Thai Journal of **Math**ematics
Volume 18 Number 4 (2020)
Pages 2107–2127

http://thaijmath.in.cmu.ac.th

ISSN 1686-0209

A Novel Twin Parametric Support Vector Machine for Large Scale Problem

Dawarwee Makmuang[1], **Rabian Wangkeeree**[1,2,*], **Cholwich Nattee**[3] **and Nirattaya Khamsemanan**[3]

[1] *Department of Mathematics, Faculty of Science, Naresuan University, Phitsanulok 65000, Thailand*
e-mail : dawrawee@gmail.com (D. Makmuang); rabianw@nu.ac.th (R. Wangkeeree)

[2] *Research Center for Academic Excellence in Mathematics, Naresuan University, Phitsanulok 65000, Thailand*

[3] *Sirindhorn International Institute of Technology, Thammasat University, Pathum Thani 12120, Thailand*
e-mail : cholwich@siit.tu.ac.th (C. Nattee); nirattaya@siit.tu.ac.th (N. Khamsemanan)

Abstract In this paper, we propose a stochastic gradient descent algorithm, called stochastic gradient descent method-based generalized pinball twin parametric support vector machine (SG-TPSVM) to solve data classification problems. This approach is developed by replacing hinge loss function in the conventional twin parametric support vector machine (TPSVM) with a generalized pinball loss function. Moreover, the numerical experiment solved by proposed method is shown.

MSC: 49K35; 47H10; 20M12
Keywords: generalized pinball loss function; large scale problems; stochastic gradient descent method; twin parametric support vector machine

Submission date: 05.03.2020 / Acceptance date: 17.04.2020

1. Introduction

Support vector machine (SVM) is a popular supervised binary classification algorithm based on statistical learning theory. SVM [1] searches parallel hyperplanes with the maximum margin between two classes of samples by minimizing structural as well as empirical risks evaluated from the given labeled training data. Within a few years after its introduction the SVM has already outperformed most other systems in a wide variety of applications like financial forecasting [2], human activity recognition [3], bioinformatics [4], and financial regression [5–8], etc.

On the one hand, a main challenge for the standard SVM is the high computational complexity of training samples, i.e. $O(m^3)$, where m is a total number of training samples, due to the standard SVM solved a single larger-sized convex quadratic programming problem (QPP) to find an optimal separating hyperplane. In contrast to the aforementioned idea of generate separating hyperplane in SVMs, Jayadeva [9] proposed a twin

*Corresponding author.

support vector machine (TSVM) that generating two nonparallel hyperplanes such that each hyperplane is closer to one of two classes and is at least one far from the other classes. The strategy of TSVM aimed at solving a pair of smaller sized QPPs, instead of solving a large one as in the classical SVM. Therefore, it makes the computational time of TSVM approximately four times faster than the standard SVM in theory. Many various of the TSVMs have been proposed, such as least square twin support vector machine [10, 11], twin support vector machine regression [12, 13], twin parametric support vector machine (TPMSVM) [14, 15], etc. Although the various forms of TSVM works faster than the standard SVM, these are not able to handle a very large number of data samples during training as solving the corresponding QPP becomes infeasible. This is due to the computation of the inverse of a large matrix, in Lagrangian dual problems of TSVMs, which needed for all dual solutions. In order to deal with the large scale problem, many improvements were proposed, such as, for SVM, sequential minimal optimization, coordinate decent method, trust region Newton and stochastic gradient descent algorithm (SGD) in [16–20], and for TSVMs, successive overrelaxation technique, Newton-Armijo algorithm, and dual coordinate decent method in [21–23]. In the spirit of the stochastic gradient algorithm (SGD), Wang [24] recently proposed a stochastic gradient twin support vector machine (SG-TSVM). This technique partitions a large scale problem into a series of subproblems by stochastic sampling with a suitable size. The SG-TSVM has been shown to be the fastest method among the TSVM-type classifiers for large scale problems.

On the other hand, the well known and common loss function that used in SVMs and TSVMs model are hinge loss functions, which makes it essentially sensitive to noise, and is not stable for resampling. The SVM model with pinball loss function (Pin-SVM) was proposed by Huang [25] to treat noise sensitivity and instability to re-sampling. However, the Pin-SVM loses sparsity needs to be corrected. To achieve sparsity, they also proposed a modified ϵ-insensitive zone into the Pin-SVM. Although ϵ-insensitive zone Pin-SVM improves loses sparsity of Pin-SVM, its formulation requires the value of ϵ to be specified beforehand and therefore a bad choice may effect its performance. Taking motivation from these developments, Reshma [26] proposed a modified (ϵ_1, ϵ_2)-insensitive zone Pin-SVM, called generalized pinball loss SVM. This generalized pinball loss SVM model is noise-insensitive as well as sparse in the solution obtained. Nevertheless, compared with the TSVMs, the generalized pinball loss SVM still needs to solve a single large QPP, which leads to a higher computational complexity and not capable to solve large scale problem.

In order to overcome the above-mentioned limitations of large scale problem and inspired by the studies of the TSVMs and the generalized pinball loss function, we formulate a twin parametric support vector machine model as a convex unconstrained minimization problem. Further, we propose to use the stochastic gradient descent algorithm for computing the solution to the above-mentioned convex unconstrained minimization problem. The proposed technique is an efficient algorithm for real-world datasets, especially large-scale ones. Moreover, we show that SG-TPSVM enjoys an expected convergence rate similar to the rate of SG-TSVM [24]. Finally, we show, by a number of numerical experiments, that the proposed SG-TPSVM approach outperforms the existing approaches in term of accuracy. The results also show the robustness of the proposed approach on noises and re-sampling.

The layout of the paper is as follows. Section 2, describes related works that forms the basis of our proposed work. In Section 3, we introduce the SG-TPSVM with the theoretical analysis. Section 4 reports experimental results on several machine learning benchmark datasets. Section 5 concludes the paper.

2. Background

The purpose of this section is to review related methods for binary classification problems. Let us consider a binary classification problem in the n-dimensional Euclidean space $\mathbb{R}^n$. Here we denote the set of training samples by $X \in \mathbb{R}^{m\times n}$, where $\mathbf{x} \in \mathbb{R}^n$ is a sample with a label $y \in \{+1, -1\}$. Note that we establish m_1 samples in class $+1$ into a matrix $A \in \mathbb{R}^{m_1\times n}$ and m_2 samples of class -1 into a matrix $B \in \mathbb{R}^{m_2\times n}$. Below, we give a brief outline of several related methods.

2.1. Support Vector Machine

The SVM model consists of maximizing the distance between the two bounding hyperplanes which bound their classes so that the SVM model is generally formulated as a convex quadratic programming problem (QPP). Let $\|\cdot\|$ denote the m_2-norm of a vector in $\mathbb{R}^n$. Given a training set $T = \{(\mathbf{x}_i^\top, y_i) \in \mathbb{R}^n \times \{1, -1\} : i = 1, 2, 3, \ldots, m\}$, the SVM model is expressed in the form

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i, \\ \text{subject to} \quad & y_i(\mathbf{x}_i^\top\mathbf{w} + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, \quad i = 1, 2, 3, \ldots, m. \end{aligned} \tag{2.1}$$

where, $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are the weight vector and bias, respectively, which define the hyperplane $f(\mathbf{x}) : \mathbf{w}^\top\mathbf{x} + b = 0$, $C > 0$ is a constant parameter and $\xi \in \mathbb{R}^m$ is slack vectors. The decision function of the above formulation is based on the sign of $\mathbf{w}^\top\mathbf{x} + b$ where x is assigned to class $+1$ if the value is positive otherwise it is assigned to class -1. In fact, the SVM problem (2.1) can rewrited into unconstrained optimization problem as follows [27, P. 207]:

$$\min_{\mathbf{w},b} \ \frac{1}{2}\left(\|\mathbf{w}\|^2 + b^2\right) + C\sum_{i=1}^{m} L_{hinge}(1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b)), \tag{2.2}$$

where $L_{hinge}(1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b)) = \max\left\{0, 1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b)\right\}$. The function $L_{hinge}(\cdot)$ is a so-called hinge loss function. This loss is related to the shortest distance between the sets and the corresponding classifier leads to its sensitivity of noise and instability for resampling. To remediate this short coming, Huang [25] suggests using a pinball loss function to the SVM classifier (Pin-SVM). This approach brings noise insensitivity. The way this model works by penalizing correctly classified samples, which is evident by the pinball loss function which is defined as follows

$$L_\tau(u) = \begin{cases} u, & \text{if } \ u \geq 0, \\ -\tau u, & \text{if } \ u < 0, \end{cases} \tag{2.3}$$

where $u = 1 - y_i(\mathbf{w}^\top\mathbf{x}_i + b)$ and $\tau \geq 0$ is a user-defined parameter. However, the above formulation attains noise- insensitivity which, in the process, it loses sparsity. This is

because the pinball loss functions sub-gradient is non-zero almost everywhere. Therefore, to get the sparsity back, in the same paper [25] proposed an ϵ-insensitive zone to Pin-SVM (ϵ-insensitive zone pinball SVM), then sub-gradient of the this loss function turns out to be zero in the range $[\tau, \epsilon]$ providing sparsity to the model. Definition of pinball loss with ϵ-insensitive zone as follows

$$L_{\tau}^{\epsilon}(u) = \begin{cases} u - \epsilon, & \text{if } \ u > \epsilon, \\ 0, & \text{if } \ -\frac{\epsilon}{\tau} \leq u \leq \epsilon, \\ -\tau(u + \frac{\epsilon}{\tau}), & \text{if } \ u < -\frac{\epsilon}{\tau}, \end{cases} \tag{2.4}$$

where $u = 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$ and $\epsilon \geq 0$, $\tau \geq 0$ are user-defined parameters. However, the ϵ-insensitive zone pinball SVM formulation requires an optimal choice of ϵ parameter needs to be prescribed. Therefore, Reshma [26] proposed the ϵ-insensitive zone Pin-SVM idea to develop the generalized pinball loss SVM. This technique allows asymmetric insensitive zone by allowing ϵ to be different. The generalized pinball loss function is a generalization of other exiting loss functions which handle the problems of noise insensitivity and instability of re-sampling as well. The definition of generalized pinball loss function is given as follows

$$L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}(u) = \begin{cases} \tau_1(u - \frac{\epsilon_1}{\tau_1}), & \text{if } \ u > \frac{\epsilon_1}{\tau_1}, \\ 0, & \text{if } \ -\frac{\epsilon_2}{\tau_2} \leq u \leq \frac{\epsilon_1}{\tau_1}, \\ -\tau_2(u + \frac{\epsilon_2}{\tau_2}), & \text{if } \ u < -\frac{\epsilon_2}{\tau_2}, \end{cases} \tag{2.5}$$

where $u = 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)$ and $\tau_1, \tau_2, \epsilon_1, \epsilon_2 \geq 0$ are user-defined parameters. With a generalized pinball loss function, the resulting formulation of SVM, leads to the following constrained optimization problem:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i, \\ \text{subject to} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \frac{1}{\tau_1}(\xi_i - \epsilon_1), \\ & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \leq 1 + \frac{1}{\tau_2}(\xi_i + \epsilon_2), \\ & \xi_i \geq 0, \quad i = 1, 2, 3, \ldots, m, \end{aligned} \tag{2.6}$$

where $C > 0$ is a constant parameter and $\xi \in \mathbb{R}^m$ is slack vectors. This problem can be formulated as an unconstrained optimization problem given as follows:

$$\min_{\mathbf{w},b} \ \frac{1}{2}\left(\|\mathbf{w}\|_2^2 + b^2\right) + C\sum_{i=1}^{m} L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)), \tag{2.7}$$

where $L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}(\cdot)$ is a generalized pinball loss function.

2.2. Twin Support Vector Machine (TSVM)

To reduce the computational complexity of SVM, Jayadeva et al. [9] proposed the twin support vector machine (TSVM) based on hinge loss function. The formulation of TSVM

solves two smaller QPPs. The TSVM classifier aims to determine two nonparallel planes:

$$f_1(\mathbf{x}) : \mathbf{w}_1^\top \mathbf{x} + b_1 = 0 \text{ and } f_2(\mathbf{x}) : \mathbf{w}_2^\top \mathbf{x} + b_2 = 0, \tag{2.8}$$

where, $\mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^n$ and $b_1, b_2 \in \mathbb{R}$ are the weight vectors and biases of the hyperplane $f_1(\mathbf{x})$ and the hyperplane $f_2(\mathbf{x})$, respectively. To find the pair of nonparallel hyperplanes, it is necessary to obtain solutions to constrained optimization problems:

$$\begin{aligned} \min_{\mathbf{w}_1, b_1, \boldsymbol{\xi}_2} \quad & \frac{1}{2}(A\mathbf{w}_1 + \mathbf{e}_1 b_1)^\top (A\mathbf{w}_1 + \mathbf{e}_1 b_1) + c_1 \mathbf{e}_2^\top \boldsymbol{\xi}_2, \\ \text{subject to} \quad & -(B\mathbf{w}_1 + \mathbf{e}_2 b_1) + \boldsymbol{\xi}_2 \geq \mathbf{e}_2, \\ & \boldsymbol{\xi}_2 \geq \mathbf{0}, \end{aligned} \tag{2.9}$$

and

$$\begin{aligned} \min_{\mathbf{w}_2, b_2, \boldsymbol{\xi}_1} \quad & \frac{1}{2}(B\mathbf{w}_2 + \mathbf{e}_2 b_2)^\top (B\mathbf{w}_2 + \mathbf{e}_2 b_2) + c_2 \mathbf{e}_1^\top \boldsymbol{\xi}_1, \\ \text{subject to} \quad & (A\mathbf{w}_2 + \mathbf{e}_1 b_2) + \boldsymbol{\xi}_1 \geq \mathbf{e}_1, \\ & \boldsymbol{\xi}_1 \geq \mathbf{0}, \end{aligned} \tag{2.10}$$

where $c_1, c_2 > 0$ are the penalty parameters, and $\boldsymbol{\xi}_1 \in \mathbb{R}^{m_1}$, $\boldsymbol{\xi}_2 \in \mathbb{R}^{m_2}$ are slack vectors. However, within the large scale classification problem, the stochastic gradient descent method has been successfully applied to support vector machines (SVMs). Recently, Wang [24] proposed a stochastic gradient TSVM (SG-TSVM) to solve (2.9) and (2.10) in TWSVM. Note that these problems can be represented in the following problems:

$$\min_{\mathbf{w}_1, b_1} \frac{1}{2}(\|\mathbf{w}_1\|^2 + b_1^2) + \frac{c_1}{2m_1}\|A\mathbf{w}_1 + b_1\|^2 + \frac{c_2}{m_2} e_2^\top L_{hinge}(\mathbf{e}_2 + B\mathbf{w}_1 + \mathbf{e}_2 b_1), \tag{2.11}$$

and

$$\min_{\mathbf{w}_2, b_2} \frac{1}{2}(\|\mathbf{w}_2\|^2 + b_2^2) + \frac{c_3}{2m_2}\|B\mathbf{w}_2 + b_2\|^2 + \frac{c_4}{m_1} e_1^\top L_{hinge}(\mathbf{e}_1 - A\mathbf{w}_2 - \mathbf{e}_1 b_2), \tag{2.12}$$

where $c_1, c_2, c_3, c_4 > 0$ are the penalty parameters. In the t-th iteration, SG-TSVM constructs a pair of momentary functions using subgradients with respect to $\mathbf{w}_1, b_1, \mathbf{w}_2$ and b_2 which is defined by a pair of samples $(\mathbf{x}_t^+, y_t^+)$ and $(\mathbf{x}_t^-, y_t^-)$ from two classes and iteratively updates the decision parameters with some predefined step sizes.

However, the SG-TSVM that based on a hinge loss function leads to its sensitivity of the noise and instability for re-sampling. Moreover, the SG-TSVM might be unprofitable if the amont of noise strongly depends on the input value.

2.3. Twin Parametric-Margin Support Vector Machine (TPSVM)

The classical SVM and TSVM assume that the probability distribution of training data is distributed uniformly. However, this probability distribution assumption is not always satisfied. To resolve this problem, the concept of TPSVM [15] was developed. The process of TPSVM is generating two nonparallel hyperplanes similar to the TSVM. Each hyperplane determines one of the parametric-margin hyperplanes. Note that $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ (defined in (2.8)) are positive and negative parametric-margin hyperplanes,

respectively. Therefore, the TPSVM separates data if and only if:

$$\mathbf{w}_1^\top \mathbf{x}_i + b_1 \geq 0 \qquad \forall i = 1, 2, 3, ..., m_1, \tag{2.13}$$

$$\mathbf{w}_2^\top \mathbf{x}_i + b_2 \leq 0 \qquad \forall i = 1, 2, 3, ..., m_2. \tag{2.14}$$

In order to obtain the positive and negative parametric-margin hyperplanes, let us consider the following pair of constrained optimization problems:

$$\begin{aligned} \min_{\mathbf{w}_1, b_1, \boldsymbol{\xi}_1} \quad & \frac{1}{2}\|\mathbf{w}_1\|^2 + \frac{\nu_1}{m_2}\mathbf{e}_2^\top \Big(B\mathbf{w}_1 + \mathbf{e}_2 b_1\Big) + \frac{c_1}{m_1}\mathbf{e}_1^\top \boldsymbol{\xi}_1, \\ \text{subject to} \quad & A\mathbf{w}_1 + b_1\mathbf{e}_1 \geq \mathbf{0} - \boldsymbol{\xi}_1, \\ & \boldsymbol{\xi}_1 \geq \mathbf{0}, \end{aligned} \tag{2.15}$$

and

$$\begin{aligned} \min_{\mathbf{w}_2, b_2, \boldsymbol{\xi}_2} \quad & \frac{1}{2}\|\mathbf{w}_2\|^2 + \frac{\nu_2}{m_1}\mathbf{e}_1^\top \Big(A\mathbf{w}_2 + \mathbf{e}_1 b_2\Big) + \frac{c_2}{m_2}\mathbf{e}_1^\top \boldsymbol{\xi}_2, \\ \text{subject to} \quad & B\mathbf{w}_2 + b_2\mathbf{e}_2 \leq \mathbf{0} + \boldsymbol{\xi}_2, \\ & \boldsymbol{\xi}_2 \geq \mathbf{0}, \end{aligned} \tag{2.16}$$

where $c_1, c_2, \nu_1, \nu_2 > 0$ are the penalty parameters, and $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2$ are slack vectors. As mentioned in [15], this TPMSVM is suitable for the data with a heteroscedastic error structure, that is, the noise strongly depends on the input value. However, it is worth noting that the TPSVM based on hinge loss function needs to be solved via quadratic programming problems (QPPs), which might lead to its sensitivity to noise, instability for re-sampling and its computational cost could leave a lot for large scale problems.

Therefore, in view of the SGD-based optimization in SG-TSVM [24] and motivated with generalized pinball loss SVM and twin parametric-margin support vector machine (TPSVM), we formulate a twin parametric support vector machine (TPMSVM) model which is based on generalized pinball loss function by using an iterative method such as stochastic gradient descent (SG-TPSVM).

3. Proposed Work

3.1. Linear SG-TPSVM

Following the method of formulating the TPSVM problems (discussed in (2.15) and (2.16)), we incorporate the generalized pinball loss function in the objective function to get the convex unconstrained minimization problems as follows:

$$\min_{\boldsymbol{\omega}_1} \ \frac{1}{2}\|\boldsymbol{\omega}_1\|^2 + \frac{\nu_1}{m_2}\sum_{j=1}^{m_2}\Big(\boldsymbol{\omega}_1^\top \mathbf{x}_j^-\Big) + \frac{c_1}{m_1}\sum_{i=1}^{m_1} L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}\Big(0 - y_i^+(\boldsymbol{\omega}_1^\top \mathbf{x}_i^+)\Big), \tag{3.1}$$

$$\min_{\boldsymbol{\omega}_2} \ \frac{1}{2}\|\boldsymbol{\omega}_2\|^2 - \frac{\nu_2}{m_1}\sum_{i=1}^{m_1}\Big(\boldsymbol{\omega}_2^\top \mathbf{x}_i^+\Big) + \frac{c_2}{m_2}\sum_{j=1}^{m_2} L_{\tau_3,\tau_4}^{\epsilon_3,\epsilon_4}\Big(0 - y_j^-(\boldsymbol{\omega}_2^\top \mathbf{x}_j^-)\Big), \tag{3.2}$$

where $\boldsymbol{\omega}_1 = [\mathbf{w}_1 \quad b_1]$, $\boldsymbol{\omega}_2 = [\mathbf{w}_2 \quad b_2]$, $\mathbf{x} = [\mathbf{x} \quad 1]$ and $\mathbf{w}_1, \mathbf{w}_2 \in \mathbb{R}^n$, $b_1, b_2 \in \mathbb{R}$ are the weight vectors and biases, respectively, and $c_1, c_2, \nu_1, \nu_2 > 0$ are the penalty parameters. For a linear case, the SG-TPSVM model finds two hyperplanes in $\mathbb{R}^n$ as follows:

$$f_1(\mathbf{x}) : \boldsymbol{\omega}_1^\top \mathbf{x} = 0 \quad \text{and} \quad f_2(\mathbf{x}) : \boldsymbol{\omega}_2^\top \mathbf{x} = 0. \tag{3.3}$$

Let us consider a model of stochastic optimization problems. Let $\theta_i = (\mathbf{x}_i^+, y_i^+)$ and $\tilde{\theta}_i = (\mathbf{x}_i^-, y_i^-)$ be a pair of training sample from class $+1$ and -1, respectively. The objective functions on these two distributions of training set θ_i and $\tilde{\theta}_i$ as follows:

$$f(\boldsymbol{\omega}_{1,i}, \theta_i) = \frac{1}{2}\|\omega_1\|^2 + \nu_1\Big(\boldsymbol{\omega}_1^\top \mathbf{x}_i^-\Big) + c_1 L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}\Big(0 - y_i^+(\boldsymbol{\omega}_1^\top \mathbf{x}_i^+)\Big), \tag{3.4}$$

$$f(\boldsymbol{\omega}_{2,i}, \tilde{\theta}_i) = \frac{1}{2}\|\boldsymbol{\omega}_2\|^2 - \nu_2\Big(\boldsymbol{\omega}_2^\top \mathbf{x}_i^+\Big) + c_2 L_{\tau_3,\tau_4}^{\epsilon_3,\epsilon_4}\Big(0 - y_i^-(\boldsymbol{\omega}_2^\top \mathbf{x}_i^-)\Big). \tag{3.5}$$

To apply the SGD algorithm, let $t > 0$ be an iteration count. We assume that at time t, a sample of random variable $\theta_t = \{(\mathbf{x}_t^+, +1)\}$ and $\tilde{\theta}_t = \{(\mathbf{x}_t^-, -1)\}$ are given. Let $\mathbf{s}_t$ and $\tilde{\mathbf{s}}_t$ be subgradients of $f(\boldsymbol{\omega}_{1,t}, \theta_t)$ and $f(\boldsymbol{\omega}_{2,t}, \theta_t)$ associated with the samples θ_t and $\tilde{\theta}_t$ at point $\boldsymbol{\omega}_{1,t}$ and point $\boldsymbol{\omega}_{2,t}$, respectively. That is,

$$\mathbf{s}_t \in \partial f(\boldsymbol{\omega}_{1,t}, \theta_t) = \boldsymbol{\omega}_{1,t} + \nu_1 \mathbf{x}_t^- - c_1 \partial L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}(0 - y_t^+(\boldsymbol{\omega}_1^\top \mathbf{x}_t^+)) y_t^+(\mathbf{x}_t^+), \tag{3.6}$$

$$\tilde{\mathbf{s}}_t \in \partial f(\boldsymbol{\omega}_{2,t}, \tilde{\theta}_t) = \boldsymbol{\omega}_{2,t} - \nu_2 \mathbf{x}_t^+ - c_2 \partial L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}(0 - y_t^-(\boldsymbol{\omega}_2^\top \mathbf{x}_t^-)) y_t^-(\mathbf{x}_t^-), \tag{3.7}$$

where

$$\partial L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}(u') = \begin{cases} \{-\tau_1\} & \text{if } u' > \frac{\epsilon_1}{\tau_1}, \\ [-\tau_1, 0] & \text{if } u' = \frac{\epsilon_1}{\tau_1}, \\ \{0\} & \text{if } -\frac{\epsilon_2}{\tau_2} \le u' \le \frac{\epsilon_1}{\tau_1}, \\ [0, \tau_2] & \text{if } u' = -\frac{\epsilon_2}{\tau_2}, \\ \{\tau_2\} & \text{if } u' < -\frac{\epsilon_2}{\tau_2}. \end{cases} \tag{3.8}$$

With the above notations and the existence of $l_t \in \partial L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}(0 - y_t^+(\boldsymbol{\omega}_1^\top \mathbf{x}_t^+))$ and $\tilde{l}_t \in \partial L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}(0 - y_t^-(\boldsymbol{\omega}_2^\top \mathbf{x}_t^-))$, sudgradients $\mathbf{s}_t$ and $\tilde{\mathbf{s}}_t$ can be rewritten as:

$$\mathbf{s}_t = \boldsymbol{\omega}_{1,t} + \nu_1 \mathbf{x}_t^- - c_1 l_t y_t^+(\mathbf{x}_t^+), \tag{3.9}$$

$$\tilde{\mathbf{s}}_t = \boldsymbol{\omega}_{2,t} - \nu_2 \mathbf{x}_t^+ - c_2 \tilde{l}_t y_t^-(\mathbf{x}_t^-). \tag{3.10}$$

The proposed SG-TPSVM starts from the initial $w_{1,t}$ and $w_{2,t}$. Then, the updates are as follows:

$$\boldsymbol{\omega}_{1,t+1} = \boldsymbol{\omega}_{1,t} - \eta_t s_t, \tag{3.11}$$

$$\boldsymbol{\omega}_{2,t+1} = \boldsymbol{\omega}_{2,t} - \eta_t \tilde{s}_t. \tag{3.12}$$

A new data sample $\mathbf{x} \in \mathbb{R}^n$ is assigned to class $y\,(y = -1, 1)$ depending on which of the two planes given by (3.3) it lies closest to, i.e.,

$$\bar{\mathrm{y}} = \arg\min_{i=1,2} \frac{|\bar{\mathbf{x}}^\top \mathbf{w}_i + b_i|}{\|\mathbf{w}_i\|}, \tag{3.13}$$

where $|.|$ denotes the absolute value and, hence, the procedure for training a SG-SPTPSVM model is summarized in Algorithm 1.

Algorithm 1 SG-TPSVM

Input: Positive class $A \in \mathbb{R}^{m_1 \times n}$, negative class $B \in \mathbb{R}^{m_2 \times n}$,
positive parameters c_1, c_2, ν_1, ν_2 and tolerance: $tol = 10^{-4}$

Set $\boldsymbol{\omega}_{1,1}$ and $\boldsymbol{\omega}_{2,1}$ to be zero;
while $\|\mathbf{s}_t\| \geq tol$ **do**
  Choose a pair of samples $\mathbf{x}_t^+$ and $\mathbf{x}_t^-$ at random from A and B, respectively.
  Compute stochastic gradient $\mathbf{s}_t$ using Eqs. (3.9)
  Update $w_{1,t+1}$ using Eqs. (3.11)
  $t = t + 1$
end
while $\|\tilde{\mathbf{s}}_t\| \geq tol$ **do**
  Choose a pair of samples $\mathbf{x}_t^+$ and $\mathbf{x}_t^-$ at random from A and B, respectively.
  Compute stochastic gradient $\tilde{\mathbf{s}}_t$ using Eqs. (3.10)
  Update $\boldsymbol{\omega}_{2,t+1}$ using Eqs. (3.12)
  $t = t + 1$
end

Output: $\boldsymbol{\omega}_1^* = \dfrac{1}{T}\sum_{t=1}^{T} \boldsymbol{\omega}_{1,t}$ and $\boldsymbol{\omega}_2^* = \dfrac{1}{T}\sum_{t=1}^{T} \boldsymbol{\omega}_{2,t}$

3.2. Non-Linear SG-TPSVM

By employing the kernel trick, we extend our SG-TPSVM to the non-linear case. Suppose that $K(\cdot,\cdot)$ is the predefined kernel function. The kernel counterparts of (3.1) and (3.2) can be formulated as

$$\min_{\boldsymbol{\omega}_1} \frac{1}{2}\|\boldsymbol{\omega}_1\|^2 + \frac{\nu_1}{m_2}\sum_{j=1}^{m_2}\Big(\boldsymbol{\omega}_1^\top K(\mathbf{x}_j^-, X)\Big) + \frac{c_1}{m_1}\sum_{i=1}^{m_1} L_{\tau_1,\tau_2}^{\epsilon_1,\epsilon_2}\Big(0 - y_i^+(\boldsymbol{\omega}_1^\top K(\mathbf{x}_i^+, X))\Big), \tag{3.14}$$

$$\min_{\boldsymbol{\omega}_2} \frac{1}{2}\|\boldsymbol{\omega}_2\|^2 - \frac{\nu_2}{m_1}\sum_{i=1}^{m_1}\Big(\boldsymbol{\omega}_2^\top K(\mathbf{x}_i^+, X)\Big) + \frac{c_2}{m_2}\sum_{j=1}^{m_2} L_{\tau_3,\tau_4}^{\epsilon_3,\epsilon_4}\Big(0 - y_j^-(\boldsymbol{\omega}_2^\top K(\mathbf{x}_j^-, X))\Big), \tag{3.15}$$

where $c_1, c_2, \nu_1, \nu_2 > 0$ are the penalty parameters and $K(\mathbf{x}, X) = [K(\mathbf{x}, X) \;\; e]$ are column augmented matrices. For a nonlinear case, the SG-TPSVM model finds two hyperplanes as the following:

$$f_1(\mathbf{x}) : \boldsymbol{\omega}_1^\top K(\mathbf{x}, X) = 0 \quad \text{and} \quad f_2(\mathbf{x}) : \boldsymbol{\omega}_2^\top K(\mathbf{x}, X) = 0, \tag{3.16}$$

where $X = \begin{bmatrix} A_{m_1 \times n} \\ B_{m_2 \times n} \end{bmatrix}$. The solution to the above-mentioned optimization problems (3.14) and (3.15) can be computed similar to the linear case using Algorithm 1. Furthermore, a new data point $\mathbf{x} \in \mathbb{R}^n$ is assigned to class $y\,(y = -1, +1)$ according to the equation (3.13).

4. Convergence Analysis

In this section, we analyze the convergence of the proposed SG-TPSVM model. For convenience, we only consider the first QPP (3.1) together with the SGD formulation of the linear SG-TPSVM. The conclusions for another QPP (3.2) and the nonlinear algorithm can be obtained similarly. From now we denote $f(\boldsymbol{\omega}_{1,t}, \theta_t) = f(\boldsymbol{\omega}_t)$. Given $\boldsymbol{\omega}_1$ and the step size $\eta_t = \frac{1}{t}$, $\boldsymbol{\omega}_{t+1}$ for $t \geq 1$ is updated by

$$\boldsymbol{\omega}_{t+1} = \boldsymbol{\omega}_t - \eta_t \mathbf{s}_t \tag{4.1}$$

i.e.,

$$\begin{aligned}\boldsymbol{\omega}_{t+1} &= \boldsymbol{\omega}_t - \eta_t \mathbf{s}_t \\ &= \boldsymbol{\omega}_t - \frac{1}{t}\Big(\boldsymbol{\omega}_t + \nu_1 \mathbf{x}_t^- - c_1 l_t y_t^+(\mathbf{x}_t^+)\Big) \\ &= (1 - \frac{1}{t})\boldsymbol{\omega}_t - \frac{1}{t}\Big(\nu_1 \mathbf{x}_t^- - c_1 l_t y_t^+(\mathbf{x}_t^+)\Big)\end{aligned} \tag{4.2}$$

To prove the convergence of **Algorithm 1**, we rely on the following lemma.

Lemma 4.1. *The sequence* $\{\|\boldsymbol{\omega}_t\| : t = 1, 2, \ldots\}$ *and the sequence* $\{\|\boldsymbol{s}_t\| : t = 1, 2, \ldots\}$ *have upper bounds, where* $\boldsymbol{\omega}_t$ *and* $\boldsymbol{s}_t$ *are defined by* (4.2) *and* (3.9)*, respectively.*

Proof. The formulation (4.2) can be rewritten as

$$\boldsymbol{\omega}_{t+1} = A_t \boldsymbol{\omega}_t + \frac{1}{t}\mathbf{v}_t \tag{4.3}$$

where $A_t = \Big(\frac{t-1}{t}\Big)I$, I is the identity matrix, and $\mathbf{v}_t = -\nu_1 \mathbf{x}_t^- + c_1 l_t y_t(\mathbf{x}_t^+)$. For $t \geq 2$, A_t is positive definite, and the largest eigenvalue λ_t of A_t is equal to $\dfrac{t-1}{t}$. From (4.3), we obtrain that

$$\boldsymbol{\omega}_{t+1} = \prod_{i=2}^{t} A_i \boldsymbol{\omega}_2 + \sum_{i=2}^{t} \bar{A}_i, \tag{4.4}$$

where

$$\bar{A}_i = \begin{cases} \dfrac{1}{i}\Big(\prod_{j=i+1}^{t} A_j\Big)\mathbf{v}_i & \text{if } \ i < t, \\ \dfrac{1}{i} v_i & \text{if } \ i = t. \end{cases} \tag{4.5}$$

For $i \geq 2$,

$$\|A_i \boldsymbol{\omega}_2\| \leq \|A_i\| \|\boldsymbol{\omega}_2\| \leq \lambda_i \|\boldsymbol{\omega}_2\| = \frac{i-1}{i}\|\boldsymbol{\omega}_{2\|}.$$

Therefore,

$$\begin{aligned}\left\|\prod_{i=2}^{t} A_i \boldsymbol{\omega}_2\right\| &= \|A_2 A_3 \cdots A_t \boldsymbol{\omega}_2\| \\ &\leq \|A_2\| \|A_3\| \cdots \|A_t\| \|\boldsymbol{\omega}_2\| \\ &\leq \frac{(2-1)}{2} \cdot \frac{(3-1)}{3} \cdot \cdots \cdot \frac{(t-1)}{t} \|\boldsymbol{\omega}_2\| \\ &\leq \frac{1}{t}\|\boldsymbol{\omega}_2\|,\end{aligned} \tag{4.6}$$

Next, we consider $\|\bar{A}_i\|$, case I, for $i < t$ we have

$$\begin{aligned}
\|\bar{A}_i\| &= \left\| \frac{1}{i}\Big(\prod_{j=i+1}^{t} A_j \Big)\mathbf{v}_i \right\| \\
&= \frac{1}{i}\|A_{i+1}\|\|A_{i+2}\|\cdots\|A_t\|\|\mathbf{v}_i\| \\
&= \frac{1}{i}\cdot\frac{(i+1)-1}{i+1}\cdot\frac{(i+2)-1}{i+2}\cdots\frac{t-1}{t}\|\mathbf{v}_i\| \\
&= \frac{1}{t}\|\mathbf{v}_i\| \\
&\leq \frac{1}{t}\max_{i<t}\|\mathbf{v}_i\|,
\end{aligned}$$

and case II, for $i = t$ we have

$$\|\bar{A}_i\| = \frac{1}{t}\|v_t\| \leq \frac{1}{t}\max_{i=t}\|\mathbf{v}_i\|.$$

From case I and case II, we obtain that

$$\|\bar{A}_i\| \leq \frac{1}{t}\max_{i\leq t}\|\mathbf{v}_i\|, \qquad \text{for } i \geq 2. \tag{4.7}$$

Thus, we have

$$\begin{aligned}
\|\boldsymbol{\omega}_{t+1}\| &= \left\| \prod_{i=2}^{t} A_i\boldsymbol{\omega}_2 + \sum_{i=2}^{t}\bar{A}_i \right\| \\
&\leq \left\| \prod_{i=2}^{t} A_i\boldsymbol{\omega}_2 \right\| + \sum_{i=2}^{t}\|\bar{A}_i\| \\
&\leq \frac{1}{t}\|\boldsymbol{\omega}_2\| + \sum_{i=2}^{t}\frac{1}{t}\max_{i\leq t}\|\mathbf{v}_i\| + \frac{1}{t}\max_{i\leq t}\|\mathbf{v}_i\| - \frac{1}{t}\max_{i\leq t}\|\mathbf{v}_i\| \\
&= \frac{1}{t}\|\boldsymbol{\omega}_2\| + t\Big(\frac{1}{t}\max_{i\leq t}\|\mathbf{v}_i\|\Big) - \Big(\frac{1}{t}\max_{i\leq t}\|\mathbf{v}_i\|\Big) \\
&= \frac{1}{t}\|\boldsymbol{\omega}_2\| + \frac{t-1}{t}\Big(\max_{i\leq t}\|\mathbf{v}_i\|\Big) \\
&\leq \|\boldsymbol{\omega}_2\| + \nu_1\max_{\mathbf{x}\in B}\|\mathbf{x}\| + c_1\max\{\tau_1,\tau_2\}\max_{\mathbf{x}\in A}\|\mathbf{x}\|.
\end{aligned} \tag{4.8}$$

Let M_1 and M_2 be the largest norms of the samples in the set A and set B, respectively, and

$$G_1 = \max\left\{\|\boldsymbol{\omega}_1\|, \|\boldsymbol{\omega}_2\| + \nu_1 M_2 + c_1\max\{\tau_1,\tau_2\}M_1\right\}.$$

Therefore G_1 is an upper bound of $\{\|\boldsymbol{\omega}_t\| : t = 1, 2, \ldots\}$. Next, we consider, for any $t \geq 1$

$$\begin{aligned}
\|\mathbf{s}_t\| &= \|\boldsymbol{\omega}_t + \nu_1\mathbf{x}_t^- - c_1 l_t y_t(\mathbf{x}_t^+)\| \\
&\leq \|\boldsymbol{\omega}_t\| + \nu_1\|\mathbf{x}_t^-\| + c_1\max\{\tau_1,\tau_2\}\|\mathbf{x}_t^+\| \\
&\leq \|\boldsymbol{\omega}_t\| + \nu_1 M_2 + c_1\max\{\tau_1,\tau_2\}M_1 \\
&\leq G_1 + \nu_1 M_2 + c_1\max\{\tau_1,\tau_2\}M_1.
\end{aligned}$$

We obtain that $G_2 = G_1 + \nu_1 M_2 + c_1 \max\{\tau_1, \tau_2\} M_1$ being an upper bound of $\{\|\mathbf{s}_t\| : t = 1, 2, \ldots\}$. ∎

Now, we can establish a convergence of **Algorithm 1**.

Theorem 4.2. *The sequence* $\{\boldsymbol{\omega}_t\}$ *generated by Algorithm* 1 *almost surely converge to an optimal solution of problem* (3.1).

Proof. On the one hand, from (4.6) and Lemma 4.1, we have

$$\lim_{t\to\infty} \left\| \prod_{i=2}^{t} A_i \boldsymbol{\omega}_2 \right\| = 0, \tag{4.9}$$

which implies that

$$\lim_{t\to\infty} \prod_{i=2}^{t} A_i \boldsymbol{\omega}_2 = 0. \tag{4.10}$$

On the other hand, from (4.7), we have

$$\begin{aligned} \sum_{i=2}^{t} \|\bar{A}_i\| &\leq \sum_{i=2}^{t} \left(\frac{1}{t} \max_{i\leq t} \|\mathbf{v}_i\| \right) \\ &\leq v_1 M_2 + c_1 \max\{\tau_1, \tau_2\} M_1. \end{aligned} \tag{4.11}$$

Since (4.11), we have $\sum_{i=2}^{t} \|\bar{A}_i\|$ is bounded. And we know that $\sum_{i=2}^{t} \|\bar{A}_i\|$ is monotone increasing, this implies that $\sum_{i=2}^{t} \|\bar{A}_i\|$ is convergent. By infinite series of vectors is convergent if its norm series is convergent, we have $\sum_{i=2}^{t} \bar{A}_i$ is convergent. Then, since

$$\boldsymbol{\omega}_{t+1} = \prod_{i=2}^{t} A_i \boldsymbol{\omega}_2 + \sum_{i=2}^{t} \bar{A}_i,$$

we have the sequence $\{\boldsymbol{\omega}_t\}$ is convergent. Then, we assume that $\{\boldsymbol{\omega}_t\}$ converges to $\boldsymbol{\omega}^*$, for some $\boldsymbol{\omega}^* \in \mathbb{R}^n$. We will show that $\boldsymbol{\omega}^*$ is an optimal solution of problem (3.1). Let us consider, for all $\boldsymbol{\omega} \in \mathbb{R}^n$. Since $\mathbf{s}_t$ is a subgradient of f at $\boldsymbol{\omega}_t$, we have

$$\begin{aligned} f(\boldsymbol{\omega}_t) - f(\boldsymbol{\omega}) &\leq \langle \mathbf{s}_t, \boldsymbol{\omega}_t - \boldsymbol{\omega} \rangle \\ &= \langle \mathbf{s}_t, \boldsymbol{\omega}_t - \boldsymbol{\omega} + \boldsymbol{\omega}^* - \boldsymbol{\omega}^* \rangle \\ &= \langle \mathbf{s}_t, \boldsymbol{\omega}_t - \boldsymbol{\omega}^* \rangle + \langle \mathbf{s}_t, \boldsymbol{\omega}^* - \boldsymbol{\omega} \rangle \\ &\leq \|\mathbf{s}_t\| \|\boldsymbol{\omega}_t - \boldsymbol{\omega}^*\| + \|\mathbf{s}_t\| \|\boldsymbol{\omega}^* - \boldsymbol{\omega}\|. \end{aligned}$$

Taking $t \to \infty$, and by Lemma 4.1, we have

$$\lim_{t\to\infty} \Big(f(\boldsymbol{\omega}_t) - f(\boldsymbol{\omega}) \Big) \leq 0. \tag{4.12}$$

Since f is continuous and $\boldsymbol{\omega}_t \to \boldsymbol{\omega}^*$, this implies that

$$f(\boldsymbol{\omega}^*) \leq f(\boldsymbol{\omega}), \quad \forall \boldsymbol{\omega} \in \mathbb{R}^n. \tag{4.13}$$

Thus, $\boldsymbol{\omega}^*$ is an optimal solution of problem (3.1). Hence, we conclude that the sequence $\{\boldsymbol{\omega}_t\}$ generated by Algorithm 1 almost surely converge to an optimal solution of problem (3.1) ∎

Theorem 4.2 says that the first of two iterative problems in the sequence generated by Algorithm 1 is convergent. The same conclusion can be obtained for the nonlinear case. Thus, we immediately have the sequences generated by Algorithm 1 convergent. Further, in order to establish a bound on the expected convergence rate of the proposed SG-TPSVM model, we need to justify the following lemma.

Lemma 4.3. *For each $t = 1, 2, \ldots, T$, assume that SGD is run for T iterations with $\eta_t = \frac{1}{t}$. For an algorithm with an initialization $\boldsymbol{\omega}_1 = \boldsymbol{0}$ we have,*

$$\sum_{t=1}^{T}\langle \boldsymbol{\omega}_t - \boldsymbol{\omega}^*, \boldsymbol{s}_t\rangle \quad \leq \quad TG_2(G_1 + \|\boldsymbol{\omega}^*\|) + \frac{1}{2}G_2^2(1 + \ln T), \tag{4.14}$$

where G_1, G_2 are an upper bound of $\{\|\boldsymbol{\omega}_t\|\}$ and $\{\|\boldsymbol{s}_t\|\}$, respectively.

Proof. Consider,

$$\langle \boldsymbol{\omega}_t - \boldsymbol{\omega}^*, \mathbf{s}_t\rangle = \frac{1}{\eta_t}\langle \boldsymbol{\omega}_t - \boldsymbol{\omega}^*, \eta_t\mathbf{s}_t\rangle \tag{4.15}$$

$$= \frac{1}{2\eta_t}\Big(\|\boldsymbol{\omega}_t - \boldsymbol{\omega}^*\|^2 - \|\boldsymbol{\omega}_{t+1} - \boldsymbol{\omega}^*\|^2 + \eta_t^2\|\mathbf{s}_t\|^2\Big). \tag{4.16}$$

Summing the equality over t and using Lemma 4.1, we obtain that

$$\begin{aligned}
\sum_{t=1}^{T}\langle \boldsymbol{\omega}_t - \boldsymbol{\omega}^*, \mathbf{s}_t\rangle &= \sum_{t=1}^{T}\Big(\frac{1}{2\eta_t}\Big(\|\boldsymbol{\omega}_t - \boldsymbol{\omega}^*\|^2 - \|\boldsymbol{\omega}_{t+1} - \boldsymbol{\omega}^*\|^2 + \eta_t^2\|\mathbf{s}_t\|^2\Big)\Big) \\
&= \frac{1}{2}\sum_{t=1}^{T}\frac{1}{\eta_t}\Big(\|\boldsymbol{\omega}_t - \boldsymbol{\omega}^*\|^2 - \|\boldsymbol{\omega}_{t+1} - \boldsymbol{\omega}^*\|^2\Big) + \frac{1}{2}\sum_{t=1}^{T}\Big(\eta_t\|\mathbf{s}_t\|^2\Big) \\
&= \frac{1}{2}\Big(\sum_{t=1}^{T}t\|\boldsymbol{\omega}_t - \boldsymbol{\omega}^*\|^2 - \sum_{t=1}^{T}t\|\boldsymbol{\omega}_{t+1} - \boldsymbol{\omega}^*\|^2\Big) + \frac{1}{2}\sum_{t=1}^{T}\Big(\eta_t\|\mathbf{s}_t\|^2\Big) \\
&= \frac{1}{2}\Big(\|\boldsymbol{\omega}_1 - \boldsymbol{\omega}^*\|^2 + 2\|u_2 - \boldsymbol{\omega}^*\|^2 + \cdots + T\|\boldsymbol{\omega}_T - \boldsymbol{\omega}^*\|^2 \\
&\quad - \|\boldsymbol{\omega}_2 - \boldsymbol{\omega}^*\|^2 - 2\|\boldsymbol{\omega}_3 - \boldsymbol{\omega}^*\|^2 - \cdots \\
&\quad - (T-1)\|\boldsymbol{\omega}_T - \boldsymbol{\omega}^*\|^2 - T\|\boldsymbol{\omega}_{T+1} - \boldsymbol{\omega}^*\|^2\Big) + \frac{1}{2}\sum_{t=1}^{T}\Big(\eta_t\|\mathbf{s}_t\|^2\Big) \\
&= \frac{1}{2}\Big(\|\boldsymbol{\omega}_1 - \boldsymbol{\omega}^*\|^2 + \|\boldsymbol{\omega}_2 - \boldsymbol{\omega}^*\|^2 + \cdots + \|\boldsymbol{\omega}_T - \boldsymbol{\omega}^*\|^2 \\
&\quad - T\|\boldsymbol{\omega}_{T+1} - \boldsymbol{\omega}^*\|^2\Big) + \frac{1}{2}\sum_{t=1}^{T}\Big(\eta_t\|\mathbf{s}_t\|^2\Big) \\
&= \frac{1}{2}\Big(\sum_{t=1}^{T}\|\boldsymbol{\omega}_t - \boldsymbol{\omega}^*\|^2 - T\|\boldsymbol{\omega}_{T+1} - \boldsymbol{\omega}^*\|^2\Big) + \frac{1}{2}\sum_{t=1}^{T}\Big(\eta_t\|\mathbf{s}_t\|^2\Big)
\end{aligned}$$

$$\leq \Big(G_1 + \|\boldsymbol{\omega}^*\|\Big) \sum_{t=1}^{T} \|\boldsymbol{\omega}_{T+1} - \boldsymbol{\omega}_t\| + \frac{1}{2} G_2^2 \Big(1 + \ln T\Big)$$

$$= \Big(G_1 + \|\boldsymbol{\omega}^*\|\Big) \sum_{t=1}^{T} \| \sum_{t=1}^{T} \frac{1}{t} \mathbf{s}_t\| + \frac{1}{2} G_2^2 \Big(1 + \ln T\Big)$$

$$\leq T G_2 \Big(G_1 + \|\boldsymbol{\omega}^*\|\Big) + \frac{1}{2} G_2^2 \Big(1 + \ln T\Big). \tag{4.17}$$

hence proving our Lemma. ∎

Theorem 4.4. *Under the assumptions of Lemma 4.3 and additional assumption that f is convex, let $\bar{\boldsymbol{\omega}} = \frac{1}{T} \sum_{t=1}^{T} \boldsymbol{\omega}_t$ be the solution output of SG-TPSVM. Then*

$$f(\bar{\boldsymbol{\omega}}) - f(\boldsymbol{\omega}^*) \quad \leq \quad G_2 \Big(G_1 + \|\boldsymbol{\omega}^*\|\Big) + \frac{1}{2T} G_2^2 \Big(1 + \ln T\Big). \tag{4.18}$$

Proof. From the definition of $\bar{\boldsymbol{\omega}}$, we have

$$f(\bar{\boldsymbol{\omega}}) - f(\boldsymbol{\omega}^*) = f\Big(\frac{1}{T} \sum_{t=1}^{T} \boldsymbol{\omega}_t\Big) - f(\boldsymbol{\omega}^*)$$

$$\leq \frac{1}{T} \sum_{t=1}^{T} \Big(f(\boldsymbol{\omega}_t)\Big) - f(\boldsymbol{\omega}^*)$$

$$= \frac{1}{T} \sum_{t=1}^{T} \Big(f(\boldsymbol{\omega}_t) - f(\boldsymbol{\omega}^*)\Big). \tag{4.19}$$

For every t, by assumption and $s_t = \nabla f(\boldsymbol{\omega}_t, \theta_t)$, we have

$$f(\boldsymbol{\omega}_t) - f(\boldsymbol{\omega}^*) \leq \langle \boldsymbol{\omega}_t - \boldsymbol{\omega}^*, \mathbf{s}_t \rangle. \tag{4.20}$$

Also, taking sum over t and divided by T, it follows that

$$\frac{1}{T} \sum_{t=1}^{T} \Big(f(\boldsymbol{\omega}_t) - f(\boldsymbol{\omega}^*)\Big) \leq \frac{1}{T} \sum_{t=1}^{T} \langle \boldsymbol{\omega}_t - \boldsymbol{\omega}^*, \mathbf{s}_t \rangle. \tag{4.21}$$

By Lemma 4.3, we have

$$\frac{1}{T} \sum_{t=1}^{T} \langle \boldsymbol{\omega}_t - \boldsymbol{\omega}^*, \mathbf{s}_t \rangle \leq G_2(G_1 + \|\boldsymbol{\omega}^*\|) + \frac{1}{2T} G_2^2 (1 + \ln T). \tag{4.22}$$

Combining the preceding we conclude that

$$f(\bar{\boldsymbol{\omega}}) - f(\boldsymbol{\omega}^*) \leq G_2(G_1 + \|\boldsymbol{\omega}^*\|) + \frac{1}{2T} G_2^2 (1 + \ln T). \tag{4.23}$$

∎

Remark 4.5. With Theorem 4.4 we have shown that in a random iteration T, the resulting expected error is bounded by $O\Big(\frac{1 + \ln T}{T}\Big)$. However, it is interesting to note that SG-TPSVM enjoys an error bound similar to the SG-TSVM [24].

5. Numerical Experiment

Here we will establish the performance of our SG-TPSVM model. The experiments have been preformed on Python3 on a macOS with an Intel i5 Processor 2.3 GHz with Memory 8 GB 2133 MHz LPDDR3. In order to evaluate the performance of classifiers, we have used a 10-fold cross validation technique for all experiments. The results of each experiment show the average CPU time and standard deviation in all tables and highlight the best one. From now on we denote $\tau_1 = \tau_3$, $\tau_2 = \tau_4$, $\epsilon_1 = \epsilon_3$, and $\epsilon_2 = \epsilon_4$. To derive the non-linear case, we use the Gaussian kernel $K(x, y) = \exp\{-\sigma\|x - y\|^2\}$.

5.1. Synthetic Dataset

The purpose of our SG-TPSVM model is to be able to handle noise-sensitive classifiers. To illustrate the noise insensitivity performance consider Figure 1 and Figure 2, where we take a two dimensional synthetic dataset with equal number of samples from two Gaussian distributions: $x_i, i \in \{i : y_i = 1\} \sim \mathcal{N}(\mu_1, \sum_1)$ and $x_i, i \ln\{i : y_i = -1\} \sim \mathcal{N}(\mu_2, \sum_2)$, where $\mu_1 = [1, -3]^\top$, $\mu_2 = [-1, 3]^\top$ and $\sum_1 = \sum_2 = \begin{bmatrix} 0.2 & 0 \\ 0 & 3 \end{bmatrix}$. We add noise to the dataset, with each noise sample drawn from the Gaussian distribution $\mathcal{N}(\mu_n, \sum_n)$ where $\mu_n = [0, 0]^\top$ and $\sum_n = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}$. From Figure 1, the bar chart demonstrates the percentages of accuracy to classify from different sectors including SG-TPSVM, SG-TSVM [24] and PEGASOS [20] during 0 (free-noise) to 30 noise points. Overall, the SG-TPSVM achieves the best results in all cases. This implies that the SG-TPSVM was the strongest candidate for the method of classifying the data with noise corrupted.

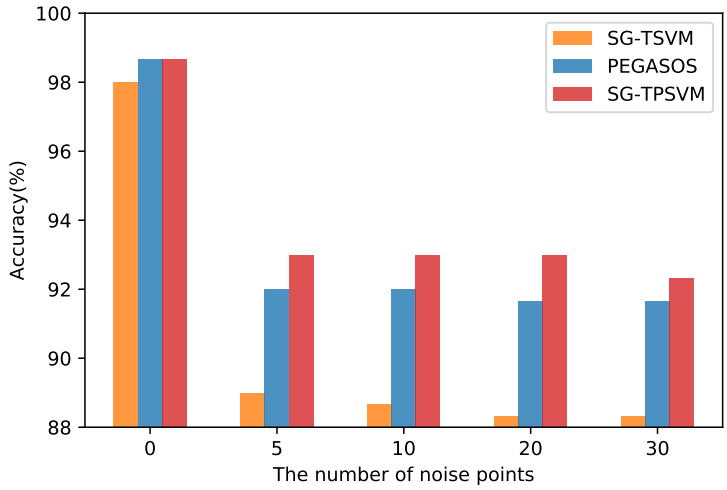
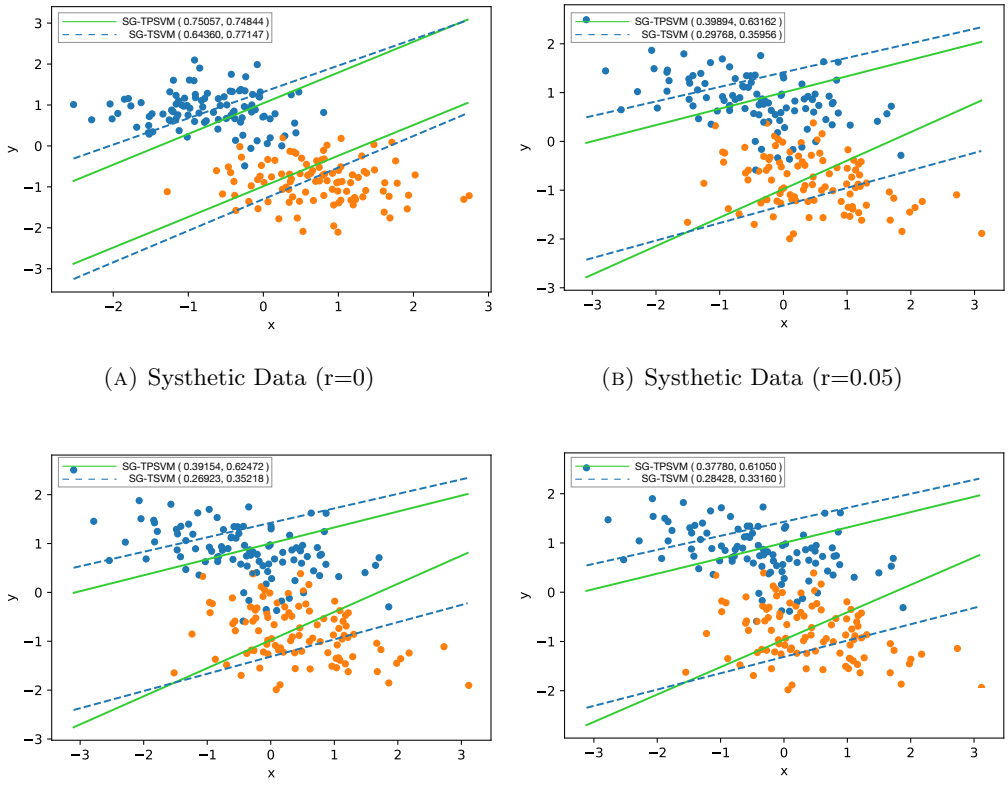


Figure 1. Bar graph of the accuracies for three algorithms on the 2-D artificial data.

Next result Figure 2 in this regard, showing when we increase the amount of noise from 0 to 20%, the hyperplanes of SG-TSVM diverge from 0.64360, 0.77147 to 0.28428, 0.33160 while the hyperplanes of our SG-TPSVM slightly changed. This means that the our SG-TPSVM model is insensitive to noise.



(A) Systhetic Data (r=0) (B) Systhetic Data (r=0.05)

(C) Systhetic Data (r=0.1) (D) Systhetic Data (r=0.2)

FIGURE 2. These figures demonstrate the noise insensitive properties possessed. we have varying number of noise samples, from $r = 0$ (noise free), $r = 0.05$, $r = 0.1$ and $r = 0.2$. Here, r is the ratio of total number of noisy samples to the total number of samples originally in the dataset (including both classes). The legend in each figure gives the slopes of the separating hyperplanes in the brackets.

5.2. UCI DATASETS

To test the performance of the proposed SG-TPSVM, experiments are carried out on eight UCI datasets from binary classification. All the data are summarized in Table 1. The features of each data set are corrupted by zero-mean Gaussian noise. For each feature, the ratio of the variance of noise denoted as r is set to be 0 (i.e., noise-free), 0.05, and 0.1.

In order to prove the efficacy of the proposed model, we have also compared the proposed formulation with SG-TSVM [24]. The comparison results are shown in Table 2. Here SG-TSVM(a), SG-TPSVM(a) and SG-TSVM(b), SG-TPSVM(b) refer to linear and nonlinear cases, respectively. One can observe from Table 2 that the proposed model achieves better results in most cases. Figure 3 and Figure 4 show the average accuracy obtained from Table 2. It is clear that our SG-TPSVM has accuracy performance better than SG-TSVM. The computational time of both models are not different. The optimal parameters used in Table 2 are summarized in Table 3.

Table 1 The details of datasets.

datasets	No. of samples	Dimension
Australian	690	14
Banknote	1,372	5
Coil2000	900	86
Heart	270	13
Monk2	432	7
Pima Indians	768	8
Titanic	2,201	41
Twonorm	7,400	21

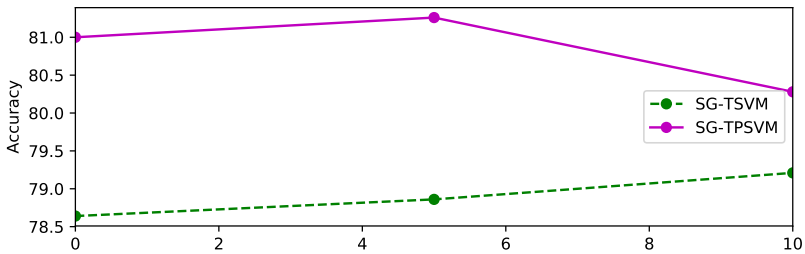


FIGURE 3. The average accuracy of SG-TSVM and SG-TPSVM (linear case) on UCI data sets, where the horizontal axis denotes the amount of noise.

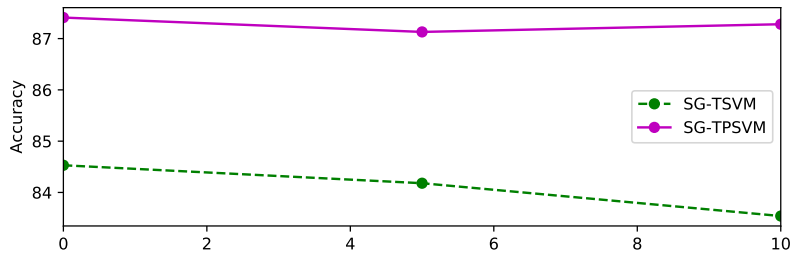


FIGURE 4. The average accuracy of SG-TSVM and SG-TPSVM (nonlinear case) on UCI data sets, where the horizontal axis denotes the amount of noise.

Table 2 Testing accuracy obtained from all discussed formulation in UCI datasets.

Datasets	r	Existing algorithms		Proposed algorithms	
		SG-TSVM(a)	SG-TSVM(b)	**SG-TPSVM(a)**	**SG-TPSVM(b)**
		Time (s)	Time (s)	Time (s)	Time (s)
Australian	0	86.81±3.27	76.09±7.73	**87.83±3.12**	84.49±3.37
		0.0639	0.2879	0.0830	0.4139
	0.05	85.65±3.52	76.67±6.53	**87.10±3.14**	84.78±3.68
		0.0627	0.2594	0.0862	0.4638
	0.1	86.23±2.69	74.78±8.58	**86.38±3.12**	84.78±3.26
		0.0602	0.2473	0.0831	0.4397
Banknote	0	83.75±2.27	97.15±2.27	84.62±2.90	**98.32±1.82**
		0.0656	0.4380	0.0432	0.7620
	0.05	88.12±2.24	96.86±2.77	84.40±2.98	**98.11±2.07**
		0.0610	0.4460	0.0417	0.4639
	0.1	88.34±2.66	96.50±2.23	83.38±3.20	**98.40±1.56**
		0.0599	0.4318	0.0402	0.4907
Coil2000	0	50.67±11.76	**93.29±0.78**	68.11±2.61	93.02±0.76
		0.0642	7.1999	0.0927	9.1320
	0.05	51.19±10.61	**94.03±0.71**	63.04±3.06	**94.03±0.71**
		0.0621	5.2486	0.0859	5.0169
	0.1	50.38±13.47	90.83±4.37	60.01±2.51	**94.03±0.17**
		0.0568	5.0152	0.0807	5.2470
Heart	0	83.33±5.30	83.70±6.46	80.00±5.79	**84.07±4.40**
		0.0627	0.3028	0.0844	0.3243
	0.05	82.59±4.98	81.48±7.03	82.96±4.44	**84.81±4.52**
		0.0665	0.3033	0.0863	0.3387
	0.1	83.33±5.80	82.59±5.98	83.70±4.12	**84.44±3.63**
		0.0612	0.2910	0.0870	0.3526
Monk2	0	78.93±8.09	85.64±6.14	79.60±8.54	**89.58±4.91**
		0.0585	0.3128	0.0788	0.3434
	0.05	78.22±7.68	84.73±7.64	82.86±6.17	**87.04±6.83**
		0.0546	0.3359	0.0782	0.3666
	0.1	79.85±9.54	84.48±5.72	80.06±7.42	**88.42±7.71**
		0.0561	0.3527	0.0845	0.3783
Pima	0	72.40±4.75	**75.64±6.33**	73.55±4.80	74.35±5.38
		0.0644	0.3690	0.0800	0.3943
	0.05	72.27±6.11	73.95±4.95	**74.73±4.80**	73.56±5.95
		0.0549	0.3522	0.0775	0.4484
	0.1	72.65±4.61	73.42±6.42	**74.47±4.97**	73.16±5.87
		0.0527	0.3461	0.0740	0.3802
Titanic	0	74.47±3.47	66.96±2.04	76.56±3.20	**77.60±2.67**
		0.0397	0.2427	0.0791	0.5115
	0.05	75.15±2.50	68.01±2.62	**77.10±2.98**	76.92±2.71
		0.0590	0.3823	0.0789	0.5621
	0.1	75.15±3.07	67.92±2.69	76.60±2.45	**77.37±2.33**
		0.0560	0.4162	0.0833	0.5481
Twonorm	0	97.73±0.71	97.78±0.66	97.80±0.56	**97.81±0.63**
		0.0613	1.6700	0.0813	1.7512
	0.05	97.69±0.67	97.72±0.56	**97.85±0.63**	97.76±0.63
		0.0651	1.8604	0.0889	1.7798
	0.1	97.73±0.60	**97.78±0.55**	97.65±0.64	97.66±0.56
		0.0659	2.9027	0.0822	1.8999

Table 3 The optimal parameters of SG-TSVM, SG-GTPPSVM.

Datasets	SG-TSVM(a)	SG-TSVM(b)	**SG-TPSVM(a)**	**SG-TPSVM(b)**
	c_1, c_2	c_1, c_2, γ	$c, \nu, \tau_1, \tau_2,$ ϵ_1, ϵ_2	$c, \nu, \tau_1, \tau_2,$ $\epsilon_1, \epsilon_2, \gamma$
Australian	1,1	0.1,1,0.01	1,1,2,0.01, 0.01,0.01	1,1,1,1, 1,1,0.1
Banknote	1,1	0.1,1,10	0.05,0.1,1,1, 1.5,0.5	1,1,1,1, 1,1,10
Coil2000	1,1	0.1,1,10	2,1.5,2,1, 1.5,0.5	1,1,1,1, 1,1,1
Heart	1,1	0.1,1,0.01	1,1,2,0.01, 0.01,0.01	1,1,1,1, 1,1,0.1
Monk2	1,1	0.1,1,1	0.01,1,2,0.01, 0.01,0.01	1,0.1,1,1, 1,1,1
Pima	1,1	0.1,1,0.1	0.01,1,2,0.01, 0.01,0.01	1,0.09,1,1, 1,1,0.1
Titanic	1,1	0.1,1,0.1	1,1,2,0.01, 0.01,0.01	1,1,1,1,1, 1,1,0.1
Twonorm	1,1	0.1,1,0.1	1,1,1,1.3, 0.01,0.01	1,1,1.3,1, 1,1,0.1

5.3. Large Scale Data Set

In this section, we show that our SG-TPSVM is capable to solve large scale problems. Note that this section has been solved on a machine equipped with an Intel CPU E5-2658 v3 at 2.20GHz and 256 GB RAM running Ubuntu Linux operating system. The usual SVMs are used with scikit-learn package [28]. The large scale datasets we used is presented in Table 2. For the nonlinear case, the reduced kernel [29] was used, and the kernel size was fixed to 100.

Table 4 The details of large scale datasets.

datasets	No. of samples	Dimension
Skin	245,057	3
SUSY	5,000,000	18
Kddcup	4,898,432	41
Hepmass	10,500,000	28

It is clear from the table 5 that the accuracy of SG-TPSVM is better than SVM and SG-TSVM. However, the experimental results on the two largest datasets, i.e. KDDCup and Hepmass, cannot be obtained due to the excessive memory requirement. This is because the implementation of SVM needs to store the entire training set in the main memory. Meanwhile SG-TSVM and SG-TPSVM only store a subset related to the iteration. Accordingly, SG-TSVM and SG-TPSVM is capable to solve largle scale problem.

Table 5 The results for the large scale datasets.

Datasets	Existing algorithms			Proposed algorithms	
	SVM	SG-TSVM(a)	SG-TSVM(b)	**SG-TPSVM(a)**	**SG-TPSVM(b)**
Skin	78.87	85.23	84.70	92.64	**92.77**
Susy	**78.52**	75.09	68.61	75.91	68.70
Kddcup	*	95.24	93.19	96.72	**99.41**
Hepmass	*	81.10	79.59	**81.92**	79.16

6. Conclusion

In this paper we have proposed a stochastic gradient descent method based on generalized pinball twin parametric support vector machine (SG-TPSVM). The efficiency of proposed method by using generated data and datasets imported from UCI is compared to SG-TSVM. The experimental results have shown that the accuracy performance of our method is better than the existing classifiers for different data scenarios. We have shown that the SG-TPSVM approach is leading to insensitivity from noise and can handle large scale problems. The results implies that the SG-TPSVM approach was the strongest candidate for the method of a binary classification problem.

For future research, we plan to consider applications of SG-TPSVM on activity recognition dataset and image retrieval dataset and also plan to improve our approach to deal with multi-category classification scenarios.

Acknowledgments

The authors would like to thank the referees for careful reading and constructive comments. This research is partially supported by the Science Achivement Scholarship of Thailand and Naresuan university.

References

[1] C. Cortes, V.N. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297.

[2] L. Cao, F.E.H. Tay, Financial forecasting using support vector machines, Neural Computional Applications 10 (2) (2001) 184–192.

[3] K.G.M. Chathuramali, R. Rodrigo, Faster human activity recognition with SVM, ICT Emerg. Regions (ICTer) (2012) 197–203.

[4] E. Byvatov, G. Schneider, Support vector machine applications in bioinformatics, Applications Bioinformat 2 (2) (2003) 67–77.

[5] Z. Huang, H. Chen, C.-J. Hsua, W.-H. Chen, S. Wu, Credit rating analysis with support vector machines and neural networks: a market comparative study, Decision Support Systems 37 (2004) 543–558.

[6] H. Ince, T.B. Trafalis, Support vector machine for regression and applications to financial forecasting, International Joint Conference on Neural Networks, IEEE-INNS-ENNS, Como, Italy (2002).

[7] S. Mukherjee, E. Osuna, F. Girosi, Nonliner prediction of chaotic time series using a support vector machine, Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing, Amelia Island, FL, USA (1997), 511–520.

[8] R. Khemchandani, Jayadeva, S. Chandra, Regularized least squares fuzzy support vector regression for financial time series forecasting, Expert Systems with Applications 36 (1) (2009) 132–138.

[9] Jayadeva, R. Khemchandani, S. Chandra, Twin support vector machines for pattern classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (5) (2007) 905–910.

[10] M.A. Kumar, M. Gopal, Least squares twin support vector machines for pattern classification, Expert Systems with Applications 36 (4) (2009) 7535–7543.

[11] Y. Xu, W. Xi, X. Lv, R. Guo, An improved least squares twin support vector machine, Information Sciences 9 (4) (2012) 1063–1071.

[12] X. Peng, TSVR: An efficient twin support vector machine for regression, Neural Networks 23 (3) (2010) 365–372.

[13] Y. Xu, L. Wang, A weighted twin support vector regression, Knowledge Based Systems 33 (2012) 92–101.

[14] P.Y. Hao, New support vector algorithms with parametric insensitive/margin model, Neural Networks 23 (1) (2010) 60–73.

[15] X. Peng, A noval twin parametric-margin support vector machine for pattern recognition, Pattern recognition 44 (2011) 2678–2692.

[16] J. Platt, Fast training of support vector machines using sequential minimal optimization, In Advances in kernel methods-support vector learning, Cambridge, MA: MIT Press (1999), 185–208.

[17] T. Joachims, Making large-scale SVM learning practical, In Advances in Kernel Methods-Support Vector Learning, Cambridge (1998), 169–184.

[18] C.C. Chang, C.J. Lin, LIBSVM: A library for support vector machines, http://www.csie.ntu.edu.tw/ cjlin (2001).

[19] R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, C.J. Lin, LIBLINEAR: a library for large linear classification, Journal of Machine Learning Research 9 (2008) 1871–1874.

[20] S. Shalev-Shwartz, Y. Singer, N. Srebro, A. Cotter, Pegasos: Primal estimated subgradient solver for SVM, Mathematical Programming 127 (1) (2011) 3–30.

[21] Y. Shao, C. Zhang, X. Wang, N. Deng, Improvements on twin support vector machines, IEEE Transactions on Neural Networks 22 (6) (2011) 962–968.

[22] Z. Wang, Y.H. Shao, T.R. Wu, A gabased model selection for smooth twin parametric margin support vector machine, Pattern Recognition 46 (8) (2013) 2267–2277.

[23] Y.J. Tian, Y. Ping, Large-scale linear nonparallel support vector machine solver, Neural Networks 50 (2014) 166–174.

[24] Z. Wang, Y. Shao, L. Bai, C. Li, L. Liu, N. Denge, Insensitive stochastic gradient twin support vector machines for large scale problems, Information Sciences 462 (2018)114–131.

[25] X. Huang, L. Shi, J.A. Suykens, Support vector machine classifier with pinball loss, IEEE Transactions on Pattern Analysis and Machine Intelligence 36 (5) (2014) 984–997.

[26] R. Reshma, R. Khemchandani, A. Pal, S. Chandra, Generalized pinball loss svms, Neurocomputing 10 (2018) 322.

[27] S. Shwartz, S. Ben-David, Understanding Machine Learning Theory Algorithms, Cambridge University Press, 2014.

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Pret- tenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[29] Y. Lee, O. Mangasarian, RSVM: Reduced support vector machines, First SIAM International Conference on Data Mining (2001) 5–7.