Soft Computing (2020) 24:6101–6117
https://doi.org/10.1007/s00500-020-04746-6

FOUNDATIONS

# Fast clustering-based weighted twin support vector regression

Binjie Gu[1] · Jianwen Fang[1] · Feng Pan[1] · Zhonghu Bai[2]

Published online: 27 February 2020

**Abstract**
Construction of an effective model for regression to fit data samples with noise or outlier is a challenging work. In this paper, in order to reduce the influence of noise or outlier on regression and further improve the prediction performance of standard twin support vector regression (TSVR), we proposed a fast clustering-based weighted TSVR, termed as FC-WTSVR. First, we use a fast clustering algorithm to quickly classify samples into different categories based on their similarities. Secondly, to reflect the prior structural information and distinguish contributions of samples located at different positions to regression, we introduce the covariance matrix and weighted diagonal matrix into the primal problems of FC-WTSVR, respectively. Finally, to shorten the training time, we adopt the successive over-relaxation algorithm to solve the quadratic programming problems. The results show that the proposed FC-WTSVR can obtain better prediction performance and anti-interference capability than some state-of-the-art algorithms.

**Keywords** Machine learning · Twin support vector regression · Fast clustering · Prior structural information · Weighted strategy

## 1 Introduction

Support vector machine (SVM) is a powerful machine learning method based on the theory of Vapnik–Chervonenkis (VC) dimension and statistical learning (Vapnik 1999; Cristianini and Shawe-Talyor 2000; Mangasarian and Musicant 2001; Deng and Tian 2009). SVM has been successfully applied in areas such as feature selection, data mining, image processing, and intrusion detection. In nature, SVM is ascribed to solve quadratic programming problems (QPPs) for obtaining the optimal solution of its dual problem. However, when training the large-scale datasets, the time cost by solving QPPs will increase rapidly. Therefore, many fast training algorithms, such as sequential minimal optimization (SMO) (Platt 2000), geometric approach (Mavroforakis and Theodoridis 2006), coordinate descent method (Chang et al. 2008), and clipping algorithm (López et al. 2011), have been specifically designed for shortening the training time of large-scale datasets.

Recently, Jayadeva et al. (2007) proposed a twin support vector machine (TSVM). Unlike standard SVM, TSVM attempts to find two non-parallel hyperplanes so that each hyperplane is closest to one class and far away from the other classes. The main difference between SVM and TSVM is that SVM needs to solve one larger-size QPPs, whereas TSVM only needs to solve two smaller-size QPPs. Therefore, TSVM performs approximately four times faster than SVM. Then, Peng (2010) extended the idea of TSVM to regression, and he proposed a twin support vector regression (TSVR). TSVR generates a pair of QPPs such that each QPP determines one of the up-bound and down-bound functions by using only one group of constraints. In comparison to standard support vector regression (SVR), TSVR has better fitting regression performance and costs lower training time. Therefore, TSVR has become a new hot topic in machine learning field. To date, many variants of TSVR have been developed, such as smooth TSVR (Chen et al. 2012), weighted TSVR (Xu and Wang 2012), ε-TSVR (Shao et al. 2013), Lagrangian TSVR (Balasundaram and Gupta 2014), K-nearest neighbor-based weighted TSVR (KNN-WTSVR) (Xu and Wang 2014), weighted

Communicated by A. Di Nola.

Binjie Gu
gubinjie1980@126.com

[1] Key Laboratory of Advanced Process Control for Light Industry, Ministry of Education, Jiangnan University, Wuxi 214122, China

[2] National Engineering Laboratory for Cereal Fermentation Technology, Jiangnan University, Wuxi 214122, China

Lagrange $\varepsilon$-TSVR (WL-$\varepsilon$-TSVR) (Ye et al. 2016), robust TSVR (López and Maldonado 2018), projection TSVR (Gu et al. 2019), and projection weighted TSVR (Wang et al. 2019).

It has been proved that the prior structural information of samples may contain some useful knowledge for training a classification or regression model. Therefore, how to skillfully apply the prior structural information of samples to construct an effective model has become a hot research topic. Yeung et al. (2007) first proposed a structured large margin machine (SLMM), and they integrate the prior structural information of each cluster into the primal problem of SVM in the form of covariance matrix. The experimental results show that the prior structural information of samples is one of the key factors which determines the classification accuracy. Then, under the concept of structural granularity, Peng et al. (2013) developed a structural twin parametric-margin support vector machine (STPMSVM). The STPMSWM classifier is proved to be superior to some other learning algorithms in terms of learning speed and generalization capacity. Qi et al. (2013) point out the drawbacks of the existing SLMM algorithms, and they designed a new structural twin support vector machine, termed as STSVM. Theoretical analysis and experimental results show that STSVM is rigidly better than other prior structural information-based algorithms in both classification accuracy and computation time. Pan et al. (2015) proposed a novel K-nearest neighbor-based structural TSVM, called KNN-STSVM. In KNN-STSVM, different weights are assigned to the samples in one class to strengthen the structural information by applying the intra-class KNN approach, while for the other classes, in order to shorten the training time, the inter-class KNN approach is adopted to delete the redundant constraints. Extensive experimental results show the efficiency of the proposed KNN-STSVM. Parastalooi et al. (2016) presented a modified TSVR (MTSVR) for regression. They add a new term in the primal problems of TSVR to reflect the prior structural information of the input data samples. Moreover, they utilize successive over-relaxation (SOR) algorithm and particle swam optimization (PSO) algorithm to accelerate the training process and determine the parameters of the proposed MTSVR model, respectively. The results on several artificial and real datasets show that the prediction performance and generalization capability of the proposed MTSVR model are greatly improved. In summary, in order to improve the performance of the classification or regression model, it is necessary to incorporate the prior structural information of samples into the model.

During the fitting process, standard TSVR adjusts the trends of the fitting regression curve according to the prediction error. This means that standard TSVR assumes that samples located at different positions have the same impact on the fitting regression curve. In most real scenarios, due to factors such as environmental change and erroneous measurement, the samples collected may contain noise. Therefore, the fitting regression curve obtained by standard TSVR may seriously deviate from the actual one, which will cause large prediction errors. Nowadays, researchers have developed two categories of methods, i.e., weighted method and loss function method, to reduce the influence of noise on regression. The main idea of the weighted method is to assign different weights to samples located at different positions based on their contributions to the fitting regression curve. The most commonly used weighted methods are distance-based method (Bruno et al. 2000), clustering-based method (Jiang et al. 2006), depth-based method (Kalidas and Chandra 2008), and K-nearest neighbor-based method (Pang et al. 2018; Pang and Xu 2019). But these weighted methods are sensitive to the dimension of samples. Density-based method (Cheng and Wang 2016) and rough set theory-based method (Xue et al. 2018) provide feasible solutions to address this difficulty. The main drawback of the density-based method is that it is sensitive to parameters that define neighbors, while the rough set theory-based method is limited to be applied in discrete situations. In addition, the selection of loss functions also plays an important role in reducing the impact of noise existed in data samples (Ye et al. 2013; Peng et al. 2016; Niu et al. 2017; Xu et al. 2017; Anagha et al. 2018; Tanveer et al. 2019a, b). The most commonly used loss functions for regression are 1-norm loss function, 2-norm loss function, $\varepsilon$-insensitive loss function, Huber loss function, and squared pinball loss function (Chen et al. 2019, Gupta and Gupta 2019; Hua et al. 2019; Tanveer et al. 2019a, b). Among these loss functions, 2-norm loss function is attractive because it is smooth. Unfortunately, because 2-norm loss function is sensitive to large error, it is not robust. The reason is that 2-norm loss function will sacrifice the errors of other samples and update toward the direction of reducing the error of noise. Compared with 2-norm loss function, 1-norm and $\varepsilon$-insensitive loss functions are more robust because they can both reduce the influence of noise. Unfortunately, they are not smooth, which restrains the application of the numerical minimization approaches. Moreover, although the Huber loss function and squared pinball loss function are both robust, in order to achieve the best fitting performance, they both need to constantly tune the hyper-parameters.

To conclude, in order to construct an effective model for regression, the prior structural information of samples should be considered, and the samples located at different positions should be assigned different weights. In addition, the samples collected may contain several outliers. Fortunately, we can adopt the fast clustering algorithm (Rodriguez and Laio 2014; Liu et al. 2018) to effectively separate outliers from normal samples. Hence, in this paper, to reduce the influence of noise and potential outliers on regression and further improve the prediction performance of standard TSVR, we investigated a fast clustering-based weighted twin support vector regression, termed as FC-WTSVR.

The main contributions of this paper are summarized as follows:

1. We utilize the fast clustering algorithm developed by Rodriguez and Laio to determine the cluster centers and outliers according to suitable principles.
2. In order to reflect the prior structural information of samples, we introduce the covariance matrix into the primal problems of FC-WTSVR.
3. To further reduce the influence of noise on regression, we design a new feedback weighted strategy and assign samples located at different positions with different penalties.
4. To accelerate the training process, the successive over-relaxation (SOR) algorithm is employed to solve the QPPs in the dual problems of FC-WTSVR.

In addition, we conduct extensive experiments on benchmark datasets, artificial datasets, and actual glutamic acid fed-batch fermentation process to demonstrate the superiorities of the proposed FC-WTSVR over some state-of-the-art algorithms in terms of prediction performance and anti-interference capability.

The remainder of this paper is arranged as follows. In Sect. 2, we briefly review the twin support vector regression in linear and nonlinear case. Section 3 introduces the proposed fast clustering-based weighted twin support vector regression in detail. Section 4 presents the experimental results and analyses. The conclusions are summarized in Sect. 5.

## 2 Twin support vector regression

A training sample set $T = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ is given, where $\boldsymbol{x}_i \in \mathbb{R}^d$ ($d$ is the number of attributions) and $y_i \in \mathbb{R}$, $i = 1, \ldots, n$ ($n$ is the number of samples) represent the input and output, respectively. Figure 1 illustrates twin support vector regression (TSVR) in linear case.

We can see from Fig. 1 that the aim of TSVR in linear case is to generate a pair of ε-insensitive up-bound function $f_1(\boldsymbol{x}) = \boldsymbol{w}_1^{\mathrm{T}}\boldsymbol{x} + b_1$ and down-bound function $f_2(\boldsymbol{x}) = \boldsymbol{w}_2^{\mathrm{T}}\boldsymbol{x} + b_2$. The regression function of TSVR in linear case is determined by the mean of the ε-insensitive up- and down-bound functions as follows (Peng 2010):

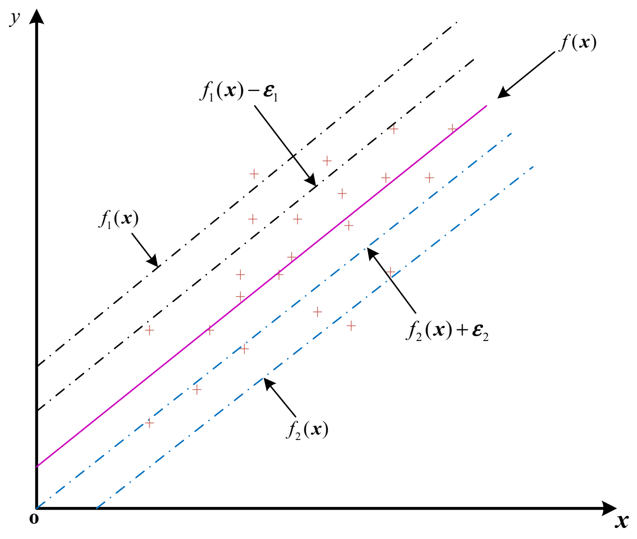


**Fig. 1** An illustration of TSVR in linear case

$$f(\boldsymbol{x}) = \frac{1}{2}(f_1(\boldsymbol{x}) + f_2(\boldsymbol{x})) = \frac{1}{2}(\boldsymbol{w}_1 + \boldsymbol{w}_2)^{\mathrm{T}}\boldsymbol{x} + \frac{1}{2}(b_1 + b_2) \tag{1}$$

where $\boldsymbol{w}_1$, $\boldsymbol{w}_2 \in \mathbb{R}^d$ are weight vectors, $b_1, b_2$ are biases.

Let $\boldsymbol{A} = [\boldsymbol{x}_1; \ldots; \boldsymbol{x}_n] \in \mathbb{R}^{n\times d}$ and $\boldsymbol{Y} = [y_1, \ldots, y_n]^{\mathrm{T}} \in \mathbb{R}^n$, the primal problems of TSVR in linear case, are expressed as follows:

$$\begin{aligned} &\min_{\boldsymbol{w}_1, b_1, \boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{e}\varepsilon_1 - (\boldsymbol{A}\boldsymbol{w}_1 + \boldsymbol{e}b_1)\|^2 + c_1\boldsymbol{e}^{\mathrm{T}}\boldsymbol{\xi} \\ &\text{s.t. } \boldsymbol{Y} - (\boldsymbol{A}\boldsymbol{w}_1 + \boldsymbol{e}b_1) \geq \boldsymbol{e}\varepsilon_1 - \boldsymbol{\xi}, \boldsymbol{\xi} \geq \boldsymbol{0} \end{aligned} \tag{2}$$

$$\begin{aligned} &\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}} \frac{1}{2}\|\boldsymbol{Y} + \boldsymbol{e}\varepsilon_2 - (\boldsymbol{A}\boldsymbol{w}_2 + \boldsymbol{e}b_2)\|^2 + c_2\boldsymbol{e}^{\mathrm{T}}\boldsymbol{\eta} \\ &\text{s.t. } (\boldsymbol{A}\boldsymbol{w}_2 + \boldsymbol{e}b_2) - \boldsymbol{Y} \geq \boldsymbol{e}\varepsilon_2 - \boldsymbol{\eta}, \boldsymbol{\eta} \geq \boldsymbol{0} \end{aligned} \tag{3}$$

where $c_1, c_2 > 0$ are penalty parameters, $\varepsilon_1, \varepsilon_2 > 0$ are insensitive loss parameters, $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ are slack vectors, $\|\cdot\|$ denotes the 2-norm, $\boldsymbol{e}$ and $\boldsymbol{0}$ represent all ones and all zeros column vectors with proper dimensions, respectively.

By introducing nonnegative Lagrangian multiplier vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, we can obtain the following dual problems of Eqs. (2) and (3):

$$\begin{aligned} &\max_{\boldsymbol{\alpha}} -\frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\boldsymbol{G}(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G})^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\alpha} + \boldsymbol{f}^{\mathrm{T}}\boldsymbol{G}(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G})^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\alpha} - \boldsymbol{f}^{\mathrm{T}}\boldsymbol{\alpha} \\ &\text{s.t. } \boldsymbol{0} \leq \boldsymbol{\alpha} \leq c_1\boldsymbol{e} \end{aligned} \tag{4}$$

$$\begin{aligned} &\max_{\boldsymbol{\beta}} -\frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{G}(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G})^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\beta} - \boldsymbol{h}^{\mathrm{T}}\boldsymbol{G}(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G})^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\beta} + \boldsymbol{h}^{\mathrm{T}}\boldsymbol{\beta} \\ &\text{s.t. } \boldsymbol{0} \leq \boldsymbol{\beta} \leq c_2\boldsymbol{e} \end{aligned} \tag{5}$$

where $\boldsymbol{G} = [\boldsymbol{A} \quad \boldsymbol{e}]$, $\boldsymbol{f} = \boldsymbol{Y} - \boldsymbol{e}\varepsilon_1$, and $\boldsymbol{h} = \boldsymbol{Y} + \boldsymbol{e}\varepsilon_2$.

Solving Eqs. (4) and (5), we can obtain the optimal solutions $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$, then the augmented vectors of TSVR in linear case can be computed as follows:

$$\begin{bmatrix} \boldsymbol{w}_1^{\mathrm{T}} & b_1 \end{bmatrix}^{\mathrm{T}} = \left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}(\boldsymbol{f} - \boldsymbol{\alpha}^*) \tag{6}$$

$$\begin{bmatrix} \boldsymbol{w}_2^{\mathrm{T}} & b_2 \end{bmatrix}^{\mathrm{T}} = \left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}(\boldsymbol{h} + \boldsymbol{\beta}^*). \tag{7}$$

By introducing the kernel function $\boldsymbol{K}(\cdot,\cdot)$, we can easily extend TSVR in linear case to nonlinear case. The most commonly used kernel functions are polynomial kernel, radial basis function (RBF) kernel, and sigmoid kernel (Peng et al. 2014; Shao et al. 2014; Chen et al. 2014).

The primal problems of TSVR in nonlinear case are expressed as follows:

$$\begin{aligned} &\min_{\boldsymbol{w}_1,b_1,\boldsymbol{\xi}} \frac{1}{2}\left\|\boldsymbol{Y} - \boldsymbol{e}\varepsilon_1 - \left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{w}_1 + \boldsymbol{e}b_1\right)\right\|^2 + c_1\boldsymbol{e}^{\mathrm{T}}\boldsymbol{\xi} \\ &\text{s.t. } \boldsymbol{Y} - \left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{w}_1 + \boldsymbol{e}b_1\right) \geq \boldsymbol{e}\varepsilon_1 - \boldsymbol{\xi}, \boldsymbol{\xi} \geq \boldsymbol{0} \end{aligned} \tag{8}$$

$$\begin{aligned} &\min_{\boldsymbol{w}_2,b_2,\boldsymbol{\eta}} \frac{1}{2}\left\|\boldsymbol{Y} + \boldsymbol{e}\varepsilon_2 - \left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{w}_2 + \boldsymbol{e}b_2\right)\right\|^2 + c_2\boldsymbol{e}^{\mathrm{T}}\boldsymbol{\eta} \\ &\text{s.t. } \left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{w}_2 + \boldsymbol{e}b_2\right) - \boldsymbol{Y} \geq \boldsymbol{e}\varepsilon_2 - \boldsymbol{\eta}, \boldsymbol{\eta} \geq \boldsymbol{0}. \end{aligned} \tag{9}$$

Similarly, by introducing nonnegative Lagrangian multiplier vectors $\boldsymbol{\mu}$ and $\boldsymbol{v}$, we can obtain the following dual problems of Eqs. (8) and (9):

$$\begin{aligned} &\max_{\boldsymbol{\mu}} -\frac{1}{2}\boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{Q}\left(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\right)^{-1}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{\mu} + \boldsymbol{f}^{\mathrm{T}}\boldsymbol{Q}\left(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\right)^{-1}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{\mu} - \boldsymbol{f}^{\mathrm{T}}\boldsymbol{\mu} \\ &\text{s.t } \boldsymbol{0} \leq \boldsymbol{\mu} \leq c_1\boldsymbol{e} \end{aligned} \tag{10}$$

$$\begin{aligned} &\max_{\boldsymbol{v}} -\frac{1}{2}\boldsymbol{v}^{\mathrm{T}}\boldsymbol{Q}\left(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\right)^{-1}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{v} - \boldsymbol{h}^{\mathrm{T}}\boldsymbol{Q}\left(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\right)^{-1}\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{v} + \boldsymbol{h}^{\mathrm{T}}\boldsymbol{v} \\ &\text{s.t. } \boldsymbol{0} \leq \boldsymbol{v} \leq c_2\boldsymbol{e} \end{aligned} \tag{11}$$

where $\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right) & \boldsymbol{e} \end{bmatrix}, \boldsymbol{f} = \boldsymbol{Y} - \boldsymbol{e}\varepsilon_1$, and $\boldsymbol{h} = \boldsymbol{Y} + \boldsymbol{e}\varepsilon_2$.

Once the optimal solutions $\boldsymbol{\mu}^*$ and $\boldsymbol{v}^*$ are obtained, we can compute the augmented vectors of TSVR in nonlinear case as follows:

$$\begin{bmatrix} \boldsymbol{w}_1^{\mathrm{T}} & b_1 \end{bmatrix}^{\mathrm{T}} = \left(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\right)^{-1}\boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{f} - \boldsymbol{\mu}^*) \tag{12}$$

$$\begin{bmatrix} \boldsymbol{w}_2^{\mathrm{T}} & b_2 \end{bmatrix}^{\mathrm{T}} = \left(\boldsymbol{Q}^{\mathrm{T}}\boldsymbol{Q}\right)^{-1}\boldsymbol{Q}^{\mathrm{T}}(\boldsymbol{h} + \boldsymbol{v}^*). \tag{13}$$

Finally, we can obtain the following regression function of TSVR in nonlinear case:

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{K}\left(\boldsymbol{x}^{\mathrm{T}},\boldsymbol{A}^{\mathrm{T}}\right)(\boldsymbol{w}_1 + \boldsymbol{w}_2) + \frac{1}{2}(b_1 + b_2). \tag{14}$$

## 3 Proposed fast clustering-based weighted twin support vector regression

In this section, we first discuss the fast clustering algorithm; then, we present the fast clustering-based weighted twin support vector regression (FC-WTSVR) in detail. Next, we utilize the successive over-relaxation (SOR) algorithm to solve the four QPPs in the dual problems of FC-WTSVR and summarize the training procedure of FC-WTSVR. Finally, we provide analysis of FC-WTSVR.

### 3.1 Fast clustering

The aim of clustering is to classify objects into different categories according to their similarities. Recently, Rodriguez and Laio proposed a novel fast clustering algorithm based on the idea that cluster centers are characterized by a higher density than their neighbors and by a relatively large distance from points with higher densities (Rodriguez and Laio 2014). The proposed algorithm is simple and efficient because it can quickly find the higher-density peak points, i.e., cluster centers, without iteratively calculating the objective function.

The data points set $S = \{\boldsymbol{x}_i\}_{i=1}^n, I_S = \{1,\ldots,n\}$ is given, where $I_S$ is the index set and $n$ is the number of data points. For each data point $\boldsymbol{x}_i$, we can compute its local density $\rho_i$ and its distance $\delta_i$ from the nearest points of higher density.

The local density $\rho_i$ is defined as follows:

$$\rho_i = \sum_{j \in I_S \setminus \{i\}} \exp\left(-\left(\frac{d_{ij}}{d_c}\right)^2\right) \tag{15}$$

where $d_{ij}$ is the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, and $d_c$ is a cutoff distance which is equivalent to the neighborhood radius of data points.

We can see from Eq. (15) that the local density $\rho_i$ is equivalent to the number of data points in $d_c$-neighborhood, which depends on the setting of the cutoff distance $d_c$. In general, $d_c$ is typically set as at most 1–2% of all $d_{ij}$. In our work, we utilize the K-nearest neighbor method to determine $d_c$ (Pang et al. 2018).

Define $Q = \{\boldsymbol{x}_i^u\}_{u=1}^k$ as the set of the $k^{th}$ neighbor of data point $\boldsymbol{x}_i$, let $k = n \times 1\%$, the cutoff distance $d_c$ is computed as follows:

$$d_c = \frac{1}{2}\left(\min\left(\mathrm{dist}\left(\boldsymbol{x}_i,\boldsymbol{x}_i^u\right)\right) + \max\left(\mathrm{dist}\left(\boldsymbol{x}_i,\boldsymbol{x}_i^u\right)\right)\right) \tag{16}$$

where $\mathrm{dist}\left(\boldsymbol{x}_i,\boldsymbol{x}_i^u\right)$ is the Euclidean distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_i^u$.

The distance $\delta_i$ is computed as follows:

$$\delta_i = \begin{cases} \max\limits_{j \geq 2}\{d_{ij}\}, & i = 1 \\ \min\limits_{j < i}\{d_j\}, & i = 2, \ldots, n \end{cases} \tag{17}$$

where $d_j$ represents the $j^{th}$ column of $d_{ij}$.

Then, we normalize $\rho_i$ and $\delta_i$ as follows:

$$\bar{\rho}_i = \frac{\rho_i - \min(\rho_i)}{\max(\rho_i) - \min(\rho_i)} \tag{18}$$

$$\bar{\delta}_i = \frac{\delta_i - \min(\delta_i)}{\max(\delta_i) - \min(\delta_i)}. \tag{19}$$

Subsequently, $\bar{\rho}_i$ and $\bar{\delta}_i$ are sorted in ascending order for all data points.

Finally, the decision value $\bar{\gamma}_i$ is computed as follows:

$$\bar{\gamma}_i = \bar{\rho}_i \times \bar{\delta}_i. \tag{20}$$

Equation (20) implies that the cluster centers are determined by the product of $\bar{\rho}_i$ and $\bar{\delta}_i$.

Figure 2 depicts the fast clustering algorithm in two dimensions (Rodriguez and Laio 2014). The data points in Fig. 2a are ranked in deceasing density order, and the decision graph for the data points in Fig. 2a is plotted in Fig. 2b. Figure 2a indicates that the data points 3, 20, and 21 are the cluster centers, whereas the data points 30, 31, and 32 are outliers. In addition, we can see from Fig. 2b that the cluster centers have large $\bar{\rho}_i$ and $\bar{\delta}_i$, whereas the outliers have small $\bar{\rho}_i$ and large $\bar{\delta}_i$.

Next, we will explain the principle of determining the cluster centers and outliers (see Fig. 3).

Because the cluster centers have large $\bar{\rho}_i$ and $\bar{\delta}_i$, a relatively great change of decision value $\bar{\gamma}_i$ from cluster centers to non-cluster centers must be occurred. Figure 3a shows the determination of cluster centers. To facilitate counting the number of cluster centers, Fig. 3a also provides the zoom in area illustration. We can see from Fig. 3a that the decision values above the red line are significantly greater than the decision values under the red line. Therefore, by detecting this change, we can easily determine the cluster centers. It is clear that there are four cluster centers in Fig. 3a.

Figure 3b shows an example of determining the outliers on decision graph. Considering that the outliers have small $\bar{\rho}_i$ and large $\bar{\delta}_i$, we adopt the following principle to determine the outliers: For each data point, if its $\bar{\rho}_i$ is less than $\bar{\rho}_{\text{outlier}}$ and its $\bar{\delta}_i$ is greater than $\bar{\delta}_{\text{outlier}}$, where $\bar{\rho}_{\text{outlier}}$ and $\bar{\delta}_{\text{outlier}}$ are defined in Eqs. (21) and (22), respectively, then the data point is classified as an outlier.

$$\bar{\rho}_{\text{outlier}} = \bar{\rho}_{\text{round}(n\times\tau)} \tag{21}$$

$$\bar{\delta}_{\text{outlier}} = \bar{\delta}_{\text{round}(n\times(1-\tau))} \tag{22}$$

where $\tau$ is a small constant.

To conclude, the fast clustering algorithm is summarized in Algorithm 1.

**Algorithm 1** The fast clustering algorithm.

```
Input: The training data points S
Output: The clustered data points C
Step1. Compute $d_c$, $\rho_i$, and $\delta_i$ according to Eqs. (16), (15) and (17), respectively;
Step2. Compute $\bar{\rho}_i$, $\bar{\delta}_i$, and $\bar{\gamma}_i$ according to Eqs. (18), (19) and (20), respectively;
Step3. Construct the decision graph;
Step4. Determine the cluster centers and outliers according to the principles described in Sect. 3.1;
Step5. Each remaining data point is assigned to its corresponding cluster center based on the principle that its nearest neighbor is of higher density.
```

## 3.2 Fast clustering-based weighted twin support vector regression

In this part, we first clarify the function of the prior structural information of samples. Then, in order to give different penalties to samples located at different positions, we introduce a new feedback weighted strategy. Finally,

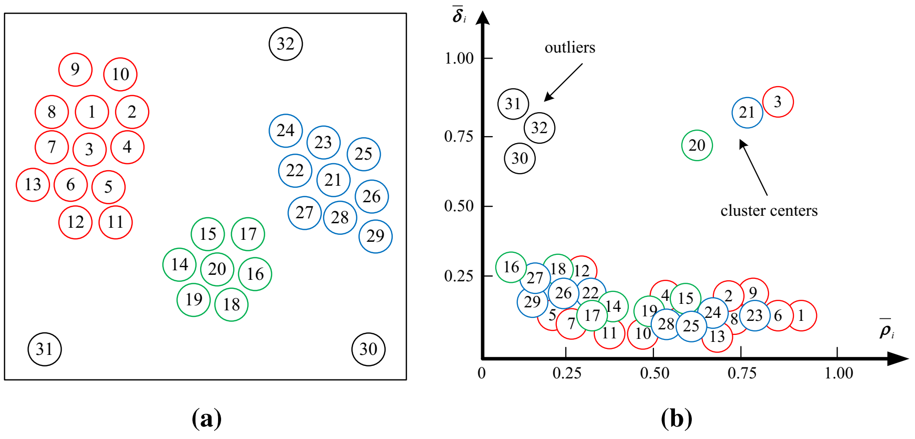


**Fig. 2** The fast clustering algorithm in two dimensions. **a** Point distribution; **b** decision graph

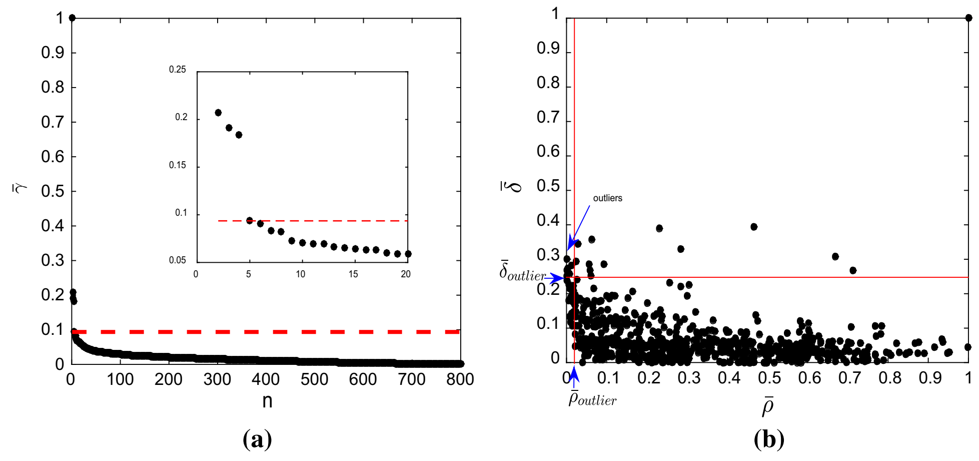


**Fig. 3** The principle of determining different categories of data points. **a** The cluster centers; **b** the outliers ($\tau = 0.1$)

we present our fast clustering-based weighted twin support vector regression (FC-WTSVR) in linear and nonlinear case.

### 3.2.1 The prior structural information of samples

In order to further improve the prediction performance of standard TSVR, motivated by Parastalooi et al. (2016), we take the prior structural information of samples into account.

In linear case, the covariance matrix $\sum$, which reflects the prior structural information of samples, can be sequentially extracted according to the following Eqs. (23) to (25):

$$\boldsymbol{v}_{\boldsymbol{u}_i} = \frac{1}{|\boldsymbol{u}_i|}\sum_{\boldsymbol{x}_j \in \boldsymbol{u}_i} \boldsymbol{x}_j, i = 1, \ldots, r \tag{23}$$

$$\Sigma_{\boldsymbol{u}_i} = \frac{1}{|\boldsymbol{u}_i|}\sum_{\boldsymbol{x}_j \in \boldsymbol{u}_i} \left\|\boldsymbol{x}_j - \boldsymbol{v}_{\boldsymbol{u}_i}\right\|^2, \begin{matrix} i = 1, \ldots, r \\ j = 1, \ldots, |\boldsymbol{u}_i| \end{matrix} \tag{24}$$

$$\Sigma = \Sigma_{\boldsymbol{u}_1} + \Sigma_{\boldsymbol{u}_2} + \cdots + \Sigma_{\boldsymbol{u}_r} \tag{25}$$

where $r$ stands for the number of clusters, $\boldsymbol{u}_i$ represents the $i$th cluster, $|\boldsymbol{u}_i|$ is the number of samples in cluster $\boldsymbol{u}_i$, and $\boldsymbol{v}_{\boldsymbol{u}_i}$ is the mean of cluster $\boldsymbol{u}_i$.

In nonlinear case, by introducing the kernel function $\boldsymbol{K}(\cdot,\cdot)$, the covariance matrix $\Sigma^{\boldsymbol{\Phi}}$ can be extracted as follows:

$$\Sigma^{\boldsymbol{\Phi}} = \Sigma^{\boldsymbol{\Phi}}_{\boldsymbol{u}_1} + \Sigma^{\boldsymbol{\Phi}}_{\boldsymbol{u}_2} + \cdots + \Sigma^{\boldsymbol{\Phi}}_{\boldsymbol{u}_r} \tag{26}$$

where $\Sigma^{\boldsymbol{\Phi}}_{\boldsymbol{u}_i} = \frac{1}{|\boldsymbol{u}_i|}\sum_{\boldsymbol{x}_j \in \boldsymbol{u}_i} \left\|\boldsymbol{K}(\boldsymbol{x}_j, \boldsymbol{A}) - \boldsymbol{K}(\boldsymbol{v}_{\boldsymbol{u}_i}, \boldsymbol{A})\right\|^2$, $\boldsymbol{v}_{\boldsymbol{u}_i} = \frac{1}{|\boldsymbol{u}_i|}\sum_{\boldsymbol{x}_j \in \boldsymbol{u}_i} \boldsymbol{x}_j, i = 1, \ldots, r,\ j = 1, \ldots, |\boldsymbol{u}_i|$, $\boldsymbol{A} = [\boldsymbol{x}_1; \cdots; \boldsymbol{x}_n]$.

### 3.2.2 Weighted strategy

In practice, samples located at different positions have different influences on regression (Xu and Wang 2012). Therefore, it is more reasonable to assign different samples with different penalties. Motivated by WL-ε-TSVR (Ye et al. 2016), we adopt the following feedback strategy to weigh samples located at different positions:

$$d_i = \begin{cases} \exp\left(-\left(\left|\frac{e_i}{\hat{s}_{dev}}\right|\right)^2\right), & \left|\frac{e_i}{\hat{s}_{dev}}\right| < Th \\ 10^{-3}, & \left|\frac{e_i}{\hat{s}_{dev}}\right| \geq Th \end{cases} \tag{27}$$

where $d_i$ is the weight; $e_i = y_i - \hat{y}_i$ is the prediction error, $\hat{y}_i$ is the prediction value of $y_i$; $\hat{s}_{dev} = 1.483\text{MAD}(e_i)$ represents the extent of the estimated error deviated from the normal distribution; MAD stands for the median absolute deviation; $Th$ is a constant threshold, which is typically set as $Th = 3$.

Equation (27) indicates that the closer $|e_i/\hat{s}_{dev}|$ to 0, the larger weight $d_i$, and if $|e_i/\hat{s}_{dev}|$ is greater than or equal to $Th$, $d_i$ is set as $10^{-3}$.

### 3.2.3 Linear case

The primal problems of FC-WTSVR in linear case are expressed as follows:

$$\begin{aligned} &\min_{\boldsymbol{w}_1, b_1, \boldsymbol{\xi}} \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{e}\varepsilon_1 - (\boldsymbol{A}\boldsymbol{w}_1 + \boldsymbol{e}b_1)\|^2 + \frac{1}{2}\boldsymbol{w}_1^{\mathrm{T}}\Sigma\boldsymbol{w}_1 + \frac{1}{2}c_1\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\xi} \\ &\text{s.t. } \boldsymbol{Y} - (\boldsymbol{A}\boldsymbol{w}_1 + \boldsymbol{e}b_1) \geq \boldsymbol{e}\varepsilon_1 - \boldsymbol{\xi}, \boldsymbol{\xi} \geq \boldsymbol{0} \end{aligned} \tag{28}$$

$$\begin{aligned} &\min_{\boldsymbol{w}_2, b_2, \boldsymbol{\eta}} \frac{1}{2}\|\boldsymbol{Y} + \boldsymbol{e}\varepsilon_2 - (\boldsymbol{A}\boldsymbol{w}_2 + \boldsymbol{e}b_2)\|^2 + \frac{1}{2}\boldsymbol{w}_2^{\mathrm{T}}\Sigma\boldsymbol{w}_2 + \frac{1}{2}c_2\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\eta} \\ &\text{s.t. } (\boldsymbol{A}\boldsymbol{w}_2 + \boldsymbol{e}b_2) - \boldsymbol{Y} \geq \boldsymbol{e}\varepsilon_2 - \boldsymbol{\eta}, \boldsymbol{\eta} \geq \boldsymbol{0} \end{aligned} \tag{29}$$

where $c_1, c_2 > 0$ are penalty parameters, $\varepsilon_1, \varepsilon_2 > 0$ are insensitive loss parameters, $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ are slack vectors, $\|\cdot\|$ denotes the 2-norm, $\boldsymbol{w}_1$, $\boldsymbol{w}_2$ are weight vectors, $b_1, b_2$ are biases, $\Sigma$ is the covariance matrix defined in Eq. (25), $\boldsymbol{D} = \text{diag}(d_1, \ldots, d_n)$ is a weighted diagonal matrix, $d_i, i =$

$1,2,\ldots,n$ are defined in Eq. (27), and $\boldsymbol{e}$ and $\boldsymbol{0}$ represent all ones and all zeros column vectors with proper dimensions, respectively.

By introducing nonnegative Lagrangian multiplier vector $\boldsymbol{\alpha}$, we can construct the following Lagrangian function of Eq. (28):

$$\begin{aligned} L(\boldsymbol{w}_1,b_1,\xi,\boldsymbol{\alpha}) = & \frac{1}{2}\|\boldsymbol{Y}-\boldsymbol{e}\varepsilon_1-(\boldsymbol{A}\boldsymbol{w}_1+\boldsymbol{e}b_1)\|^2 \\ & +\frac{1}{2}\boldsymbol{w}_1^{\mathrm{T}}\Sigma\boldsymbol{w}_1+\frac{1}{2}c_1\xi^{\mathrm{T}}\boldsymbol{D}\xi- \\ & \boldsymbol{\alpha}^{\mathrm{T}}(\boldsymbol{Y}-\boldsymbol{e}\varepsilon_1-(\boldsymbol{A}\boldsymbol{w}_1+\boldsymbol{e}b_1)+\xi). \end{aligned} \tag{30}$$

Then, differentiating Eq. (30) with respect to $\boldsymbol{w}_1$,$b_1$, and $\xi$, we can obtain the following Karush–Kuhn–Tucker (KKT) conditions:

$$\begin{aligned} \frac{\partial L(\boldsymbol{w}_1,b_1,\xi,\boldsymbol{\alpha})}{\partial \boldsymbol{w}_1} & = -\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{Y}-\boldsymbol{e}\varepsilon_1-(\boldsymbol{A}\boldsymbol{w}_1+\boldsymbol{e}b_1))+\boldsymbol{w}_1\Sigma \\ & \quad +\boldsymbol{A}^{\mathrm{T}}\boldsymbol{\alpha} \\ & = \boldsymbol{0} \end{aligned} \tag{31}$$

$$\frac{\partial L(\boldsymbol{w}_1,b_1,\xi,\boldsymbol{\alpha})}{\partial b_1} = -\boldsymbol{e}^{\mathrm{T}}(\boldsymbol{Y}-\boldsymbol{e}\varepsilon_1-(\boldsymbol{A}\boldsymbol{w}_1+\boldsymbol{e}b_1))+\boldsymbol{e}^{\mathrm{T}}\boldsymbol{\alpha}=0 \tag{32}$$

$$\frac{\partial L(\boldsymbol{w}_1,b_1,\xi,\boldsymbol{\alpha})}{\partial \xi} = c_1\boldsymbol{D}\xi-\boldsymbol{\alpha}=\boldsymbol{0} \tag{33}$$

$$\boldsymbol{Y}-(\boldsymbol{A}\boldsymbol{w}_1+\boldsymbol{e}b_1)\geq \boldsymbol{e}\varepsilon_1-\xi,\xi\geq\boldsymbol{0} \tag{34}$$

$$\boldsymbol{\alpha}^{\mathrm{T}}(\boldsymbol{Y}-\boldsymbol{e}\varepsilon_1-(\boldsymbol{A}\boldsymbol{w}_1+\boldsymbol{e}b_1)+\xi)=0,\boldsymbol{\alpha}\geq\boldsymbol{0}. \tag{35}$$

Further, Eqs. (31) and (32) can be written into the following matrix form:

$$\begin{aligned} & \begin{bmatrix}\boldsymbol{A}^{\mathrm{T}}\\ \boldsymbol{e}^{\mathrm{T}}\end{bmatrix}(\boldsymbol{e}\varepsilon_1-\boldsymbol{Y})+\begin{bmatrix}\boldsymbol{A}^{\mathrm{T}}\\ \boldsymbol{e}^{\mathrm{T}}\end{bmatrix}[\boldsymbol{A}\quad \boldsymbol{e}]\begin{bmatrix}\boldsymbol{w}_1\\ b_1\end{bmatrix}+\begin{bmatrix}\Sigma & \boldsymbol{0}\\ \boldsymbol{0}^{\mathrm{T}} & 0\end{bmatrix}\begin{bmatrix}\boldsymbol{w}_1\\ b_1\end{bmatrix} \\ & +\begin{bmatrix}\boldsymbol{A}^{\mathrm{T}}\\ \boldsymbol{e}^{\mathrm{T}}\end{bmatrix}\boldsymbol{\alpha} \\ & =\boldsymbol{0}. \end{aligned} \tag{36}$$

Define $\boldsymbol{G}=[\boldsymbol{A}\quad \boldsymbol{e}]$, $\boldsymbol{f}=\boldsymbol{Y}-\boldsymbol{e}\varepsilon_1$, $\boldsymbol{J}=\begin{bmatrix}\Sigma & \boldsymbol{0}\\ \boldsymbol{0}^{\mathrm{T}} & 0\end{bmatrix}$, and $\boldsymbol{u}_1=\begin{bmatrix}\boldsymbol{w}_1\\ b_1\end{bmatrix}$; we can obtain the following augmented vector:

$$\boldsymbol{u}_1=\left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}+\boldsymbol{J}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}(\boldsymbol{f}-\boldsymbol{\alpha}). \tag{37}$$

Then, substituting Eqs. (33) and (37) into Eq. (30), we can obtain the dual problem of Eq. (28) as follows:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} & -\frac{1}{2}\boldsymbol{\alpha}^{\mathrm{T}}\left(\frac{\boldsymbol{D}^{-1}}{c_1}+\boldsymbol{G}\left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}+\boldsymbol{J}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\right)\boldsymbol{\alpha} \\ & +\boldsymbol{f}^{\mathrm{T}}\boldsymbol{G}\left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}+\boldsymbol{J}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\alpha}-\boldsymbol{f}^{\mathrm{T}}\boldsymbol{\alpha} \\ \text{s.t.} & \quad \boldsymbol{0}\leq\boldsymbol{\alpha}. \end{aligned} \tag{38}$$

Subsequently, we can obtain the augmented vector $\boldsymbol{u}_2$ as follows:

$$\boldsymbol{u}_2=\left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}+\boldsymbol{J}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}(\boldsymbol{h}+\boldsymbol{\beta}) \tag{39}$$

where $\boldsymbol{u}_2=\begin{bmatrix}\boldsymbol{w}_2\\ b_2\end{bmatrix}$, $\boldsymbol{h}=\boldsymbol{Y}+\boldsymbol{e}\varepsilon_2$, and $\boldsymbol{\beta}$ is nonnegative Lagrangian multiplier vector.

Similarly, the dual problem of Eq. (29) can be represented as follows:

$$\begin{aligned} \max_{\boldsymbol{\beta}} & -\frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\left(\frac{\boldsymbol{D}^{-1}}{c_2}+\boldsymbol{G}\left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}+\boldsymbol{J}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\right)\boldsymbol{\beta} \\ & -\boldsymbol{h}^{\mathrm{T}}\boldsymbol{G}\left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}+\boldsymbol{J}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{\beta}+\boldsymbol{h}^{\mathrm{T}}\boldsymbol{\beta} \\ \text{s.t.} & \quad \boldsymbol{0}\leq\boldsymbol{\beta}. \end{aligned} \tag{40}$$

Once Eqs. (38) and (40) are solved, we can obtain the optimal Lagrangian multiplier vectors $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$. Then, we can compute the augmented vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ according to Eqs. (37) and (39), respectively. Finally, the regression function of FC-WTSVR in linear case can be constructed by Eq. (1).

### 3.2.4 Nonlinear case

By introducing the kernel function $\boldsymbol{K}(\cdot,\cdot)$, the FC-WTSVR in linear case can be easily extended to nonlinear case. The primal problems of FC-WTSVR in nonlinear case are expressed as follows:

$$\begin{aligned} \min_{\boldsymbol{w}_1,b_1,\xi} & \frac{1}{2}\left\|\boldsymbol{Y}-\boldsymbol{e}\varepsilon_1-\left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{w}_1+\boldsymbol{e}b_1\right)\right\|^2 \\ & +\frac{1}{2}\boldsymbol{w}_1^{\mathrm{T}}\Sigma^{\boldsymbol{\Phi}}\boldsymbol{w}_1+\frac{1}{2}c_1\xi^{\mathrm{T}}\boldsymbol{D}\xi \\ \text{s.t.} & \quad \boldsymbol{Y}-\left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{w}_1+\boldsymbol{e}b_1\right)\geq\boldsymbol{e}\varepsilon_1-\xi,\xi\geq\boldsymbol{0} \end{aligned} \tag{41}$$

$$\begin{aligned} \min_{\boldsymbol{w}_2,b_2,\boldsymbol{\eta}} & \frac{1}{2}\left\|\boldsymbol{Y}+\boldsymbol{e}\varepsilon_2-\left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{w}_2+\boldsymbol{e}b_2\right)\right\|^2 \\ & +\frac{1}{2}\boldsymbol{w}_2^{\mathrm{T}}\Sigma^{\boldsymbol{\Phi}}\boldsymbol{w}_2+\frac{1}{2}c_2\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\eta} \\ \text{s.t.} & \quad \left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{w}_2+\boldsymbol{e}b_2\right)-\boldsymbol{Y}\geq\boldsymbol{e}\varepsilon_2-\boldsymbol{\eta},\boldsymbol{\eta}\geq\boldsymbol{0} \end{aligned} \tag{42}$$

where $\Sigma^{\boldsymbol{\Phi}}$ is the covariance matrices defined in Eq. (26); other symbols have the same meaning as in Eqs. (28) and (29).

Similar with the derivation of FC-WTSVR in linear case, by introducing nonnegative Lagrangian multiplier vectors $\boldsymbol{\mu}$ and $\boldsymbol{v}$, we can obtain the following dual problems of Eqs. (41) and (42), respectively:

$$\begin{aligned}\max_{\mu}\ & -\frac{1}{2}\mu^{\mathrm{T}}\left(\frac{D^{-1}}{c_1}+Q\left(Q^{\mathrm{T}}Q+J^{\Phi}\right)^{-1}Q^{\mathrm{T}}\right)\mu \\ & +f^{\mathrm{T}}Q\left(Q^{\mathrm{T}}Q+J\right)^{-1}Q^{\mathrm{T}}\mu-f^{\mathrm{T}}\mu \\ \text{s.t. }\ & 0\le\mu\end{aligned} \tag{43}$$

$$\begin{aligned}\max_{v}\ & -\frac{1}{2}v^{\mathrm{T}}\left(\frac{D^{-1}}{c_2}+Q\left(Q^{\mathrm{T}}Q+J^{\Phi}\right)^{-1}Q^{\mathrm{T}}\right)v \\ & -h^{\mathrm{T}}Q\left(Q^{\mathrm{T}}Q+J\right)^{-1}Q^{\mathrm{T}}v+h^{\mathrm{T}}v \\ \text{s.t. }\ & 0\le v\end{aligned} \tag{44}$$

where $Q=\left[K\left(A,A^{\mathrm{T}}\right)\quad e\right]$, $f=Y-e\varepsilon_1$, $h=Y+e\varepsilon_2$, and $J^{\Phi}=\begin{bmatrix}\Sigma^{\Phi} & 0\\ 0^{\mathrm{T}} & 0\end{bmatrix}$.

Once Eqs. (43) and (44) are solved, we can obtain the optimal Lagrangian multiplier vectors $\mu^*$ and $v^*$. Then, we can compute the augmented vectors $\vartheta_1$ and $\vartheta_2$ as follows:

$$\vartheta_1=\left(Q^{\mathrm{T}}Q+J^{\Phi}\right)^{-1}Q^{\mathrm{T}}(f-\mu^*) \tag{45}$$

$$\vartheta_2=\left(Q^{\mathrm{T}}Q+J^{\Phi}\right)^{-1}Q^{\mathrm{T}}(h+v^*) \tag{46}$$

where $\vartheta_1=\begin{bmatrix}w_1\\ b_1\end{bmatrix}$ and $\vartheta_2=\begin{bmatrix}w_2\\ b_2\end{bmatrix}$.

The final regression function of FC-WTSVR in nonlinear case can be constructed by Eq. (14).

## 3.3 Solving QPPs by successive over-relaxation algorithm

In this section, we adopt the successive over-relaxation (SOR) algorithm developed by Quan et al. to solve QPPs. The SOR algorithm has been proved to be of much faster convergence speed than other iterative algorithms such as gradient descending algorithm in solving QPPs (Quan et al. 2004).

In FC-WTSVR, four QPPs, i.e., Equations (38), (40), (43), and (44), should be solved. They can be rewritten into the following unified form:

$$\begin{aligned}\min_{z} f(z) &= \frac{1}{2}z^{\mathrm{T}}Pz-\epsilon^{\mathrm{T}}z \\ \text{s.t.}\, z &\ge 0\end{aligned} \tag{47}$$

where $z\in\mathbb{R}^{(n+1)}$ is a Lagrangian coefficient vector, $P\in\mathbb{R}^{(n+1)\times(n+1)}$ is a positive definite matrix, and $\epsilon\in\mathbb{R}^{(n+1)}$ is a vector.

For instance, if we let $z=\alpha$, $P=\frac{D^{-1}}{c_1}+G\left(G^{\mathrm{T}}G+J\right)^{-1}G^{\mathrm{T}}$, $\epsilon=f^{\mathrm{T}}G\left(G^{\mathrm{T}}G+J\right)^{-1}G^{\mathrm{T}}-f^{\mathrm{T}}$, then Eq. (47) equals to Eq. (38).

In Eq. (47), the iterative equation for updating $z^{i+1}$ based on $z^i$ is expressed as follows:

$$z^{i+1}=z^i-tE^{-1}\left(Pz^i-\epsilon+L\left(z^{i+1}-z^i\right)\right), i=0,1,\ldots \tag{48}$$

where interval $t\in(0,2)$, $z^i$ stands for the $i^{th}$ iteration of $z$, and $L\in\mathbb{R}^{(n+1)\times(n+1)}$ and $E\in\mathbb{R}^{(n+1)\times(n+1)}$ stand for the strictly lower triangular part and diagonal part of matrix $P$, respectively.

The SOR algorithm is iteratively computed according to Eq. (48) until $\left\|z^{i+1}-z^i\right\|$ is less than a predefined tolerance. In our work, we set the tolerance as $10^{-3}$. Similarly, the SOR algorithm can be used to solve Eqs. (40), (43), and (44).

## 3.4 Training procedure of FC-WTSVR

The whole training procedure of FC-TWSVR in linear case is summarized in Algorithm 2.

**Algorithm 2** The training procedure of FC-WTSVR in linear case (High-level summary).

```
Input: The training sample set T, parameters $c_1, c_2, \varepsilon_1, \varepsilon_2$, nearest neighbor parameters $k$, constant $\tau$, threshold $Th$ and interval $t$.
Output: The regression function $f(x)$.
Step1. Cluster data using Algorithm 1;
Step2. Remove outliers;
Step3. Compute the covariance matrix $\sum$ according to Eq. (25), set the initial weighted diagonal matrix as $D = diag(1,\cdots,1)$;
Step4. Solving Eqs. (38) and (40) by the SOR algorithm described in Sect. 3.3 to obtain the optimal parameters $c_1^*, c_2^*$;
Step5. Construct the regression function $f(x)$ according to Eq. (1);
Step6. Compute the prediction error $e_i = y_i - f(x_i)$, then update $d_i$ according to Eq. (27) and set $D^* = diag(d_1,\cdots,d_n)$;
Step7. Use the SOR algorithm to compute the final optimal solutions $\alpha^*$ and $\beta^*$ based on the optimal parameters $c_1^*, c_2^*$ and $D^*$;
Step8. Construct the final regression function $f^*(x)$ according to Eq. (1).
```

Algorithm 2 can be easily extended to nonlinear case; it is omitted here.

## 3.5 Analysis of FC-WTSVR

### 3.5.1 Time complexity of FC-WTSVR

According to Algorithm 2, the whole time complexity of the proposed FC-WTSVR is $O\left(n^3+2n^2+4n\log_2^n+5n\right)$. Obviously, the time complexity of FC-WTSVR is greater than that of standard TSVR ($O(n^3)$). However, with the increasing of the number of training samples, the time complexity of FC-WTSVR will approximate to $O(n^3)$. This implies that the time complexity of fast clustering (Algorithm 1) is negligible when training large-scale datasets.

### 3.5.2 Anti-interference capability of FC-WTSVR

In the clustering step, we determine the cluster centers and outliers based on appropriate principles. Then, to further improve the prediction performance, we utilize the covariance matrix to reflect the prior structural information of samples. In addition, to reduce the impact of noise on regression, a new weighed strategy is employed to assign weight based on proper principle. As a result, the final regression function constructed by FC-WTSVR is expected to be more robust, i.e., less sensitive to noise and outlier. This means that the proposed FC-WTSVR has powerful anti-interference capability.

## 4 Experimental results and analyses

In this section, we first present the experimental design. Then, we discuss the parameter selection of different algorithms. Finally, we conduct extensive experiments on benchmark datasets, artificial datasets, and actual glutamic acid fed-batch fermentation process.

### 4.1 Experimental design

In order to validate the superiorities of the proposed FC-WTSVR, we compared it with TSVR (Peng 2010), ε-TSVR (Shao et al. 2013), KNN-WTSVR (Xu and Wang 2014), and WL-ε-TSVR (Ye et al. 2016) on benchmark datasets, artificial datasets, and actual glutamic acid fed-batch fermentation process, respectively.

The following three criteria, i.e., the root mean squared error (RMSE), the mean absolute error (MAE), and the coefficient of determination ($R^2$), are used to evaluate the prediction performance of all algorithms:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|^2} \tag{49}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{50}$$

$$R^2 = \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|^2}{\sum_{i=1}^{n}|y_i - \bar{y}|^2} \tag{51}$$

where $n$ is the number of samples, $y_i$ is the actual output, $\hat{y}_i$ is the predicted value of $y_i$, and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ is the mean of the output.

Note that the smaller is the RMSE, the better is the prediction performance; the smaller is the MAE, the smaller is the prediction error; the closer is the $R^2$ to 1, the better is the fitting performance of the algorithm.

In addition, we also record the elapsed CPU time of TSVR, ε-TSVR, KNN-TSVR, WL-ε-TSVR, and FC-WTSVR, respectively. All experiments are implemented in MATLAB 2014a platform on a PC with 2.2 GHz Intel® Core™ i5-5200U Processor and 4 GB RAM, and all results are averaged in 20 independent trials. To make the results more convincing, we employ standard fivefold cross-validation (CV) method.

### 4.2 Parameter selection

In order to obtain good prediction performance, selecting suitable parameters for each regression algorithm is very important. In all experiments, we utilize the grid search algorithm to select the optimal parameters for different regression algorithms.

In all regression algorithms, because the setting of $\varepsilon_1$ and $\varepsilon_2$ cannot greatly influence the prediction performance (Chen et al. 2012; Shao et al. 2013; Xu and Wang 2014; Ye et al. 2016; López and Maldonado 2018; Fang et al. 2019), we set $\varepsilon_1 = 0.1$ and $\varepsilon_2 = 0.1$, and $c_1$ and $c_2$ are selected from the same set $\{2^i | i = -8, \cdots, 0, \cdots, 8\}$. In KNN-TSVR, we set $k = 10$. In ε-TSVR and WL-ε-TSVR, to make the comparison fair with other three regression algorithms, we set $c_1 = c_3$ and $c_2 = c_4$. In FC-WTSVR, we set $k = 10$, $\tau = 0.1$, $\mathrm{Th} = 3$, and $t = 0.9$.

In nonlinear case, we use the RBF kernel, i.e., $\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-(\boldsymbol{x}_i, \boldsymbol{x}_j)^2 / 2\sigma^2\right)$ and the kernel width parameter $\sigma$ is selected from the set $\{2^i | i = -4, \ldots, 0, \ldots, 4\}$.

### 4.3 Experiments on benchmark datasets

The UCI benchmark datasets used in our experiments are listed in Table 1, whose sizes vary from 102 to 1260. They can be downloaded from UCI machine learning repository.

Table 2 summarizes the average RMSE, MAE, $R^2$, and elapsed CPU time in 20 independent trials using different regression algorithms. The best values are marked with bold font.

From Table 2, we can find that, except for the elapsed CPU time, the proposed FC-WTSVR has similar or better prediction performance in comparison to other regression algorithms. The underlying cause is that the introduction of

**Table 1** The nine different UCI benchmark datasets used in our experiments

| Datasets | Number of samples | Attributions |
|---|---|---|
| Concrete slump test | 102 | 10 |
| Servo | 166 | 3 |
| Computer hardware | 208 | 8 |
| Yacht hydrodynamics | 308 | 7 |
| Auto MPG | 398 | 10 |
| Boston housing | 506 | 13 |
| DrivFace | 604 | 18 |
| Concrete compressive | 800 | 9 |
| Airfoil self-noise | 1260 | 6 |

prior structural information of samples and feedback weighted strategy is helpful in improving the prediction performance. Specially, the RMSE and MAE of FC-WTSVR are significantly better than other algorithms on Computer hardware and Yacht hydrodynamics datasets. This is due to the fact that there are several outliers in these two datasets, and our FC-WTSVR can remove outliers effectively so as to improve the prediction performance.

Table 2 also indicates that the elapsed CPU time of ε-TSVR is the shortest except for the Airfoil self-noise dataset. This is because the dual problems of ε-TSVR are strictly positive definite QPPs, which cost much less time in training compared with other algorithms. In addition, the elapsed CPU time of KNN-WTSVR and WL-ε-TSVR are less than that of standard TSVR on five benchmark datasets. This is due to the fact that the introduction of weighted matrix eliminates the redundant constraints and reduces the time complexity. At last, the elapsed CPU time of FC-WTSVR is the longest expect for the Yacht hydrodynamics dataset. This is caused by the extra time of implementing fast clustering algorithm.

Next, we will discuss the relationships between constant $\tau$ and RMSE in the proposed FC-WTSVR, and the results are presented in Fig. 4. Note that the bigger is the constant $\tau$, the more samples are determined as outliers and removed from the training data samples.

Figure 4a, e indicates that the RMSE gradually becomes larger with the increase in constant $\tau$. This means that the samples of significant contributions to regression may have been removed, and there may be no outliers in Concrete slump test and Auto MPG datasets.

Then, we can see from Fig. 4b that the RMSE first becomes smaller and then becomes larger with the increase in constant $\tau$. This implies that there are potential outliers in Servo dataset, and they have great influences on regression. Similar situations can be found in Fig. 4f–h. Moreover, Fig. 4c, d shows that the RMSE monotonically decreases with the increase in constant $\tau$. This indicates that there may be several outliers in Computer hardware and Yacht hydrodynamics datasets. Therefore, after removing the outliers, the prediction performance of our FC-WTSVR is much better than the other four regression algorithms. Finally, Fig. 4i demonstrates that the RMSE almost remains unchanged with the increase in constant $\tau$. The reason for interpreting this phenomenon is that all potential outliers fall into the top left corner of decision graph, and they are independent of the selection of constant $\tau$.

To conclude, the RMSE fluctuates with the selection of constant $\tau$. Hence, we should select the constant $\tau$ carefully. In our experiments, we set the constant $\tau$ as a relatively reasonable value, i.e., $\tau = 0.1$.

## 4.4 Experiments on artificial datasets

We conduct simulation experiments on the following two artificial nonlinear functions:

$$A : y_i = \sin c(x_i) + n_i = \frac{\sin(x_i)}{x_i} + n_i, x_i \in [-5, +5] \tag{52}$$

$$B : y_i = x_i^{2/3} + n_i, x_i \in [-3, +3] \tag{53}$$

where $x_i$ is the input, $y_i$ is the output, and $n_i$ is the normally distributed noise with zero mean and variance $0.1^2$, i.e., $n_i \sim N(0, 0.1^2)$.

Figures 5 and 6 present the fitting regression curves of the two nonlinear functions using different regression algorithms.

We randomly generate 80 data samples $(x_i, y_i), i = 1, \ldots, 80$. Furthermore, to test the anti-interference capability of different regression algorithms, we intentionally add four different outliers in Eqs. (52) and (53). Half of the data samples are randomly selected for training, and the remaining half are selected for testing.

We can see clearly from Figs. 5 and 6 that the proposed FC-WTSVR outperforms the other four algorithms in terms of anti-interference ability. Firstly, compared with standard TSVR, the anti-interference ability our FC-WTSVR is far superior. The reason is that our FC-WTSVR removes potential outliers according to appropriate principle. Secondly, the anti-interference ability of WL-ε-TSVR is better than that of KNN-WTSVR. This can be explained by the fact that KNN-WTSVR assigns equal weight to each sample in the $k$-nearest neighbor range, whereas WL-ε-TSVR weighs each sample according to the density of samples. In comparison to WL-ε-TSVR, FC-WTSVR removes outliers instead of assigning tiny weights for potential outliers. Therefore, the anti-interference ability FC-WTSVR is superior to WL-ε-TSVR. Finally, standard TSVR and ε-TSVR are sensitive to noise and outlier. The

**Table 2** The results of average RMSE, MAE, $R^2$, and elapsed CPU time on UCI benchmark datasets using different regression algorithms

| Datasets | Algorithms | RMSE | MAE | $R^2$ | CPU time (s) |
|---|---|---|---|---|---|
| Concrete slump test | TSVR | 2.7694 | 2.1572 | 2.2116 | 1.2860 |
| | ε-TSVR | 2.7687 | **1.9710** | 0.3995 | **0.5781** |
| | KNN-WTSVR | 2.6730 | 2.0283 | 0.4407 | 1.9547 |
| | WL-ε-TSVR | **2.6115** | 2.0484 | **0.4955** | 1.7756 |
| | FC-WTSVR | 2.7159 | 2.2038 | 0.3986 | 3.6563 |
| Servo | TSVR | 1.2308 | 0.8928 | 0.8260 | 3.3653 |
| | ε-TSVR | 1.2253 | 0.8186 | 1.1446 | **2.6093** |
| | KNN-WTSVR | 1.2089 | 0.9215 | 0.8460 | 4.4452 |
| | WL-ε-TSVR | 1.2126 | **1.0240** | 0.8503 | 3.3437 |
| | FC-WTSVR | **1.1287** | 0.9583 | **0.8973** | 6.9687 |
| Computer hardware | TSVR | 38.2803 | 24.8779 | 0.1776 | 5.4531 |
| | ε-TSVR | 39.4950 | 27.0523 | 0.1596 | **1.2028** |
| | KNN-WTSVR | 35.8257 | 23.6840 | 0.1886 | 3.9876 |
| | WL-ε-TSVR | 39.6133 | 26.2237 | 0.1987 | 3.4531 |
| | FC-WTSVR | **28.1649** | **21.1926** | **0.2569** | 11.6719 |
| Yacht hydrodynamics | TSVR | 10.3484 | 7.2022 | 0.7343 | 26.5781 |
| | ε-TSVR | 9.0528 | 6.6976 | **1.0308** | **5.3125** |
| | KNN-WTSVR | 9.1757 | 7.3108 | 0.7271 | 7.5625 |
| | WL-ε-TSVR | 8.9709 | 6.7360 | 0.7345 | 5.8281 |
| | FC-WTSVR | **5.9343** | **3.8772** | 0.8569 | 19.2981 |
| Auto MPG | TSVR | 0.6250 | 0.4650 | 1.2390 | 22.8125 |
| | ε-TSVR | 0.6375 | 0.4583 | 0.7441 | **7.1994** |
| | KNN-WTSVR | 0.6235 | 0.4679 | 0.8199 | 13.625 |
| | WL-ε-TSVR | **0.6192** | **0.4502** | **0.8905** | 7.8281 |
| | FC-WTSVR | 0.6475 | 0.4618 | 0.8646 | 29.7188 |
| Boston housing | TSVR | 5.1122 | 3.4555 | 1.4470 | 18.8906 |
| | ε-TSVR | 5.0056 | 3.8761 | 0.6752 | **11.2344** |
| | KNN-WTSVR | 4.9499 | 3.4019 | 0.6529 | 17.8281 |
| | WL-ε-TSVR | **4.8529** | **3.1632** | **0.6805** | 13.5781 |
| | FC-WTSVR | 5.0231 | 3.3355 | 0.6335 | 41.2813 |
| DrivFace | TSVR | 1.6631 | 1.2620 | 0.9224 | 43.8594 |
| | ε-TSVR | 1.7430 | 1.3442 | 0.8866 | **18.6352** |
| | KNN-WTSVR | 1.7326 | 1.3378 | 0.9078 | 44.3690 |
| | WL-ε-TSVR | **1.6300** | **1.2023** | **0.9536** | 35.3594 |
| | FC-WTSVR | 1.6422 | 1.2678 | 0.9382 | 75.5781 |
| Concrete compressive | TSVR | 11.041 | 8.7213 | 1.3175 | 51.0805 |
| | ε-TSVR | 11.026 | 8.6861 | 1.3016 | **26.0737** |
| | KNN-WTSVR | 10.988 | 8.6265 | 1.3146 | 26.5695 |
| | WL-ε-TSVR | 11.069 | 8.5716 | 1.3556 | 32.3281 |
| | FC-WTSVR | **10.675** | **7.9064** | **1.2305** | 54.3594 |
| Airfoil self-noise | TSVR | 5.1529 | 4.1936 | 0.9785 | 60.6909 |
| | ε-TSVR | 4.9396 | 3.8922 | **0.9816** | 51.3082 |
| | KNN-WTSVR | 5.0049 | 3.8652 | 0.9753 | **44.6554** |
| | WL-ε-TSVR | **4.8542** | **3.7004** | 0.9708 | 85.7344 |
| | FC-WTSVR | 4.9647 | 3.7845 | 0.9598 | 112.8590 |

potential outliers will make the fitting regression curve greatly deviate from the actual one. The reason for interpreting this phenomenon is that the final fitting regression curve is determined by the minimum fitting regression error.

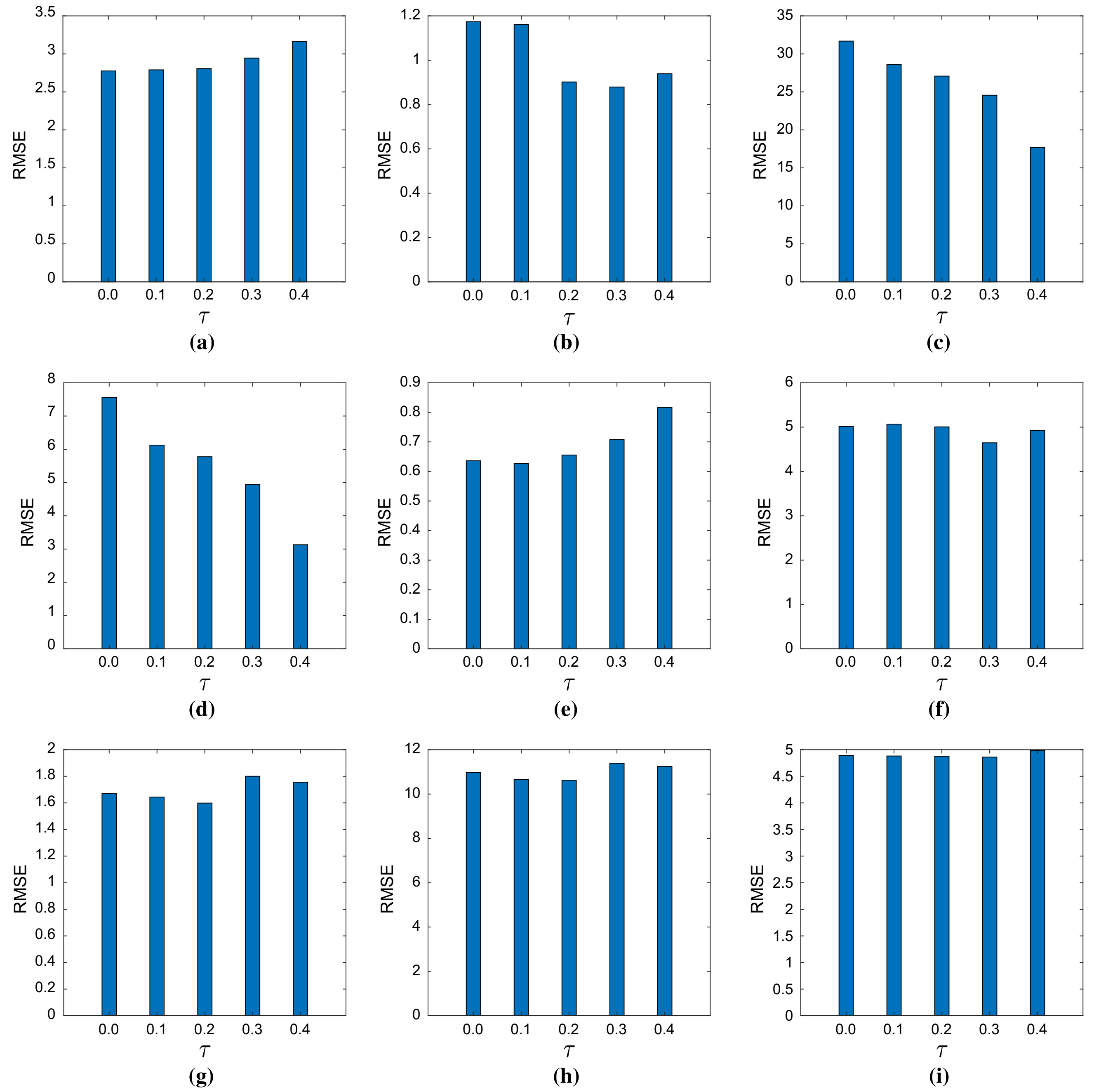


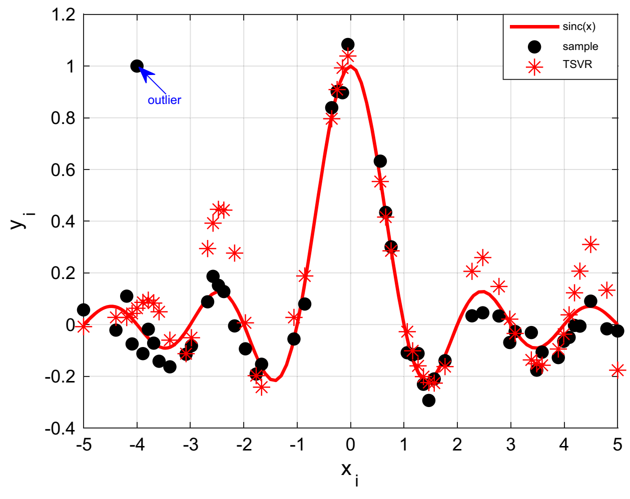
**Fig. 4** The relationships between constant $\tau$ and RMSE in the proposed FC-WTSVR. **a** Concrete slump test; **b** Servo; **c** Computer hardware; **d** Yacht hydrodynamics; **e** Auto MPG; **f** Boston housing; **g** DrivFace; **h** Concrete compressive; **i** Airfoil self-noise

Next, in order to further distinguish the anti-interference capability of our FC-WTSVR. We conduct experiments on $y_i = \sin c(x_i) + n_i, x_i \in [-5, +5]$ with two categories of noise $n_i$, i.e., uniformly distributed (UD) noise and normally distributed (ND) noise with zero mean and varying variance, respectively.

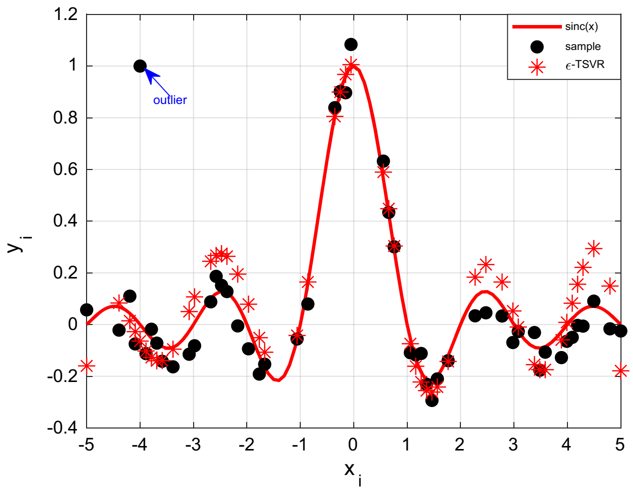
Once again, we randomly generate data 80 samples $(x_i, y_i), i = 1, \ldots, 80$. Half of the data samples are randomly selected for training, and the remaining half are selected for testing. The artificial test functions used in our experiments are listed in Table 3.

Table 4 summarizes the average RMSE, MAE, and $R^2$ in 20 independent trials on six artificial test functions using different regression algorithms. The best values are marked with bold font.

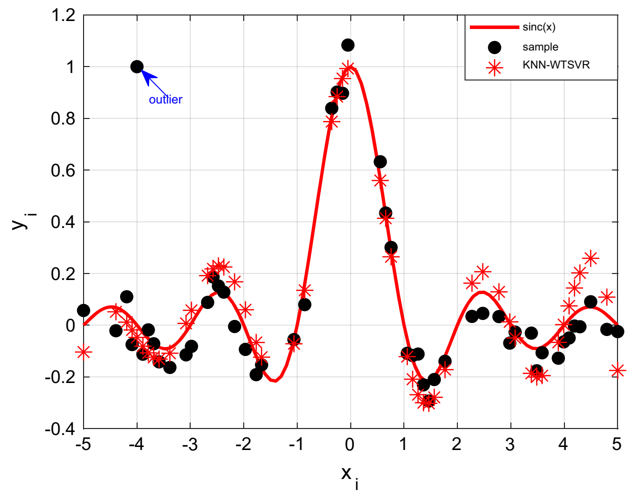
We can see from Table 4 that WL-ε-TSVR and FC-WTSVR alternatively outperform TSVR, ε-TSVR, and KNN-WTSVR in terms of RMSE, MAE, and $R^2$ on test



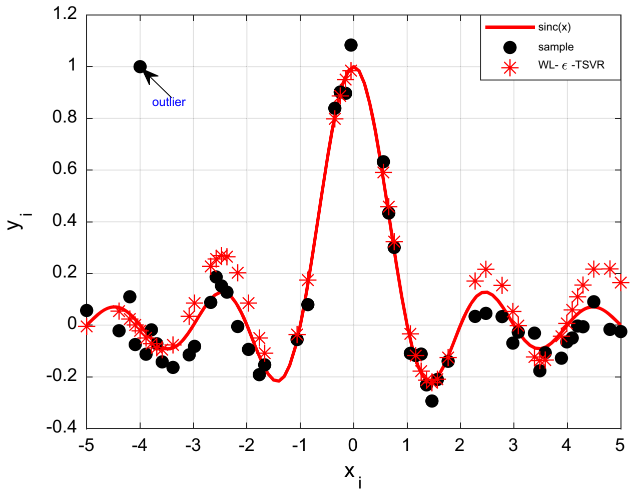
(a)



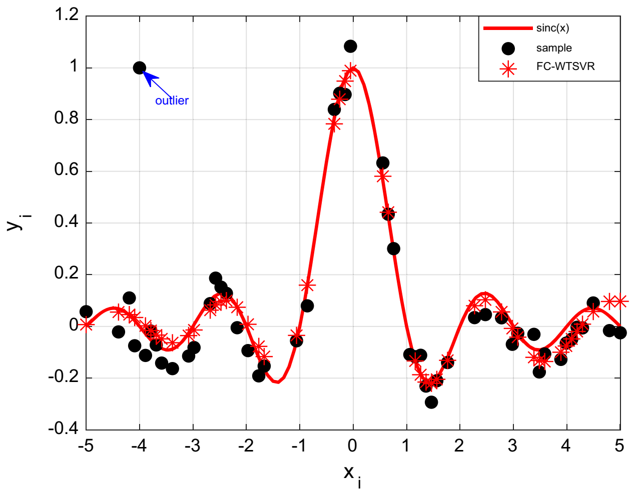
(b)



(c)



(d)



(e)

**Fig. 5** The fitting regression curves of nonlinear function $A$ using different regression algorithms. **a** TSVR; **b** ε-TSVR; **c** KNN-WTSVR; **d** WL-ε-TSVR; **e** FC-WTSVR

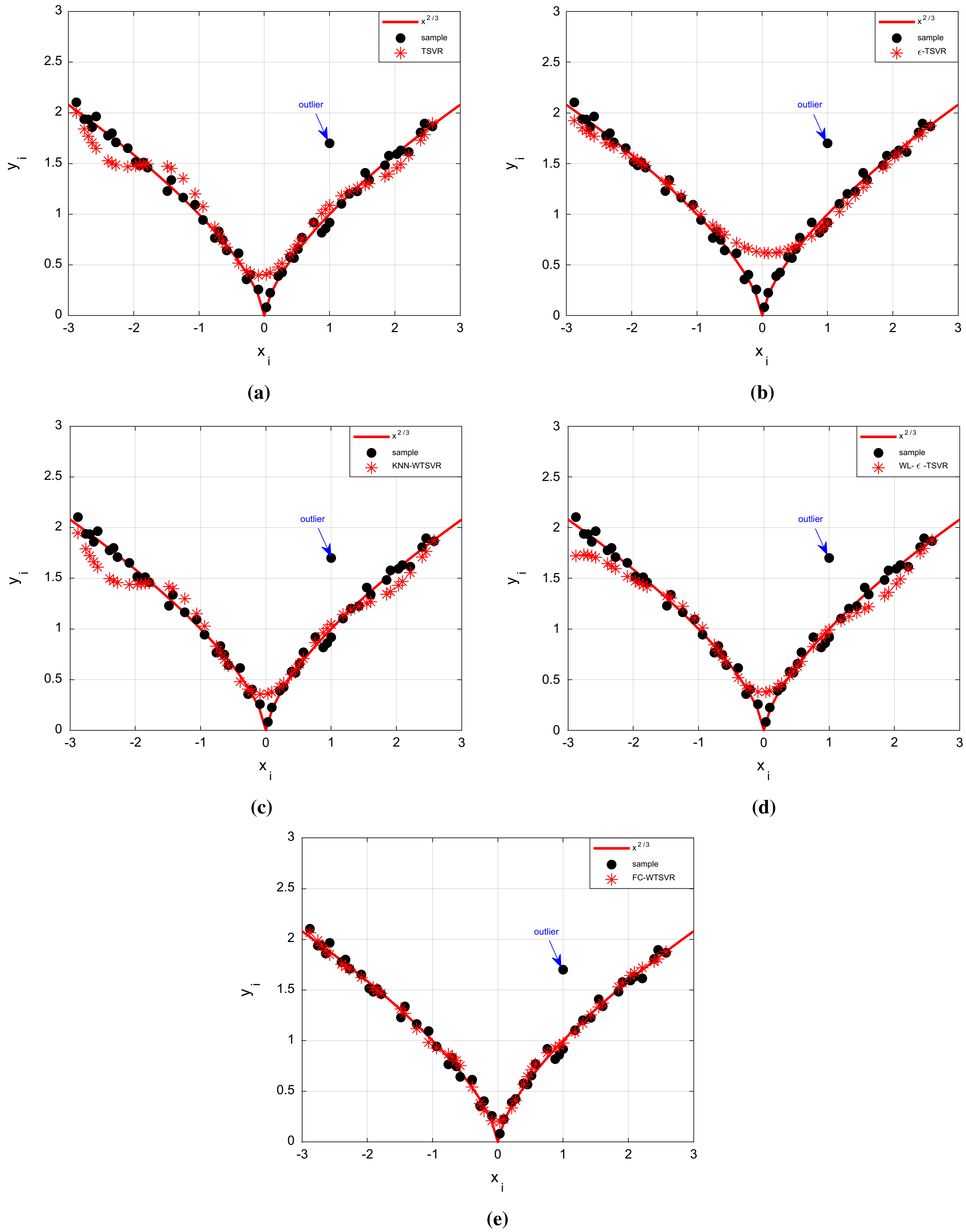


**Fig. 6** The fitting regression curves of nonlinear function *B* using different regression algorithms. **a** TSVR; **b** ε-TSVR; **c** KNN-WTSVR; **d** WL-ε-TSVR; **e** FC-WTSVR

**Table 3** The artificial test functions with different categories of noise and different numbers of outliers

| No. | Category of noise | Variance | Number of outliers |
|---|---|---|---|
| A1 | UD | 0.1 | 0 |
| A2 | UD | 0.2 | 0 |
| A3 | ND | $0.1^2$ | 0 |
| A4 | ND | $0.2^2$ | 0 |
| A5 | UD | 0.1 | 4 |
| A6 | ND | $0.1^2$ | 4 |

functions A1 to A4 without outliers. This indicates that both WL-ε-TSVR and FC-WTSVR have good prediction performance and powerful anti-noise capability. In addition, compared with the other four regression algorithms,

**Table 4** The results of average RMSE, MAE, and $R^2$ on artificial test functions using different regression algorithms

| No. | Algorithms | RMSE | MAE | $R^2$ |
|---|---|---|---|---|
| A1 | TSVR | 0.2074 | 0.0494 | 0.6112 |
| | ε-TSVR | 0.1903 | 0.0487 | 0.6052 |
| | KNN-WTSVR | 0.1872 | 0.0520 | 0.5496 |
| | WL-ε-TSVR | 0.1846 | 0.0483 | 0.6426 |
| | FC-WTSVR | **0.1834** | **0.0478** | **0.6505** |
| A2 | TSVR | 0.2213 | 0.0988 | 0.4412 |
| | ε-TSVR | 0.2264 | 0.1045 | 0.4228 |
| | KNN-WTSVR | 0.2163 | 0.0936 | 0.4407 |
| | WL-ε-TSVR | **0.2074** | **0.0836** | **0.4955** |
| | FC-WTSVR | 0.2093 | 0.0867 | 0.4686 |
| A3 | TSVR | 0.2612 | 0.2048 | 0.3702 |
| | ε-TSVR | 0.2624 | 0.2077 | 0.3756 |
| | KNN-WTSVR | 0.2592 | 0.2037 | 0.3608 |
| | WL-ε-TSVR | **0.2491** | 0.1881 | **0.4036** |
| | FC-WTSVR | 0.2561 | **0.1834** | 0.3877 |
| A4 | TSVR | 0.4357 | 0.3138 | 0.4509 |
| | ε-TSVR | 0.4332 | 0.3129 | 0.4468 |
| | KNN-WTSVR | 0.4345 | 0.3175 | 0.4543 |
| | WL-ε-TSVR | **0.4218** | 0.3194 | **0.4826** |
| | FC-WTSVR | 0.4256 | **0.3112** | 0.4667 |
| A5 | TSVR | 0.3389 | 0.2712 | 0.6725 |
| | ε-TSVR | 0.3428 | 0.2736 | 0.6658 |
| | KNN-WTSVR | 0.3390 | 0.2689 | 0.6686 |
| | WL-ε-TSVR | 0.3301 | 0.2579 | 0.6805 |
| | FC-WTSVR | **0.2694** | **0.2208** | **0.7515** |
| A6 | TSVR | 0.4161 | 0.3188 | 0.5231 |
| | ε-TSVR | 0.4123 | 0.3180 | 0.5218 |
| | KNN-W TSVR | 0.4098 | 0.3061 | 0.5536 |
| | WL-ε-TSVR | 0.4034 | 0.2997 | 0.5807 |
| | FC-WTSVR | **0.3092** | **0.2486** | **0.6875** |

the prediction performance and anti-interference capability of the proposed FC-WTSVR are the best on test functions A5 and A6 with outliers. This can be explained by the fact that FC-WTSVR not only removes potential outliers according to appropriate principle, but also introduces the covariance matrix and weighted diagonal matrix into the primal problems.

In a word, the proposed FC-WTSVR is less sensitive to noise and outlier compared with some state-of-the-art algorithms.

## 4.5 Experiments on actual glutamic acid fed-batch fermentation process

To further validate the advantages of the proposed FC-WTSVR, we conduct experiments on actual glutamic acid fed-batch fermentation process. The experimental data are supported by the key laboratory of industrial biotechnology, ministry of education, Jiangnan University. In nature, the glutamic fed-batch fermentation process is a quite complicated nonlinear regression process (Gu and Pan 2015). In general, the whole glutamic acid fed-batch fermentation process includes three different types of variable, i.e., physical variable, chemical variable, and biological variable. Among these variables, biological variable, such as biomass concentration (g/L), glutamic acid concentration (g/L), and substrate concentration (g/L), is the key factor of measuring whether the fermentation process is successful or not. However, in practice, these biological variables can only be offline measured by special instruments. Recently, with the help of soft sensor modeling, they can be online measured and adjusted. First, we use the proposed FC-WTSVR to construct the soft sensor model of the glutamic fed-batch fermentation process. Then, we focus on the glutamic acid concentration.

Six batch experiments are carried out under the condition of keeping 10%, 20%, 30%, and 50% dissolved oxygen (DO) concentration, respectively. Each batch of data can represent the whole glutamic fed-batch fermentation process. The data include offline analyzed data and online measured data, and they are normalized to cancel the influence of dimension. Five batches are used for training, and the remaining one batch is used for testing. We use the same parameter setting as in nonlinear case. Table 5 lists the results of average RMSE, MAE, $R^2$, the elapsed CPU time, and the optimal parameters using different regression algorithms. The best values are marked with bold font. We can see clearly from Table 5 that the RMSE, MAE, and $R^2$ of FC-WTSVR are better than those of the other four regression algorithms. This means that the prediction performance of FC-WTSVR is the best. However, the elapsed CPU time of FC-WTSVR is the longest. Fortunately, the

Table 5 The results of average RMSE, MAE, $R^2$, the elapsed CPU time, and the optimal parameters $(c_1, c_2, \sigma)$ using different regression algorithms

| Algorithms | RMSE | MAE | $R^2$ | CPU time (s) | $(c_1, c_2, \sigma)$ |
|---|---|---|---|---|---|
| TSVR | $2.600 \times 10^{-4}$ | $1.002 \times 10^{-4}$ | 1.3897 | 50.2855 | $(2^{-8}, 2^{-8}, 2^{-2})$ |
| $\varepsilon$-TSVR | $1.410 \times 10^{-3}$ | $8.669 \times 10^{-4}$ | 1.5786 | **29.5680** | $(2^{-8}, 2^{-8}, 2^{0})$ |
| KNN-TSVR | $2.241 \times 10^{-3}$ | $1.896 \times 10^{-3}$ | 1.6896 | 44.3699 | $(2^{+6}, 2^{-2}, 2^{0})$ |
| WL-$\varepsilon$-TSVR | $7.221 \times 10^{-6}$ | $5.668 \times 10^{-6}$ | 1.2033 | 40.9663 | $(2^{-8}, 2^{-6}, 2^{-4})$ |
| FC-WTSVR | $\mathbf{4.620 \times 10^{-6}}$ | $\mathbf{3.821 \times 10^{-6}}$ | **1.1036** | 62.3002 | $(2^{-6}, 2^{-8}, 2^{-4})$ |

elapsed CPU time of FC-WTSVR is still within acceptable range.

## 5 Conclusions

In this paper, we present an effective regression model, i.e., fast clustering-based weighted twin support vector regression (FC-WTSVR), to fit data samples with noise or outlier. First, a fast clustering algorithm is utilized to classify different categories of samples. Then, by introducing the covariance matrix and weighed diagonal matrix, the primal problems of the proposed FC-WTSVR not only reflect the prior structural information of samples, but also assign samples located at different positions with different penalties. Finally, the SOR algorithm is adopted to speed up the training process. Experimental results indicate that the proposed FC-WTSVR is better than some state-of-the-art algorithms in both prediction performance and anti-interference capability.

In fact, the idea of the proposed FC-WTSVR is expected to be applied in $\nu$-twin support vector regression ($\nu$-TSVR) and least squares twin support vector regression (LS-TSVR).

However, the main drawback of the proposed FC-WTSVR is that it costs more training time than other algorithms. In addition, the principle of determining the potential outliers and the weighted strategy used in our paper may be not the best one. We hope these questions will be addressed in our future work.

**Acknowledgement** This research was funded by the National Natural Science Foundation of China (21878124, 31771680 and 61773182).

## Compliance with ethical standards

**Conflict of interest** The authors declare no conflict of interests.

**Ethical approval** This article does not contain any studies with human participants or animals performed by any of the authors.

**Informed consent** Informed consent was obtained from all individual participants included in the study.

## References

Anagha P, Balasundaram S, Meena Y (2018) On robust twin support vector regression in primal using squared pinball loss. J Intell Fuzzy Syst 35(5):5231–5239

Balasundaram S, Gupta D (2014) Training Lagrangian twin support vector regression via unconstrained convex minimization. Knowl Based Syst 59:85–96

Bruno WJ, Socci ND, Halpern AL (2000) Weighted neighbor joining: a likelihood-based approach to distance-based phylogeny reconstruction. Mol Biol Evol 17(1):189–197

Chang KW, Hsieh CJ, Lin CJ (2008) Coordinate descent method for large-scale $L_2$-loss linear support vector machines. J Mach Learn Res 9(3):1369–1398

Chen XB, Yang J, Liang J, Ye QL (2012) Smooth twin support vector regression. Neural Comput Appl 21(3):505–513

Chen XB, Yang J, Chen L (2014) An improved robust and sparse twin support vector regression via linear programming. Soft Comput 18(12):2335–2348

Chen SG, Gao JF, Huang Z (2019) Weighted linear loss projection twin support vector machine for pattern classification. IEEE Access 7:57349–57360

Cheng HX, Wang J (2016) Density-weighted twin support vector regression. Control Decis 31(4):755–758

Cristianini N, Shawe-Talyor J (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge

Deng NY, Tian YJ (2009) Support vector machine: theory, algorithm and extension. Science Press, Beijing

Fang JW, Pan F, Gu BJ (2019) Twin support vector regression based on fruit fly optimization algorithm. Int J Innov Comput Inf Control 15(5):1851–1864

Gu BJ, Pan F (2015) A soft sensor modelling of biomass concentration during fermentation using accurate incremental online ν-support vector regression learning algorithm. Am J Biochem Biotechnol 1(3):149–159

Gu BJ, Shen GL, Pan F, Chen H (2019) Least squares twin projection support vector regression. Int J Innov Comput Inf Control 15(6):2275–2288

Gupta U, Gupta D (2019) An improved regularization based Lagrangian asymmetric ν-twin support vector regression using pinball loss function. Appl Intell 49(10):3606–3627

Hua XP, Xu S, Gao J, Ding SF (2019) $L_1$-norm loss-based projection twin support vector machine for binary classification. Soft Comput 23(21):10649–10659

Jayadeva KR, Chandra S (2007) Twin support vector machines for pattern classification. IEEE Trans Pattern Anal Mach Intell 29(5):905–910

Jiang SY, Song XY, Wang H, Han JJ, Li QH (2006) A clustering-based method for unsupervised intrusion detections. Pattern Recogn Lett 27(7):802–810

Kalidas Y, Chandra N (2008) Pocketdepth: a new depth based algorithm for identification of ligand binding sites in proteins. J Struct Biol 161(1):0–42

Liu R, Wang H, Yu XM (2018) Shared-nearest-neighbor-based clustering by fast search and find of density peaks. Inf Sci 450:200–226

López J, Maldonado S (2018) Robust twin support vector regression via second-order cone programming. Knowl Based Syst 152:83–93

López J, Barbero Á, Dorronsoro JR (2011) Clipping algorithms for solving the nearest point problem over reduced convex hulls. Pattern Recogn 44(3):607–614

Mangasarian OL, Musicant DR (2001) Lagrangian support vector machines. J Mach Learn Res 1(3):161–177

Mavroforakis ME, Theodoridis S (2006) A geometric approach to support vector machine (SVM) classification. IEEE Trans Neural Netw 17(3):671–682

Niu JY, Chen J, Xu YT (2017) Twin support vector regression with Huber loss. J Intell Fuzzy Syst 32(6):4247–4258

Pan XL, Luo Y, Xu YT (2015) *K*-nearest neighbor based structural twin support vector machine. Knowl Based Syst 88:34–44

Pang XY, Xu YT (2019) A safe screening rule for accelerating weighted twin support vector machine. Soft Comput 23(17):7725–7739

Pang XY, Xu C, Xu YT (2018) Scaling KNN multi-class twin support vector machine via safe instance reduction. Knowl Based Syst 148:17–30

Parastalooi N, Amiri A, Aliheydari P (2016) Modified twin support vector regression. Neurocomputing 211:84–97

Peng XJ (2010) TSVR: an efficient twin support vector machine for regression. Neural Netw 23(3):365–372

Peng XJ, Wang YF, Xu D (2013) Structural twin parametric-margin support vector machine for binary classification. Knowl Based Syst 49:63–72

Peng XJ, Xu D, Shen JD (2014) A twin projection support vector machine for data regression. Neurocomputing 138:131–141

Peng XJ, Xu D, Kong LY, Chen DJ (2016) $L_1$-norm loss based twin support vector machine for data recognition. Inf Sci 340–341:86–103

Platt J (2000) Fast training of support vector machines using sequential minimal optimization. MIT Press, Cambridge

Qi ZQ, Tian YJ, Shi Y (2013) Structural twin support vector machine for classification. Knowl Based Syst 43:74–81

Quan Y, Yang J, Yao LX, Ye CZ (2004) Successive over-relaxation for support vector regression. J Softw 15(2):200–206

Rodriguez A, Laio A (2014) Clustering by fast search and find of density peaks. Science 344(6191):1492–1496

Shao YH, Zhang CH, Yang ZM, Ling J, Deng NY (2013) An ε-twin support vector machine for regression. Neural Comput Appl 23(1):175–185

Shao YH, Chen WJ, Zhang JJ, Wang Z, Deng NY (2014) An efficient weighted Lagrangian twin support vector machine for imbalanced data classification. Pattern Recogn 47(9):3158–3167

Tanveer M, Sharma A, Suganthan PN (2019a) General twin support vector machine with pinball loss function. Inf Sci 494:311–327

Tanveer M, Tiwari A, Choudhary R, Jalan S (2019b) Sparse pinball twin support vector machines. Appl Soft Comput J 78:164–175

Vapnik VN (1999) An overview of statistical learning theory. IEEE Trans Neural Netw 10(5):988–999

Wang LD, Gao C, Zhao NN, Chen XB (2019) A projection wavelet weighted twin support vector regression and its primal solution. Appl Intell 49(8):3061–3081

Xu YT, Wang LS (2012) A weighted twin support vector regression. Knowl Based Syst 33:92–101

Xu YT, Wang LS (2014) *K*-nearest neighbor-based weighted twin support vector regression. Appl Intell 41(1):299–309

Xu GB, Cao Z, Hu BG, Principe JC (2017) Robust support vector machines based on the rescaled hinge loss function. Pattern Recogn 63:139–148

Xue ZX, Zhang RX, Qin CD, Zeng XQ (2018) A rough υ-twin support vector regression machine. Appl Intell 48(11):4023–4046

Ye YF, Cao H, Bai L, Wang Z, Shao YH (2013) Exploring determinants of inflation in china based on $L_1$-ε-twin support vector regression. Proc Comput Sci 17:514–522

Ye YF, Bai L, Hua XY, Shao YH, Wang Z et al (2016) Weighted Lagrange ε-twin support vector regression. Neurocomputing 197:53–68

Yeung DS, Wang DF, Ng WWY, Tsang ECC, Wang XZ (2007) Structured large margin machines: sensitive to data distributions. Mach Learn 68(2):171–200