	
Vote	17.92%	24.24%	17.21%	24.84%	48.86%	70.11%	22.81%	18.36%	16.57%	24.65%	<b>100.00%</b>	<b>100.00%</b>	44.47%	46.00%	<b>100.00%</b>	94.16%
Wdbc	80.16%	67.21%	88.29%	70.81%	95.88%	<b>95.91%</b>	82.71%	59.67%	65.33%	54.85%	94.75%	89.25%	55.43%	56.57%	<b>99.94%</b>	65.80%

undoubtedly prove that C-STSVm is significantly better than other compared methods in feature selection performance and the classification accuracy.

## 4 Conclusion

This paper proposes a novel sparse twin support vector machine with the correntropy-induced loss (C-STSVm). Because the correntropy-induced loss has favorable robustness to outliers and the  $\ell_1$ -norm regularization can induce a sparse model, we expect that C-STSVm has a satisfactory robustness to outliers and a sparseness solution to implement feature selection. To validate the robustness of C-STSVm, we carry out experiments on nine UCI datasets with 0% and 10% label noise. To validate the performance of C-STSVm to select features, we conduct experiments on nine UCI datasets with 50 noise features. Experiments results confirm that C-STSVm is significantly better than other compared methods in robustness to outliers, feature selection performance, and classification accuracy.

Since C-STSVm achieves a good performance in the binary classification tasks, we plan to extend C-STSVm to regression estimation and multi-class classification tasks in future.

## References

1. Adankon, M.M., Cheriet, M.: Model selection for LS-SVM: application to handwriting recognition. *Pattern Recogn.* **42**(12), 3264–3270 (2009)
2. Jain, A.K., Robert, P.W., Duin, J.M.: Statistical pattern recognition: a review. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(1), 4–37 (2000)
3. Arjunan, S.P., Kumar, D.K., Naik, G.R.: A machine learning based method for classification of fractal features of forearm sEMG using twin support vector machines. In: *Proceedings of the IEEE Annual International Conference of the Engineering in Medicine and Biology Society*, pp. 4821–4824 (2010)
4. Borwein, J., Lewis, A.: *Convex Analysis and Nonlinear Optimization: Theory and Examples*, 2nd edn. Springer, New York (2006). <https://doi.org/10.1007/978-0-387-31256-9>
5. Boyd, S.P., Vandenberghe, L.: *Convex Optimization*. Cambridge University, Cambridge (2004)
6. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. *Comput. Electr. Eng.* **40**, 16–28 (2014)
7. Chen, S., Wu, X.: A new fuzzy twin support machine for pattern classification. *Int. J. Mach. Learn. Cybernet.* **9**, 1553–1564 (2018)
8. Chen, S., Wu, X., Xu, J.: locality preserving projection least squares twin support vector machine for pattern classification. *Pattern Anal. Appl.* **23**, 1–13 (2020)
9. Chen, W., Shao, Y., Li, C., Liu, M., Wang, Z., Deng, N.:  $v$ -projection twin support vector machine for pattern classification. *Neurocomputing* **376**, 10–24 (2020)
10. Cristianini, N., Shawe-Taylor, J.: *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University, Cambridge (2000)
11. Dheeru, D., Karra Taniskidou, E.: UCI machine learning repository (2017). <http://archive.ics.uci.edu/ml>

12. Filzmoser, P., Liebmann, B., Varmuza, K.: Repeated double cross validation. *J. Chemom.* **23**(4), 160–171 (2008)
13. Hua, X., Ding, S.: Locality preserving twin support vector machines. *J. Comput. Res. Dev.* **51**(3), 590–597 (2014)
14. Isabelle Guyon, A.E.: An introduction to variable and feature selection. *J. Mach. Learn. Res.* **3**, 1157–1182 (2003)
15. Jayadeva, Khemchandani, R., Chandra, S.: Twin support vector machine for pattern classification. *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(5), 905–910 (2007)
16. Khan, N., Ksantini, R., Ahmad, I., Oufama, B.: A novel SVM+NDA model for classification with an application to face recognition. *Pattern Recogn.* **45**(1), 66–79 (2012)
17. Kohavi, R., John, G.H.: Wrappers for feature subset selection. *Artif. Intell.* **97**, 273–324 (1997). [https://doi.org/10.1016/S0004-3702\(97\)00043-X](https://doi.org/10.1016/S0004-3702(97)00043-X)
18. Kumar, M.A., Gopall, M.: Least squares twin support vector machine for pattern classification. *Expert Syst. Appl.* **36**, 7535–7543 (2009)
19. Liu, M., Dai, B., Xie, Y., Ya, Z.: Improved GMM-UBM/SVM for speaker verification. In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 1925–1928 (2006)
20. Shao, Y., Zhang, C., Wang, X., Deng, N.: Improvements on twin support vector machine. *IEEE Trans. Neural Netw.* **22**(6), 962–968 (2011)
21. Singh, A., Principe, J.C.: A loss function for classification based on a robust similarity metric. In: *Proceedings of IEEE International Joint Conference on Neural Network*, pp. 1–6 (2010)
22. Tanveer, M.: Robust and sparse linear programming twin support vector machines. *Cogn. Comput.* **7**(1), 137–149 (2015)
23. Tian, Y., Ju, X., Qi, Z.: Efficient sparse nonparallel support vector machines for classification. *Neural Comput. Appl.* **24**(5), 1089–1099 (2013). <https://doi.org/10.1007/s00521-012-1331-5>
24. Wang, X., Wang, T., Bu, J.: Color image segmentation using pixel wise support vector machine classification. *Pattern Recogn.* **44**(4), 777–787 (2011)
25. Xu, G., Hu, B., Principe, J.C.: Robust C-loss kernel classifiers. *IEEE Trans. Neural Netw. Learn. Syst.* **29**(3), 510–522 (2018)
26. Xu, Y., Yang, Z., Pan, X.: A novel twin support-vector machine with pinball loss. *IEEE Trans. Neural Netw. Learn. Syst.* **28**(2), 359–370 (2017)
27. Xue, Z., Ming, D., Song, W., Wan, B., Jin, S.: Infrared gait recognition based on wavelet transform and support vector machine. *Pattern Recogn.* **43**(8), 2904–2910 (2010)
28. Zhang, L., Zhou, W.: On the sparseness of 1-norm support vector machines. *Neural Netw.* **23**(3), 373–385 (2010)
29. Zhang, X., Wu, G., Dong, Z., Curran, C.: Embedded feature-selection support vector machine for driving pattern recognition. *J. Franklin Inst.* **352**, 669–685 (2015)
30. Zhang, Z., Zhen, L., Deng, N., Tan, J.: Sparse least square twin support vector machine with adaptive norm. *Appl. Intell.* **41**(4), 1097–1107 (2014). <https://doi.org/10.1007/s10489-014-0586-1>
31. Zhu, J., Rosset, S., Hastie, T., Tibshirani, R.: 1-norm support vector machines. In: *Proceedings of the 16th International Conference on Neural Information Processing Systems*, vol. 16, no. 1, pp. 49–56 (2003)