International Journal of Innovative
Computing, Information and Control
Volume **17**, Number **2**, April **2021**

 ISSN 1349-4198
pp. 565–579

ISOLATION FOREST-BASED LEAST SQUARES TWIN MARGIN DISTRIBUTION SUPPORT VECTOR REGRESSION

Wei Feng, Geliang Shen, Benye Xu and Binjie Gu*

Key Laboratory of Advanced Process Control for Light Industry, Ministry of Education
Jiangnan University
No. 1800, Lihu Road, Wuxi 214122, P. R. China
fengwei@jiangnan.edu.cn; 17851315189@163.com; 6191905036@stu.jiangnan.edu.cn
*Corresponding author: gubinjie1980@126.com

Received August 2020; revised January 2021

Abstract. *The least squares twin support vector regression (LSTSVR) is a powerful tool for regression estimation. LSTSVR implements faster than twin support vector regression (TSVR) and fits for training large-scale data. However, LSTSVR is sensitive to outliers. In this paper, we present an isolation forest-based least squares twin margin distribution support vector regression (IFLSTMDSVR). First, we utilize the isolation forest approach to explicitly isolate the potential outliers and assign them with suitable anomaly scores. Next, because the margin distribution information is closely related to the generalization capability of regression model, we integrate it into the objective functions of IFLSTMDSVR. Finally, we conduct extensive simulation experiments on UCI benchmark datasets and synthetic test function. The results show that IFLSTMDSVR is less sensitive to outliers and performs better than several state-of-the-art algorithms in terms of generalization capability.*

Keywords: Machine learning, Twin support vector regression (TSVR), Least squares, Isolation forest, Margin distribution

1. **Introduction.** Support vector machine (SVM) is an effective machine learning tool based on the theory of statistical learning [1-5]. The goal of SVM is to minimize the structural risks by maximizing the margin between different classes. Therefore, the model constructed by SVM is of good learning and generalization capabilities. In recent decades, many variants of SVM have been developed and successfully applied in forest fires burned area prediction [6], target tracking [7], pedestrian detection [8], stock price modeling [9], and so on.

Recently, Jayadeva et al. investigated a new model for classification, called twin support vector machine (TSVM) [10]. TSVM builds two non-parallel hyperplanes and reduces to solve two small-scale quadratic programming problems (QPPs) rather than one large-scale QPP. In theory, TSVM runs approximately four times faster than SVM. Hence, TSVM has become a new hot topic. Inspired by TSVM, Peng developed a twin support vector regression (TSVR). Experimental results show that TSVR is superior to support vector regression (SVR) in both training time and generalization capability [11]. To date, many variants of TSVR have been exploited, such as twin parametric insensitive SVR (TPISVR) [12], ε-TSVR [13], Lagrangian TSVR [14], twin projection support vector regression (TPSVR) [15], and weighted TSVR [16,17].

Although TSVR and its variants perform faster than SVR, it is inefficient when training large-scale dataset. Fortunately, the least squares approach provides an effective way

DOI: 10.24507/ijicic.17.02.565

to address this issue. Huang et al. designed a primal least squares twin support vector regression (LSTSVR) [18]. Because the inequality constraints are replaced with equality ones, LSTSVR is simplified to solve two linear equations in the primal space. Hence, the training speed of LSTSVR is greatly accelerated. Therefore, LSTSVR is suitable for training large-scale data. Next, Ding and Huang exploited a least squares twin parametric insensitive support vector regression (LSTPISVR) [19]. Extensive experimental results indicate that LSTPISVR not only has faster training speed, but also has better generalization capability. Huang et al. proposed a novel regressor, i.e., sparse method for least squares twin support vector regression [20]. The proposed regressor can yield very sparse solutions. Zhang et al. developed a p-norm least square twin support vector regression, termed as PLSTSVR [21]. The parameter p is adjustable in the range of $0 < p \leq 2$ and can be automatically chosen by data. The results on UCI benchmark datasets and synthetic datasets verified the efficacy of PLSTSVR. Recently, in order to further promote the prediction performance of LSTSVR, Gu et al. investigated a least squares twin projection support vector regression, named LSTPSVR [22]. By minimizing the variance of the projected data, LSTPSVR can find a suitable projection axis. The results of simulation demonstrate that LSTPSVR performs better than several state-of-the-art regression models.

In real scenarios, due to the influence of measuring instrument and environment, the actual sampled data inevitably contains outliers, i.e., the sample points which seriously deviate from other observed values in the sample set [23]. In general, outliers are those sample points with larger loss. In LSTSVR, because all the training samples are support vectors (SVs), outliers will involve in determining the decision function [24,25]. As a result, the decision hyperplane will undoubtedly orient to the direction of outliers. This is the reason why LSTSVR is sensitive to outliers. If there are potential outliers in training data, the generalization capability of LSTSVR will be declined. In order to remove the potential outliers, Ye et al. proposed a localized version of least squares twin support vector machine (LSTSVM) classification via maximum one-class within-class variance, called LMWSVM for short [26]. Based on the principle that the samples containing larger noise should be assigned with smaller weights whereas the samples containing smaller noise with larger weights, Mu et al. developed a classification with noise via weighted LSTSVM. The simulation results disclose that the developed classification model lessens the influence of noise to a certain extent [27]. Tanveer et al. investigated a robust energy-based LSTSVM, and they employed energy parameters to reduce the effect of noise and outliers [28]. However, all the aforementioned models ignore the influence of the margin distribution information on regression. In fact, the margin distribution information is critical to the generalization capability of regression model [29,30].

To conclude, for one thing, LSTSVR is sensitive to potential outliers in the sampled data. For another, LSTSVR ignores the impact of the margin distribution information on regression model. Therefore, it is of great significance to suppress the influence of outliers and take the margin distribution information into account. In this paper, we investigate an isolation forest-based least squares twin margin distribution support vector regression, named IFLSTMDSVR for short. The main contributions of our work are summarized as below.

1) In order to effectively remove the influence of potential outliers on regression, we first adopt the isolation forest approach to explicitly isolate outliers instead of profile normal samples, and then we construct a reasonable diagonal impact factor matrix based on the anomaly scores.

2) In order to further improve the generalization capability of LSTSVR, we integrate the margin distribution information, which is characterized by the margin mean and the margin variance, into the objective functions of our IFLSTMDSVR, respectively.

The rest of this paper is organized as follows. Section 2 briefly reviews the least squares twin support vector regression (LSTSVR) in linear and nonlinear cases. Section 3 describes our work in detail. The experimental results and analyses are presented in Section 4. Section 5 draws the conclusion of our work.

2. **Least Squares Twin Support Vector Regression.** Suppose a training set is represented by $T = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$, where $\boldsymbol{x}_i \in R^m$ is the input with m attributions, $y_i \in R$ is the output, and n is the number of the training sets. Then, for simplicity, the input matrix and output vector are denoted as $\boldsymbol{A} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^{\mathrm{T}} \in R^{n\times m}$ and $\boldsymbol{Y} = [y_1, y_2, \ldots, y_n]^{\mathrm{T}} \in R^n$, respectively.

The aim of the least squares twin support vector regression (LSTSVR) is to search a pair of non-parallel bound functions, i.e., the down-bound function $f_1(\boldsymbol{x}) = \boldsymbol{\omega}_1^{\mathrm{T}}\boldsymbol{x} + b_1$ and the up-bound function $f_2(\boldsymbol{x}) = \boldsymbol{\omega}_2^{\mathrm{T}}\boldsymbol{x} + b_2$, where $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2 \in R^m$ are weight vectors, $b_1, b_2 \in R$ are biases and T stands for transpose.

As for linear regression, the primal optimization problems are listed as follows [18]:

$$\begin{aligned} \min_{\boldsymbol{\omega}_1, b_1} \quad & \frac{1}{2}||\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e})||^2 + \frac{C_1}{2}\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} \\ \text{s.t.} \quad & \boldsymbol{Y} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e}) = \varepsilon_1\boldsymbol{e} - \boldsymbol{\xi} \end{aligned} \tag{1}$$

and

$$\begin{aligned} \min_{\boldsymbol{\omega}_2, b_2} \quad & \frac{1}{2}||\boldsymbol{Y} + \varepsilon_2\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_2 + b_2\boldsymbol{e})||^2 + \frac{C_2}{2}\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{\eta} \\ \text{s.t.} \quad & (\boldsymbol{A}\boldsymbol{\omega}_2 + b_2\boldsymbol{e}) - \boldsymbol{Y} = \varepsilon_2\boldsymbol{e} - \boldsymbol{\eta} \end{aligned} \tag{2}$$

where $||\cdot||$ stands for the 2-norm, $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ are non-negative slack vectors, $C_1, C_2 > 0$ are penalty factors, $\varepsilon_1, \varepsilon_2 > 0$ are insensitive loss parameters, and $\boldsymbol{e}$ is a unit column vector with proper dimensions.

The optimal solutions of Equations (1) and (2) are

$$\boldsymbol{u}_1 = \left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{f} \tag{3}$$

and

$$\boldsymbol{u}_2 = (\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G})^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{h} \tag{4}$$

where $\boldsymbol{G} = [\ \boldsymbol{A} \quad \boldsymbol{e}\ ]$, $\boldsymbol{f} = \boldsymbol{Y} - \varepsilon_1\boldsymbol{e}$, $\boldsymbol{h} = \boldsymbol{Y} + \varepsilon_2\boldsymbol{e}$, $\boldsymbol{u}_1 = \left[\ \boldsymbol{\omega}_1^{\mathrm{T}} \quad b_1\ \right]^{\mathrm{T}}$ and $\boldsymbol{u}_2 = \left[\ \boldsymbol{\omega}_2^{\mathrm{T}} \quad b_2\ \right]^{\mathrm{T}}$.

Further, to avoid the ill-conditioned matrix, a regularization term $\vartheta\boldsymbol{I}$ is intentionally added to Equations (3) and (4), respectively.

Hence, we have

$$\boldsymbol{u}_1 = \left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G} + \vartheta\boldsymbol{I}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{f} \tag{5}$$

and

$$\boldsymbol{u}_2 = \left(\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G} + \vartheta\boldsymbol{I}\right)^{-1}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{h} \tag{6}$$

where $\vartheta = 10^{-6}$ and $\boldsymbol{I}$ is a unit matrix with proper dimensions.

Then, the regression function in linear case is estimated by

$$f(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2)^{\mathrm{T}}\boldsymbol{x} + \frac{1}{2}(b_1 + b_2) \tag{7}$$

As for nonlinear regression, the training samples are mapped into a higher dimensional (maybe infinite) feature space by the kernel function $\boldsymbol{K}(\cdot,\cdot)$. In practice, the radial basis function (RBF) kernel is widely used. In this case, the down-bound and up-bound

functions transform into $f_1(\boldsymbol{x}) = \boldsymbol{K}\left(\boldsymbol{x}^{\mathrm{T}}, \boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_1 + b_1$ and $f_2(\boldsymbol{x}) = \boldsymbol{K}\left(\boldsymbol{x}^{\mathrm{T}}, \boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_2 + b_2$, respectively.

Let $\boldsymbol{G} = \left[\begin{array}{cc} \boldsymbol{K}\left(\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\right) & \boldsymbol{e} \end{array}\right]$, the optimal solutions can also be determined by Equations (5) and (6). Finally, the regression function in nonlinear case is estimated by

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{K}\left(\boldsymbol{x}^{\mathrm{T}}, \boldsymbol{A}^{\mathrm{T}}\right)(\boldsymbol{\omega}_1 + \boldsymbol{\omega}_2) + \frac{1}{2}(b_1 + b_2) \tag{8}$$

3. **Proposed Isolation Forest-Based Least Squares Twin Margin Distribution Support Vector Regression.** In this part, we first discuss how to separate potential outliers and normal samples based on the isolation forest approach developed from Liu et al. Then, we consider the impact of the margin distribution information on improving the generalization capability of existing LSTSVR models. Finally, we present our work in detail.

3.1. **Isolation forest.** Isolation forest (iForest) is an effective tool for detecting outliers. Even if there are no outliers in the training set, iForest can also work well. Different from most existing statistical approaches, iForest isolates outliers instead of profiling normal samples.

By constructing isolation tree (iTree), each sample can be recursively partitioned. Because outliers are more susceptible to isolation than normal samples, they are isolated closer to the root of iTree. The iForest approach only needs to select two parameters: the number of trees t and the size of sub-sampling ψ [25].

The process of detecting outliers using iForest includes two different phases. The aim of the first phase is to construct iTrees based on the sub-samples of the training set. In the second phase, each sample is assigned with an anomaly score according to the iTrees constructed in the first phase.

The anomaly score s_i of each sample $\boldsymbol{x}_i$ is defined as follows [25]:

$$s_i = 2^{-E(h(\boldsymbol{x}_i))/c(n)} \tag{9}$$

where $h(\boldsymbol{x}_i)$ is the path length of sample $\boldsymbol{x}_i$, $E(h(\boldsymbol{x}_i))$ is the mean of $h(\boldsymbol{x}_i)$ in a group of iTrees, and $c(n)$ is the mean of $h(\boldsymbol{x}_i)$ given n (n is the number of samples).

Here, $c(n)$ is used to normalize $h(\boldsymbol{x}_i)$, and it is defined as follows:

$$c(n) = 2H(n-1) - \frac{2(n-1)}{n} \tag{10}$$

where the harmonic number $H(n-1)$ can be estimated by $\ln(n-1) + 0.5772156649$.

Based on Equation (9), we can make the following assessments: (a) if samples return s_i very close to 1, then they are definitely outliers, (b) if samples have s_i much smaller than 0.5, then they are quite safe to be treated as normal samples, and (c) if all the samples return $s_i \approx 0.5$, then the entire sample does not really contain any distinct outlier [25].

Then, we define the following impact factor:

$$IF_i = 1 - s_i \tag{11}$$

where IF_i, $i = 1, 2, \ldots, n$ is the impact factor of sample $\boldsymbol{x}_i$.

The research of Liu et al. has proved that samples can be identified as outliers when $s_i \geq 0.6$ [25], because the greater anomaly score s_i than 0.5, the more likely it is an outlier. Furthermore, in order to separate outliers and normal samples more efficiently, in this paper, we set the critical anomaly score s_i as 0.65.

Finally, the diagonal impact factor matrix Σ is defined as follows:

$$\Sigma = \begin{cases} 10^{-6}, & \text{if } s_i \geq 0.65 \\ IF_i, & \text{otherwise} \end{cases} \tag{12}$$

3.2. **Margin distribution.** The research of Gao and Zhou [29] indicates that the margin distribution information greatly influences the generalization capability of regression model. In general, the margin distribution information is characterized by the first-order and second-order statistical properties of samples. In this paper, inspired by the work of Cheng and Wang [30], we adopt the margin mean and margin variance to measure the margin distribution information. The margin mean $\bar{\mu}_i$ and the margin variance $\hat{\mu}_i$ of samples are measured as follows:

$$\bar{\mu}_i = \frac{1}{n}\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{\omega}_i + b_i\boldsymbol{e}), \quad i = 1, 2 \tag{13}$$

$$\hat{\mu}_i = \frac{1}{n^2}\left[n(\boldsymbol{A}\boldsymbol{\omega}_i + b_i\boldsymbol{e})^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{\omega}_i + b_i\boldsymbol{e}) - (\boldsymbol{A}\boldsymbol{\omega}_i + b_i\boldsymbol{e})^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{\omega}_i + b_i\boldsymbol{e})\right], \quad i = 1, 2 \tag{14}$$

Here, the meaning of symbols $\boldsymbol{A}$, $\boldsymbol{Y}$, $\boldsymbol{e}$, n, $\boldsymbol{\omega}_i$, b_i, $i = 1, 2$ is the same as those in Equations (1) and (2).

3.3. **Our work.** In this part, we present our isolation forest-based least squares twin margin distribution support vector regression, i.e., IFLSTMDSVR, in linear and nonlinear cases, respectively.

3.3.1. *Linear case.* In linear case, the primal optimization problems of IFLSTMDSVR are constructed as follows:

$$\begin{aligned} \min_{\boldsymbol{\omega}_1, b_1} \quad & \frac{1}{2}[\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e})]^{\mathrm{T}}\Sigma[\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e})] + \frac{\lambda_1}{2}\hat{\mu}_1 - \lambda_3\bar{\mu}_1 \\ & + \frac{1}{2}C_1\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} \\ \text{s.t.} \quad & \boldsymbol{Y} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e}) = \varepsilon_1\boldsymbol{e} - \boldsymbol{\xi} \end{aligned} \tag{15}$$

and

$$\begin{aligned} \min_{\boldsymbol{\omega}_2, b_2} \quad & \frac{1}{2}[\boldsymbol{Y} + \varepsilon_2\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_2 + b_2\boldsymbol{e})]^{\mathrm{T}}\Sigma[\boldsymbol{Y} + \varepsilon_2\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_2 + b_2\boldsymbol{e})] + \frac{\lambda_2}{2}\hat{\mu}_2 - \lambda_4\bar{\mu}_2 \\ & + \frac{1}{2}C_2\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{\eta} \\ \text{s.t.} \quad & (\boldsymbol{A}\boldsymbol{\omega}_2 + b_2\boldsymbol{e}) - \boldsymbol{Y} = \varepsilon_2\boldsymbol{e} - \boldsymbol{\eta} \end{aligned} \tag{16}$$

where $\Sigma \in R^{n\times n}$ is the diagonal impact factor matrix defined in Equation (12), $\bar{\mu}_i$, $\hat{\mu}_i \in R$, $i = 1, 2$ are the margin mean and the margin variance defined in Equations (13) and (14), $\lambda_1, \lambda_2 > 0$ and $\lambda_3, \lambda_4 > 0$ are penalty parameters used to counterbalance the margin variance $\hat{\mu}_i$ and the margin mean $\bar{\mu}_i$, respectively.

The first term in the objective function of Equations (15) and (16) is used to discriminate the influence of outliers and normal samples on regression, the second term and the third term are employed to simultaneously minimize the margin variance and maximize the margin mean.

Substituting the equality constraints into the objective function of Equation (15), we can obtain the following Lagrangian function:

$$\begin{aligned} L(\boldsymbol{\omega}_1, b_1) = & \frac{1}{2}\left[\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e})\right]^{\mathrm{T}}\Sigma\left[\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e})\right] \\ & + \frac{\lambda_1}{2}\hat{\mu}_1 - \lambda_3\bar{\mu}_1 + \frac{1}{2}C_1||(\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e}) - \boldsymbol{Y} + \varepsilon_1\boldsymbol{e}||^2 \end{aligned} \tag{17}$$

Next, setting the partial derivatives with respect to $\boldsymbol{\omega}_1$ and b_1, we can obtain the following Karush-Kuhn-Tucker (KKT) conditions:

$$\begin{aligned}\frac{\partial L(\boldsymbol{\omega}_1, b_1)}{\partial \boldsymbol{\omega}_1} &= -\boldsymbol{A}^{\mathrm{T}}\Sigma\left[\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e})\right] + \left(\frac{\lambda_1}{n}\boldsymbol{A}^{\mathrm{T}} - \frac{\lambda_1}{n^2}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}\right)(\boldsymbol{A}\boldsymbol{\omega}_1 \\ &\quad + b_1\boldsymbol{e}) - \frac{\lambda_3}{n}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{Y} + C_1\boldsymbol{A}^{\mathrm{T}}\left[(\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e}) - \boldsymbol{Y} + \varepsilon_1\boldsymbol{e}\right] = \boldsymbol{0}\end{aligned} \tag{18}$$

$$\begin{aligned}\frac{\partial L(\boldsymbol{\omega}_1, b_1)}{\partial b_1} &= -\boldsymbol{e}^{\mathrm{T}}\Sigma[\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - (\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e})] + \left(\frac{\lambda_1}{n}\boldsymbol{e}^{\mathrm{T}} - \frac{\lambda_1}{n^2}\boldsymbol{e}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}\right)(\boldsymbol{A}\boldsymbol{\omega}_1 \\ &\quad + b_1\boldsymbol{e}) - \frac{\lambda_3}{n}\boldsymbol{e}^{\mathrm{T}}\boldsymbol{Y} + C_1\boldsymbol{e}^{\mathrm{T}}[(\boldsymbol{A}\boldsymbol{\omega}_1 + b_1\boldsymbol{e}) - \boldsymbol{Y} + \varepsilon_1\boldsymbol{e}] = 0\end{aligned} \tag{19}$$

Further, combining Equations (18) and (19), we have

$$\begin{aligned}&-\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{e}\end{bmatrix}^{\mathrm{T}}\Sigma\left((\boldsymbol{Y} - \varepsilon_1\boldsymbol{e}) - \begin{bmatrix}\boldsymbol{A} & \boldsymbol{e}\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_1\\ b_1\end{bmatrix}\right) \\ &+\left(\frac{\lambda_1}{n}\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{e}\end{bmatrix}^{\mathrm{T}} - \frac{\lambda_1}{n^2}\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{e}\end{bmatrix}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}\right)\left(\begin{bmatrix}\boldsymbol{A} & \boldsymbol{e}\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_1\\ b_1\end{bmatrix}\right) \\ &-\frac{\lambda_3}{n}\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{e}\end{bmatrix}^{\mathrm{T}}\boldsymbol{Y} + C_1\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{e}\end{bmatrix}^{\mathrm{T}}\left(\begin{bmatrix}\boldsymbol{A} & \boldsymbol{e}\end{bmatrix}\begin{bmatrix}\boldsymbol{\omega}_1\\ b_1\end{bmatrix} - (\boldsymbol{Y} - \varepsilon_1\boldsymbol{e})\right) = \boldsymbol{0}\end{aligned} \tag{20}$$

Let $\boldsymbol{G} = \begin{bmatrix}\boldsymbol{A} & \boldsymbol{e}\end{bmatrix}$, $\boldsymbol{f} = \boldsymbol{Y} - \varepsilon_1\boldsymbol{e}$, $\boldsymbol{u}_1 = \begin{bmatrix}\boldsymbol{\omega}_1^{\mathrm{T}} & b_1\end{bmatrix}^{\mathrm{T}}$, Equation (20) is simplified into the following matrix form:

$$-\boldsymbol{G}^{\mathrm{T}}\Sigma(\boldsymbol{f} - \boldsymbol{G}\boldsymbol{u}_1) + \left(\frac{\lambda_1}{n}\boldsymbol{G}^{\mathrm{T}} - \frac{\lambda_1}{n^2}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}\right)\boldsymbol{G}\boldsymbol{u}_1 - \frac{\lambda_3}{n}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{Y} + C_1\boldsymbol{G}^{\mathrm{T}}(\boldsymbol{G}\boldsymbol{u}_1 - \boldsymbol{f}) = \boldsymbol{0} \tag{21}$$

The solution of Equation (21) is

$$\boldsymbol{u}_1 = \left[\left(C_1 + \frac{\lambda_1}{n}\right)\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G} + \boldsymbol{G}^{\mathrm{T}}\Sigma\boldsymbol{G} - \frac{\lambda_1}{n^2}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{G}\right]^{-1}\left[\left(\boldsymbol{G}^{\mathrm{T}}\Sigma + C_1\boldsymbol{G}^{\mathrm{T}}\right)\boldsymbol{f} + \frac{\lambda_3}{n}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{Y}\right] \tag{22}$$

Similarly, we can obtain

$$\boldsymbol{u}_2 = \left[\left(C_2 + \frac{\lambda_2}{n}\right)\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G} + \boldsymbol{G}^{\mathrm{T}}\Sigma\boldsymbol{G} - \frac{\lambda_2}{n^2}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Y}^{\mathrm{T}}\boldsymbol{G}\right]^{-1}\left[\left(\boldsymbol{G}^{\mathrm{T}}\Sigma + C_2\boldsymbol{G}^{\mathrm{T}}\right)\boldsymbol{g} + \frac{\lambda_4}{n}\boldsymbol{G}^{\mathrm{T}}\boldsymbol{Y}\right] \tag{23}$$

where $\boldsymbol{g} = \boldsymbol{Y} + \varepsilon_2\boldsymbol{e}$ and $\boldsymbol{u}_2 = \begin{bmatrix}\boldsymbol{\omega}_2^{\mathrm{T}} & b_2\end{bmatrix}^{\mathrm{T}}$.

Then, we can build the down-bound function $f_1(\boldsymbol{x}) = \boldsymbol{\omega}_1^{\mathrm{T}}\boldsymbol{x} + b_1$ and the up-bound function $f_2(\boldsymbol{x}) = \boldsymbol{\omega}_2^{\mathrm{T}}\boldsymbol{x} + b_2$, respectively. Finally, the regression function of IFLSTMDSVR in linear case can be estimated by Equation (7).

3.3.2. *Nonlinear case.* By introducing the kernel function $\boldsymbol{K}(\cdot,\cdot)$, the proposed IFLSTMDSVR can be easily extended to nonlinear case. In this case, the down-bound function and the up-bound function change into $f_1(\boldsymbol{x}) = \boldsymbol{K}\left(\boldsymbol{x}^{\mathrm{T}}, \boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_1 + b_1$ and $f_2(\boldsymbol{x}) = \boldsymbol{K}\left(\boldsymbol{x}^{\mathrm{T}}, \boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_2 + b_2$, respectively.

In nonlinear case, the primal optimization problems of IFLSTMDSVR are constructed as follows:

$$\begin{aligned}\min_{\boldsymbol{\omega}_1, b_1}\quad &\frac{1}{2}\left[\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - \left(\boldsymbol{K}(\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}})\boldsymbol{\omega}_1 + b_1\boldsymbol{e}\right)\right]^{\mathrm{T}}\Sigma\left[\boldsymbol{Y} - \varepsilon_1\boldsymbol{e} - \left(\boldsymbol{K}\left(\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_1\right.\right. \\ &\left.\left. + b_1\boldsymbol{e}\right)\right] + \frac{\lambda_1}{2}\hat{\mu}_1 - \lambda_3\bar{\mu}_1 + \frac{1}{2}C_1\boldsymbol{\xi}^{\mathrm{T}}\boldsymbol{\xi} \\ \text{s.t.}\quad &\boldsymbol{Y} - \left(\boldsymbol{K}\left(\boldsymbol{A}, \boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_1 + b_1\boldsymbol{e}\right) = \varepsilon_1\boldsymbol{e} - \boldsymbol{\xi}\end{aligned} \tag{24}$$

and

$$\begin{aligned}\min_{\boldsymbol{\omega}_2,b_2} \quad & \frac{1}{2}\left[\boldsymbol{Y}+\varepsilon_2\boldsymbol{e}-\left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_2+b_2\boldsymbol{e}\right)\right]^{\mathrm{T}}\Sigma\left[\boldsymbol{Y}+\varepsilon_2\boldsymbol{e}-\left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_2\right.\right. \\ & \left.\left.+b_2\boldsymbol{e}\right)\right]+\frac{\lambda_2}{2}\hat{\mu}_2-\lambda_4\bar{\mu}_2+\frac{1}{2}C_2\boldsymbol{\eta}^{\mathrm{T}}\boldsymbol{\eta} \\ \text{s.t.} \quad & \left(\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right)\boldsymbol{\omega}_2+b_2\boldsymbol{e}\right)-\boldsymbol{Y}=\varepsilon_2\boldsymbol{e}-\boldsymbol{\eta}\end{aligned} \tag{25}$$

Similar to the derivation of linear case, let $\boldsymbol{G}=\left[\begin{array}{cc}\boldsymbol{K}\left(\boldsymbol{A},\boldsymbol{A}^{\mathrm{T}}\right) & \boldsymbol{e}\end{array}\right]$, the optimal solutions of Equations (24) and (25) can also be obtained by Equations (22) and (23), respectively. Accordingly, the regression function can be estimated by Equation (8).

3.4. **The pseudocode of IFLSTMDSVR.** The whole process of the proposed IFLSTMDSVR in linear case is summarized in Algorithm 1.

Algorithm 1 The procedure of IFLSTMDSVR in linear case: high-level summary

Input: the training set T
Output: the regression function $f(\boldsymbol{x})$
Step 1: Set the penalty factors C_1, C_2, the penalty parameters λ_1, λ_2, λ_3, λ_4, the insensitive loss parameters ε_1, ε_2, the number of trees t and the size of sub-sampling ψ;
Step 2: Compute the anomaly score s_i of each sample $\boldsymbol{x}_i$ by Equation (9) and define the diagonal impact factor matrix Σ as in Equation (12);
Step 3: Measure the margin mean $\bar{\mu}_i$ and the margin variance $\hat{\mu}_i$ as in Equations (13) and (14), respectively;
Step 4: Construct the primal problems as in Equations (15) and (16), respectively;
Step 5: Compute the optimal solution $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ by Equations (22) and (23), respectively;
Step 6: Estimate the regression function $f(\boldsymbol{x})$ by Equation (7).

Algorithm 1 can be easily generalized to nonlinear case, which is omitted here.

4. **Results and Analyses.** In this part, we first describe the experimental design and parameter setting. Then, we launch experiments on UCI benchmark datasets and synthetic test function to demonstrate the superiorities of our work.

4.1. **Experimental design.** In order to show the strengths of the proposed IFLSTMDSVR, we compared it with four state-of-the-art regression algorithms, i.e., weighted least squares SVR (WLSSVR) [31], TSVR [11], LSTSVR [18] and LSTPISVR [19]. The following three criteria, i.e., the root mean square error (RMSE), mean absolute error (MAE) and ET, are adopted to comprehensively assess the prediction performance of all algorithms:

$$\mathrm{RMSE}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i-\hat{y}_i\right)^2} \tag{26}$$

$$\mathrm{MAE}=\frac{1}{n}\sum_{i=1}^{n}\left|y_i-\hat{y}_i\right| \tag{27}$$

$$\mathrm{ET}=\frac{\sum_{i=1}^{n}\left(y_i-\hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i-\bar{y}\right)^2} \tag{28}$$

where n is the number of samples, y_i is the actual value, $\hat{y}_i$ is the predicted value of y_i, and $\bar{y}$ is the mean of y_i given n.

In general, the smaller RMSE, the better generalization capability; the smaller MAE, the smaller prediction error; the smaller ET, the better consistency of predicted value with actual value.

Furthermore, we also report the elapsed training time of all algorithms. All experiments are carried out in the MATLAB R2015a platform on a PC with 3.40 GHz Intel®Core™i5-7500 CPU and 8GB RAM, and the results are averaged in 20 independent trials. In addition, we adopt the 10-fold cross validation to guarantee that the results are more reliable.

4.2. **Parameter setting.** It is well-known that parameter setting is closely related to the performance of algorithms. Therefore, setting appropriate parameters is critical. In our work, we adopt the grid search technique to determine the optimal parameters of all algorithms.

For all algorithms, because the setting of insensitive loss parameters ε, ε_1 and ε_2 cannot influence the generalization capability greatly [10-24,32], we set the insensitive parameters as the same fixed values, i.e., $\varepsilon = \varepsilon_1 = \varepsilon_2 = 0.1$. In addition, to keep the comparison fair and save time on parameter optimization, we set the same penalty factors C, C_1, C_2 and they are selected from the same set $\{2^i | i = -9, \ldots, 0, \ldots, 9\}$ by grid search technique. For the same reason, in LSTPISVR, we set the same parametric values ($v_1 = v_2$) and they are selected from the same set as the penalty factor. Meanwhile, in our IFLSTMDSVR, we set $\lambda_1 = \lambda_2$, $\lambda_3 = \lambda_4$ and they are also selected from the same set as the penalty factor. Besides, we set the number of trees as $t = 100$ and the size of sub-sampling as $\psi = 256$, respectively [25].

In addition, for nonlinear test, we adopt the RBF kernel and the width parameter σ is selected from the set $\{2^i | i = -4, -3, -2, -1, 0, 1, 2, 3, 4\}$.

4.3. **Experiments on UCI benchmark datasets.** Table 1 lists the seven UCI benchmark datasets used in our experiments and they can be accessed from UCI machine learning repository. The scale of the dataset varies from 103 to 1503 and all the datasets are linearly normalized to the closed interval within $[0, 1]$.

TABLE 1. The UCI benchmark datasets used in our experiments

Dataset	**Number of training samples**	**Attributions**
Slump	103	8
Yacht	308	7
Stock	315	12
Boston	506	14
Enb_2012	768	7
Concrete	1030	9
Airfoil	1503	6

The average RMSE, MAE, ET and training time in 20 independent trials on UCI benchmark datasets are summarized in Table 2. We indicate the best results with bold for clarity.

Table 2 demonstrates that, on most benchmark datasets, the proposed IFLSTMDSVR performs better than the other four algorithms in terms of RMSE, MAE and ET. This can be explained by the fact that, for one thing, iTree isolates potential outliers closer to the root of the tree as compared to normal samples, this unique characteristic allows iForest to detect outliers effectively, and the strategy of assigning potential outliers with tiny impact factors lightens their influences on regression; for another, the integration of

TABLE 2. The results of RMSE, MAE, ET and training time on UCI benchmark datasets

Datasets	Algorithms	RMSE	MAE	ET	Training time (s)
Slump	WLSSVR	7.5681	5.9780	0.7661	3.6369×10^{-2}
	TSVR	7.8536	6.4740	0.6390	0.0401
	LSTSVR	7.5417	6.4029	0.7130	$\mathbf{0.1371 \times 10^{-3}}$
	LSTPISVR	7.7147	5.9678	**0.6168**	0.2725×10^{-3}
	IFLSTMDSVR	**7.1260**	**5.8378**	0.7474	1.8240×10^{-3}
Yacht	WLSSVR	9.1784	6.8724	0.3585	0.3743
	TSVR	9.1124	7.2665	0.3736	0.1872
	LSTSVR	8.9466	7.3464	0.3815	0.2524×10^{-3}
	LSTPISVR	8.7573	**6.6778**	0.3657	$\mathbf{0.2141 \times 10^{-3}}$
	IFLSTMDSVR	**8.6188**	6.9471	**0.3577**	3.2350×10^{-3}
Stock	WLSSVR	**0.1036**	0.0887	0.7338	0.3841
	TSVR	0.1140	0.0887	0.7555	0.1722
	LSTSVR	0.1140	0.0883	0.7378	$\mathbf{0.1704 \times 10^{-3}}$
	LSTPISVR	0.1176	0.0915	0.6737	0.1909×10^{-3}
	IFLSTMDSVR	0.1121	**0.0851**	**0.6162**	3.4840×10^{-3}
Boston	WLSSVR	4.7352	3.2043	0.2953	1.1425
	TSVR	4.7670	3.3984	0.2997	0.4545
	LSTSVR	**4.7334**	3.3697	0.2884	0.3288×10^{-3}
	LSTPISVR	4.9217	3.3323	**0.2678**	$\mathbf{0.3237 \times 10^{-3}}$
	IFLSTMDSVR	4.7541	**3.1894**	0.2723	7.4940×10^{-3}
Enb_2012	WLSSVR	3.0429	2.0728	0.0844	4.2634
	TSVR	2.9331	2.0780	0.0918	1.3626
	LSTSVR	2.9599	2.0969	0.0838	$\mathbf{0.2974 \times 10^{-3}}$
	LSTPISVR	**2.8636**	2.0583	0.0884	0.3390×10^{-3}
	IFLSTMDSVR	2.8712	**2.0346**	**0.0802**	1.5227×10^{-2}
Concrete	WLSSVR	10.5856	**8.0901**	0.4049	12.9903
	TSVR	10.5927	8.2907	0.3956	2.9822
	LSTSVR	10.6183	8.3261	0.3869	0.3646×10^{-3}
	LSTPISVR	10.3831	8.4270	0.3874	$\mathbf{0.3341 \times 10^{-3}}$
	IFLSTMDSVR	**10.2873**	8.1053	**0.3783**	2.7626×10^{-2}
Airfoil	WLSSVR	4.7754	3.7410	0.4955	43.9760
	TSVR	4.7795	3.7423	0.4942	6.2883
	LSTSVR	4.7906	3.7668	0.4814	$\mathbf{0.3797 \times 10^{-3}}$
	LSTPISVR	4.7952	**3.6634**	0.4787	0.5110×10^{-3}
	IFLSTMDSVR	**4.7295**	3.6640	**0.4723**	6.3652×10^{-2}

the margin mean and the margin variance reflects the margin distribution information of samples, which improves the generalization capability.

Table 2 also discloses that LSTSVR, LSTPISVR and IFLSTMDSVR run much faster than WLSSVR and TSVR. This is because only simple linear equations need to be solved in the primal space instead of dual problems. However, the training time of IFLSTMDSVR is a little longer than LSTSVR and LSTPISVR. This is due to the fact that the integration of diagonal impact factor matrix and margin distribution information slightly increases the algorithmic complexity of solving linear equations. In short, except for the training speed, our IFLSTMDSVR has better generalization capability than the other four algorithms.

4.4. **Experiments on synthetic test function.** In this part, we carry out experiments on synthetic test function to check the performance of different regression algorithms in nonlinear case. We adopt the following synthetic test function:

$$y_i = \mathrm{sinc}(x_i) + n_i = \frac{\sin x_i}{x_i} + n_i, \quad x_i \in [-3\pi, +3\pi] \tag{29}$$

where x_i is the input, y_i is the output and n_i is the noise.

In order to test the anti-interference capability, we add the following two different types of noise in Equation (29), i.e., type A: $n_i = (0.5 - |x_i|/8\pi) \times \zeta_i$, $\zeta_i \sim U[-0.5, 0.5]$ and type B: $n_i = (0.5 - |x_i|/8\pi) \times \zeta_i$, $\zeta_i \sim N\left[0, 0.25^2\right]$, where $\zeta_i \sim U[-0.5, 0.5]$ and $\zeta_i \sim N\left[0, 0.25^2\right]$ means that the variables ζ_i are subjected to uniform distribution within the closed interval $[-0.5, 0.5]$ and normal distribution with zero mean and variance 0.25^2, respectively.

We randomly generate 37 training samples and 100 testing samples mixed with two different types of noise. In addition, we artificially add three different outliers in Equation (29). Table 3 provides the average RMSE, MAE, ET and training time in 20 independent trials on synthetic test function. Once again, the best results are indicated with bold for clarity.

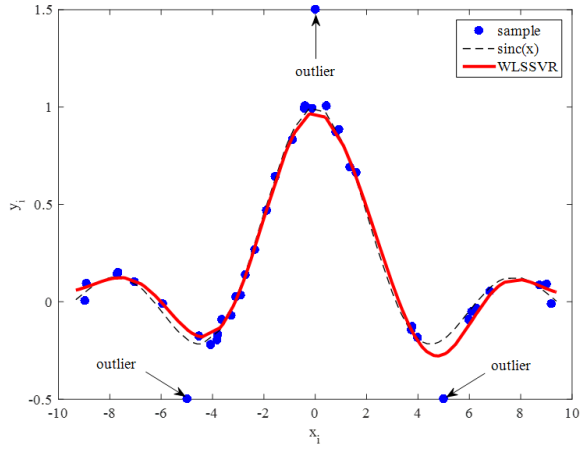
Table 3. The results of RMSE, MAE, ET and training time on synthetic test function

Noise type	**Algorithms**	**RMSE**	**MAE**	**ET**	**Training time (s)**
type A	WLSSVR	0.0289	**0.0152**	**0.0057**	1.8677×10^{-2}
	TSVR	0.0478	0.0407	0.0187	2.1497×10^{-2}
	LSTSVR	0.0423	0.0354	0.0162	$\mathbf{6.3200 \times 10^{-4}}$
	LSTPISVR	0.0389	0.0311	0.0141	2.6646×10^{-3}
	IFLSTMDSVR	**0.0270**	0.0286	0.0072	2.9640×10^{-3}
type B	WLSSVR	0.0252	0.0200	0.0066	2.0330×10^{-2}
	TSVR	0.0559	0.0416	0.0231	2.2066×10^{-2}
	LSTSVR	0.0333	0.0337	0.0136	$\mathbf{1.8250 \times 10^{-3}}$
	LSTPISVR	0.0476	0.0330	0.0127	2.5130×10^{-3}
	IFLSTMDSVR	**0.0245**	**0.0170**	**0.0062**	2.9420×10^{-3}

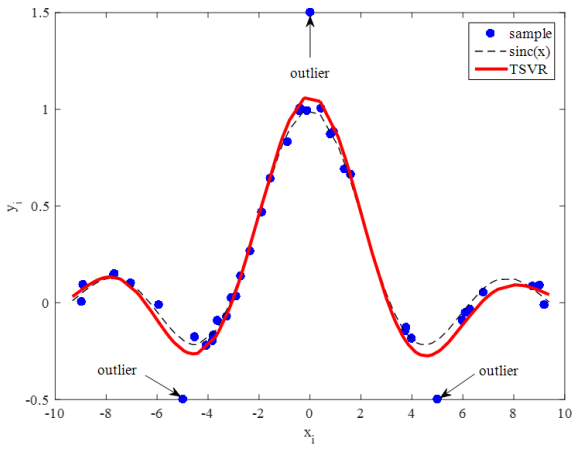
We can see clearly from Table 3 that our IFLSTMDSVR has similar or better performance compared with the other four algorithms, especially in the case of normally-distributed noise. Therefore, the proposed IFLSTMDSVR has stronger anti-interference capability and better generalization capability. This is ascribed to the tricks of using the iForest approach to reduce the influence of anomalies and introducing the margin distribution information of samples.

Furthermore, from Table 3, we can find that LSTSVR costs the least training time. The underlying cause is that LSTSVR reduces to solve two small-scale linear equations, whereas WLSSVR and TSVR should solve one large-scale linear equation and two small-scale quadratic programming problems, respectively. However, because the diagonal impact factor matrix and the margin distribution information are integrated into the objective functions of the proposed IFLSTMDSVR, the training time of IFLSTMDSVR slightly increases compared with LSTSVR and LSTPISVR.

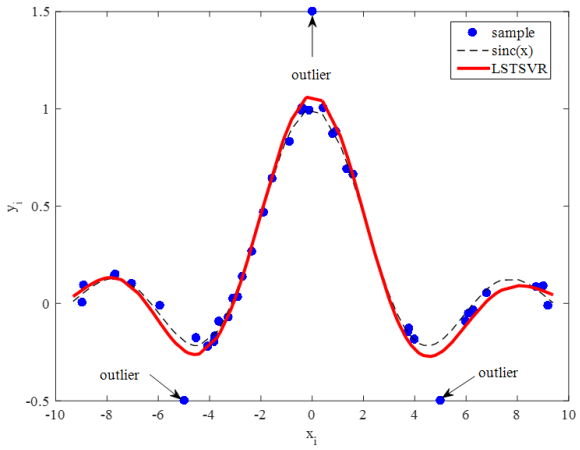
Figures 1 and 2 depict the fitting curves estimated by different regression algorithms on the synthetic test function disturbed by artificial outliers and different types of noise. We can see clearly from Figures 1 and 2 that the fitting curve of IFLSTMDSVR approximates



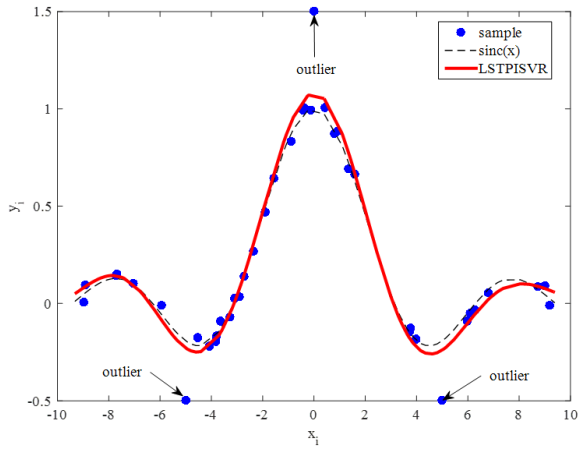
(a) WLSSVR



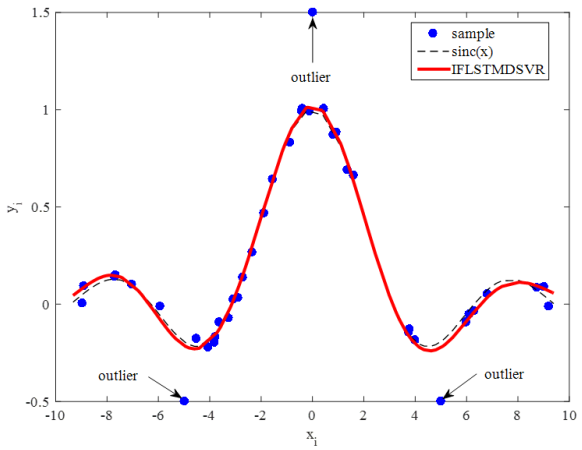
(b) TSVR



(c) LSTSVR

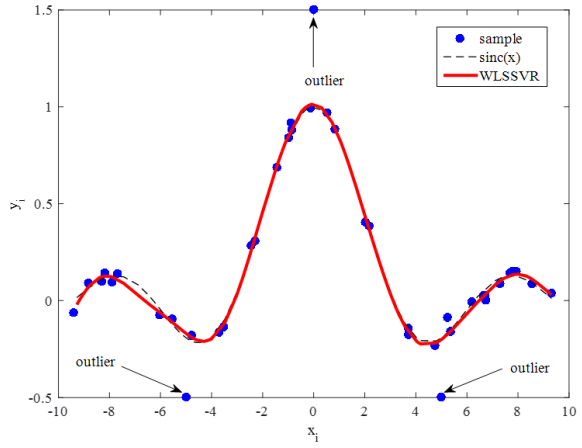


(d) LSTPISVR

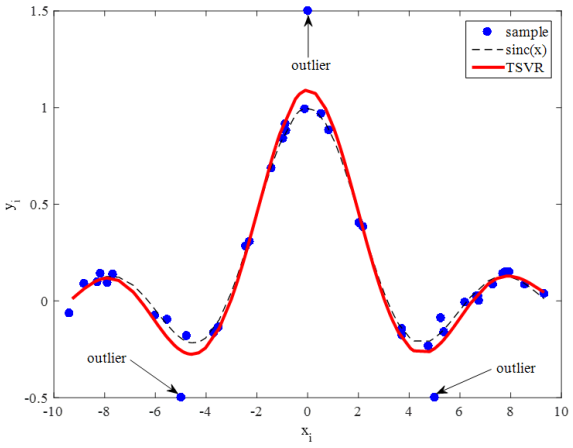


(e) IFLSTMDSVR

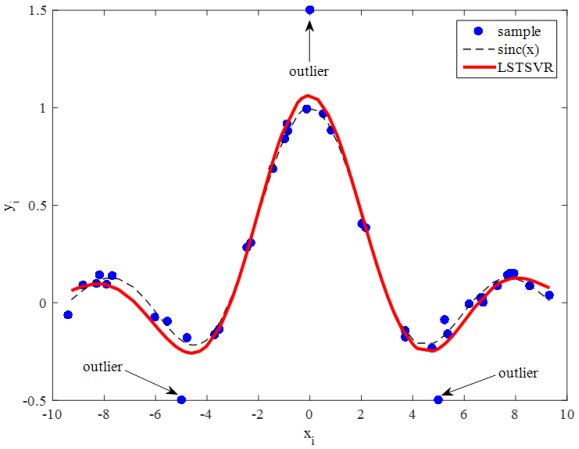
FIGURE 1. Fitting capacity of different algorithms under uniformly-distributed noise



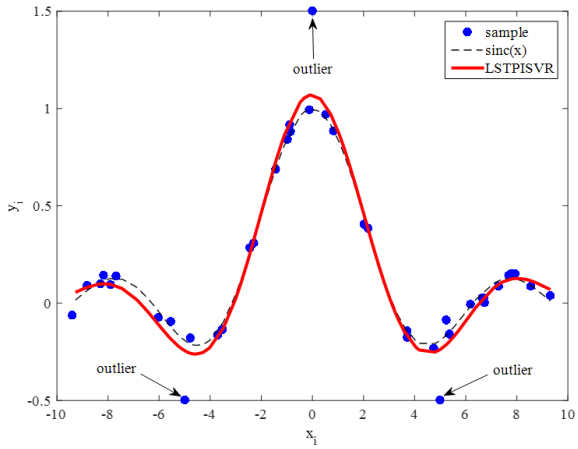
(a) WLSSVR



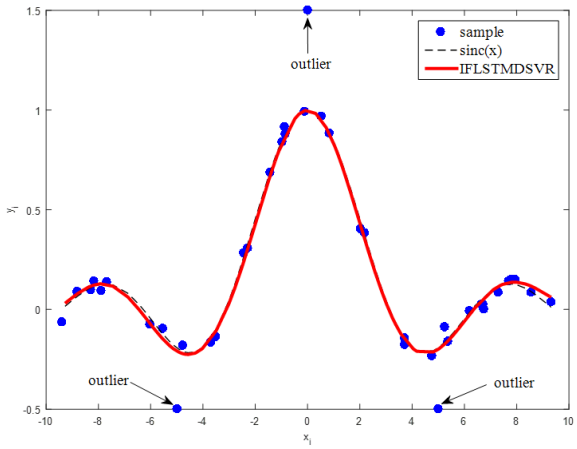
(b) TSVR



(c) LSTSVR



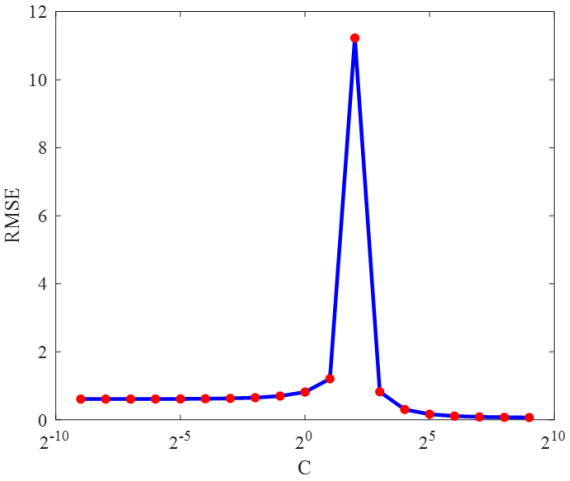
(d) LSTPISVR



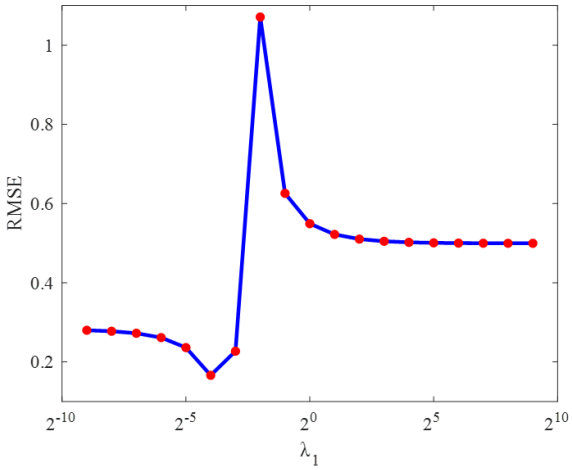
(e) IFLSTMDSVR

FIGURE 2. Fitting capacity of different algorithms under normally-distributed noise

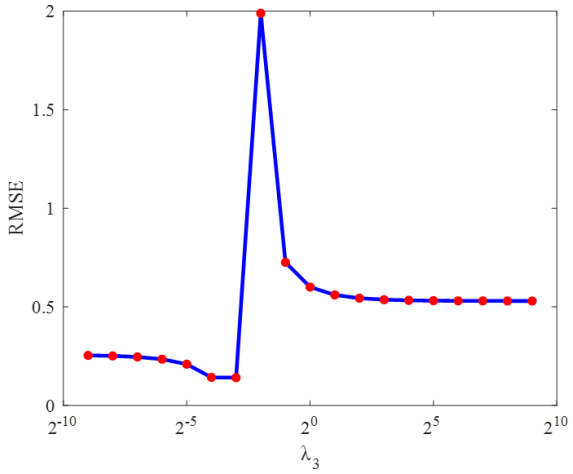
to the real synthetic test function most, regardless of the noise distribution and location of outliers. The reason is that our IFLSTMDSVR utilizes the iForest approach to remove the influence of noise and potential outliers on regression. This guarantees that the proposed IFLSTMDSVR is more robust to noise and outliers than other algorithms.



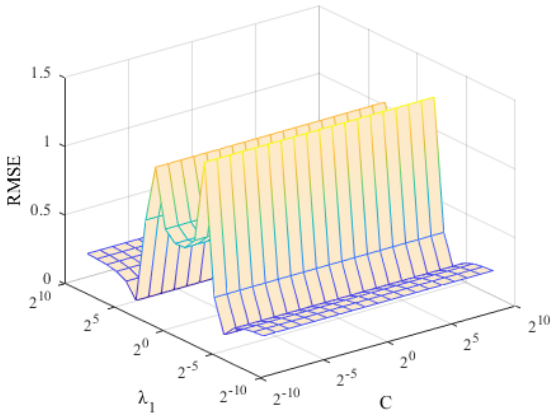
(a) C versus RMSE



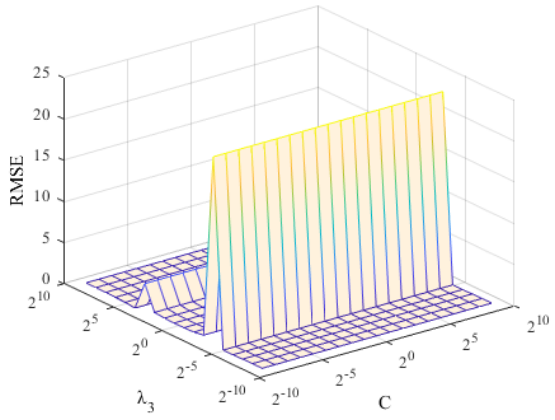
(b) λ_1 versus RMSE



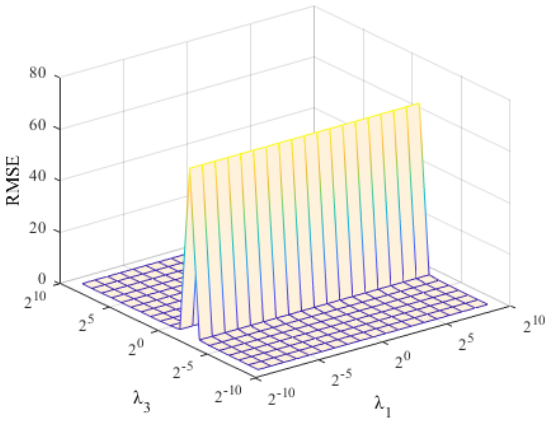
(c) λ_3 versus RMSE



(d) C and λ_1 versus RMSE



(e) C and λ_3 versus RMSE



(f) λ_1 and λ_3 versus RMSE

FIGURE 3. The relationship of different parameter settings versus RMSE

In addition, because WLSSVR also selects appropriate weights for each sample, the fitting result of WLSSVR is also satisfactory. By contrast, the fitting curves estimated by TSVR, LSTSVR and LSTPISVR orient towards anomalies and get worse. This is because anomalies involve in determining the decision hyperplane, which forces the fitting curve toward the orientation of anomalies.

In our work, we set different penalty parameters λ for the margin variance and the margin mean. Next, we briefly discuss the relationship of C and λ versus RMSE in the case of normally-distributed noise and the results are illustrated in Figure 3 (Note that the remaining parameters are fixed).

Figure 3 implies that the RMSE of the proposed IFLSTMDSVR stabilizes on the premise of setting smaller or larger λ and C. The relationship of C and λ versus MAE or ET is similar to RMSE, which is omitted here. Hence, setting appropriate C and λ is critical to generalization capability.

To sum up, the IFLSTMDSVR developed in this paper has significant robustness to noise and outliers, and it also improves generalization capability. Furthermore, our IFLSTMDSVR stabilizes when given relatively small or large penalty parameters.

5. Conclusion.

In this paper, we developed an isolation forest-based least squares twin margin distribution support vector regression (IFLSTMDSVR). The strategy of assigning potential outliers with tiny impact factors using the iForest approach effectively removes their influences on regression, and the introduction of the margin distribution information in the form of the margin mean and margin variance promotes the generalization capability. The experimental results on several UCI benchmark datasets and synthetic test function validate the superiorities of our IFLSTMDSVR in terms of generalization capability and anomaly insensitivity.

However, the iForest approach may be not the best choice to remove anomalies, and the critical anomaly score 0.65 may be not the best choice. Besides, the grid search technique is inefficient. We hope these issues can be settled in our future work.

Acknowledgment. This research was supported by the National Natural Science Foundation of China under Grant Numbers 31771680 and 61773182.

REFERENCES

[1] C. Cortes and V. Vapnik, Support vector networks, *Machine Learning*, vol.20, no.3, pp.273-297, 1995.

[2] V. N. Vapnik, An overview of statistical learning theory, *IEEE Trans. Neural Networks*, vol.10, no.5, pp.988-999, 1999.

[3] J. Ruan, X. Wang and Y. Shi, Developing fast predictors for large-scale time series using fuzzy granular support vector machines, *Applied Soft Computing*, vol.13, no.9, pp.3981-4000, 2013.

[4] J. Fang, F. Pan and B. Gu, Twin support vector regression based on fruit fly optimization algorithm, *International Journal of Innovative Computing, Information and Control*, vol.15, no.5, pp.1851-1864, 2020.

[5] G. Shen, B. Gu and F. Pan, Twin support vector regression based on grey wolf optimization algorithm, *Journal of Nanjing University of Science and Technology*, vol.44, no.2, pp.202-208, 2020.

[6] J. Ruan, Y. Shi and J. Yang, Forest fires burned area prediction based on support vector machines with feature selection, *ICIC Express Letters*, vol.5, no.8(A), pp.2597-2603, 2011.

[7] Z. Hu, Y. Xu, X. Zhao, J. He and Y. Zhou, Multi-feature selection tracking based on support vector machine, *Journal of Applied Sciences*, vol.33, no.5, pp.502-517, 2015.

[8] L. Gan and M. Yang, Pedestrian detection method based on ensemble SVM classifier, *Computer Engineering and Applications*, vol.55, no.7, pp.194-198, 2019.

[9] H. Yasin, R. Caraka, A. Hoyyi and Sugito, Stock price modeling using localized multiple kernel learning support vector machine, *ICIC Express Letters, Part B: Applications*, vol.11, no.4, pp.333-339, 2020.

[10] Jayadeva, R. Khemchandani and S. Chandra, Twin support vector machines for pattern classification, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol.29, no.5, pp.905-910, 2007.
[11] X. Peng, TSVR: An efficient twin support vector machine for regression, *Neural Networks*, vol.23, no.3, pp.365-372, 2010.
[12] X. Peng, Efficient twin parametric insensitive support vector regression model, *Neurocomputing*, vol.79, pp.26-38, 2012.
[13] Y. Shao, C. Zhang, Z. Yang, L. Jing and N. Deng, An ε-twin support vector machine for regression, *Neural Computing & Applications*, vol.23, no.1, pp.175-185, 2013.
[14] S. Balasundaram and M. Tanveer, On Lagrangian twin support vector regression, *Neural Computing and Applications*, vol.22, pp.S257-S267, 2013.
[15] X. Peng, D. Xu and J. Shen, A twin projection support vector machine for data regression, *Neurocomputing*, vol.138, pp.131-141, 2014.
[16] Y. Xu and L. Wang, K-nearest neighbor-based weighted twin support vector regression, *Applied Intelligence*, vol.41, no.1, pp.299-309, 2014.
[17] B. Gu, J. Fang, F. Pan and Z. Bai, Fast clustering-based weighted twin support vector regression, *Soft Computing*, vol.24, no.8, pp.6101-6117, 2020.
[18] H. Huang, S. Ding and Z. Shi, Primal least squares twin support vector regression, *Journal of Zhejiang University: Science C: Computers and Electronics*, vol.14, no.9, pp.722-732, 2013.
[19] S. Ding and H. Huang, Least squares twin parametric insensitive support vector regression, *Journal of Software*, vol.28, no.12, pp.3146-3155, 2017.
[20] H. Huang, X. Wei and Y. Zhou, A sparse method for least squares twin support vector regression, *Neurocomputing*, vol.211, pp.150-158, 2016.
[21] Z. Zhang, T. Lv, H. Wang, L. Liu and J. Tan, A novel least square twin support vector regression, *Neural Processing Letters*, vol.48, no.2, pp.1187-1200, 2018.
[22] B. Gu, G. Shen, F. Pan and H. Chen, Least squares twin projection support vector regression, *International Journal of Innovative Computing, Information and Control*, vol.15, no.6, pp.2275-2288, 2020.
[23] K. Wang, J. Ma and X. Ding, Robust least square support vector regression, *Journal of Computer Applications*, vol.31, no.8, pp.2111-2114, 2011.
[24] J. Nasiri, N. Charkari and K. Mozafari, Energy-based model of least squares twin support vector machines for human action recognition, *Signal Processing*, vol.104, no.6, pp.248-257, 2014.
[25] F. Liu, K. Ting and Z. Zhou, Isolation-based anomaly detection, *ACM Trans. Knowledge Discovery from Data*, vol.6, no.1, pp.1-39, 2012.
[26] Q. Ye, C. Zhao and N. Ye, Least squares twin support vector machine classification via maximum one-class within class variance, *Optimization Methods and Software*, vol.27, no.1, pp.53-69, 2012.
[27] X. Mu, J. Li and L. Chen, Classification with noise via weighted least squares twin support vector regression, *Computer Simulation*, vol.31, no.5, pp.288-292, 2014.
[28] M. Tanveer, M. A. Khan and S. S. Ho, Robust energy-based least squares twin support vector machines, *Applied Intelligence*, vol.45, no.1, pp.174-186, 2016.
[29] W. Gao and Z. Zhou, On the doubt about margin explanation of boosting, *Artificial Intelligence*, vol.203, no.5, pp.1-18, 2013.
[30] H. Cheng and J. Wang, A novel twin large margin distribution machine, *Control and Decision*, vol.31, no.5, pp.949-952, 2016.
[31] J. A. K. Suykens, J. D. Brabanter, L. Lukas and J. Vandewalle, Weighted least squares support vector machines: Robustness and sparse approximation, *Neurocomputing*, vol.48, pp.85-105, 2002.
[32] G. Shen, B. Gu and F. Pan, Twin support vector regression based on grey wolf optimization algorithm, *Journal of Nanjing University of Science and Technology*, vol.44, no.2, pp.202-208, 2020.