The Influences of Smooth Approximation Functions for SPTSVM

Xinxin Zhang
Liaocheng University
School of Mathematics Sciences
Liaocheng, 252059
P.R. China
ldzhangxin2008@126.com

Liya Fan
Liaocheng University
School of Mathematics Sciences
Liaocheng, 252059
P.R. China
fanliya63@126.com

Abstract: The recently proposed smooth projection twin support vector machine(SPTSVM) gains a good generalization ability and is suitable for many binary classification problems. But we know that different smooth approximation functions may bring different classification accuracies. In order to study the influence of smooth approximation functions for SPTSVM, in this paper, we first overview eight known smooth approximation functions and describe their differentiability and error ranges by five lemmas and one theorem. Then, we perform a series of comparative experiments on classification accuracy and running time by using SPTSVM with Newton-Armijo method on 10 UCI datasets and 6 NDC datasets. From experiment results, we can get a choice order of the eight approximation functions in generally.

Key–Words: Smooth projection TSVM; plus function; smooth approximation function; choice order

1 Introduction

Recently, nonparallel hyperplane support vector machine (NHSVM) classification methods, as the extension of the classical SVM, have become the researching hot spots in the field of machine learning. The study of NHSVM classification methods originates from generalized eigenvalue proximal SVM (GEPSVM) [1], twin support vector machine (TSVM) [2] and projection twin support vector machine (PTSVM) [3]. For binary data classification problems, NHSVM methods aim to find a hyperplane for each class, such that each hyperplane is proximal to the data points of one class and far from the data points of the other class. GEPSVM, TSVM and PTSVM are three representative algorithms of NHSVM, and all the other NHSVM methods are improved versions based on them.

GEPSVM obtains each of nonparallel hyperplanes by solving the eigenvector corresponding to a smallest eigenvalue of a generalized eigenvalue problem, so that each hyperplane is as close as possible to the points of its own class and as far as possible from the points of the other class, in the meantime. Twin support vector machine (TSVM) constructs a pair of nonparallel hyperplanes by solving two smaller size QPPs rather than a single quadratic programming problem (QPP) such that each one is as close as possible to one class, and as far as possible from the other class. A new input will be assigned to one of the classes depending on its proximity to which hyperplane. Experiments show that TSVM is faster than SVM [2,4]. Different from GEPSVM and TSVM, the central idea in PTSVM is to find a projection axis for each class, such that within-class variance of the projected samples of its own class is minimized; meanwhile, the projected samples of the other class scatter away as far as possible. PTSVM is an improvement and extension of multi-weight vector projection SVM (MVSVM) [5].

In order to further enhance the performance of PTSVM, Shao et al. [6] proposed a least squares version of PTSVM, called least squares PTSVM (LSPTSVM). LSPTSVM works extremely faster than PTSVM because the solutions of LSPTSVM can be attained by solving two systems of linear equations, whereas PTSVM needs to solve two QPPs. Because of this, the least squares method is also received great attention in support tensor machine. Later, Shao et al. [7-9] proposed a simple and reasonable variant of PTSVM from theoretical point

of view, called PTSVM with regularization term (RPTSVM), in which the regularized risk principle is implemented and the nonlinear classification ignored in PTSVM is also considered in RPTSVM. Ding and Hua [10] formulated a nonlinear version of LSPTSVM for binary nonlinear classification by introducing nonlinear kernel into LSPTSVM. This formulation leads to a novel nonlinear algorithm, called nonlinear L-SPTSVM (NLSPTSVM). Ding et al. reviewed many known nonparallel hyperplane support vector machine algorithms in [11]. In addition, by means of the idea of smooth TSVM in [12], the authors of the paper introduce smoothing technique into PTSVM and propose smooth PTSVM (SPTSVM) in [13].

We know that by using smoothing techniques, we can solve primal unconstrained differentiable optimization problems rather than dual QPPs, which results that many optimization methods can be used in smooth versions of various variants of TSVM, such as Newton method, quasi Newton method, Newton-Armijo method and so on. But we discover that different smooth approximation functions have different impacts for classification results even using the same classifier. So, in this paper, we first overview eight smooth approximation functions proposed in [14-22] and then compare their influences for SPTSVM on 16 datasets taken from UCI database and NDC database. Taking into account the length of the paper, we only discuss the influences of smooth approximation functions for linear version of SPTSVM. By means of kernel skill, we can discuss the influences of smooth approximation functions for nonlinear SPTSVM by using the similar way.

2 Linear PTSVM and SPTSVM

In this section, we recall linear PTSVM and linear SPTSVM briefly, for details see [3,13]. Let $T = \{(x_j^{(i)}, y_j^{(i)})\}_{j=1}^{m_i}, i = 1, 2$ be a set of data samples for a binary classification problem, where $i = 1$ denotes the positive class, $i = 2$ denotes the negative class, m_i denotes the number of samples belonging to class i, $x_j^{(i)} \in R^n$ and $y_j^{(i)} \in \{\pm 1\}$ are respectively the input and class label of jth sample in class i. Let $\mu^{(i)} = \frac{1}{m_i}\sum_{j=1}^{m_i} x_j^{(i)}$ be the mean of class i for $i = 1, 2$ and $A = [x_1^{(1)}, \cdots, x_{m_1}^{(1)}]^T \in R^{m_1 \times n}$ and $B = [x_1^{(2)}, \cdots, x_{m_2}^{(2)}]^T \in R^{m_2 \times n}$ denote the input matrices of positive and negative classes, respectively, and $m = m_1 + m_2$. Let $e_1 \in R^{m_1}$ and $e_2 \in R^{m_2}$ be vectors of ones.

2.1 Linear PTSVM

The central idea of linear PTSVM is to find a projection axis for each class such that within-class variance of the projected samples of its own class is minimized meanwhile the projected samples of the other class scatter away as far as possible. This leads to the following two optimization problems:

$$\begin{aligned} \min_{w_1,\xi_k}\ & \tfrac{1}{2}\textstyle\sum_{i=1}^{m_1}\left(w_1^T x_i^{(1)} - w_1^T\mu^{(1)}\right)^2 + c_1\sum_{k=1}^{m_2}\xi_k \\ s.t.\ & w_1^T x_k^{(2)} - w_1^T\mu^{(1)} + \xi_k \geq 1, \\ & \xi_k \geq 0,\ k = 1, 2, \cdots, m_2, \end{aligned} \tag{1}$$

$$\begin{aligned} \min_{w_2,\eta_k}\ & \tfrac{1}{2}\textstyle\sum_{i=1}^{m_2}\left(w_2^T x_i^{(2)} - w_2^T\mu^{(2)}\right)^2 + c_2\sum_{k=1}^{m_1}\eta_k \\ s.t.\ & -(w_2^T x_k^{(1)} - w_2^T\mu^{(2)}) + \eta_k \geq 1, \\ & \eta_k \geq 0,\ k = 1, 2, \cdots, m_1, \end{aligned} \tag{2}$$

where $c_1, c_2 > 0$ are trade-off parameters and $\{\xi_k\}_{k=1}^{m_2}$ and $\{\eta_k\}_{k=1}^{m_1}$ are slack variables. Put

$$S_1 = \textstyle\sum_{j=1}^{m_1}(x_i^{(1)} - \mu^{(1)})(x_i^{(1)} - \mu^{(1)})^T \in R^{n\times n},$$
$$S_2 = \textstyle\sum_{j=1}^{m_2}(x_i^{(2)} - \mu^{(2)})(x_i^{(2)} - \mu^{(2)})^T \in R^{n\times n},$$

then the problems (1) and (2) can be written as the following matrix forms, respectively:

$$\begin{aligned} \min_{w_1,\xi}\ & \tfrac{1}{2}w_1^T S_1 w_1 + c_1 e_2^T\xi \\ s.t.\ & Bw_1 - \tfrac{1}{m_1}e_2 e_1^T A w_1 + \xi \geq e_2, \\ & \xi \geq 0. \end{aligned} \tag{3}$$

$$\begin{aligned} \min_{w_2,\eta}\ & \tfrac{1}{2}w_2^T S_2 w_2 + c_2 e_1^T\eta \\ s.t.\ & -(Aw_2 - \tfrac{1}{m_2}e_1 e_2^T B w_2) + \eta \geq e_1, \\ & \eta \geq 0. \end{aligned} \tag{4}$$

By solving the Wolfe dual problems of the problems (3) and (4), respectively,

$$\begin{aligned} \min_{\alpha}\ & \tfrac{1}{2}\alpha^T(B - \tfrac{1}{m_1}e_2 e_1^T A)S_1^{-1}(B^T - \tfrac{1}{m_1}A^T e_1 e_2^T)\alpha \\ & -e_2^T\alpha \\ s.t.\ & 0 \leq \alpha \leq c_1 e_2, \end{aligned}$$

$$\begin{aligned} \min_{\beta}\ & \tfrac{1}{2}\beta^T(A - \tfrac{1}{m_2}e_1 e_2^T B)S_2^{-1}(A^T - \tfrac{1}{m_2}B^T e_2 e_1^T)\beta \\ & -e_1^T\beta \\ s.t.\ & 0 \leq \beta \leq c_2 e_1, \end{aligned}$$

we can obtain the optimal Lagrange multipliers vectors α^* and β^*. Without loss of generality, we can let S_1 and S_2 are nonsingular matrices. Otherwise, since they are symmetric nonnegative definite matrices, we can regularize them by using $S_1 + \varepsilon I_n$ and $S_2 + \varepsilon I_n$ to replace S_1 and S_2, respectively, where $\varepsilon > 0$ is a sufficient small number and I_n denotes the n order unit matrix. Consequently, we can deduce that

$$w_1^* = S_1^{-1}(B^T - \tfrac{1}{m_1}A^T e_1 e_2^T)\alpha^*,$$
$$w_2^* = S_2^{-1}(A^T - \tfrac{1}{m_2}B^T e_2 e_1^T)\beta^*,$$

and then the class label of a new input $x \in R^n$ can be assigned by

$$\text{class}(x) = \arg\min_{i=1,2} \left|(w_i^*)^T x - (w_i^*)^T \mu^{(i)}\right|.$$

2.2 Linear SPTSVM

The main idea of linear SPTSVM is to introduce smoothing technique into PTSVM, which results in solving a pair of primal unconstraint differentiable optimization problems rather than a pair of dual QPPs. By introducing the plus functions:

$$x_+ = \max\{x, 0\}, \ \forall x \in R,$$
$$x_+ = ((x_1)_+, \cdots, (x_n)_+)^T, \ \forall x \in R^n,$$

the constraints of the primal problems (1) and (2) can be rewritten as follows, respectively,

$$\begin{aligned}
\xi_k &= (1 - w_1^T x_k^{(2)} + w_1^T \mu^{(1)})_+, \\
&\quad k = 1, 2, \cdots, m_2, \\
\eta_k &= (1 + w_2^T x_k^{(1)} - w_2^T \mu^{(2)})_+, \\
&\quad k = 1, 2, \cdots, m_1, \\
\xi &= (\xi_1, \cdots, \xi_{m_2})^T \\
&= (e_2 + e_2\tilde{A}w_1 - Bw_1)_+ \in R^{m_2}, \\
\eta &= (\eta_1, \cdots, \eta_{m_1})^T \\
&= (e_1 - e_1\tilde{B}w_2 + Aw_2)_+ \in R^{m_1}.
\end{aligned}$$

where $\tilde{A} = (\mu^{(1)})^T$ and $\tilde{B} = (\mu^{(2)})^T$. In order to avoid the singularities of the matrices S_1 and S_2 involved in linear PTSVM, we adding the generalization terms $\frac{c_3}{2}\|w_1\|^2$ and $\frac{c_4}{2}\|w_2\|^2$ in problems (1) and (2), respectively. In addition, for obtaining differentiable optimization problems, we replace one penalty by two penalty for slack vectors ξ and η. Consequently, we get two improved unconstraint optimization problems:

$$\begin{aligned}
\min_{w_1} \ & \tfrac{c_1}{2}\left\|(e_2 + e_2\tilde{A}w_1 - Bw_1)_+\right\|^2 \\
& + \tfrac{1}{2}\left\|Aw_1 - e_1\tilde{A}w_1\right\|^2 + \tfrac{c_3}{2}\|w_1\|^2,
\end{aligned} \tag{5}$$

$$\begin{aligned}
\min_{w_2} \ & \tfrac{c_2}{2}\left\|(e_1 - e_1\tilde{B}w_2 + Aw_2)_+\right\|^2 \\
& + \tfrac{1}{2}\left\|Bw_2 - e_2\tilde{B}w_2\right\|^2 + \tfrac{c_4}{2}\|w_2\|^2.
\end{aligned} \tag{6}$$

Because the plus function $(x)_+$ for $x \in R$ is nondifferentiable, for effectively and quickly solving the problems (5) and (6) by using the known optimization methods, we need to introduce a smoothing approximation function $\rho(x, \zeta)$ for the plus function $(x)_+$, where ζ is a smoothing parameter. Consequently, the problems (5) and (6) can be further improved as the following two unconstraint differentiable optimization problems:

$$\begin{aligned}
\min_{w_1} f_1(w_1) = & \tfrac{1}{2}\left\|Aw_1 - e_1\tilde{A}w_1\right\|^2 \\
& + \tfrac{c_1}{2}\left\|\rho(e_2 + e_2\tilde{A}w_1 - Bw_1, \zeta)\right\|^2 \\
& + \tfrac{c_3}{2}\|w_1\|^2,
\end{aligned} \tag{7}$$

$$\begin{aligned}
\min_{w_2} f_2(w_2) = & \tfrac{1}{2}\left\|Bw_2 - e_2\tilde{B}w_2\right\|^2 \\
& + \tfrac{c_2}{2}\left\|\rho(e_1 + Aw_2 - e_1\tilde{B}w_2, \zeta)\right\|^2 \\
& + \tfrac{c_4}{2}\|w_2\|^2.
\end{aligned} \tag{8}$$

In this paper, we mainly use Newton-Armijo method for solving the problems (7) and (8).

2.3 Newton-Armijo method

Newton-Armijo method is one of the most popular iterative algorithms for solving unconstraint smooth optimization problems and has been shown to be quadratically convergent (see [15]). In order to use Newton-Armijo method, firstly we need to calculate the gradient vectors $\nabla f_i(w_i)$ and Hessian matrices $\nabla^2 f_i(w_i)$ of the objective functions of the problems (7) and (8):

$$\begin{aligned}
&\nabla f_1(w_1) = \\
&\quad c_1 \textstyle\sum_{i=1}^{m_2} \rho(z_{1i}, \zeta)\rho'(z_{1i}, \zeta)(\tilde{A}^T - x_i^{(2)}) \\
&\quad + (A - e_1\tilde{A})^T(A - e_1\tilde{A})w_1 + c_3 w_1, \\
&\nabla^2 f_1(w_1) = \\
&\quad c_1 \textstyle\sum_{i=1}^{m_2} (\rho'(z_{1i}, \zeta)^2 + \rho(z_{1i}, \zeta)\rho''(z_{1i}, \zeta)) \\
&\quad \cdot(\tilde{A}^T - x_i^{(2)})(\tilde{A}^T - x_i^{(2)})^T \\
&\quad + (A - e_1\tilde{A})^T(A - e_1\tilde{A}) + c_3 I, \\
&\nabla f_2(w_2) = \\
&\quad c_2 \textstyle\sum_{i=1}^{m_1} \rho(z_{2i}, \zeta)\rho'(z_{2i}, \zeta)(x_i^{(1)} - \tilde{B}^T) \\
&\quad + (B - e_2\tilde{B})^T(B - e_2\tilde{B})w_2 + c_4 w_2, \\
&\nabla^2 f_2(w_2) = \\
&\quad c_2 \textstyle\sum_{i=1}^{m_1} (\rho'(z_{2i}, \zeta)^2 + \rho(z_{2i}, \zeta)\rho''(z_{2i}, \zeta)) \\
&\quad \cdot(x_i^{(1)} - \tilde{B}^T)(x_i^{(1)} - \tilde{B}^T)^T \\
&\quad + (B - e_2\tilde{B})^T(B - e_2\tilde{B}) + c_4 I,
\end{aligned}$$

where $z_{1i} = 1 + \tilde{A}w_1 - (x_i^{(2)})^T w_1, z_{2i} = 1 - \tilde{B}w_2 - (x_i^{(1)})^T w_2$ and I is the identity matrix of appropriate dimension. Then we calculate the search direction d_t by Newton method (called Newton direction) and search stepsize λ_t by Armijo method (called Armijo stepsize) for t-th step iteration. The specific procedure is as follows, in which we only solve the problem (7). With the similar way, we can solve the problem (8).

Algorithm 1. The Newton-Armijo algorithm for solving linear SPTSVM

Step 1. Initialization. For given parameter values c_1, c_3 and the maximum number of iterations T, let $t = 0$ and $\varepsilon > 0$ be small enough and take arbitrarily nonzero vector $w_1^t \in R^n$.

Step 2. Calculate Newton direction d_t by solving the system of linear equations $\nabla^2 f_1(w_1^t)d_t = -\nabla f_1(w_1^t)$.

Step 3. Calculate Armijo stepsize λ_t by inexact linear searching, that is, choose $\lambda_t = \max\left\{1, \frac{1}{2}, \frac{1}{4}, \cdots\right\}$ satisfying

$$f_1(w_1^t) - f_1(w_1^t + \lambda_t d_t) \geq -\tfrac{\lambda_t}{4}\nabla f_1(w_1^t)^T d_t.$$

Step 4. Update w_1^t. Calculate the next iterative point by formula $w_1^{t+1} = w_1^t + \lambda_t d_t$.

Step 5. If $\left\|w_1^{t+1} - w_1^t\right\| < \varepsilon$ or the maximum number of iterations T is achieved, stop iteration and take $w_1^* = w_1^{t+1}$; otherwise, put $t \leftarrow t + 1$ and return to step 2.

Step 6. The class label of a new input $x \in R^n$ is assigned by

$$\text{class}(x) = \arg\min_{i=1,2}\left|(w_i^*)^T x - (w_i^*)^T \frac{1}{m_i}\sum\nolimits_{j=1}^{m_i} x_j^{(i)}\right|.$$

3 Smooth approximation functions

In this section, we briefly overview eight smooth approximation functions for the plus function x_+, which are taken from [14-22], and describe the differentiability and error ranges of these approximation functions as five lemmas and one theorem. In all approximation functions, $\zeta > 0$ denotes the smooth parameter.

In 1980, Zhang [14] introduced a smooth approximation function as the integral of the sigmoid function for the plus function x_+, which is defined as follows.

$$\rho_1(x, \zeta) = x + \frac{1}{\zeta}\ln(1 + e^{-\zeta x}), \forall x \in R. \quad (9)$$

Later, this approximation function is used in many SVM models, such as [23-25]. It is evident that

$$\lim_{\zeta \to +\infty} \rho_1(x, \zeta) = x_+, \forall x \in R.$$

This indicates that the approximation effect will be better and better with the increase of the value of ζ. The first- and second-order derivatives of $\rho_1(x, \zeta)$ are respectively

$$\rho_1{}'(x, \zeta) = \frac{1}{1 + e^{-\zeta x}},$$

and

$$\rho_1{}''(x, \zeta) = \frac{\zeta e^{-\zeta x}}{(1 + e^{-\zeta x})^2},$$

where $\ln(\cdot)$ is the natural logarithms and e is the base of natural logarithms.

Lemma 3.1. [22] Let $\rho_1(x, \zeta)$ be defined by (9). Then

(1) $\rho_1(x, \zeta)$ is arbitrary order smooth with respect to x;

(2) $\rho_1(x, \zeta) \geq x_+, \forall x \in R$;

(3) for arbitrarily $k > 0$ with $|x| < k$, one has $\rho_1(x, \zeta)^2 - x_+^2 \leq (\frac{\ln 2}{\zeta})^2 + (\frac{2k}{\zeta})\ln 2$.

In 2005, Yuan et al. [15] proposed the following quadratic piecewise polynomial smooth approximation function for x_+ and got a quadratic polynomial smooth support vector machine (QPSSVM) model:

$$\rho_2(x, \zeta) = \begin{cases} x, x \geq \frac{1}{\zeta}, \\ \frac{\zeta}{4}x^2 + \frac{1}{2}x + \frac{1}{4\zeta}, -\frac{1}{\zeta} < x < \frac{1}{\zeta}, \\ 0, x \leq -\frac{1}{\zeta}. \end{cases} \quad (10)$$

The first- and second-order derivatives of $\rho_2(x, \zeta)$ are respectively

$$\rho_2{}'(x, \zeta) = \begin{cases} 1, x \geq \frac{1}{\zeta}, \\ \frac{\zeta}{2}x + \frac{1}{2}, -\frac{1}{\zeta} < x < \frac{1}{\zeta}, \\ 0, x \leq -\frac{1}{\zeta}, \end{cases}$$

and

$$\rho_2{}''(x, \zeta) = \begin{cases} \frac{\zeta}{2}, |x| < \frac{1}{\zeta}, \\ 0, |x| \geq \frac{1}{\zeta}. \end{cases}$$

In the same year, Yuan et al. [16] introduced the following fourth piecewise polynomial smooth approximation function for x_+ and got

a fourth polynomial smooth support vector machine (FPSSVM) model:

$$\rho_3(x,\zeta)=\begin{cases} x, x\geq \frac{1}{\zeta},\\ -\frac{1}{16\zeta}(\zeta x+1)^3(\zeta x-3), -\frac{1}{\zeta}<x<\frac{1}{\zeta},\\ 0, x\leq -\frac{1}{\zeta}. \end{cases} \tag{11}$$

The first- and second-order derivatives of $\rho_3(x,\zeta)$ are respectively

$$\rho_3{}'(x,\zeta)=\begin{cases} 1, x\geq \frac{1}{\zeta},\\ -\frac{1}{8}(\zeta x+1)^2(\zeta x-5), -\frac{1}{\zeta}<x<\frac{1}{\zeta},\\ 0, x\leq -\frac{1}{\zeta}, \end{cases}$$

and

$$\rho_3{}''(x,\zeta)=\begin{cases} -\frac{3}{8}\zeta(\zeta x+1)(\zeta x-3), |x|<\frac{1}{\zeta},\\ 0, |x|\geq \frac{1}{\zeta}. \end{cases}$$

Lemma 3.2 [16] Let $\rho_2(x,\zeta)$ and $\rho_3(x,\zeta)$ be defined by (10) and (11), respectively. Then

(1) $\rho_2(x,\zeta)$ is 1-order smooth and $\rho_3(x,\zeta)$ is 2-order smooth with respect to x;

(2) $\rho_2(x,\zeta)\geq x_+$ and $\rho_3(x,\zeta)\geq x_+$ for all $x\in R$;

(3) for any $x\in R$, one has $\rho_2(x,\zeta)^2-x_+^2\leq \frac{1}{11\zeta^2}$ and $\rho_3(x,\zeta)^2-x_+^2\leq \frac{1}{19\zeta^2}$.

In 2007, Xiong et al. [17] derived an important recursive equation (12) and proposed a class of smooth approximation functions using the interpolation technique.

$$\begin{cases} \rho_4^d(x,\zeta)=a\int I_{d-1}dx\\ I_{d-1}=\frac{x(x^2-\frac{1}{\zeta^2})^{d-1}}{2d-1}-\frac{2(d-1)}{(2d-1)\zeta^2}I_{d-2}, d=2,3,\cdots, \end{cases} \tag{12}$$

where d is the number of iterations and $a\in R$ is a parameter. For example, taking $d=2$ and $a=3\zeta^2$, we can calculate that

$$I_1=\tfrac{1}{3}x^3-\tfrac{1}{3\zeta^2}x-\tfrac{2}{3\zeta^2}=\tfrac{1}{3\zeta^2}(\zeta^2x^3-x-2),$$

then

$$\rho_4^2(x,\zeta)=\frac{\zeta^2}{4}x^4-\frac{1}{2}x^2-2x, \forall x\in R.$$

The first- and second-order derivatives of $\rho_4^2(x,\zeta)$ are respectively

$$(\rho_4^2)'(x,\zeta)=\zeta^2x^3-x-2,$$

and

$$(\rho_4^2)''(x,\zeta)=3\zeta^2x^2-1.$$

It notes that the larger the parameter d is, the higher the approximation accuracy is, but that will generate the additional computation cost. So, we only consider the case of $d=2$, that is, $\rho_4^2(x,\zeta)$.

In the same year, Yuan et al. [19] proposed a three-order spline interpolation polynomial approximation function and obtained a three-order spline smooth support vector machine (TSSVM) model:

$$\rho_5(x,\zeta)=\begin{cases} x, x>\frac{1}{\zeta},\\ -\frac{\zeta^2}{6}x^3+\frac{\zeta}{2}x^2+\frac{1}{2}x+\frac{1}{6\zeta}, 0<x\leq \frac{1}{\zeta},\\ \frac{\zeta^2}{6}x^3+\frac{\zeta}{2}x^2+\frac{1}{2}x+\frac{1}{6\zeta}, -\frac{1}{\zeta}<x\leq 0,\\ 0, x\leq -\frac{1}{\zeta}. \end{cases} \tag{13}$$

The first- and second-order derivatives of $\rho_5(x,\zeta)$ are respectively

$$\rho_5{}'(x,\zeta)=\begin{cases} 1, x>\frac{1}{\zeta},\\ -\frac{\zeta^2}{2}x^2+\zeta x+\frac{1}{2}, 0<x\leq \frac{1}{\zeta},\\ \frac{\zeta^2}{2}x^2+\zeta x+\frac{1}{2}, -\frac{1}{\zeta}<x\leq 0,\\ 0, x\leq -\frac{1}{\zeta}, \end{cases}$$

and

$$\rho_5{}''(x,\zeta)=\begin{cases} -\zeta^2|x|+\zeta>0, |x|<\frac{1}{\zeta},\\ 0, |x|\geq \frac{1}{\zeta}. \end{cases}$$

Lemma 3.3 [18] Let $\rho_5(x,\zeta)$ be defined by (13). Then

(1) $\rho_5(x,\zeta)$ is 2-order smooth with respect to x;

(2) $\rho_5(x,\zeta)\geq x_+, \forall x\in R$;

(3) for any $x\in R$, one has $\rho_5(x,\zeta)^2-x_+^2\leq \frac{1}{24\zeta^2}$.

In 2013, Wu et al. [19] introduced a three order piecewise polynomial approximation function:

$$\rho_6(x,\zeta)=\begin{cases} 0, x<-\frac{\zeta}{4},\\ \frac{8(x+\frac{\zeta}{4})^3}{3\zeta^2}, -\frac{\zeta}{4}\leq x\leq 0,\\ x+\frac{8(\frac{\zeta}{4}-x)^3}{3\zeta^2}, 0<x\leq \frac{\zeta}{4},\\ x, x>\frac{\zeta}{4}. \end{cases} \tag{14}$$

The first- and second-order derivatives of $\rho_6(x,\zeta)$ are respectively

$$\rho_6{}'(x,\zeta)=\begin{cases} 0, x<-\frac{\zeta}{4},\\ \frac{8(x+\frac{\zeta}{4})^2}{\zeta^2}, -\frac{\zeta}{4}\leq x\leq 0,\\ 1-\frac{8(x-\frac{\zeta}{4})^2}{\zeta^2}, 0<x\leq \frac{\zeta}{4},\\ 1, x>\frac{\zeta}{4}, \end{cases}$$

and

$$\rho_6{}''(x,\zeta)=\begin{cases}0, |x|>\frac{\zeta}{4},\\ \frac{16(\frac{\zeta}{4}-|x|)}{\zeta^2}\geq 0, |x|\leq\frac{\zeta}{4}.\end{cases}$$

Lemma 3.4 [19] Let $\rho_6(x,\zeta)$ be defined by (14). Then

(1) $\rho_6(x,\zeta)$ is 2-order smooth with respect to x;

(2) $\rho_6(x,\zeta)\geq x_+, \forall x\in R$;

(3) for any $x\in R$, one has $\rho_6(x,\zeta)^2-x_+^2\leq \frac{\zeta^2}{385}$.

In the same year, Ding et al. [20] introduced a cluster of polynomial approximation functions and obtained a polynomial smooth twin support vector regression:

$$\rho_7^n(x,\zeta)=\begin{cases}x, x\geq\frac{1}{\zeta},\\ \frac{1}{2\zeta}(\frac{1+\zeta^2x^2}{2}-\sum_{l=2}^n\frac{(2l-3)!!}{(2l)!!}(1+\zeta^2x^2)^l)\\ +\frac{x}{2}, |x|<\frac{1}{\zeta},\\ 0, x\leq-\frac{1}{\zeta},\end{cases}\tag{15}$$

where $n=2,3,\cdots$. The first- and second-order derivatives of $\rho_n(x,\zeta)$ are respectively

$$(\rho_7^n)'(x,\zeta)=\begin{cases}1, x\geq\frac{1}{\zeta},\\ \frac{\zeta x}{2}(1+\sum_{l=2}^n\frac{(2l-3)!!}{(2l)!!}(1-\zeta^2x^2)^{l-1})\\ +\frac{1}{2}, |x|<\frac{1}{\zeta},\\ 0, x\leq-\frac{1}{\zeta},\end{cases}$$

and

$$(\rho_7^n)''(x,\zeta)=\begin{cases}x, x\geq\frac{1}{\zeta},\\ \frac{\zeta}{2}(1+\sum_{l=2}^n\frac{(2l-3)!!}{(2l-2)!!}(1-\zeta^2x^2)^{l-1})\\ -\frac{\zeta^2x^2}{2}\sum_{l=2}^n\frac{(2l-3)!!}{(2l-4)!!}(1-\zeta^2x^2)^{l-2},\\ |x|<\frac{1}{\zeta},\\ 0, x\leq-\frac{1}{\zeta},\end{cases}$$

Lemma 3.5 [25] Let $\rho_7^n(x,\zeta)$ be defined by (15). Then

(1) $\rho_7^n(x,\zeta)$ is n-order smooth with respect to x;

(2) $\lim_{n\to\infty}\max(\rho_7^n(x,\zeta)-x_+)=0$.

In 2014, a quadratic polynomial smooth approximation function was proposed in [22] as follows:

$$\rho(x,\alpha)=\frac{1}{4|\alpha|}x^2+\frac{1}{2}x+\frac{|\alpha|}{4}, \forall x\in R,$$

where $\alpha\in R:\alpha\neq 0$ is a smooth parameter. If letting $\zeta=\sqrt{|\alpha|}>0$, one has

$$\begin{aligned}\rho_8(x,\zeta)&=\tfrac{1}{4\zeta^2}x^2+\tfrac{1}{2}x+\tfrac{\zeta^2}{4}\\ &=\tfrac{1}{4}(\tfrac{1}{\zeta}x+\zeta)^2,\ \forall x\in R.\end{aligned}\tag{16}$$

The first- and second-order derivatives of $\rho_8(x,\zeta)$ are respectively

$$\rho_8{}'(x,\zeta)=\frac{1}{2\zeta}(\frac{1}{\zeta}x+\zeta),$$

and

$$\rho_8{}''(x,\zeta)=\frac{1}{2\zeta^2}.$$

Theorem 3.1 Let $\rho_8(x,\zeta)$ be defined by (16). Then

(1) $\rho_8(x,\zeta)$ is 1-order smooth with respect to x;

(2) $\rho_8(x,\zeta)\geq x_+, \forall x\in R$;

(3) $\lim_{|x|\to\zeta^2}(\rho_8(x,\zeta)-x_+)=0$.

Proof. The first conclusion is obvious. Since

$$\rho_8(x,\zeta)-x_+=\begin{cases}\frac{1}{4}(\frac{1}{\zeta}x-\zeta)^2, x>0,\\ \frac{1}{4}(\frac{1}{\zeta}x+\zeta)^2, x\leq 0,\end{cases}\tag{17}$$

we can obtain the second conclusion. From (17), we can get

$$\lim_{x\to\zeta^2}(\rho_8(x,\zeta)-x_+)=\lim_{x\to\zeta^2}\frac{1}{4}(\frac{1}{\zeta}x-\zeta)^2=0$$

when $x>0$, and

$$\lim_{x\to-\zeta^2}(\rho_8(x,\zeta)-x_+)=\lim_{x\to\zeta^2}\frac{1}{4}(\frac{1}{\zeta}x+\zeta)^2=0$$

when $x\leq 0$, which indicates that the third conclusion is true.

4 The Influences for SPTSVM

In this section, in order to illustrate the influences of eight smooth approximation functions for linear SPTSVM, we perform a series of comparative experiments of binary classification problems on classification accuracy and running time by using 10 datasets taken from UCI database [26] and 6 datasets taken from NDC database [27] are listed in Table 1. In 10 UCI datasets, Iris, Vehicle, Waveform and Balance four datasets all have 3

Table 1: Description of NDC datasets

Dataset	Training data	Test data	Features
NDC-200	200	40	32
NDC-500	500	100	32
NDC-700	700	140	32
NDC-1000	1000	200	32
NDC-2000	2000	400	32
NDC-3000	3000	600	32

classes, we choose the later two classes for experiments, respectively.

All experiments are implemented in Matlab (7.11.0) R2010b environment on a PC with an Intel P4 processor (2.30 GHz) with 4 GB RAM and SPTSVM is implemented by Newton-Armijo algorithm, that is, Algorithm 1, and the five-fold cross-validation method. We know that the choice of parameters have great impact on the performance of a classifier, in order to facilitate comparison, we take $\varepsilon = 10^{-3}, T = 50$ in Algorithm 1 and $c_1 = c_2 = c_3 = c_4 = 1$ after grid searching from $\{2^{-8}, \cdots, 2^8\}$. The classification accuracy is defined by

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN},$$

where TP, TN, FP and FN denote the numbers of true positive, true negative, false positive and false negative, respectively.

The experiment results on 10 UCI datasets are listed in Table 2 and on 6 NDC datasets are listed in Table 3, in which the seventh smooth approximation function $\rho_7^n(x,\zeta)$ is taken as $\rho_7^4(x,\zeta)$. In addition, in order to explain the influence of order n for $\rho_7^n(x,\zeta)$, we perform comparative experiments with $n = 2, 3, 4$, respectively, and the experiment results are listed in Table 4. It should also be pointed out that the fourth smooth approximation function $\rho_4^d(x,\zeta)$ are not commonly used, so we only take $d = 2$, that is $\rho_4^2(x,\zeta)$.

From Table 2, we can see that on the classification accuracy

(1) $\rho_1(x,\zeta)$ is the best on Breast and Waveform two datasets and the next best on the rest datasets except to Liver dataset;

(2) $\rho_2(x,\zeta)$ and $\rho_3(x,\zeta)$ are completely the same, which are the best on Breast, Pima and Waveform three datasets, and are slightly worse than $\rho_1(x,\zeta)$ on the rest datasets, and are the worst on Liver dataset;

(3) $\rho_4^2(x,\zeta)$ is the best on Balance, Liver, Iris and Vehicle four datasets and is the worst on Breast, Pima and Waveform three datasets;

(4) $\rho_5(x,\zeta)$ is almost the same as $\rho_2(x,\zeta)$ and $\rho_3(x,\zeta)$;

(5) $\rho_7^4(x,\zeta)$ is the same as $\rho_2(x,\zeta)$ and $\rho_3(x,\zeta)$ except to Liver dataset and clearly better than $\rho_2(x,\zeta)$, $\rho_3(x,\zeta)$ and $\rho_5(x,\zeta)$ on Liver dataset;

(6) although $\rho_6(x,\zeta)$ and $\rho_8(x,\zeta)$ are the best on WBC and Vote two datasets, respectively, but generally speaking, $\rho_6(x,\zeta)$, $\rho_8(x,\zeta)$ and $\rho_7^4(x,\zeta)$ are comparable.

On the running time, $\rho_8(x,\zeta)$ costs the least time among these datasets except for Vehicle and Waveform two datasets and $\rho_4^2(x,\zeta)$ costs the most time except for Liver dataset.

From Table 3, we can see that

(1) $\rho_1(x,\zeta)$ has the highest classification accuracy on all datasets;

(2) $\rho_2(x,\zeta)$, $\rho_3(x,\zeta)$, $\rho_5(x,\zeta)$ and $\rho_7^4(x,\zeta)$ have the same the classification accuracies on all datasets, which are slightly lower than the corresponding classification accuracies of $\rho_1(x,\zeta)$, respectively;

(3) $\rho_6(x,\zeta)$ and $\rho_8(x,\zeta)$ have the almost same the classification accuracies on all datasets, which are comparable with the classification accuracies of $\rho_2(x,\zeta)$, $\rho_3(x,\zeta)$, $\rho_5(x,\zeta)$ and $\rho_7^4(x,\zeta)$;

(4) $\rho_4^2(x,\zeta)$ has the worst classification accuracies except for NDC-700 dataset and the longest running times except for NDC-700 and NDC-2000 datasets;

(5) $\rho_2(x,\zeta)$ and $\rho_3(x,\zeta)$ have the shortest running times on NDC-200, NDC-700 and NDC-3000 three datasets.

From Table 4, we can see that $\rho_7^2(x,\zeta)$, $\rho_7^3(x,\zeta)$ and $\rho_7^4(x,\zeta)$ have the same classification accuracies on 10 UCI datasets, just the running times are different, which indicates that the classification accuracy of SPTSVM may have only small changes with increasing of the order n.

On the basis of the above analysis, we can conclude that when choosing a smooth approximation function in order to improve the classification accuracy of SPTSVM, in general, we can firstly consider $\rho_1(x,\zeta)$, secondly consider one of $\rho_2(x,\zeta)$, $\rho_3(x,\zeta)$ and $\rho_5(x,\zeta)$, thirdly consider

Table 2: Comparison results on 10 UCI datasets

Dataset	$\rho_1(x,\zeta)$ Accuracy (%) Time(s)	$\rho_2(x,\zeta)$ Accuracy (%) Time(s)	$\rho_3(x,\zeta)$ Accuracy (%) Time(s)	$\rho_4^2(x,\zeta)$ Accuracy (%) Time(s)	$\rho_5(x,\zeta)$ Accuracy (%) Time(s)	$\rho_6(x,\zeta)$ Accuracy (%) Time(s)	$\rho_8(x,\zeta)$ Accuracy (%) Time(s)	$\rho_7^4(x,\zeta)$ Accuracy (%) Time(s)
Balance	91.2281	90.3509	90.3509	**92.9825**	90.3509	90.8772	91.2281	90.3509
(625 × 4)	1.6683	0.9795	1.1642	3.5855	0.8727	1.4629	0.7013	0.8203
Breast	**72.3637**	71.6464	71.6364	59.6364	72.0000	71.6364	70.5455	71.6364
(277 × 9)	0.8860	0.6187	0.7097	1.6413	0.5080	0.6294	0.4153	1.0817
Heart	82.5000	**82.9167**	**82.9167**	75.83333	**82.9167**	81.6667	81.6667	**82.9167**
(303 × 13)	0.8132	1.8574	0.5638	1.5075	0.5008	0.6890	0.4611	0.9584
Pima	76.0784	**76.2092**	**76.2092**	72.9412	**76.2092**	75.2941	75.4248	**76.2092**
(768 × 8)	1.9162	1.0741	1.2668	3.2495	1.0680	1.8637	0.7220	2.6805
Vote	96.0000	94.0000	94.0000	94.0000	94.0000	96.0000	**96.50000**	94.0000
(435 × 16)	0.6796	0.5051	0.5667	1.3384	0.4773	0.5500	0.4432	1.2900
Liver	59.6552	53.7931	53.7931	**62.7586**	54.1379	60.0000	60.3448	60.0000
(345 × 6)	1.5990	1.0328	1.4103	0.9762	2.9713	0.8570	0.7598	1.3702
WBC	97.1429	96.4706	96.4706	95.1261	96.4706	**97.3109**	44.0336	96.4706
(600 × 9)	4.1595	0.7249	0.7241	1.9256	0.6794	1.1052	0.5944	1.7458
Iris	93.0000	93.0000	93.0000	**96.0000**	93.0000	92.0000	92.0000	93.0000
(150 × 4)	0.1968	0.2665	0.1788	0.5141	0.2774	0.2170	0.2155	0.2750
Vehicle	86.0000	86.0000	86.0000	**93.0000**	86.0000	86.0000	86.0000	86.0000
(150 × 18)	0.5104	0.4405	0.3836	1.4558	0.4668	0.5163	0.5629	0.6887
Waveform	**93.0000**	**93.0000**	**93.0000**	90.0000	**93.0000**	92.0000	92.0000	**93.0000**
(150 × 21)	0.8221	0.4066	0.7379	0.9567	0.7906	0.4984	0.4839	0.5858

Table 3: Comparison results on NDC datasets with eight smoothing approximation functions

Dataset	$\rho_1(x,\zeta)$ Accuracy (%) Time(s)	$\rho_2(x,\zeta)$ Accuracy (%) Time(s)	$\rho_3(x,\zeta)$ Accuracy (%) Time(s)	$\rho_4^2(x,\zeta)$ Accuracy (%) Time(s)	$\rho_5(x,\zeta)$ Accuracy (%) Time(s)	$\rho_6(x,\zeta)$ Accuracy (%) Time(s)	$\rho_8(x,\zeta)$ Accuracy (%) Time(s)	$\rho_7^4(x,\zeta)$ Accuracy (%) Time(s)
NDC-200	**94.3590**	92.3077	92.3077	82.0513	92.3077	91.2821	91.2821	92.3077
	0.7782	0.5599	0.5541	1.0489	0.6099	0.8651	0.5971	0.6719
NDC-500	**93.9394**	92.3232	92.3232	91.7172	92.3232	93.1313	92.7273	92.3232
	2.2987	1.2045	1.3064	3.0931	1.3178	1.3987	1.1931	1.7377
NDC-700	**95.8571**	94.7143	94.7143	95.7143	94.7143	95.0000	94.7143	94.7143
	2.4246	1.3848	1.1932	3.9989	6.1389	5.8373	4.3381	2.1963
NDC-1000	**96.4824**	96.2814	96.2814	93.9698	96.2814	96.3819	95.9799	96.2814
	3.7086	4.3996	5.9918	19.0389	6.4249	7.9403	1.8923	2.1920
NDC-2000	**97.0500**	96.60000	96.6000	82.0513	96.6000	96.5000	96.5500	96.6000
	10.9988	3.2119	3.1624	3.0184	12.0087	28.1650	15.6249	9.2376
NDC-3000	**97.5626**	97.0284	97.0284	95.8264	97.0284	97.2621	97.0952	97.0284
	27.7454	16.1330	17.0292	47.5982	33.4184	34.4930	19.2605	37.2665

Table 4: Comparison results with ρ_7^2, ρ_7^3 and ρ_7^4

Dataset	$\rho_7^2(x,\zeta)$ Accuracy(%) Time(s)	$\rho_7^3(x,\zeta)$ Accuracy(%) Time(s)	$\rho_7^4(x,\zeta)$ Accuracy(%) Time(s)
Balance	90.3509	90.3509	90.3509
(625 × 4)	1.1242	1.9469	0.8203
Breast	71.6364	71.6364	71.6364
(277 × 9)	0.4381	0.6462	1.0871
Heart	82.9167	82.9167	82.9167
(303 × 13)	0.7610	1.4943	0.9585
Pima	76.2092	76.2092	76.2092
(768 × 8)	2.0812	2.1044	2.6805
Vote	94.0000	94.0000	94.0000
(435 × 16)	0.7866	0.7216	1.2900
Liver	60.0000	60.0000	60.0000
(345 × 6)	1.5035	1.3319	1.3702
WBC	96.4706	96.4706	96.4706
(600 × 9)	1.5481	1.5814	1.7459
Iris	93.0000	93.0000	93.0000
(150 × 4)	0.2557	0.2673	0.2750
Vehicle	86.0000	86.0000	86.0000
(150 × 18)	0.5719	0.6552	0.6887
Waveform	93.0000	93.0000	93.0000
(150 × 21)	0.6079	0.6053	0.5858

one of $\rho_6(x,\zeta)$, $\rho_8(x,\zeta)$ and $\rho_7^n(x,\zeta)$ and finally consider $\rho_4^2(x,\zeta)$. Of course, different smooth approximation functions will bring different classification accuracies. So, we should choose a suitable smooth approximation function for underlying dataset.

5 Conclusions

In this paper, we study the influence of eight smooth approximation functions for SPTSVM on classification accuracy and running time by means of 10 UCI datasets and 6 NDC datasets. From experiment results, we can get a choice order of the eight approximation functions in generally. But we know that different approximation functions may bring different classification accuracies, we should choose a suitable smooth approximation function for underlying dataset.

As stated in Introduction, GEPSVM, TSVM and PTSVM are three representative methods of NHSVM and all the other NHSVM methods are improved versions based on them. In this paper, we only discuss the influence of eight known smooth approximation functions for smooth version of PTSVM and only discuss the linear version of SPTSVM. In the next step of work, we should do: firstly, we will investigate the influence of these approximation functions for smooth versions of GEPSVM and TSVM, respectively; secondly, we will consider the nonlinear version of SPTSVM; thirdly, we will be committed to finding more smooth approximation functions and compare the accuracies of approximating with them to the plus function.

References:

[1] O.L. Mangasarian, E.W. Wild, Multisurface proximal support vector machine classification via generalized eigenvalues, *IEEE Trans Pattern Anal Mach Intell*. 28(1), 2006, pp. 69-74.

[2] Jayadeva, R. Khemchandani, S. Chandra, Twin support vector machines for pattern classification, *IEEE Trans Pattern Anal Mach Intell*. 29(5), 2007, pp. 905-910.

[3] X. Chen, J. Yang, Q. Ye, J. Liang, Recursive projection twin support vector machine via withinclass variance minimization, *Pattern Recognition*. 44, 2011, pp. 2643-2655.

[4] Y.H. Shao, C.H. Zhang, X.B. Wang, N.Y. Deng, Improvements on twin support vector machines, *IEEE Transactions on Neural Networks*. 22(6), 2011, pp. 962-968.

[5] Q. Ye, C. Zhao, N. Ye, Y. Chen, Multi-weight vector projection support vector machines, *Pattern Recognition Letters*. 31 (13), 2010, pp. 2006-2011.

[6] Y.H. Shao, N.Y. Deng, Z.M. Yang, Least squares recursive projection twin support vector machine for classification, *Pattern Recogn*. 45(6), 2012, pp. 2299-2307.

[7] Y.H. Shao, W.J. Chen, W.B. Huang, Z.M. Yang, N.Y. Deng, The best separating decision tree twin support vector machine for multi-class classification, *Procedia Comput Sci*. 17, 2013, pp. 1032-1038.

[8] Y.H. Shao, Z. Wang, W.J. Chen, N.Y. Deng, A regularization for the projection twin support vector machine, *Knowl-Based Syst*. 37, 2013, pp. 203-210.

[9] Y.H. Shao, C.H. Zhang, Z.M. Yang, L. Jing, N.Y. Deng, An etwin support vector machine for regression, *Neural Comput Applic* 23(1), 2013, pp. 175-185.

[10] S.F. Ding, X.P. Hua, Recursive least squares projection twin support vector machines for nonlinear classification, *Neurocomputing*. 130(23), 2014, pp. 3-9.

[11] S.F. Ding, X.P. Hua, J.Z. Yu, An overview on nonparallel hyperplane support vector machine algorithms, *Neural Comput and Applic*. 25, 2014, pp. 975-982.

[12] M. Arun Kumar, M. Gopal, Application of smoothing technique on twin support vector machine, *Pattern Recogration Letters*. 29, 2008, pp. 1842-1848.

[13] X.X. Zhang, L.Y. Fan, Application of smoothing technique on projective TSVM, *International Journal of Applied Mathematics and Machine Learning*. 2(1), 2015, pp. 27-45.

[14] I. Zhang, A smoothing-out technique for min-max optimization, *Math. Program*. 19, 1980, pp. 61-77.

[15] Y.B. Yuan, J. Yan, C.X. Xu, Polynomial smooth support vector machine, *Chinese Journal of Computers*. 28(1), 2005, pp. 9-17.

[16] Y.B. Yuan, T.Z. Huang, A polynomial smooth support vector machine for classification, *Proceedings of the 1st International Conference on Advanced Data Mining and Applications (ADMA'05)*. 2005, pp.157-164.

[17] J.Z. Xiong, J.L. Hu, H.Q. Yuan, Research on a new class of functions for smoothing support vector machines, *Acta Electronica Sinica*. 35(2), 2007, pp. 366-370.

[18] Y.B. Yuan, W.G. Fan, D.M. Pu, Spline function smooth support vector machine for classification, *Journal of Industrial and Management Optimization*. 3(3), 2007, pp. 529-542.

[19] Q. Wu, J.L. Fan, Smooth support vector machine based on piecewise function, *ScienceDirect*. 20(5), 2013, pp. 122-128.

[20] S.F. Ding, H.J. Huang, R. N, Forecasting method of stock price based on polynomial smooth twin support vector regression, *Springer-Verlag Berlin Heidelberg*. 7995, 2013, pp. 96-105.

[21] S. Balasundaram, Deepak Gupta, Kapil, Lagrangian support vector regression via unconstrained convex minimization, *Neural Networks*. 51, 2014, pp. 67-79.

[22] Lee, Y.J. Mangasarian, A smooth support vector machine for classification, *Computational Optimization and Applications*. 20(1), 2001, pp. 5-22.

[23] X. Chen, J. Yang, J. Liang, Q. Ye, Smooth twin support vector regression, *Neural Computation*. 21(3), 2012, pp. 505-513.

[24] Z. Wang, Y. Shao, T. Wu, A GA-based model selection for smooth twin parametric-margin support vector machine, *Pattern Recogration*. 46, 2013, pp. 2267-2277.

[25] Y.Q. Liu, S.Y. Liu, M.G. Gu, Self-training polynomial support smooth semi-supervised support vetor machines, *Journal of System Simulation*. 21(18), 2009, pp. 5740-5743.

[26] C.L. Blake, C.J. Merz, UCI Repository for Machine Learning Databases, 1998. http://www.ics.uci.edu/mlearn/MLRepository.html.

[27] D.R. Musicant, NDC: Normally distributed clustered datasets, 1998. http://www.cs.wisc.edu/ musicant/data/ ndc.