Article

Stochastic Subgradient for Large-Scale Support Vector Machine Using the Generalized Pinball Loss Function

Wanida Panup¹ and Rabian Wangkeeree^{1,2,*}

¹ Department of Mathematics, Faculty of Science, Naresuan University, Phitsanulok 65000, Thailand; wanidap56@nu.ac.th

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

* Correspondence: rabianw@nu.ac.th

Abstract: In this paper, we propose a stochastic gradient descent algorithm, called stochastic gradient descent method-based generalized pinball support vector machine (SG-GPSVM), to solve data classification problems. This approach was developed by replacing the hinge loss function in the conventional support vector machine (SVM) with a generalized pinball loss function. We show that SG-GPSVM is convergent and that it approximates the conventional generalized pinball support vector machine (GPSVM). Further, the symmetric kernel method was adopted to evaluate the performance of SG-GPSVM as a nonlinear classifier. Our suggested algorithm surpasses existing methods in terms of noise insensitivity, resampling stability, and accuracy for large-scale data scenarios, according to the experimental results.

Keywords: support vector machine; generalized pinball loss function; large-scale problems; stochastic gradient descent algorithm; feature noise

Citation: Panup, W.; Wangkeeree, R. Stochastic Subgradient for Large-Scale Support Vector Machine Using Generalized Pinball Loss Function. *Symmetry* **2021**, *13*, 1652. <https://doi.org/10.3390/sym13091652>

Academic Editor: Juan Luis García Guirao

Received: 18 July 2021

Accepted: 1 September 2021

Published: 8 September 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Support vector machine (SVM) is a popular supervised binary classification algorithm based on statistical learning theory. Initially proposed by Vapnik [1–3], it has been gaining more and more attention. There have been many algorithmic and modeling variations of it, and it is a powerful pattern classification tool that has had many applications in various fields during recent years, including face detection [4,5], text categorization [6], electroencephalogram signal classification [7], financial regression [8–11], image retrieval [12], remote sensing [13], feature extraction [14], etc.

The concept of the conventional SVM is to find an optimal hyperplane that results in the greatest separation of the different categories of observations. The conventional SVM uses the well-known and common hinge loss function; however, it is sensitive to noise, especially noise around the decision boundary, and is not stable in resampling [15]. As a result, many researcher have proposed new SVM methods by changing the loss function. The SVM model with the pinball loss function (Pin-SVM) was proposed by Huang [16] to treat noise sensitivity and instability in resampling. The outcome is less sensitive and is related to the quantile distance. On the other hand, Pin-SVM cannot achieve sparsity. To achieve sparsity, a modified ϵ -insensitive zone for Pin-SVM was proposed. This method does not consider the patterns that lie in the insensitive zone while building the classifier, and its formulation requires the value of ϵ to be specified beforehand; therefore, a bad choice may affect its performance. Motivated by these developments, Rastogi [17] recently proposed the modified (ϵ_1, ϵ_2) -insensitive zone support vector machine. This method is an extension of existing loss functions that account for noise sensitivity and resampling stability.

However, practical problems require processing large-scale datasets, while the existing solvers are not computationally efficient. Since the generalized pinball loss SVM (GPSVM) still needs to solve a large quadratic programming problem and the dual problem of

GPSVM requires solving the quadratic programming problem (QPP), these techniques have difficulty handling large-scale problems [18,19]. Different solving methods have been proposed for large-scale SVM problems. The optimization techniques such as sequential minimal optimization (SMO) [20], successive over-relaxation (SOR) [21], and the dual coordinate descent method (DCD) [22] have been proposed to solve the SVM problem and the dual problem of SVM. However, these methods necessitate computing the inverse of large matrices. Consequently, the dual solutions of SVM cannot be effectively applied to large-scale problems. On the one hand, a main challenge for the conventional SVM is the high computational complexity of the training samples, i.e., $O(m^3)$, where m is the total number of training samples. As a result, the stochastic gradient descent algorithm (SGD) [23,24] has been proposed to solve the primal problem of SVM. As an application of the stochastic subgradient descent method, Pegasos [25] obtained an ϵ -accuracy solution for the primal problem in $\tilde{O}(1/\epsilon)$ iterations with the cost per iteration being $O(n)$, where n is the feature dimension. This technique partitions a large-scale problem into a series of subproblems by stochastic sampling with a suitable size. The SGD for SVM has been shown to be the fastest method among the SVM-type classifiers for large-scale problems [23,26–28].

In order to overcome the above-mentioned limitations of large-scale problems and inspired by the studies of SVM and the generalized pinball loss function, we propose a novel stochastic subgradient descent method with generalized pinball support vector machine (SG-GPSVM). The proposed technique is an efficient method for real-world datasets, especially large-scale ones. Furthermore, we prove the theorem that guarantees the approximation and convergence of our proposed algorithm. Finally, the experimental results show that the proposed SG-GPSVM approach outperforms the existing approaches in terms of accuracy, and the proposed SG-GPSVM may result in a more efficient and faster process of convergence rather than using the conventional SVM. The results also show that our proposed SG-GPSVM is noise insensitive and more stable in resampling and can handle large-scale problems.

The structure of the paper is as follows. Section 2 outlines the background. SG-GPSVM is proposed in Section 3, which includes both linear and nonlinear cases. In Section 4, we theoretically compare the proposed SG-GPSVM with two other algorithms, i.e., conventional SVM and Pegasos [25]. Experiments on benchmark datasets, available in the UCI Machine Learning Repository [29], with different noises are conducted to verify the effectiveness of our SG-GPSVM. The results are given in Section 5. Finally, the conclusion is given in Section 6.

2. Related Work and Background

The purpose of this section is to review related methods for binary classification problems. This section contains some notations and definitions that are used throughout the paper. Let us consider a binary classification problem of m data points and n -dimensional Euclidean space $\mathbb{R}^n$. Here, we denote the set of training samples by $X \in \mathbb{R}^{n \times m}$, where $x \in \mathbb{R}^n$ is a sample with a label $y \in \{1, -1\}$. Below, we give a brief outline of several related methods.

Support Vector Machine

The SVM model consists of maximizing the distance between the two bounding hyperplanes that bound the classes, so the SVM model is generally formulated as a convex quadratic programming problem. Let $\|\cdot\|$ denote the Euclidean norm, or two-norm, of a vector in $\mathbb{R}^n$. Given a training set $S = \{(x_i, y_i) \in \mathbb{R} \times \{1, -1\} : i = 1, 2, 3, \dots, m\}$, the strategy of SVM is to find the maximum margin separating hyperplane $w^\top x + b = 0$ between two classes by solving the following problem:

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

where C is a penalty parameter and ξ_i are the slack variables. By introducing the Lagrangian multipliers α_i , we derive its dual QPP as follows:

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m y_i y_j (x_i^\top x_j) \alpha_i \alpha_j - \sum_{i=1}^m \alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^m y_i \alpha_i = 0, \\ & 0 \leq \alpha_i \leq C, \quad i = 1, \dots, m. \end{aligned} \quad (2)$$

After optimizing this dual QPP, we obtain the following decision function:

$$f(x) = \text{sign} \left(\sum_{i=1}^{N_{SV}} \alpha_i^* y_i (x_i^\top x) + b \right) \quad (3)$$

where α^* is the solution of the dual problem (Equation (2)) and N_{SV} represents the number of support vectors satisfying $0 < \alpha < C$. In fact, the SVM problem (Equation (1)) can be rewritten as the unconstrained optimization problem as follows: [30]:

$$\min_{w,b} \quad \frac{1}{2} \|w\|^2 + \frac{C}{m} \sum_{i=1}^m \mathcal{L}_{\text{hinge}}(1 - y_i(w^\top x_i + b)), \quad (4)$$

where $\mathcal{L}_{\text{hinge}}(1 - y_i(w^\top x_i + b)) = \max\{0, 1 - y_i(w^\top x_i + b)\}$ is the so-called hinge loss function. This loss is related to the shortest distance between the sets, and the corresponding classifier leads to its sensitivity to noise and instability in resampling [31]. To address the noise sensitivity, Huang [16] proposed the pinball loss function instead of the hinge loss function by using the pinball loss function with the SVM classifier (Pin-SVM). This model works by penalizing correctly classified samples, which is evident from the pinball loss function, which is defined as follows:

$$\mathcal{L}_{\text{pin}}(v) = \begin{cases} v, & v \geq 0, \\ -\tau v, & v < 0, \end{cases} \quad (5)$$

where $v = (1 - y_i(w^\top x_i + b))$ and $\tau \geq 0$. Although the pinball loss function achieves noise insensitivity, in the process, it cannot achieve sparsity. This is because the pinball loss function's subgradient is nonzero almost everywhere. Therefore, to achieve sparsity, in the same publication, Huang [16] proposed the ϵ -insensitive pinball loss function, which is insensitive to noise and stable in resampling. The ϵ -insensitive pinball loss function is defined by:

$$\mathcal{L}_{\text{pin}}^\epsilon(v) = \begin{cases} v - \epsilon, & v > \epsilon, \\ 0, & -\frac{\epsilon}{\tau} \leq v \leq \epsilon, \\ -\tau(v + \frac{\epsilon}{\tau}), & v < -\frac{\epsilon}{\tau}, \end{cases} \quad (6)$$

where $\tau \geq 0$ and $\epsilon \geq 0$ are user-defined parameters. On the other hand, the ϵ -insensitive pinball loss function necessitates the selection of an ideal parameter. Rastogi [17] introduced the (ϵ_1, ϵ_2) -insensitive zone pinball loss function, also known as the generalized pinball loss function. The generalized pinball loss function is defined by:

$$\mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}(v) = \begin{cases} \tau_1(v - \frac{\epsilon_1}{\tau_1}), & v > \frac{\epsilon_1}{\tau_1}, \\ 0, & -\frac{\epsilon_2}{\tau_2} \leq v \leq \frac{\epsilon_1}{\tau_1}, \\ -\tau_2(v + \frac{\epsilon_2}{\tau_2}), & v < -\frac{\epsilon_2}{\tau_2}, \end{cases} \quad (7)$$

where $\tau_1, \tau_2, \epsilon_1, \epsilon_2$ are user-defined parameters. After employing the generalized pinball loss function instead of the hinge loss function in problem (1), we obtain the following optimization problem:

$$\begin{aligned} \min_{w, b, \xi} \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \xi_i \\ \text{s.t.} \quad & y_i(w^T x_i + b) \geq 1 - \frac{1}{\tau_1} (\xi_i + \epsilon_1), \quad i = 1, 2, 3, \dots, m, \\ & y_i(w^T x_i + b) \leq 1 + \frac{1}{\tau_2} (\xi_i + \epsilon_2), \quad i = 1, 2, 3, \dots, m, \\ & \xi_i \geq 0, \quad i = 1, 2, 3, \dots, m. \end{aligned} \quad (8)$$

By introducing the Lagrangian multipliers α_i, β_i , we derive its dual QPP as follows:

$$\begin{aligned} \min_{\alpha, \beta} \quad & \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m (\alpha_i - \beta_i)(\alpha_j - \beta_j) y_i y_j (x_i^T x_j) - (1 - \frac{\epsilon_1}{\tau_1}) \sum_{i=1}^m \alpha_i + (1 + \frac{\epsilon_2}{\tau_2}) \sum_{i=1}^m \beta_i \\ \text{s.t.} \quad & \sum_{i=1}^m (\alpha_i - \beta_i) y_i = 0, \\ & 0 \leq \frac{\alpha_i}{\tau_1} + \frac{\beta_i}{\tau_2} \leq C, \quad i = 1, \dots, m. \end{aligned} \quad (9)$$

A new data point $x \in \mathbb{R}^n$ is classified as positive or negative according to the final decision function:

$$f(x) = \text{sign} \left(\sum_{i=1}^m (\alpha_i^* - \beta_i^*) y_i (x_i^T x) + b \right) \quad (10)$$

where α^* and β^* are the solution to the dual problem (Equation (9)).

In the next section, we go over the core of our proposed technique and present its pseudocode.

3. Proposed Stochastic Subgradient Generalized Pinball Support Vector Machine

For our SG-GPSVM formulation, which is based on the generalized pinball loss function, we apply the stochastic subgradient approach in this section. Our SG-GPSVM can be used in both linear and nonlinear cases.

3.1. Linear Case

Following the method of formulating SVM problems (discussed in Equation (4)), we incorporated the generalized pinball loss function (Equation (7)) in the objective function to obtain the convex unconstrained minimization problems:

$$\min_u f(u) = \frac{1}{2} \|u\|^2 + \frac{C}{m} \sum_{i=1}^m \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2} (1 - y_i u^T z_i), \quad (11)$$

where $u = (w^T, b)^T, z = (x^T, 1)^T$ and $w \in \mathbb{R}^n, b \in \mathbb{R}$ are the weight vectors and biases, respectively, and $C > 0$ are the penalty parameters. To apply the stochastic subgradient approach, for each iteration t , we propose a more general method that uses k samples where $1 \leq k \leq m$. We chose a subset $A_t \subseteq [m]$ with $|A_t| = k$ where k samples are drawn uniformly at random from the training set and $[m] = \{1, 2, 3, \dots, m\}$. Let us consider a model of stochastic optimization problems. Let u_t denote the current hyperplane iterate, and we obtain an approximate objective function:

$$f(u_t) = \frac{1}{2} \|u_t\|^2 + \frac{C}{k} \sum_{i \in A_t} \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2} (1 - y_i u_t^T z_i). \quad (12)$$

When $k = m$, each iteration t , an approximate objective function is the original objective function in problem (11). Let ∇_t be subgradients of $f(u_t)$ associated with the index set of minibatches A_t at point u_t , that is:

$$\nabla_t \in \partial f(u_t) = u_t - \frac{C}{k} \sum_{i \in A_t} y_i z_i \partial \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}(1 - y_i u_t^\top z_i), \quad (13)$$

where:

$$\partial \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}(v) = \begin{cases} \{\tau_1\}, & v > \frac{\epsilon_1}{\tau_1}, \\ [0, \tau_1], & v = \frac{\epsilon_1}{\tau_1}, \\ \{0\}, & -\frac{\epsilon_2}{\tau_2} < v < \frac{\epsilon_1}{\tau_1}, \\ [-\tau_2, 0], & v = -\frac{\epsilon_2}{\tau_2}, \\ \{-\tau_2\}, & v < -\frac{\epsilon_2}{\tau_2}. \end{cases} \quad (14)$$

With the above notations and the existence of $\rho_i \in \partial \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}(1 - y_i u_t^\top z_i)$, ∇_t can be written as:

$$\nabla_t = u_t - \frac{C}{k} \sum_{i \in A_t} \rho_i y_i z_i. \quad (15)$$

Then, $u_{t+1} = u_t - \eta_t \nabla_t$ is updated using a step size of $\eta_t = \frac{1}{t}$. Additionally, a new sample z is predicted by:

$$y = \text{sgn}(u^\top z). \quad (16)$$

The above steps can be outlined as Algorithm 1.

Algorithm 1 SG-GPSVM.

Input: Training samples are represented by $X \in \mathbb{R}^{n \times m}$, positive parameters c_1, c_2, τ_1, τ_2 , $k \in \{1, 2, \dots, m\}$, and tolerance tol ; typically, $tol = 10^{-4}$.

- 1: Set u_1 to zero;
- 2: **while** $\|\nabla_t\| \geq tol$ **do**
- 3:   Choose $A_t \subseteq [m]$, where $|A_t| = k$, uniformly at random.
- 4:   Compute stochastic subgradient ∇_t using Equation (15).
- 5:   Update u_{t+1} by $u_{t+1} = u_t - \eta_t \nabla_t$.
- 6: $t = t + 1$
- 7: **end**

Output: Optimal hyperplane parameters $u = \frac{1}{T} \sum_{t=1}^T u_t$.

3.2. Nonlinear Case

Support vector machine has the advantage of being able to be employed using symmetric kernels rather than having direct access to the feature vectors x , that is instead of considering predictors that are linear functions of the training samples x themselves, predictors that are linear functions of some implicit mapping $\phi(x)$ of the instances are considered. In order to extend the linear SG-GPSVM to the nonlinear case by a symmetric kernel trick [32,33], the symmetric-kernel-generated surfaces are considered instead of hyperplanes and are given by:

$$w^\top \phi(x) + b = 0.$$

Then, the primal problem for the nonlinear SG-GPSVM is as follows:

$$\min_u f(u) = \frac{1}{2} \|u\|^2 + \frac{C}{m} \sum_{i=1}^m \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}(1 - y_i(w^\top \phi(x_i) + b)), \quad (17)$$

where $\phi(x)$ is representative of a nonlinear mapping function, which maps x into a higher-dimensional feature space. To apply the stochastic subgradient approach, for each iteration t , we propose a more general method that uses k samples where $1 \leq k \leq m$. We chose a subset $A_t \subseteq [m]$ with $|A_t| = k$ where k samples are drawn uniformly at random from the training set and $[m] = \{1, 2, 3, \dots, m\}$. Consider an approximate objective function:

$$f(u_t) = \frac{1}{2} \|u_t\|^2 + \frac{C}{k} \sum_{i \in A_t} \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}(1 - y_i(w_t^\top \phi(x_i) + b)), \quad (18)$$

Then, consider the subgradient of the above approximate objective, and let ∇_t be the subgradient of f at u_t , that is:

$$\nabla_t \in \partial f(u_t) = u_t - \frac{C}{k} \sum_{i \in A_t} y_i \phi(x_i) \partial \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}(1 - y_i(w_t^\top \phi(x_i) + b)), \quad (19)$$

where $\partial \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}$ is defined in Equation (14), and similar to the linear case, $\rho_i \in \partial \mathcal{L}_{\tau_1, \tau_2}^{\epsilon_1, \epsilon_2}(1 - y_i(w^\top \phi(x_i) + b))$ exists. Then, ∇_t can be written as:

$$\nabla_t = u_t - \frac{C}{k} \sum_{i \in A_t} \rho_i y_i \phi(x_i). \quad (20)$$

Then, update $u_{t+1} = u_t - \eta_t \nabla_t$ using a step size of $\eta_t = \frac{1}{t}$. Additionally, a new sample x can be predicted by:

$$y = \text{sgn}(w^\top \phi(x) + b).$$

The above steps can be outlined as Algorithm 2.

Algorithm 2 Nonlinear SG-GPSVM.

Input: Training samples are represented by $X \in \mathbb{R}^{n \times m}$,
positive parameters c_1, c_2, τ_1, τ_2 ,
 $k \in \{1, 2, \dots, m\}$, and tolerance tol ; typically,
 $tol = 10^{-4}$.

- 1: Set u_1 to zero;
- 2: **while** $\|\nabla_t\| \geq tol$ **do**
- 3:   Choose $A_t \subseteq [m]$, where $|A_t| = k$, uniformly at random.
- 4:   Compute stochastic subgradient ∇_t using Equation (20).
- 5:   Update u_{t+1} by $u_{t+1} = u_t - \eta_t \nabla_t$.
- 6: $t = t + 1$
- 7: **end**

Output: Optimal hyperplane parameters $u = \frac{1}{T} \sum_{t=1}^T u_t$.

However, the mapping $\phi(x)$ is never specified explicitly, but rather through a symmetric kernel operator $K(x, x^\top) = \phi(x)^\top \phi(x^\top)$ yielding the inner products after the mapping $\phi(x)$.

4. Convergence Analysis

In this section, we analyze the convergence of the proposed SG-GPSVM model. For convenience, we only consider the optimization problem (Equation (11)) together with the

conclusions for another nonlinear algorithm, which can be obtained similarly. u_1 and the step size $\eta_t = \frac{1}{t}$, u_{t+1} for $t \geq 1$ are updated by:

$$u_{t+1} = u_t - \eta_t \nabla_t, \quad (21)$$

i.e.,

$$\begin{aligned} u_{t+1} &= u_t - \eta_t \nabla_t \\ &= u_t - \frac{1}{t} \left(u_t - \frac{C}{k} \sum_{i \in A_t} \rho_i y_i z_i \right) \\ &= \left(1 - \frac{1}{t} \right) u_t + \frac{C}{kt} \sum_{i \in A_t} \rho_i y_i z_i. \end{aligned} \quad (22)$$

To prove the convergence of Algorithm 1, consider the boundedness of $\|u_t\|$ first. In fact, we have the following lemma.

Lemma 1. The sequences $\{\|\nabla_t\| : t = 1, 2, \dots\}$ and $\{\|u_t\| : t = 1, 2, \dots\}$ have upper bounds, where ∇_t and u_t are defined by Equations (15) and (22), respectively.

Proof of Lemma 1. Equation (22) can be rewritten as:

$$u_{t+1} = G_t u_t + \frac{1}{t} v_t \quad (23)$$

where $G_t = (1 - \frac{1}{t})I$, I is the identity matrix, and $v_t = \frac{C}{k} \sum_{i \in A_t} \rho_i y_i x_i$. For $t \geq 2$, G_t is positive definite, and the largest eigenvalue λ_t of G_t is equal to $\frac{t-1}{t}$. From Equation (23), we have:

$$u_{t+1} = \prod_{i=2}^t G_i u_2 + \sum_{i=2}^t \bar{G}_i, \quad (24)$$

where:

$$\bar{G}_i = \begin{cases} \frac{1}{i} \left(\prod_{j=i+1}^t u_j \right) v_i & \text{if } i < t, \\ \frac{1}{i} u_i & \text{if } i = t. \end{cases} \quad (25)$$

For any $i \geq 2$,

$$\|G_i u_2\| \leq \lambda_i \|u_2\| \leq \frac{i-1}{i} \|u_2\|.$$

Therefore,

$$\begin{aligned} \left\| \prod_{i=2}^t G_i u_2 \right\| &= \|G_2 G_3 \cdots G_t u_2\| \\ &\leq \|G_2\| \|G_3\| \cdots \|G_t\| \|u_2\| \\ &\leq \frac{2-1}{2} \cdot \frac{3-1}{3} \cdots \frac{(t-1)}{t} \|u_2\| \\ &= \frac{1}{t} \|u_2\|. \end{aligned} \quad (26)$$

Next, consider $\|\bar{G}_i\|$. In Case I for $i < t$, we have:

$$\|\bar{G}_i\| = \left\| \frac{1}{i} \left(\prod_{j=i+1}^t G_j \right) v_i \right\| \quad (27)$$

$$\begin{aligned} &= \left\| \frac{1}{i} G_{i+1} G_{i+2} \cdots G_t v_i \right\| \\ &\leq \frac{1}{i} \|G_{i+1}\| \|G_{i+2}\| \cdots \|G_t\| \|v_i\| \\ &\leq \frac{1}{i} \cdot \frac{(i+1)-1}{i+1} \cdot \frac{(i+2)-1}{i+2} \cdots \frac{t-1}{t} \|v_i\| \\ &= \frac{1}{t} \|v_i\| \\ &\leq \frac{1}{t} \max_{i < t} \|v_i\|, \end{aligned} \quad (28)$$

and in Case II, for $i = t$, we have:

$$\|\bar{G}_i\| = \left\| \frac{1}{i} u_i \right\| = \frac{1}{t} \|u_t\| \leq \frac{1}{t} \max_{i \leq t} \|v_i\|. \quad (29)$$

From Case I and Case II, we have:

$$\|\bar{G}_i\| \leq \frac{1}{t} \max_{i \leq t} \|v_i\|. \quad (30)$$

Thus,

$$\begin{aligned} \|u_{t+1}\| &= \left\| \prod_{i=2}^t G_i u_2 + \sum_{i=2}^t \bar{G}_i \right\| \\ &\leq \frac{1}{t} \|u_2\| + \sum_{i=2}^t \frac{1}{t} \max_{i \leq t} \|v_i\| + \frac{1}{t} \max_{i \leq t} \|v_i\| - \frac{1}{t} \max_{i \leq t} \|v_i\| \\ &= \frac{1}{t} \|u_2\| + t \left(\frac{1}{t} \max_{i \leq t} \|v_i\| \right) - \left(\frac{1}{t} \max_{i \leq t} \|v_i\| \right) \\ &= \frac{1}{t} \|u_2\| + \frac{t-1}{t} \max_{i \leq t} \|v_i\| \\ &\leq \|u_2\| + C \max\{\tau_1, \tau_2\} \max_{i \in [m]} \|z_i\|. \end{aligned} \quad (31)$$

Let M_0 be the largest norm of the samples in the dataset and:

$$M_1 = \max\{\|u_1\|, \|u_2\| + C \max\{\tau_1, \tau_2\} M_0\},$$

that is $\|u_t\| \leq M_1, \forall t = 1, 2, \dots$ For $t \geq 1$,

$$\begin{aligned} \|\nabla_t\| &= \left\| u_t - \frac{C}{k} \sum_{i \in A_t} \rho_i y_i z_i \right\| \\ &\leq \|u_t\| + \left\| \frac{C}{k} \sum_{i \in A_t} \rho_i y_i z_i \right\| \\ &\leq M_1 + C \max\{\tau_1, \tau_2\} M_0 = M_2. \end{aligned}$$

□

The following theorem demonstrates that we derive the proof of the convergence property of Algorithm 1 by using Lemma 1.

Theorem 1. When using Algorithm 1 to solve the optimization problem Equation (11), the iterative Equation (22) is convergent.

Proof of Theorem 1. From Equation (26) in the proof of Lemma 1, we therefore obtain:

$$\lim_{t \rightarrow \infty} \left\| \prod_{i=2}^t G_i u_2 \right\| = 0, \quad (32)$$

which implies that:

$$\lim_{t \rightarrow \infty} \prod_{i=2}^t G_i u_2 = 0. \quad (33)$$

By using Equation (30), we have:

$$\|\bar{G}_i\| \leq \frac{1}{t} \max_{i \leq t} \|v_i\| \leq C \max\{\tau_1, \tau_2\} \max_{i \in [m]} \|z_i\|, \quad (34)$$

which indicates that:

$$\lim_{t \rightarrow \infty} \sum_{i=2}^t \|\bar{G}_i\| < \infty. \quad (35)$$

Note that an infinite series of vectors is convergent if its norm series is convergent [34]. Therefore, the following limit exists:

$$\lim_{t \rightarrow \infty} \sum_{i=2}^t \bar{G}_i < \infty. \quad (36)$$

Combining Equations (33) and (36), we conclude that the series of u_{t+1} is convergent if $t \rightarrow \infty$. $\square$

The following theorem gives the relation between GPSVM and our proposed SG-GPSVM.

Theorem 2. Let f be a convex function and u_t be defined by Equations (11) and (21), respectively. If $\bar{u} = \frac{1}{T} \sum_{t=1}^T u_t$ is the solution output of SG-GPSVM, then:

$$f(\bar{u}) \leq f(u^*) + M_2(M_1 + \|u^*\|) + \frac{M_2^2}{2T}(1 + \ln T)$$

where M_1 and M_2 are the upper bounds of $\|u_t\|$ and $\|\nabla_t\|$, respectively.

Proof of Theorem 2. By the definition of $\bar{u}$, we have:

$$\begin{aligned} f(\bar{u}) - f(u^*) &= f\left(\frac{1}{T} \sum_{t=1}^T u_t\right) - f(u^*) \\ &\leq \frac{1}{T} \sum_{t=1}^T f(u_t) - f(u^*) \\ &= \frac{1}{T} \sum_{t=1}^T (f(u_t) - f(u^*)). \end{aligned} \quad (37)$$

As f is convex and ∇_t is the subgradient of f at u_t , we have:

$$f(u_t) - f(u^*) \leq (u_t - u^*)^\top \nabla_t. \quad (38)$$

By summing over $t = 1$ to T and dividing by T , we obtain the following inequality:

$$\frac{1}{T} \sum_{t=1}^T (f(u_t) - f(u^*)) \leq \frac{1}{T} \sum_{t=1}^T (u_t - u^*)^T \nabla_t. \quad (39)$$

Since $u_{t+1} = u_t - \eta_t \nabla_t$, we have:

$$(u_t - u^*)^T \nabla_t = \frac{1}{2\eta_t} (\|u_t - u^*\|^2 - \|u_{t+1} - u^*\|^2) + \frac{\eta_t}{2} \|\nabla_t\|^2. \quad (40)$$

Summing over $t = 1$ to T , we obtain:

$$\begin{aligned} \sum_{t=1}^T ((u_t - u^*)^T \nabla_t) &= \frac{1}{2} \sum_{t=1}^T \frac{1}{\eta_t} (\|u_t - u^*\|^2 - \|u_{t+1} - u^*\|^2) + \frac{1}{2} \sum_{t=1}^T (\eta_t \|\nabla_t\|^2) \\ &= \frac{1}{2} \left(\sum_{t=1}^T \frac{1}{\eta_t} (\|u_t - u^*\|^2 - \|u_{t+1} - u^*\|^2) \right) + \frac{1}{2} \sum_{t=1}^T (\eta_t \|\nabla_t\|^2) \\ &= \frac{1}{2} \left(\sum_{t=1}^T \frac{1}{\eta_t} (\|u_t - u^*\|^2 - \|u_{T+1} - u^*\|^2) \right) + \frac{1}{2} \sum_{t=1}^T (\eta_t \|\nabla_t\|^2) \\ &\leq (M_1 + \|u^*\|) \sum_{t=1}^T \|u_{t+1} - u_t\| + \frac{1}{2} M_2^2 (1 + \ln T) \\ &= (M_1 + \|u^*\|) \sum_{t=1}^T \left\| \sum_{i=1}^T \frac{1}{i} \nabla_i \right\| + \frac{1}{2} M_2^2 (1 + \ln T) \\ &\leq T M_2 (M_1 + \|u^*\|) + \frac{1}{2} M_2^2 (1 + \ln T). \end{aligned} \quad (41)$$

Multiplying Equation (41) by $1/T$, we have:

$$\frac{1}{T} \sum_{t=1}^T ((u_t - u^*)^T \nabla_t) \leq M_2 (M_1 + \|u^*\|) + \frac{1}{2T} M_2^2 (1 + \ln T). \quad (42)$$

By Equations (37), (39), and (42), we have our result:

$$f(\bar{u}) \leq f(u^*) + M_2 (M_1 + \|u^*\|) + \frac{1}{2T} M_2^2 (1 + \ln T). \quad (43)$$

□

With Theorem 2, we showed that in a random iteration T , the resulting expected error is bounded by $O(\frac{1+\ln T}{T})$, and the above theorem provides the approximations of $f(u^*)$ by $f(\bar{u})$, that is the average instantaneous objective of SG-GPSVM correlates with the objective of GPSVM.

5. Numerical Experiments

In this section, to demonstrate the validity of our proposed SG-GPSVM, we compare SG-GPSVM with the conventional SVM [35] and Pegasos [25] using artificial datasets and the UCI Machine Learning Repository [29] with noises of different variances. All of experiments were performed with Python 3.6.3 on a Windows 8 machine with an Intel i5 Processor 2.50 GHz with 4 GB RAM.

5.1. Artificial Datasets

We conducted experiments on a two-dimensional example, for which the samples came from two Gaussian distributions with equal probability: $x_i, i \in \{i : y_i = 1\} \sim \mathcal{N}(\mu_1, \Sigma_1)$ and $x_i, i \in \{i : y_i = -1\} \sim \mathcal{N}(\mu_2, \Sigma_2)$ where $\mu_1 = [1, -3]^\top$, $\mu_2 = [-1, 3]^\top$ and $\Sigma_1 = \Sigma_2 = \begin{bmatrix} 0.2 & 0 \\ 0 & 3 \end{bmatrix}$. We added noise to the dataset. The labels of the noise points

were selected from $\{1, -1\}$ with equal probabilities. Each noisy sample was drawn from a Gaussian distribution $\mathcal{N}(\mu_n, \Sigma_n)$ where $\mu_n = [0, 0]^\top$ and $\Sigma_n = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix}$. This noise affects the labels around the boundary. The level of noise was controlled by the ratio of the noisy data in the training set, denoted by r . The value of r was fixed at $r = 0$ (i.e., noise-free), 0.05, 0.1, and 0.2. All the results of the two algorithms, SG-GPSVM and Pegasos, were derived using the two-dimensional dataset shown in Figure 1. Here, green circles denote samples from Class 1 and red triangles represent data samples from Class -1 . From Figure 1, we can see that the noisy samples affect the labels around the decision boundary. As we increase the amount of noise from $r = 0\%$ to $r = 20\%$, the hyperplanes of Pegasos start deviating from the ideal slope of 2.14, whereas the deviation in the slopes of the hyperplanes is significantly less in our SG-GPSVM. This implies that our purposed algorithm is insensitive to noise around the decision boundary.

Figure 1. SG-GPSVM and Pegasos on a noisy artificial dataset. The above four figures demonstrate the noise-insensitive properties of our purposed SG-GPSVM compared to Pegasos when we have: (a) $r = 0$ (noise free); (b) $r = 0.05$; (c) $r = 0.1$; (d) $r = 0.2$.

5.2. UCI Datasets

We also performed experiments on 11 benchmark datasets available in the UCI Machine Learning Repository. Table 1 lists the datasets and their descriptions. For the benchmarks, we compared SG-GPSVM with two other SVM-based classical classifiers: conventional SVM and Pegasos. Among these SVM models, Pegasos and SG-GPSVM have extra hyperparameters, in addition to the hyperparameter C . The performance of different algorithms depends on the choices of the parameters [33,36,37]. All the hyperparameters were chosen through the grid search method and manual adjustment. The 10-fold cross-validation evaluation method was also employed. In each algorithm, the optimal parameter C was searched from set $\{10^i | i = -2, -1, 0, 1, 2\}$, and other parameters were searched from set $\{0.1, 0.25, 0.5, 0.75, 1\}$. Further, the kernel method was adopted to evaluate the performance of SG-GPSVM as a nonlinear classifier, where the RBF kernel was adopted. The parameter of the RBF kernel is γ , searched from set $\{10^i | i = -2, -1, 0, 1, 2\}$.

The results of the numerical experiments are shown in Table 2, and the optimal parameters used in Table 2 are summarized in Table 3.

Table 1. Description of the UCI datasets.

Datasets	# of Samples	# of Features	# in Negative Class	# in Positive Class
Appendicitis	106	7	85	21
Ionosphere	351	33	126	225
Monk-2	432	6	204	228
Monk-3	432	6	204	228
Saheart	462	9	302	160
WDBC	569	30	357	212
Australian	690	14	383	307
Spambase	4597	57	2785	1812
Phoneme	5404	5	3818	1586
Twonorm	7400	21	3703	3697
Coil2000	9822	86	9236	586

Table 2. Accuracy obtained on the UCI datasets using the linear and nonlinear (*) kernel.

Datasets	r	SVM Time (s)	Pegasos Time (s)	SG-GPSVM Time (s)	SG-GPSVM * Time (s)
Appendicitis	0	85.82 ± 14.33	83.82 ± 13.72	84.82 ± 13.02	87.91 ± 10.45
		0.0008	0.3198	0.4800	0.0899
	0.05	83.00 ± 14.17	81.82 ± 15.16	84.73 ± 11.34	88.91 ± 10.74
		0.0020	0.3099	0.4321	0.0883
	0.10	84.00 ± 8.61	85.55 ± 14.96	85.64 ± 13.33	86.91 ± 9.69
		0.0020	0.3169	0.4393	0.0887
	0.20	80.18 ± 13.51	79.18 ± 12.74	84.82 ± 13.55	86.82 ± 12.38
		0.0015	0.2962	0.4972	0.0880
Phoneme	0	77.48 ± 1.41	75.98 ± 1.96	77.31 ± 1.40	77.15 ± 2.64
		0.5502	0.3102	0.4015	0.0893
	0.05	77.42 ± 1.44	77.15 ± 2.020	77.44 ± 1.56	76.87 ± 2.07
		0.5153	0.3490	0.4197	0.0891
	0.10	77.18 ± 1.58	76.59 ± 1.71	76.48 ± 1.81	76.61 ± 2.15
		0.5020	0.3087	0.4083	0.0891
	0.20	76.76 ± 1.69	75.61 ± 1.63	76.33 ± 1.57	76.44 ± 2.33
		0.5210	0.3016	0.3917	0.0893
Monk-2	0	80.32 ± 8.96	79.85 ± 8.12	80.80 ± 7.66	81.48 ± 7.54
		0.0134	0.2993	0.4268	0.0891
	0.05	80.55 ± 8.96	79.38 ± 7.93	79.16 ± 6.33	79.86 ± 7.98
		0.0027	0.2835	0.4038	0.0894
	0.10	79.39 ± 8.55	76.60 ± 6.33	80.54 ± 7.82	80.53 ± 6.56
		0.0023	0.2886	0.3876	0.0894
	0.20	79.40 ± 7.07	77.09 ± 7.56	80.32 ± 8.14	80.11 ± 5.54
		0.0025	0.3099	0.3815	0.0900
Monk-3	0	80.53 ± 4.61	77.75 ± 5.68	80.30 ± 4.50	84.47 ± 4.82
		0.0047	0.2835	0.4899	0.0947
	0.05	80.35 ± 4.75	78.01 ± 4.03	80.30 ± 4.38	84.23 ± 8.64
		0.0067	0.2932	0.4856	0.0970
	0.10	78.69 ± 3.46	79.40 ± 4.59	80.07 ± 4.50	83.31 ± 8.07
		0.0031	0.2974	0.4675	0.0956
	0.20	77.99 ± 4.86	77.32 ± 6.28	80.32 ± 3.36	81.93 ± 6.99
		0.0016	0.2875	0.4918	0.0994

Table 2. Cont.

Datasets	r	SVM Time (s)	Pegasos Time (s)	SG-GPSVM Time (s)	SG-GPSVM * Time (s)
Saheart	0	71.83 $\pm$ 7.22 0.0047	70.52 $\pm$ 6.62 0.3071	72.49 $\pm$ 7.45 0.4038	72.32 $\pm$ 4.42 1.4542
	0.05	71.39 $\pm$ 7.66 0.0047	71.84 $\pm$ 5.01 0.2989	72.48 $\pm$ 6.70 0.3816	71.87 $\pm$ 3.91 1.4453
	0.10	72.27 $\pm$ 7.11 0.0052	67.98 $\pm$ 4.19 0.3249	71.38 $\pm$ 8.65 0.4107	71.00 $\pm$ 5.38 1.4201
	0.20	72.50 $\pm$ 7.65 0.0051	72.47 $\pm$ 8.19 0.3160	71.82 $\pm$ 10.10 0.3830	70.80 $\pm$ 6.23 1.4503
Ionosphere	0	88.32 $\pm$ 6.94 0.0080	84.02 $\pm$ 8.32 0.3541	86.60 $\pm$ 6.15 0.3913	91.46 $\pm$ 5.25 0.0944
	0.05	87.46 $\pm$ 5.75 0.0046	86.03 $\pm$ 5.35 0.3176	86.88 $\pm$ 6.30 0.4110	91.45 $\pm$ 2.86 0.0937
	0.10	83.74 $\pm$ 7.15 0.0076	84.60 $\pm$ 6.31 0.3160	86.03 $\pm$ 7.83 0.8683	91.17 $\pm$ 3.25 0.0941
	0.20	82.31 $\pm$ 7.78 0.0079	85.46 $\pm$ 4.88 0.3233	87.73 $\pm$ 5.74 0.4122	90.03 $\pm$ 5.12 0.0936

Note: * Nonlinear case.

Table 2 shows the results for a linear and nonlinear kernel on six different UCI datasets by applying conventional SVM, Pegasos, and our proposed SG-GPSVM. From the experimental outcomes, it can be seen that the classification performance of SG-GPSVM is better than that of the conventional SVM and Pegasos on most of the datasets in terms of accuracy. From Table 2, our proposed SG-GPSVM yielded the best prediction accuracy on four of six datasets. In addition, when we varied the number of noisy samples from $r = 0$ (noise-free) to $r = 0.2$, our proposed SG-GPSVM exhibited better classification accuracy and stability than the other algorithms no matter the noise factor value, as shown in Figure 2. On the Appendicitis dataset, when the noise factor increased, the accuracy of the conventional SVM and Pegasos fluctuated greatly. On the contrary, the classification accuracies of our proposed SG-GPSVM were stable. Similarly, on the other datasets, such as Monk-3 and Appendicitis, the accuracies of the conventional SVM and Pegasos not only varied greatly, but they also performed worse than our proposed algorithm. This demonstrates that our proposed SG-GPSVM is relatively stable, regardless of the noise factor. In terms of computational time, our proposed algorithm cost nearly the same as Pegasos. The computational time of Pegasos and our proposed SG-GPSVM on the largest dataset was longer than the computational time of the conventional SVM on the smaller datasets.

Table 3. The optimal parameters of conventional SVM, Pegasos, SG-GPSVM, and SG-GPSVM *.

Datasets	SVM	Pegasos	SG-GPSVM	SG-GPSVM *
	C	C	$C, \tau_1, \tau_2, \epsilon_1, \epsilon_2$	$C, \tau_1, \tau_2, \epsilon_1, \epsilon_2, \gamma$
Appendicitis	1	10	100, 0.5, 0.75, 0.1, 1	10, 1, 1, 0.5, 0.5, 0.1
Phoneme	1	1	10, 0.5, 0.5, 0.1, 1	100, 1, 1, 0.25, 0.25, 1
Monk-2	1	0.01	100, 1, 0.25, 0.1, 0.25	100, 0.25, 0.25, 0.1, 0.25, 0.1
Monk-3	1	0.1	1, 1, 0.25, 0.5, 0.5	10, 1, 0.75, 0.25, 0.25, 0.1
Saheart	1	1	1, 0.5, 0.25, 0.25, 0.1	100, 1, 0.75, 0.25, 0.25, 0.1
Ionosphere	1	1	1, 0.75, 0.1, 0.25, 0.1	100, 1, 1, 1, 0.5, 0.1

Note: * Nonlinear case.

The experimental results of three different minibatches for SG-GPSVM (1, 32, 512) and the accuracy and computational time results of the methods are illustrated in Figure 3. Subfigure (a) shows how the accuracy changed in terms of the batch size, and the curves in Subfigure (b) show how the computational time changed in terms of the batch size. We can see that when the size of the minibatch increased, the accuracy increased, and the computational time increased approximately linearly when the batch size increased. The datasets Spambase and WDBC were taken into consideration. It is evident that if the minibatch size k is small enough, $k = 32$ is more accurate than $k = 512$. However, when using a large minibatch size of $k = 512$, the accuracy decreased and the computational time increased. This suggests that in the scenarios in which we care more about the accuracy, we should choose a relatively medium batch size (it does not need to be large).

Figure 2. The accuracy of four algorithms with different noise factors on six datasets: (a) Appendicitis; (b) Ionosphere; (c) Monk-2; (d) Monk-3; (e) Phoneme; (f) Saheart.

Figure 3. Classification results on the UCI datasets with minibatch SG-GPSVM. (a) Accuracy of minibatch SG-GPSVM. (b) Computational time of minibatch SG-GPSVM.

5.3. Large-Scale Dataset

In order to validate the classification efficiency of SG-GPSVM, we conducted comparisons of our methods with the two most related methods (SVM and Pegasos) on four large-scale UCI datasets. We show that our SG-GPSVM is capable of solving large-scale problems. Note that the results of this section were solved on a machine equipped with an Intel CPU E5-2658 v3 at 2.20 GHz and 256 GB RAM running the Ubuntu Linux operating system. The scikit-learn package was used for the conventional SVM [35]. The large-scale datasets are presented in Table 4.

Table 4. The details of the large-scale datasets.

Datasets	# of Samples	# of Features	#in Negative Class	# in Positive Class
Credit card	30,000	324	23,364	6636
Skin	245,057	3	194,198	50,859
Kddcup	4,898,431	41	972,781	3,925,650
SUSY	5,000,000	18	2,712,173	2,287,827

For the four large-scale datasets, we compare the performance of SG-GPSVM against the conventional SVM and Pegasos in Table 5. The reduced kernel [38] was employed in the nonlinear situation, and the kernel size was set to 100. From the results in Table 5, SG-GPSVM outperformed the conventional SVM and Pegasos on three out of the four datasets in terms of accuracy. The best ones are highlighted in bold. However, SG-GPSVM's accuracy on some datasets, such as Credit card, was not the best. On the Skin dataset, the conventional SVM performed much worse than SG-GPSVM and Pegasos, and it was not possible to use the conventional SVM on the Kddcup and Susy datasets due to the high memory requirements. This was because SVM requires the complete training set to be stored in the main memory. On the Credit card dataset, our proposed SG-GPSVM took almost the same amount of time as the conventional SVM and Pegasos. Our suggested approach, SG-GPSVM, took almost the same amount of time as Pegasos on the Kddcup and Susy datasets.

Table 5. Classification results on the large-scale UCI datasets.

Datasets	Mean Accuracy in %			
	Credit Card	Skin	Kddcup	Susy
SVM	80.82 ± 0.90	92.90 ± 0.16	*	*
Time (s)	0.4398	289.4454	*	*
Pegasos	79.74 ± 2.38	93.22 ± 0.78	95.31 ± 0.31	99.25 ± 1.37
Time (s)	0.2479	0.2676	1.9823	4.5164
SG-GPSVM	80.72 ± 0.95	93.98 ± 0.38	98.82 ± 0.60	99.97 ± 0.06
Time (s)	0.4417	0.3751	2.0474	5.3744
SG-GPSVM *	77.88 ± 0.87	94.06 ± 1.13	96.92 ± 0.35	99.99 ± 0.02
Time (s)	1.0881	0.9738	2.8125	2.3218

Note: * Nonlinear case.

6. Conclusions

In this paper, we used the generalized pinball loss function in SVM to perform classification and proposed the SG-GPSVM classifier. This paper adapted the stochastic subgradient descent method as stochastic subgradient descent method-based generalized pinball SVM (SG-GPSVM). Compared to the hinge loss SVM and Pegasos, the major advantage of our proposed method is that SG-GPSVM is less sensitive to noise, especially the feature noise around the decision boundary. In addition, we investigated the convergence of SG-GPSVM and the theoretical approximation between GPSVM and SG-GPSVM. The validity of our proposed SG-GPSVM was demonstrated by numerical experiments on artificial datasets and datasets from UCI with noises of different variances. The experimental

results clearly showed that our suggested SG-GPSVM outperformed the existing classifier approach in terms of accuracy, and SG-GPSVM has a significant advantage in handling large-scale classification problems. The results imply that the SG-GPSVM approach is the strongest candidate for solving binary classification problems.

In further work, we would like to consider applications of SG-GPSVM to activity recognition datasets and image retrieval datasets, and we also plan to improve our approach to deal with multicategory classification scenarios.

Author Contributions: Conceptualization, W.P. and R.W.; writing—original draft, W.P.; writing—review and editing, W.P. and R.W. Both authors read and agreed to the published version of the manuscript.

Funding: This research was funded by the NSRF and NU, Thailand, with Grant Number R2564B024.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to thank the referees for careful reading and constructive comments. This research is partially supported by the Development and Promotion of the Gifted in Science and Technology Project and Naresuan University.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Vapnik, V.N. *The Nature of Statistical Learning Theory*; Springer: New York, NY, USA, 1995.
2. Vapnik, V.N. *Statistical Learning Theory*; Wiley: New York, NY, USA, 1998.
3. Vapnik, V.N. An overview of statistical learning theory. *IEEE Trans. Neural Netw.* **1999**, *10*, 988–999. [[CrossRef](#)] [[PubMed](#)]
4. Colmenarez, A.J.; Huang, T.S. Face Detection With Information-Based Maximum Discrimination. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, PR, USA, 17–19 June 1997; pp. 782–787.
5. Osuna, E.; Freund, R.; Girosi, F. Training support vector machines: An application to face detection. In Proceedings of the IEEE Computer Vision and Pattern Recognition, San Juan, PR, USA, 17–19 June 1997.
6. Joachims, T.; Ndellec, C.; Rouveriol, C. Text categorization with support vector machines: Learning with many relevant features. In Proceedings of the European Conference on Machine Learning, Chemnitz, Germany, 21–23 April 1998; pp. 137–142.
7. Richhariya, B.; Tanveer, M. EEG signal classification using universum support vector machine. *Expert Syst. Appl.* **2018**, *106*, 169–182. [[CrossRef](#)]
8. Mukherjee, S.; Osuna, E.; Girosi, F. Nonlinear prediction of chaotic time series using a support vector machine. In Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing, Amelia Island, FL, USA, 24–26 September 1997.
9. Ince, H.; Trafalis, T.B. Support vector machine for regression and applications to financial forecasting. In Proceedings of the International Joint Conference on Neural Networks (IEEE-INNSNNS), Como, Italy, 27 July 2000.
10. Huang, Z.; Chen, H.; Hsua, C.-J.; Chen, W.-H.; Wu, S. Credit rating analysis with support vector machines and neural networks: A market comparative study. *Decis. Support Syst.* **2004**, *37*, 543–558. [[CrossRef](#)]
11. Khemchandani, R.; Jayadeva; Chandra, S. Regularized least squares fuzzy support vector regression for nancial time series forecasting. *Expert Syst. Appl.* **2009**, *36*, 132–138. [[CrossRef](#)]
12. Tao, D.; Tang, X.; Li, X.; Wu, X. Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *28*, 1088–1099.
13. Pal, M.; Mather, P. Support vector machines for classification in remote sensing. *Int. J. Remote Sens.* **2005**, *26*, 1007–1011. [[CrossRef](#)]
14. Li, Y.Q.; Guan, C.T. Joint feature re-extraction and classification using an iterative semi-supervised support vector machine algorithm. *Mach. Learn.* **2008**, *71*, 33–53. [[CrossRef](#)]
15. Bi, J.; Zhang, T. Support vector classification with input data uncertainty. In Proceedings of the Neural Information Processing Systems, Vancouver, BC, Canada, 13–18 December 2004.
16. Huang, X.; Shi, L.; Suykens, J.A.K. Support vector machine classifier with pinball loss. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 984–997. [[CrossRef](#)]
17. Rastogi, R.; Pal, A.; Chandra, S. Generalized pinball loss SVMs. *Neurocomputing* **2018**, *322*, 151–165. [[CrossRef](#)]
18. Nanculef, R.; Frandi, E.; Sartori, C.; Allende, H. A Novel Frank-Wolfe Algorithm. Analysis and Applications to Large-Scale SVM Training. *Inf. Sci.* **2014**, *285*, 66–99. [[CrossRef](#)]
19. Xu, J.; Xu, C.; Zou, B.; Tang, Y.Y.; Peng, J.; You, X. New incremental learning algorithm with support vector machines. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *49*, 2230–2241. [[CrossRef](#)]

20. Platt, J. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods—Support Vector Learning*; MIT Press: Cambridge, MA, USA, 1999; pp. 185–208.
21. Mangasarian, O.; Musicant, D. Successive overrelaxation for support vector machines. *IEEE Trans. Neural Netw.* **1999**, *10*, 1032–1037. [[CrossRef](#)]
22. Fan, R.; Chang, K.; Hsieh, C.; Wang, X.; Lin, C. LIBLINEAR: A library for large linear classification. *J. Mach. Learn. Res.* **2008**, *9*, 1871–1874.
23. Zhang, T. Solving large-scale linear prediction problems using stochastic gradient descent algorithms. In Proceedings of the International Conference on Machine Learning, Banff, AB, Canada, 4–8 July 2004.
24. Xu, W. Towards optimal one pass large-scale learning with averaged stochastic gradient descent. *arXiv* **2011**, arXiv:1107.2490.
25. Shai, S.; Singer, Y.; Srebro, N.; Cotter, A. Pegasos: Primal estimated subgradient solver for SVM. *Math. Program.* **2011**, *127*, 3–30.
26. Alencar, M.; Oliveira, D.J. Online learning early skip decision method for the hevc inter process using the SVM-based Pegasos algorithm. *Electron. Lett.* **2016**, *52*, 1227–1229.
27. Reyes-Ortiz, J.; Oneto, L.; Anguita, D. Big data analytics in the cloud: Spark on Hadoop vs MPI/OpenMP on Beowulf. *Procedia Comput. Sci.* **2015**, *53*, 121–130. [[CrossRef](#)]
28. Sopyla, K.; Drozd, P. Stochastic gradient descent with barzilaiborwein update step for SVM. *Inf. Sci.* **2015**, *316*, 218–233. [[CrossRef](#)]
29. Dua, D.; Taniskidou, E.K. *UCI Machine Learning Repository*; University of California, School of Information and Computer Science: Irvine, CA, USA, 2019. Available online: <http://archive.ics.uci.edu/ml> (accessed on 24 September 2018).
30. Shwartz, S.; Ben-David, S. *Understanding Machine Learning Theory Algorithms*; Cambridge University Press: Cambridge, MA, USA, 2014; 207p.
31. Xu, Y.; Yang, Z.; Pan, X. A novel twin support-vector machine with pinball loss. *IEEE Trans. Neural Netw. Learn. Syst.* **2017**, *28*, 359–370. [[CrossRef](#)]
32. Shao, Y.; Zhang, C.; Wang, X.; Deng, N. Improvements on twin support vector machines. *IEEE Trans. Neural Netw.* **2011**, *22*, 962–968. [[CrossRef](#)] [[PubMed](#)]
33. Khemchandani, R.; Jayadeva; Chandra, S. Optimal kernel selection in twin support vector machines. *Optim. Lett.* **2009**, *3*, 77–88. [[CrossRef](#)]
34. Rudin, W. *Principles of Mathematical Analysis*, 3rd ed.; McGraw-Hill: New York, NY, USA, 1964.
35. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
36. Cristianini, N.; Shawe-Taylor, J. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*; Cambridge University Press: Cambridge, UK, 2000.
37. Shawe-Taylor, J.; Cristianini, N. *Kernel Methods for Pattern Analysis*; Cambridge University Press: Cambridge, UK, 2004.
38. Lee, Y.; Mangasarian, O. RSVM: Reduced support vector machines. In Proceedings of the First SIAM International Conference on Data Mining, Chicago, IL, USA, 5–7 April 2001.