Ann Oper Res (2019) 276:155–168
https://doi.org/10.1007/s10479-017-2724-8

S.I.: COMPUTATIONAL BIOMEDICINE

An improvement on parametric ν-support vector algorithm for classification

Saeed Ketabchi[1] · **Hossein Moosaei**[2] · **Mohamad Razzaghi**[1] · **Panos M. Pardalos**[3]

Published online: 18 December 2017

Abstract One effective technique that has recently been considered for solving classification problems is parametric ν-support vector regression. This method obtains a concurrent learning framework for both margin determination and function approximation and leads to a convex quadratic programming problem. In this paper we introduce a new idea that converts this problem into an unconstrained convex problem. Moreover, we propose an extension of Newton's method for solving the unconstrained convex problem. We compare the accuracy and efficiency of our method with support vector machines and parametric ν-support vector regression methods. Experimental results on several UCI benchmark data sets indicate the high efficiency and accuracy of this method.

Keywords Classification · Support vector regression · ν-support vector machines · Parametric ν-support vector machines · Generalized Newton method · Parametric margin

✉ Saeed Ketabchi
sketabchi@guilan.ac.ir

Hossein Moosaei
hmoosaei@gmail.com; moosaei@ub.ac.ir

Mohamad Razzaghi
razzaghim@phd.guilan.ac.ir; razzaghi.mohamad@gmail.com

Panos M. Pardalos
pardalos@ise.ufl.edu

[1] Department of Applied Mathematics, Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran

[2] Department of Mathematics, Faculty of Science, University of Bojnord, Bojnord, Iran

[3] Department of Industrial and Systems Engineering, "Center for Applied Optimization", University of Florida, Gainesville, FL, USA

1 Introduction

Classification has numerous practical applications including medical science, letter and number recognition, voice recognition, face recognition and hand-writing (Joachims 1998; Cao and Tay 2001; Osuna et al. 1997; Ivanciuc 2007). The first idea of classification as a support vector machine (SVM) was introduced by Vapnik and Chervonenkis (1974). A new method to obtain the separating hyperplane has recently been considered, the parametric v-support vector classification (Par ν-SVC) (Hao 2010).

To date, many methods have been proposed for the classification of data by SVM as a hyperplane with maximum margin [The maximum margin hyperplane was shown to minimize an upper bound of the generalization error according to the Vapnik theory (Vapnik and Chervonenkis 1974; Bennett and Bredensteiner 2000)], regression classifier, etc (Deng et al. 2012; Pappu et al. 2015; Xanthopoulos et al. 2014). The ν-support vector regression is a new class of SVM. It can handle both classification and regression (Schölkopf et al. 2000; Schölkopf and Smola 2001; Pontil et al. 1998). Schölkopf et al. introduced a new parameter ν which can control the number of support vectors and training errors.

Finally, a new method for obtaining the regression line that has recently been considered is the parametric ν-support vector regression (Par ν-SVR) (Hao 2010; Wang et al. 2014).

All the above-mentioned methods are used to solve constrained quadratic problems.

In this paper, we introduce a new idea for converting the constrained convex quadratic problem into an unconstrained convex problem.There are several approaches for solving unconstrained convex optimization problems (Resende and Pardalos 2002). One important and fast established methods for convex unconstrained problems is Newton's method. Because in our case the objective function is not twice differentiable, we use the generalized Newton's method.

Our notations are described as follows:

Let $a = [a_i]$ be a vector in R^n. By a_+ we mean a vector in R^n whose ith entry is 0 if $a_i < 0$ and equals a_i if $a_i \geq 0$. If f is a real valued function defined on the n-dimensional real space R^n, the gradient of f at x is denoted by $\bigtriangledown f(x)$ which is a column vector in R^n, and the $n \times n$ Hessian matrix of second partial derivatives of f at x is denoted by $\bigtriangledown^2 f(x)$. By A^T we mean the transpose of matrix A, and $\nabla f(x)^T d$ is called directional derivative of f at x in direction d. For the two vectors x and y in the $n-$dimensional real space, $x^T y$ denotes the scalar product. For $x \in R^n$, $\|x\|$ denotes $2-$norm. A column vector of ones of arbitrary dimension will be indicated by e. For $A \in R^{m\times n}$ and $B \in R^{n\times l}$; the kernel $K(A; B)$ is an arbitrary function which maps $R^{m\times n} \times R^{n\times l}$ into $R^{m\times l}$. In particular, if x and y are column vectors in R^n then, $K(x^T; y)$ is a real number, $K(x^T; A^T)$ is a row vector in R^m , and $K(A; A^T)$ is an $m \times m$ matrix. The convex hull of a set S has been shown by $co\{S\}$. The identity $n \times n$ matrix will be denoted by $I_{n\times n}$.

2 Parametric ν-support vector classification

For a classification problem, a data set (x_i, y_i) is given for training with the input $x_i \in R^n$ and the corresponding target value or label $y_i = 1$ or -1 i.e.:

$$(x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^n \times \{\pm 1\}.$$

A classification problem finds the unique hyperplane $w^T x + b = 0$ $(w, x \in R^n,\ b \in R)$ that best separates the two classes of data.

Schölkopf et al. proposed a new class of support vector machines which called ν-support vector machine or ν-support vector classification (ν-SVC) (Schölkopf et al. 2000; Schölkopf and Smola 2001). In ν-SVC, there is a parameter ν for controlling the number of support vectors and this parameter also can eliminate one of the other free parameters of the original support vector algorithms (Schölkopf and Smola 2001).

A modification of the ν-SVC algorithm, called Par ν-SVC, which considers a parametric-margin model of arbitrary shape (Hao 2010). In fact, in the Par ν-SVC we consider a parametric margin $g(x) = c^T x + d$ and hyperplane $f(x) = w^T x + b$ that classify data if and only if:

$$y_i \left(w^T x_i + b \right) \geq c^T x_i + d, \quad y_i \in \{\pm 1\}, \quad i = 1, \ldots, n. \tag{1}$$

For finding function $f(x)$ and $g(x)$ as follows using the minimization problem (Hao 2010)

$$\begin{aligned} &\min_{w,b,c,d,\xi} \frac{1}{2}\|w\|^2 + C\left(-\nu\left(\frac{1}{2}\|c\|^2 - d\right) + \frac{1}{n}\sum_{i=1}^{n}\xi_i\right) \\ &s.t.\ y_i \left(w^T x_i + b \right) \geq (c^T x_i + d) - \xi_i, \\ &\xi_i \geq 0, \quad i = 1, \ldots, n, \end{aligned} \tag{2}$$

where C and ν are the penalty parameters.

In Par ν-SVC, a margin of separation between the two pattern classes is maximized, and the solutions are those examples that lie closest to this margin (Boser et al. 1992).

Also, it is obvious that the objective function of Par ν-SVC is an non-convex function and so we are motivated to consider other techniques for finding an approximate solution of (2).

We know that the regression is more general than classification and if we apply Par ν-SVR to a binary classification dataset, then under some conditions, Par ν-SVR gives the same solution as Par ν-SVC. For this reason, we review ν-SVR and Par ν-SVR formulation for a linear two-class classifier.

In the ε-support vector regression (ε-SVR) (Schölkopf and Smola 2001) for classification, our goal is to solve the following constrained optimization problem

$$\begin{aligned} &\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ &s.t.\ \left(w^T x_i + b \right) - y_i \leq \varepsilon + \xi_i, \\ &y_i - \left(w^T x_i + b \right) \leq \varepsilon + \xi_i^*, \\ &\xi_i^*, \xi_i \geq 0. \quad i = 1, \ldots, n \end{aligned} \tag{3}$$

The ν-SVR algorithm alleviates the problem (3) by considering ε part of the optimization problem because it is difficult to select appropriate value of the ε in ε-SVR (Schölkopf et al. 2000; Schölkopf and Smola 2001).

Then the minimization problem of ν-SVR is as follows:

$$\begin{aligned} &\min_{w,b,\xi,\xi^*,\varepsilon} \frac{1}{2}\|w\|^2 + C\left(\nu\varepsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*)\right) \\ &s.t.\ \left(w^T x_i + b \right) - y_i \leq \varepsilon + \xi_i, \end{aligned}$$

$$y_i - \left(w^T x_i + b\right) \leq \varepsilon + \xi_i^*,$$
$$\xi_i^*, \xi_i \geq 0. \quad i = 1, \ldots, n$$

Everything above ε is captured in slack variables ξ_i and ξ_i^*, which are penalized in the objective function via a regularization constant C, chosen a priori (Vapnik 1998). The size of ε is traded off against model complexity and slack variables via a constant $\nu > 0$.

In a Par ν-SVR, we consider a parametric margin $g(x) = c^T x + d$ instead of ε in ν-SVR. Especially the hyperplane $f(x) = w^T x + b$ classifies data if and only if (Hao 2010; Wang et al. 2014):

$$\left(w^T x_i + b\right) \geq c^T x_i + d \quad for \;\; y_i = +1,$$
$$\left(w^T x_i + b\right) \leq -c^T x_i - d \quad for \;\; y_i = -1.$$

Using the following minimization problem, we find $f(x)$ and $g(x)$ simultaneously

$$\begin{aligned} \min_{w,b,c,d,\xi^*,\xi} \quad & \frac{1}{2}\|w\|^2 + C\left(\nu\left(\frac{1}{2}\|c\|^2 + d\right) + \frac{1}{n}\sum_{i=1}^{n}\left(\xi_i + \xi_i^*\right)\right) \\ s.t. \quad & \left(w^T x_i + b\right) + \left(c^T x_i + d\right) \geq y_i - \xi_i, \\ & \left(c^T x_i + d\right) - \left(w^T x_i + b\right) \leq y_i + \xi_i^*, \\ & \xi_i^*, \xi_i \geq 0, \; i = 1, \ldots, n, \end{aligned} \tag{4}$$

where C and ν are positive penalty parameters.

The point that is important here is that the Par ν-SVR for classification leads to a convex problem. Figure 1 illustrates the Par ν-SVR for classification graphically.

The Lagrangian corresponding to the problem (4) is given by

$$\begin{aligned} L(w, b, c, d, \alpha, \alpha^*, \beta, \beta^*, \xi, \xi^*) = {} & \frac{1}{2}\|w\|^2 + C\left(\nu\left(\frac{1}{2}\|c\|^2 + d\right) + \frac{1}{n}\sum_{i=1}^{n}\left(\xi_i + \xi_i^*\right)\right) \\ & - \sum_{i=1}^{n}\alpha_i\left[\left(w^T x_i + b\right) + \left(c^T x_i + d\right) - y_i + \xi_i\right] \\ & - \sum_{i=1}^{n}\alpha_i^*\left[\left(w^T x_i + b\right) - \left(c^T x_i + d\right) + y_i + \xi_i^*\right] \\ & - \sum_{i=1}^{n}\beta_i\xi_i - \sum_{i=1}^{n}\beta_i^*\xi_i^*, \end{aligned}$$

where α_i , $\alpha_i{}^*$, β_i and $\beta_i{}^*$ are the nonnegative Lagrange multipliers.

By using the Karush–Kuhn–Tucker (KKT) conditions, we obtain the dual optimization problem of (4) as (Boyd and Vandenberghe 2004)

$$\begin{aligned} \max_{\alpha,\alpha^*} \quad & -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\alpha_i - \alpha_i^*\right)\left(\alpha_j - \alpha_j^*\right) x_i{}^T x_j \\ & -\frac{1}{2C\nu}\sum_{i=1}^{n}\sum_{j=1}^{n}\left(\alpha_i + \alpha_i^*\right)\left(\alpha_j + \alpha_j^*\right) x_i{}^T x_j + \sum_{i=1}^{n}\left(\alpha_i - \alpha_i^*\right) y_i \end{aligned}$$

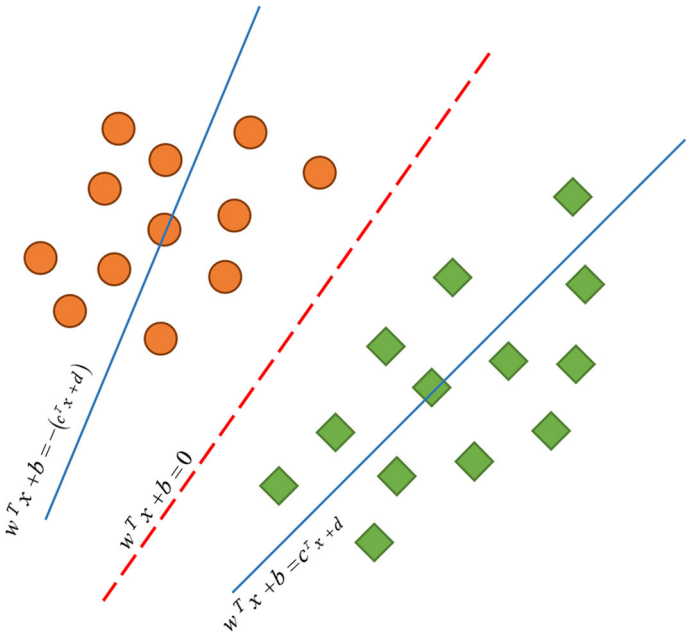

Fig. 1 Illustration of parametric ν-SVR for classification

$$
\begin{aligned}
s.t.\ & \sum_{i=1}^{n} \left(\alpha_i - \alpha_i^*\right) = 0, \\
& \sum_{i=1}^{n} \left(\alpha_i + \alpha_i^*\right) = C\nu, \\
& 0 \le \alpha_i \le \frac{C}{n},\ 0 \le \alpha_i^* \le \frac{C}{n}.\ i = 1, \ldots, n
\end{aligned}
$$

By solving the above dual problem, we obtain the Lagrange multipliers α_i and α_i^*, which give the weight vector w and c as a linear combination of x_i:

$$
\begin{aligned}
w &= \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) x_i, \\
c &= \frac{1}{C\nu} \sum_{i=1}^{n} (\alpha_i + \alpha_i^*) x_i,
\end{aligned}
$$

while the bias terms b and d are determined by exploiting the KKT conditions, which are

$$
\begin{aligned}
b &= \frac{-1}{2} \left(w^T x_i + w^T x_j + c^T x_i - c^T x_j - y_i - y_j \right), \\
d &= \frac{-1}{2} \left(w^T x_i - w^T x_j + c^T x_i + c^T x_j - y_i + y_j \right),
\end{aligned}
$$

for some i, j such that $\alpha_i, \alpha_j^* \in (0, \frac{C}{n})$.

The regression function $f(x)$ and the corresponding parametric insensitive function $g(x)$ can be obtained as follows [see Chen et al. (2012a)]:

$$f(x) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)(x_i^T x) + b,$$

$$g(x) = \frac{1}{Cv}\sum_{i=1}^{n}(\alpha_i + \alpha_i^*)(x_i^T x) + d.$$

In the next section, we focus on solving the minimization problem (4).

3 Solving quadratic constrained programming problem

We see that the classical Par v-SVR formulation is equivalent to finding the $f(x)$ and $g(x)$ simultaneously . Using 2-norm slack variables ξ_i and ξ_i^* in objective function of (4) leads to the following minimization problem (Lee and Mangasarian 2001).

$$\begin{aligned}
&\min_{w,b,c,d,\xi^*,\xi} \quad \frac{1}{2}\|w\|^2 + C\left(v\left(\frac{1}{2}\|c\|^2 + d\right) + \frac{1}{n}\sum_{i=1}^{n}\left(\|\xi_i\|^2 + \left\|\xi_i^*\right\|^2\right)\right)\\
&s.t.\left(w^T x_i + b\right) + \left(c^T x_i + d\right) \geq y_i - \xi_i,\\
&-\left(c^T x_i + d\right) + \left(w^T x_i + b\right) \leq y_i + \xi_i^*,\\
&\xi_i^*, \xi_i \geq 0.\ i = 1, \ldots, n
\end{aligned} \tag{5}$$

With respect to the Lagrangian function of (5) and KKT condition we have

$$\xi \geq Y - \left[\left(A^T w + be\right) + \left(A^T c + de\right)\right], \tag{6}$$

$$\xi^T\left(\xi - Y + \left(A^T w + be\right) + \left(A^T c + de\right)\right) = 0, \tag{7}$$

$$\xi \geq 0, \tag{8}$$

where $\xi = [\xi_1, \ldots, \xi_n]^T$, $Y = [y_1, \ldots, y_n]^T$ and $A = [x_1, \ldots, x_n]^T$. According to the inequalities (6)–(8) we have (Lee and Mangasarian 2001)

$$\xi = \left(Y - \left[\left(A^T w + be\right) + \left(A^T c + de\right)\right]\right)_+,$$

and similarly, we can show

$$\xi^* = \left(\left[\left(A^T w + be\right) - \left(A^T c + de\right)\right] - Y\right)_+.$$

Thus the problem (5) is equivalent to the following problem:

$$\begin{aligned}
\min_{w,b,c,d}\ \varphi(w,b,c,d) = \min_{w,b,c,d}\ &\frac{1}{2}\|w\|^2 + Cv\left(\frac{1}{2}\|c\|^2 + d\right)\\
&+ \frac{C}{n}\left\|\left(Y - \left[\left(A^T w + be\right) + \left(A^T c + de\right)\right]\right)_+\right\|^2\\
&+ \frac{C}{n}\left\|\left(\left[\left(A^T w + be\right) - \left(A^T c + de\right)\right] - Y\right)_+\right\|^2.
\end{aligned} \tag{9}$$

In this way, we made some modifications of Par ν-SVR that led to unconstrained convex problem (9) which we call Par ν-SVRC$^+$.

The main advantage of Par ν-SVRC$^+$ over Par ν-SVC (Hao 2010) and Par ν-SVR is solving an unconstrained convex problem rather than a large complexity of quadratic programming problem (QPP).

Our goal here is to solve the unconstrained problem (9). The objective function to problem (9) is piecewise quadratic, convex, and differentiable, but it is not twice differentiable (Chen et al. 2012b; Ketabchi and Moosaei 2012; Pardalos et al. 2014). To solve the problem (9), we have provided some definitions that deal with the objective function of this problem.

Class LC^1 of functions is defined as follows (Hiriart-Urruty et al. 1984):

Definition 1 A function f is said to be an LC^1 function on an open set A if:

1. f is continuously differentiable on A,
2. ∇f is locally Lipschitz on A.

We know that if f is a LC^1 function on an open set A , then the ∇f is differentiable almost everywhere in A, and its generalized Jacobian in Clarkes sense can be defined (Clarke 1990).

Now, the generalized Hessian of f at x to be the set $\partial^2 f(x)$ of $n \times n$ matrices is defined by:

$\partial^2 f(x) = co\{H \in R^{n\times n} : \exists x_k \to x \;\; with \;\; \nabla \, f \; differentiable \; at \; x_k \; and \; \partial^2 \, f(x_k) \to H\}$.

By considering (9), we have

$$\begin{aligned}
\frac{\partial \varphi}{\partial w} &= w + \frac{2}{n}(-A)\left(\left(Y - \left[\left(A^T w + be\right) + \left(A^T c + de\right)\right]\right)_+ \right.\\
&\quad \left. - \left(\left[\left(A^T w + be\right) - \left(A^T c + de\right)\right] - Y\right)_+\right),\\
\frac{\partial \varphi}{\partial b} &= \frac{2}{n}\left(-e^T\right)\left(\left(Y - \left[\left(A^T w + be\right) + \left(A^T c + de\right)\right]\right)_+ \right.\\
&\quad \left. - \left(\left[\left(A^T w + be\right) - \left(A^T c + de\right)\right] - Y\right)_+\right),\\
\frac{\partial \varphi}{\partial c} &= C\nu c + \frac{2}{n}(-A)\left(\left(Y - \left[\left(A^T w + be\right) + \left(A^T c + de\right)\right]\right)_+ \right.\\
&\quad \left. + \left(\left[\left(A^T w + be\right) - \left(A^T c + de\right)\right] - Y\right)_+\right),\\
\frac{\partial \varphi}{\partial d} &= C\nu + \frac{2}{n}(-e^T)\left(\left(Y - \left[\left(A^T w + be\right) + \left(A^T c + de\right)\right]\right)_+ \right.\\
&\quad \left. + \left(\left[\left(A^T w + be\right) - \left(A^T c + de\right)\right] - Y\right)_+\right).
\end{aligned}$$

The formulation $\frac{\partial \varphi}{\partial w}$ can be written

$$\frac{\partial \varphi}{\partial w} = T_1 u - \frac{2}{n} A\left((Y - T_2 u)_+ - (T_3 u - Y)_+\right), \tag{10}$$

where $T_1 = [I_{n\times n} \;\; 0_{n\times n} \;\; 0_{n\times 1} \;\; 0_{n\times 1}]$, $T_2 = [A \;\; A^T \;\; e \;\; e]$, $T_3 = [A \;\; -A^T \;\; e \;\; -e]$ and $u = [w^T \;\; c^T \;\; b^T \;\; d^T]^T$.

Note, we know (10) is not differentiable, but it satisfies Lipschitz conditions.

Theorem 1 $\frac{\partial \varphi}{\partial w}$ *is globally Lipschitz.*

Proof From (10) we have that

$$\begin{aligned}\left\|\frac{\partial \varphi}{\partial w}(s)-\frac{\partial \varphi}{\partial w}(z)\right\| &= \Big\|T_1 s-\frac{2}{n}A\left((Y-T_2 s)_+-(T_3 s-Y)_+\right)-T_1 z \\ &\quad +\frac{2}{n}A\left((Y-T_2 z)_+ +(T_3 z-Y)_+\right)\Big\| \\ &\le \|T_1(s-z)-\frac{2}{n}A\left[(Y-T_2 s)-(T_3 s-Y)\right]+\frac{2}{n}A\left((Y-T_2 z)+(T_3 z-Y)\right)\| \\ &\le \|T_1(s-z)\|+\frac{2}{n}\|A\|\|T_2(s-z)+T_3(s-z)\| \\ &\le \left(\|T_1\|+\frac{2}{n}\|A^T\|\|T_2+T_3\|\right)\|s-z\|. \end{aligned} \tag{11}$$

Then from (11) we conclude that $\frac{\partial \varphi}{\partial w}$ is globally Lipschitz with constant $K=\|T_1\|+\frac{2}{n}\|A\|\|T_2+T_3\|$. □

Similarly, $\frac{\partial \varphi}{\partial b}$, $\frac{\partial \varphi}{\partial c}$ and $\frac{\partial \varphi}{\partial d}$ are globally Lipschitz.

Theorem 2 $\nabla\varphi(u)$ *is globally Lipschitz continuous and the generalized Hessian of* $\varphi(u)$ *is* $\partial^2\varphi(u)=(T_1+AD_1(u)T_2+AD_2(u)T_3)$ *where* $D_1(u)$ *denotes the diagonal matrix whose* i*th diagonal entry* u_i *is equal to* 1 *if* $(Y-T_2u)_i>0$ *;* u_i *is equal to* 0 *if* $(Y-T_2u)_i\le 0$ *and* $D_2(u)$ *also denotes the diagonal matrix whose* i*th diagonal entry* u_i *is equal to* 1 *if* $(T_3u-Y)_i>0$*;* u_i *is equal to* 0 *if* $(T_3u-Y)_i\le 0$.

Proof See (Hiriart-Urruty et al. 1984). □

From the previous discussion and according to the above theorem, we know that the $\nabla\varphi$ is differentiable almost everywhere, and the generalized Hessian of φ exists everywhere.

Therefore, to solve unconstrained problem (9), we can use the generalized Newton method.

3.1 A brief expression for nonlinear par ν-SVRC$^+$

In the nonlinear case, we have the following minimization problem (Hao 2010):

$$\begin{aligned}&\min_{w,b,c,d,\xi,\xi^*} \quad \frac{1}{2}\|w\|^2+C\left(\nu\left(\frac{1}{2}\|c\|^2+d\right)+\frac{1}{n}\left(\|\xi\|^2+\left\|\xi^*\right\|^2\right)\right)\\ &s.t.\left(K(A,D)^T w+be\right)+\left(K(A,D)^T c+de\right)\ge Y-\xi,\\ &-\left(K(A,D)^T c+de\right)+\left(K(A,D)^T w+be\right)\le Y+\xi^*,\\ &\xi^*,\ \xi\ge 0,\end{aligned}$$

where $K(.,.)$ is any arbitrary kernel function and $D=[A;B]$. Similarly, this constrained problem can be considered an unconstrained problem as follows:

$$\begin{aligned}\min_{w,b,c,d}\ \varphi(w,b,c,d) = \min_{w,b,c,d}\ &\frac{1}{2}\|w\|^2+C\nu\left(\frac{1}{2}\|c\|^2+d\right)\\ &+\frac{C}{n}\left\|\left(Y-\left[\left(K(A,D)^T w+be\right)+\left(K(A,D)^T c+de\right)\right]\right)_+\right\|^2\\ &+\frac{C}{n}\left\|\left(\left[\left(K(A,D)^T w+be\right)-\left(K(A,D)^T c+de\right)\right]-Y\right)_+\right\|^2.\end{aligned}$$

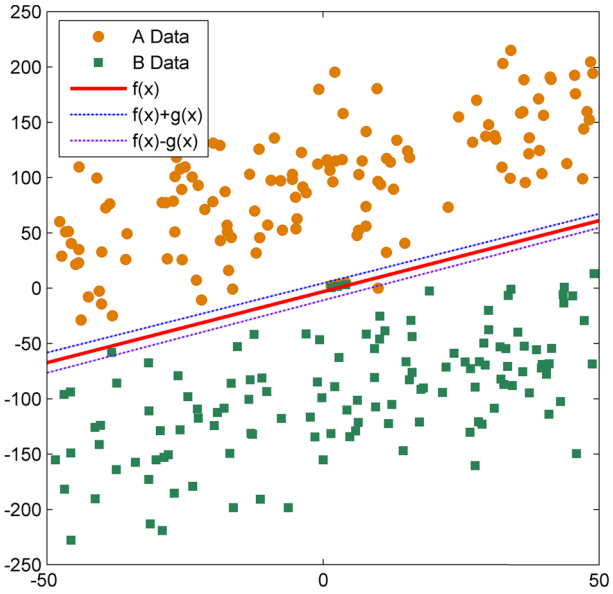

Fig. 2 Classification results of linear Par ν-SVRC$^+$ on generated dataset

4 Numerical experiments

In this section, we discuss our approach using two different performances: the accuracy and the learning speed of a classifier. Throughout this experimental part, we used the Gaussian kernel (i.e. $K(x, y) = exp(-\gamma \|x - y\|^2), \gamma > 0$) for all data. The method was implemented in MATLAB 8 and carried out on a PC with Corei5 2310 (2.9 GHz) and 8 GB main memory. In order to examine the efficiency of Par ν-SVRC$^+$, two samples of n-dimensional problems are given and we derive the separating hyperplanes by means of aforesaid algorithm. In the first problem, we determine randomly some arbitrary points in two classes of A and B which are approximately separated from each other linearly based on given MATLAB code in the "Appendix A" (here, we created 150 points for class A and 100 points for class B). These data are produced randomly within the interval $[-50, 50]$. In Fig. 2, the given separating hyperplane has been shown by means of rendering Par ν-SVRC$^+$ with red color and also the parametric margin hyperplane are indicated by blue and violet color. The accuracy rate of separating in this problem is 99.61%.

It is noted that by means of MATLAB code the problems with large-scale size are produced and the separating hyperplanes are derived by implementing the Par ν-SVRC$^+$.

The average accuracy of separation is approximately 99%.

In another example, Ripleys synthetic standard data set have been adapted (Ripley 1996). These data comprise of 250 data samples out of which 125 data are placed in class of A and the next 125 of them in class B and they are not linearly separated (see Fig. 3). In Fig. 3 the separating hyperplane has been shown by red color. Likewise, the parametric margin hyperplanes, which have been derived by rendering this program, are identified by blue and violet dotted line. The accuracy rate of separating in this problem is 84.80%.

In the following, demonstrate applications to two real data expression profiles for lung cancer and colon tumor. Lung cancer data set was used by Hong and Young to show the power

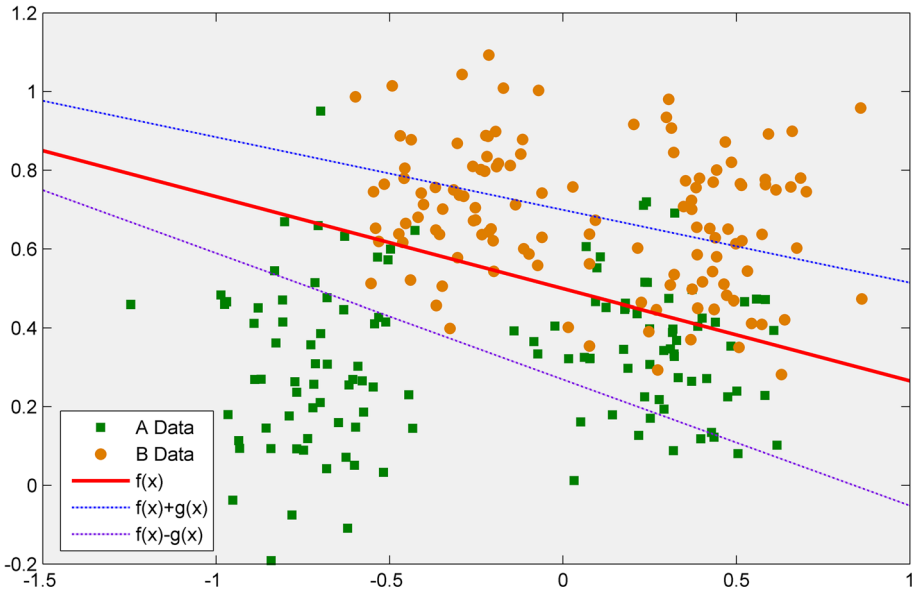

Fig. 3 Classification results of linear Par ν-SVRC$^+$ on Ripleys dataset

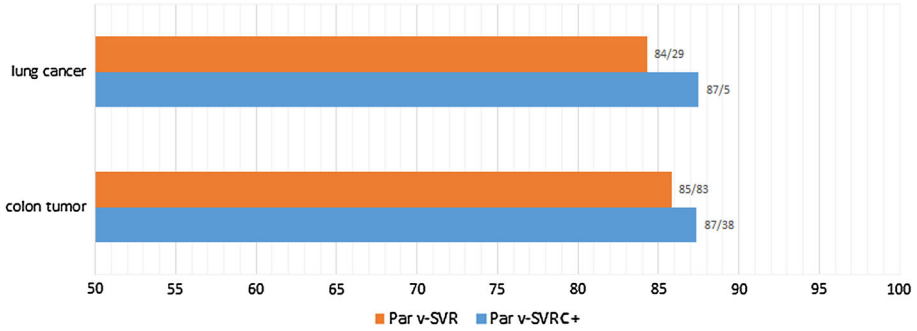

Fig. 4 Performance comparisons of lung cancer and colon tumor data across Par ν-SVR and Par ν-SVRC$^+$ methods

of the optimal discriminant plane even in ill-posed settings. Applying the KNN method in the resulting plane gave 77% accuracy (Hong and Yang 1991). Colon tumor data set contains 62 samples collected from colon-cancer patients. Among them, 40 tumor biopsies are from tumors (labeled as "negative") and 22 normal (labeled as "positive") biopsies are from healthy parts of the colons of the same patients. 2000 out of around 6500 genes were selected (Alon et al. 1999).

When we apply our method on these data sets we gain an accuracy of 87.50% for lung cancer and 87.38% colon tumor while MATLAB Quadprog gain 85.83% and 84.29%, respectively. In Fig. 4 accuracy between the proposed method and the method of quadratic programming in MATLAB can be seen.

To further test the performance of Par ν-SVRC$^+$, we run this algorithm on several UCI benchmark data sets (Lichman 2013). We tested 13 UCI benchmark data sets, which are

Table 1 Comparison of linear SVM, Par ν-SVR and Par ν-SVRC$^+$ on benchmark data sets

Dataset	SVM		Par ν-SVR		Par ν-SVRC$^+$	
size	C	Acc(%) Time (s)	C, ν	Acc (%) Time (s)	C, ν	Acc (%) Time(s)
House-votes	10^2	94.96 ± 5.04	10^{20}, 0.1	78.62 ± 5.54	10^6, 0.1	$\mathbf{95.65 \pm 4.34}$
435×16		1.52		23.14		0.43
Spect	10^9	72.63 ± 17.07	10^{15}, 0.1	53.21 ± 6.03	10^6, 0.1	$\mathbf{73.44 \pm 7.32}$
237×22		1.12		4.51		0.54
Australian	10	85.65 ± 7.68	10^{20}, 0.9	83.90 ± 4.66	10^8, 0.5	$\mathbf{86.36 \pm 7.8}$
690×14		2.18		389.72		0.65
Diabetes	10^3	$\mathbf{77.86 \pm 4.34}$	10^6 , 0.1	60.54 ± 4.39	10^{10}, 0.9	77.73 ± 5.89
768×8		1.93		596.21		1.24
Heart	10^4	84.81 ± 10.37	10^{20} , 0.1	82.96 ± 0.74	10^7, 0.1	$\mathbf{85.18 \pm 11.11}$
270×14		1.07		112.55		0.51
Haberman	10	73.53 ± 4.22	10^7, 0.1	65.21 ± 9.49	10^6, 0.1	$\mathbf{75.48 \pm 4.51}$
306×3		0.81		48.81		0.53
F-Diagnosis	10^3	$\mathbf{88.14 \pm 6.32}$	10^{20}, 0.1	44.18 ± 15.81	10^6, 0.1	$\mathbf{88.14 \pm 1.85}$
100×9		0.49		0.87		0.5
German	10^6	77.00 ± 7	10^{15} , 0.9	$\mathbf{79.47 \pm 4.5}$	10^6, 0.1	76.80 ± 4.20
1000×24		3.69		1352.82		0.81
Ionosphere	10	$\mathbf{88.59 \pm 10.81}$	10^{20} , 0.9	76.03 ± 10.76	10^8,0.1	88.01 ± 7.59
351×34		13.69		37.06		0.54
Bupa	10^3	69.85 ± 11.03	10^{10} , 0.9	71.84 ± 7.14	10^7, 0.2	$\mathbf{71.88 \pm 9.63}$
345×6		0.68		38.49		0.46
Sonar	10^0	$\mathbf{79.98 \pm 16.34}$	10^{20}, 0.1	70.72 ± 11.22	10^9, 0.1	77.01 ± 3.53
208×60		0.95		8.49		0.67
Splice	10^6	80.90 ± 5.14	10^{20} , 0.1	63.50 ± 1.44	10^6, 0.1	$\mathbf{81.5 \pm 4.49}$
1000×60		6.98		2052.50		1.53
Wdbc	10^6	$\mathbf{95.58 \pm 4.42}$	10^{12} , 0.1	91.91 ± 4.36	10^{10}, 0.1	95.06 ± 4.94
569×30		3.34		37.81		1.38

shown in Tables 1 and 2. To accelerate model selection, we tuned a set comprising randomly 20% of the training samples to select optimal parameters.

As we noted in the discussion on Par ν-SVRC$^+$, the generalization errors of the classifier depend on the values of the kernel parameter γ, the regularization parameter C, and parameter ν. Tenfold cross-validation was used to evaluate the performance of the classifier and estimate the accuracy. Tenfold cross-validation followed these steps

- The datasets were divided into ten disjoint subsets of equal size.
- The classifier was trained on all the subsets except one.
- The validation error was computed by testing it on the omitted subset left out.
- This process was repeated for ten trials.

Tables 1 and 2, respectively, give the average accuracies, times, and kernel operations, of this method in the linear and nonlinear case of classification. In Par ν-SVR we solve a

Table 2 Comparison of non-linear SVM, Par ν-SVR and Par ν-SVRC$^+$ on benchmark data sets

Dataset	SVM		Par ν-SVR		Par ν-SVRC$^+$	
	C γ	Acc (%) Time (s)	C, ν γ	Acc (%) Time (s)	C, ν γ	Acc (%) Time (s)
House-votes	10^6	94.91 ± 5.09	10^6, 0.1	76.58 ± 7.50	10^6, 0.1	$\mathbf{97.00 \pm 2.99}$
435×16	1×10^{-2}	19.65	5×10^{-2}	59.11	1×10^{-1}	3.45
Spect	10^0	71.16 ± 9.60	10^{20}, 0.1	58.06 ± 11.16	10^6, 0.1	$\mathbf{75.71 \pm 18.03}$
237×22	1×10^{-1}	4.04	1×10^{-3}	91.13	4×10^{-2}	2.12
Australian	10^{10}	$\mathbf{82.17 \pm 7.82}$	10^{10}, 0.9	67.66 ± 10.90	10^6, 0.1	73.76 ± 1.94
690×14	1×10^{-6}	106.34	1×10^{-4}	115.60	5×10^{-4}	11.09
Diabetes	10^1	75.25 ± 6.83	10^6 , 0.1	69.27 ± 4.75	10^6, 0.1	$\mathbf{76.56 \pm 7.98}$
768×8	1×10^{-4}	72.43	5×10^{-4}	328.23	9×10^{-5}	12.16
Heart	10^6	$\mathbf{81.48 \pm 14.81}$	10^{15}, 0.1	61.11 ± 1.85	10^6, 0.1	71.11 ± 2.14
270×14	1×10^{-5}	7.60	1×10^{-6}	133.89	5×10^{-4}	1.28
Haberman	10^0	73.52 ± 2.55	10^7, 0.1	71.27 ± 6.13	10^6, 0.1	$\mathbf{76.17 \pm 7.80}$
306×3	1×10^{-5}	7.55	5×10^{-6}	55.47	5×10^{-5}	1.74
F-Diagnosis	10^1	88.14 ± 6.32	10^{10}, 0.1	60.31 ± 19.68	10^6, 0.1	$\mathbf{88.14 \pm 1.85}$
100×9	1×10^{-1}	0.85	1×10^{-1}	3.11	9×10^{-2}	0.80
German	10^3	76.00 ± 6.00	10^{20}, 0.9	71.80 ± 3.20	10^6, 0.1	$\mathbf{76.00 \pm 4.00}$
1000×24	1×10^{-4}	147.96	1×10^{-3}	620.94	5×10^{-3}	21.12
Ionosphere	10^2	92.57 ± 4.64	10^{10}, 0.1	89.14 ± 2.51	10^6, 0.1	$\mathbf{96.58 \pm 3.42}$
351×34	1×10^{-3}	10.66	5×10^{-1}	41.97	34×10^{-2}	4.20
Bupa	10^3	73.31 ± 17.42	10^{20}, 0.9	72.20 ± 10.15	10^6, 0.1	$\mathbf{73.36 \pm 15.85}$
345×6	1×10^{-5}	8.23	1×10^{-3}	31.04	3×10^{-4}	1.77
Sonar	10^2	83.64 ± 12.21	10^{13}, 0.8	74.03 ± 7.69	10^6, 0.1	$\mathbf{89.90 \pm 10.10}$
208×60	1×10^{-1}	3.01	1×10^{-1}	53.12	2×10^0	0.98
Splice	10^6	60.10 ± 3.53	10^{20}, 0.9	64.79 ± 5.00	10^6, 0.1	$\mathbf{88.70 \pm 4.98}$
1000×60	1×10^{-1}	82.03	9×10^{-3}	3361.91	2×10^{-2}	26.25
Wdbc	10^2	$\mathbf{94.55 \pm 3.69}$	10^{20}, 0.1	86.28 ± 3.36	10^6, 0.1	94.03 ± 4.21
569×30	1×10^{-5}	35.06	1×10^{-7}	859.85	5×10^{-5}	5.48

constrained convex quadratic programming problem by using dual problem with MATLAB Quadprog optimization toolbox (the bolded values in the Tables 1, 2 represent highest accuracies obtained by corresponding classifiers).

In Table 1, we see that the accuracy of our linear Par ν-SVRC$^+$ is higher than linear Par ν-SVR on various datasets. For example, for House-votes dataset the accuracy of our Par ν-SVRC$^+$ is 95.65% , while the accuracy of SVM is 94.96% and Par ν-SVR is 78.62%. Beside on the another example, for Sonar datasets the accuracy of our Par ν-SVRC$^+$ is 77.01% , while the accuracy of SVM is 79.98% and Par ν-SVR is 70.72%. Although SVM win in the accuracy, so our method is still winner in time. In other datasets, we also obtain similar results. Our Par ν-SVRC$^+$ is much far faster than the original SVM and Par ν-SVR, indicating that the unconstrained optimization technique can improve the calculation speed.

It can also be see that our Par ν-SVRC$^+$ is the fastest on all of datasets. The results in Table 2 are better condition in time and accuracy with that appeared in Table 1, and therefore confirm the above conclusions further.

As mentioned above, we have solved an unconstrained convex minimization problem instead of a constrained convex one. The Experimental results in Tables 1 and 2 demonstrate the high speed, efficiency and accuracy of the proposed method.

5 Concluding remarks

In this paper, we presented a new idea for solving the Par ν-SVR classification problem. By using 2-norm of the slack variables in the objective function of (4) and the KKT conditions associated with this obtained problem, we converted the constrained quadratic minimization problem (4) into an unconstrained convex problem. Since the objective function of Par ν-SVRC$^+$ is an LC^1 function, the generalized Newton method was proposed for solving it. In this way, we have derived much faster and accurately method than Par ν-SVR which solves a constrained quadratic problem. The experimental results on several UCI benchmark data sets have shown that this method has high efficiency and accuracy both in the linear and nonlinear case.

A Matlab code

```
% Generate random M,N;
%Input: m1,m2 n; Output:M N
pl=inline('(abs(x)+x)/2');
M=rand(m1,n); M=100*(M-0.5*spones(M));
M(:,2)=M(:,1)+1*ones(m1,1)+100*rand(m1,1)+100*rand(m1,1);
N=rand(m2,n); N=100*(N-0.5*spones(N));
N(:,2)=N(:,1)-1*ones(m2,1)-100*rand(m2,1)-100*rand(m2,1);
uu=5*rand(3,n); uu1=uu;uu1(:,2)= uu1(:,1)+1*ones(3,1);
uu2=uu;uu2(:,2)= uu2(:,1)-1*ones(3,1);
M=[M;uu1;10 0]; N=[N;uu2;30 -20];m1=m1+4;m2=m2+4;m=m1+m2;
xM=[-50:40*rand: 50];yM=xM+1;xN=[-50:20*rand:50];yN=xN-1;
plot(M(:,1),M(:,2),'oblack',N(:,1),N(:,2),'*bl');
axis square
format short ;[m1 m2 n toc],[max(M(:,1)) min( N(:,1))]
```

References

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., et al. (1999). Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, *96*(12), 6745–6750.

Bennett, K. P., & Bredensteiner, E. J. (2000). Duality and geometry in SVM classifiers. In *Proceedings of the seventeenth international conference on machine learning* (pp. 57–64). San Francisco.

Boser, B. E., Guyon, I. M., & Vapnik. V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on computational learning theory (COLT '92)* (pp. 144–152). ACM, New York, NY, USA.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. New York: Cambridge University Press.

Cao, L., & Tay, E. F. (2001). Financial forecasting using support vector machines. *Neural Computing and Applications*, *10*(2), 184–192.
Chen, X., Yang, J., & Liang, J. (2012a). A flexible support vector machine for regression. *Neural Computing and Applications*, *21*(8), 2005–2013.
Chen, X., Yang, J., Liang, J., & Ye, Q. (2012b). Smooth twin support vector regression. *Neural Computing and Applications*, *21*(3), 505–513.
Clarke, F. (1990). *Optimization and nonsmooth analysis*. Philadelphia: Society for Industrial and Applied Mathematics.
Deng, N., Tian, Y., & Zhang, C. (2012). *Support vector machines: Optimization based theory, algorithms, and extensions* (1st ed.). Boca Raton: Chapman and Hall/CRC.
Hao, P. Y. (2010). New support vector algorithms with parametric insensitive/margin model. *Neural Networks*, *23*(1), 60–73.
Hiriart-Urruty, J.-B., Strodiot, J.-J., & Nguyen, V. H. (1984). Generalized hessian matrix and second-order optimality conditions for problems with $C^{1,1}$ data. *Applied Mathematics and Optimization*, *11*(1), 43–56.
Hong, Z.-Q., & Yang, J.-Y. (1991). Optimal discriminant plane for a small number of samples and design method of classifier on the plane. *Pattern Recognition*, *24*(4), 317–324.
Ivanciuc, O. (2007). *Reviews in computational chemistry*. London: Wiley.
Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European conference on machine learning, ECML'98* (pp. 137–142). Springer, London, UK.
Ketabchi, S., & Moosaei, H. (2012). Minimum norm solution to the absolute value equation in the convex case. *Journal of Optimization Theory and Applications*, *154*(3), 1080–1087.
Lee, Y.-J., & Mangasarian, O. (2001). SSVM: A smooth support vector machine for classification. *Computational Optimization and Applications*, *20*(1), 5–22.
Lichman, M. (2013). *UCI machine learning repository*. http://archive.ics.uci.edu/ml.
Osuna, E., Freund, R., & Girosit, F. (1997). Training support vector machines: An application to face detection. In *Proceedings of the 1997 IEEE computer society conference on computer vision and pattern recognition* (pp. 130–136).
Pappu, V., Panagopoulos, O. P., Xanthopoulos, P., & Pardalos, P. M. (2015). Sparse proximal support vector machines for feature selection in high dimensional datasets. *Expert Systems with Applications*, *42*(23), 9183–9191.
Pardalos, P. M., Ketabchi, S., & Moosaei, H. (2014). Minimum norm solution to the positive semidefinite linear complementarity problem. *Optimization*, *63*(3), 359–369.
Pontil, M., Rifkin, R., & Evgeniou, T. (1998). *From regression to classification in support vector machines*. Technical Report. Massachusetts Institute of Technology, Cambridge, MA, USA.
Resende, M. G. C., & Pardalos, P. M. (2002). *Handbook of applied optimization*. Oxford: Oxford University Press.
Ripley, B. (1996). *Pattern recognition and neural networks datasets collection*. www.stats.ox.ac.uk/pub/PRNN/.
Schölkopf, B., & Smola, A. J. (2001). *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. Cambridge: MIT Press.
Schölkopf, B., Smola, A. J., Williamson, R. C., & Bartlett, P. L. (2000). New support vector algorithms. *Neural Computation*, *12*(5), 1207–1245.
Vapnik, V. (1998). *Statistical learning theory*. New York: Wiley.
Vapnik, V., & Chervonenkis, A. (1974). *Theory of pattern recognition*. Moscow: Nauka. (in Russian).
Wang, Z., Shao, Y., & Wu, T. (2014). Proximal parametric-margin support vector classifier and its applications. *Neural Computing and Applications*, *24*(3–4), 755–764.
Xanthopoulos, P., Guarracino, M. R., & Pardalos, P. M. (2014). Robust generalized eigenvalue classifier with ellipsoidal uncertainty. *Annals of Operations Research*, *216*(1), 327–342.