Applied Intelligence (2019) 49:3606–3627
https://doi.org/10.1007/s10489-019-01465-w

An improved regularization based Lagrangian asymmetric ν-twin support vector regression using pinball loss function

Umesh Gupta [1] · Deepak Gupta [1]

Published online: 25 April 2019

Abstract
In twin support vector regression (TSVR), one can notice that the samples are having the same importance even they are laying above the up-bound and below the down-bound on the estimation function for regression problem. Instead of giving the same emphasis to the samples, a novel approach Asymmetric ν-twin support vector regression (Asy-ν-TSVR) is suggested in this context where samples are having different influences with the estimation function based on samples distribution. Inspired by this concept, in this paper, we propose a new approach as improved regularization based Lagrangian asymmetric ν-twin support vector regression using pinball loss function (LAsy-ν-TSVR) which is more effective and efficient to deal with the outliers and noise. The solution is obtained by solving the simple linearly convergent approach which reduces the computational complexity of the proposed LAsy-ν-TSVR. Also, the structural risk minimization principle is implemented to make the problem strongly convex and more stable by adding the regularization term in their objective functions. The superiority of proposed LAsy-ν-TSVR is justified by performing the various numerical experiments on artificial generated datasets with symmetric and heteroscedastic structure noise as well as standard real-world datasets. The results of LAsy-ν-TSVR compares with support vector regression (SVR), TSVR, TSVR with Huber loss (HN-TSVR) and Asy-ν-TSVR, regularization on Lagrangian TSVR (RLTSVR) for the linear and Gaussian kernel which clearly demonstrates the efficacy and efficiency of the proposed algorithm LAsy-ν-TSVR.

Keywords Support vector regression · Twin support vector regression · Huber loss function · Asymmetric-ν-TSVR · Pinball loss function · Heteroscedastic noise structure

1 Introduction

In the machine learning computational world, support vector machine (SVM) is a very popular and safe algorithm for binary classification [1]. Later, it is extended for regression problem also that is known as support vector regression (SVR) [2]. According to statistical learning theory, SVM follows the structural risk minimization (SRM) principle that solves a single large size quadratic programming problem (QPP) to get the optimal solution. SRM principle is basically for model selections which broadly explains the details of capacity control of model and maintain balance between VC dimension of function and empirical error.

SVM for regression is globally accepted due to superior forecasting performance in many research fields such as predict the popularity of online video [3], productiveness of higher education system [4], energy utilization in heat equalization [5], software enhancement effort [6], electric load [7], velocity of wind [8], flow of river [9], snow depth [10], neutral profiles from laser induced fluorescence data [11] and stock price [12]. The disadvantage of SVM is high training cost i.e. $O(m^3)$. So many significant improvements have been done by the researchers to lessen the training cost and complexity of SVM such as ν-SVR [13], SVMTorch [14], Bayesian SVR [15], geometric approach to SVR [16], active set SVR [17], heuristic training for SVR [18], smooth ε-SVR [19], fuzzy weighted support vector regression with fuzzy partition [20] etc.

A remarkable enhancement has been done in standard SVM by Jayadeva et al. [21] to propose a novel approach termed as twin support vector machine (TWSVM) which finds two non-parallel hyperplanes that are nearer to one of

✉ Deepak Gupta
deepakjnu85@gmail.com; deepak@nitap.ac.in

Umesh Gupta
er.umeshgupta@gmail.com

[1] Department of Computer Science & Engineering, National Institute of Technology Arunachal Pradesh, Yupia, India

the class either positive or negative and also sustains unit difference between each other. In comparison to SVM, TWSVM has shown good generalization ability and lesser computation time. Motivated by the concept of TWSVM, a non-parallel twin support vector regression (TSVR) is proposed by Peng [22] in which two unknown optimal regression such as ε−insensitive down- and up- bound functions are determined. TSVR has better prediction performance and accelerated training speed over the SVR [22]. Many other variants of TSVR have been suggested such as reduced TSVR [23] which applied the concept of rectangle kernels to obtain significant improvement in learning time on TSVR, weighted TSVR [24] reduces the problem of overfitting by assigned different penalties to each sample. Twin least square SVR [25] takes the concept of TSVR and least square SVR [26] to improve the prediction performance with training speed. Linearly convergent based Lagrangian TSVR [27] has been proposed to improve the generalization performance and learning speed. Unconstrained based Lagrangian TSVR [28] has been suggested to reduce the complexity of model and improve the learning speed via solving the unconstrained minimization problems. Niu et al. [29] has combined the TSVR with Huber loss function (HN-TSVR) for handling the Gaussian noise. Tanveer & Shubham [30] has proposed a new algorithm termed as regularization on Lagrangian twin support vector regression (RLTSVR) which solves the regression problem very effectively. There are many variants of SVM exists in the literature based on pinball loss function for the classification problems like Huang has applied pin ball loss function in SVM and suggested an approach as pin-SVM [31] to handle the noisy data; Huang et al. [32] has proposed sequential minimization optimization for SVM with truncated pinball loss along with its sparse version that enhances the generalization efficiency of pin-SVM; Pin-M^{3}HM [33] has improved the twin hyper-sphere SVM (THSVM) [34] using pin ball loss that avoids noise and error in very effective manner; Xu et al. [35] has proposed a new approach TWSVM with pin ball loss that indulge with quantile distance which is performed well for noisy data and related to this for more study, see [36–44]. It actually gives the active research direction in forward way.

One can notice that very few literatures are available on SVR with pinball loss function for the regression problems. In spite of considering ε-insensitive loss function, Huang has proposed a novel approach termed as asymmetric ν-tube support vector regression (Asy-ν-SVR) [45] based on ν-SVR with pinball loss function to divide the outliers asymmetrically over above and below of the tube and improved the computational complexity. Similarly, one can observe that we have assigned same penalties to every point above the up-bound and below the down-bound in TSVR. But each sample may not give same effect in order to determine the regression function. So, asymmetric ν-twin support vector regression (Asy-ν-TSVR) [35] has been suggested to give different effects on the regression function by using the pin-ball loss function. Motivated by the above studies, we propose a new approach as improved regularization based Lagrangian asymmetric ν-twin support vector regression (LAsy-ν-TSVR) using pinball loss function in this paper where the end regression function is determined by solving the linearly convergent iterative approach unlike solve the QPPs in case of SVR, TSVR, HN-TSVR and Asy-ν-TSVR. This approach reduces the computational cost of the model. Another advantage of our proposed LAsy-ν-TSVR is that it follows the SRM principle which yields the existence of global solution and improves the generalization ability. The characteristics of proposed LAsy-ν-TSVR are as follows:

- To make the problem strongly convex and find the unique global solution, 2-norm of vector of slack variables is included in the objective functions of proposed LAsy-ν-TSVR.
- Regularization terms are added in the objective functions of LAsy-ν-TSVR to implement the SRM principle which makes the model well pose.
- The solution of proposed LAsy-ν-TSVR is obtained by using the linearly convergent iterative schemes which improve the computational cost.

Further, to check the effectiveness and applicability of proposed LAsy-ν-TSVR, numerical experiments are conducted on artificial generated datasets having symmetric and asymmetric structure of noise and also on real-world benchmark datasets. The experimental results of proposed LAsy-ν-TSVR are compared with SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR in this paper.

This paper is organised as follows. Section 2 outlines briefly about SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR. The formulation of proposed LAsy-ν-TSVR is described in Section 3. In Section 4, the numerical experiments are performed on artificially generated and real-world datasets in detail. At last, conclusion and future work is presented in Section 5.

2 Background

In this paper, consider the training data as $\{(x_i, y_i)\}_{i=1}^{m}$ where i^{th} -input data sample is shown as $x_i = (x_{i1}, ..., x_{in}) \in R^n$ and $y_i \in R$ is the observed outcome of corresponding input sample. Let consider A is a $m \times n$ matrix in which m is the number of input samples and n is the number of attributes in such a way that $A = (x_1, ..., x_m)^t \in R^{m \times n}$ and $y = (y_1, ..., y_m)^t \in R^m$. The 2-norm of a vector x will be represented by $\|x\|$. The plus function x_+ is given by $\max\{0, x_i\}$ for $i = 1, ..., m$.

2.1 Support vector regression (SVR)

In the linear regression [2], the main aim is to find the optimal linear regression estimating function of the form as

$$f(x) = w^t x + b$$

where, $w \in R^n$, $b \in R$.

The formulation of linear SVR as constrained minimization problem [46] is given as

$$\min \frac{1}{2}\|w\|^2 + C(e^t\xi_1 + e^t\xi_2)$$

subject to

$$\begin{aligned} &y - (Aw + be) \le \varepsilon e + \xi_{1i}, \xi_{1i} \ge 0 \\ &(Aw + be) - y \le \varepsilon e + \xi_{2i}, \xi_{2i} \ge 0 \text{ for } i = 1, ..., m \end{aligned} \tag{1}$$

where, the vectors of slack variables $\xi_1 \ge (\xi_{11}, ..., \xi_{1m})^t$ and $\xi_2 \ge (\xi_{21}, ..., \xi_{2m})^t$; $C > 0$, $\varepsilon > 0$ are the input parameters and $e \in R^m$ is the vector of one's.

Now introduce the Lagrangian multipliers $\alpha_1 = (\alpha_{11}, ..., \alpha_{1m})^t, \beta_1 = (\beta_{11}, ..., \beta_{1m})^t$ and further apply the Karush–Kuhn–Tucker (KKT) conditions, the dual QPP of (1) is given as:

$$\begin{aligned} \min &\frac{1}{2}\sum_{i,j=1}^{m} (\alpha_{1i} - \beta_{1i})\left(x_i^t x_j\right)(\alpha_{1j} - \beta_{1j}) \\ &+ \varepsilon \sum_{i=1}^{m} (\alpha_{1i} + \beta_{1i}) - \sum_{i=1}^{m} y_i(\alpha_{1i} - \beta_{1i}) \end{aligned}$$

subject to

$$\sum_{i=1}^{m} e^t(\alpha_{1i} - \beta_{1i}) = 0, 0 \le \alpha_1, \beta_1 \le Ce. \tag{2}$$

The decision function $f(.)$ will be obtained from (2) [46] for any test data $x \in R^n$ as

$$f(x) = \sum_{i=1}^{m} (\alpha_{1i} - \beta_{1i})(x^t x_i) + b$$

For nonlinear SVR, we assume the nonlinear function of the given form as.

$f(x) = w^t\phi(x) + b$

where, $\phi(.)$ is a nonlinear mapping which consider the input space into feature space in the high dimension. The formulation of nonlinear constrained QPP [2, 46] is considered as

$$\min \frac{1}{2}\|w\|^2 + C(e^t\xi_1 + e^t\xi_2)$$

subject to

$$\begin{aligned} &y - (\varphi(x_i)w + be) \le \varepsilon e + \xi_{1i}, \xi_{1i} \ge 0 \\ &(\varphi(x_i)w + be) - y \le \varepsilon e + \xi_{2i}, \xi_{2i} \ge 0 \text{ for } i = 1, ..., m \end{aligned} \tag{3}$$

Now, the dual QPP of the primal problem (3) is determined by using the Lagrangian multipliersα_1, β_1 and further apply the KKT conditions. We get

$$\begin{aligned} \min &\frac{1}{2}\sum_{i,j=1}^{m} (\alpha_{1i} - \beta_{1i})k\left(x_i, x_j\right)(\alpha_{1j} - \beta_{1j}) \\ &+ \varepsilon \sum_{i=1}^{m} (\alpha_{1i} + \beta_{1i}) - \sum_{i=1}^{m} y_i(\alpha_{1i} - \beta_{1i}) \end{aligned}$$

subject to

$$\sum_{i=1}^{m} e^t(\alpha_{1i} - \beta_{1i}) = 0, 0 \le \alpha_1, \beta_1 \le Ce. \tag{4}$$

where, the kernel function $k(x_i, x_j) = \phi(x_i)^t\phi(x_j)$. The decision function $f(.)$ will be obtained [46] for any test data $x \in R^n$ from (4) as

$$f(x) = \sum_{i=1}^{m} (\alpha_{1i} - \beta_{1i})k(x, x_i) + b$$

2.2 Twin support vector regression (TSVR)

Twin support vector regression (TSVR) [22] is an effective approach which is influenced from TWSVM [21] to predict the two nonparallel ε-insensitive down-bound function $f_1(x) = w_1^t x + b_1$ and ε-insensitive up-bound function $f_2(x) = w_2^t x + b_2$. Here, w_1, $w_2 \in R^n$ and b_1, $b_2 \in R$ are unknowns. In linear TSVR, the regression functions are determined by solving the following QPPs in such a way:

$$\min \frac{1}{2}\|y - \varepsilon_1 e - (Aw_1 + b_1 e)\|^2 + C_1 e^t\xi_1$$

subject to

$$y - (Aw_1 + b_1 e) \ge \varepsilon_1 e - \xi_1, \xi_1 \ge 0 \tag{5}$$

and

$$\min \frac{1}{2}\|y + \varepsilon_2 e - (Aw_2 + b_2 e)\|^2 + C_2 e^t\xi_2$$

subject to

$$(Aw_2 + b_2 e) - y \ge \varepsilon_2 e - \xi_2, \xi_2 \ge 0 \tag{6}$$

where, input parameters are C_1, $C_2 > 0$ and ε_1, $\varepsilon_2 > 0$; the vectors of slack variables are ξ_1 and ξ_2.

Now introduce the Lagrangian multipliers in the above problems (5) and (6) is shown as

$$\begin{aligned} L(w_1, b_1, \xi_1, \alpha_1, \beta_1) = &\frac{1}{2}\|(y - e\varepsilon_1 - (Aw_1 + eb_1))\|^2 \\ &+ C_1 e^t\xi_1 - \alpha_1^t(y - (Aw_1 + eb_1) - e\varepsilon_1 + \xi_1) - \beta_1^t\xi_1 \end{aligned}$$

and

$$L(w_2,b_2,\xi_2,\alpha_2,\beta_2)=\frac{1}{2}\|(y+\varepsilon_2 e-(Aw_2+eb_2))\|^2 + C_2e^t\xi_2-\alpha_2^t((Aw_2+eb_2)-y-e\varepsilon_2+\xi_2)-\beta_2^t\xi_2$$

where, $\alpha_1=(\alpha_{11},...,\alpha_{1m})^t$, $\alpha_2=(\alpha_{21},...,\alpha_{2m})^t$ are the vectors of Lagrangian multipliers. After, we get the Wolfe dual QPPs of the above primal problems by using the KKT conditions as

$$\max -\frac{1}{2}\alpha_1^t S(S^tS)^{-1}S^t\alpha_1 + (y-\varepsilon_1 e)^t S(S^tS)^{-1}S^t\alpha_1-(y-\varepsilon_1 e)^t\alpha_1$$

subject to

$$0\le\alpha_1\le C_1 e \tag{7}$$

and

$$\max -\frac{1}{2}\alpha_2^t S(S^tS)^{-1}S^t\alpha_2+(y+\varepsilon_2 e)^t S(S^tS)^{-1}S^t\alpha_2 + (y+\varepsilon_2 e)^t\alpha_2$$

subject to

$$0\le\alpha_2\le C_2 e \tag{8}$$

where, $S=[A\ \ e]$ is the augmented matrix.

After solving the above pair of dual QPPs (7) and (8) for α_1 and α_2, one can derive the values as:

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (S^tS)^{-1}S^t(y-\varepsilon_1 e-\alpha_1)$$

and

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (S^tS)^{-1}S^t(y+\varepsilon_2 e+\alpha_2)$$

Then, the final estimated regression function is obtained as

$$f(x)=\frac{1}{2}(f_1(x)+f_2(x)) \tag{9}$$

In nonlinear case of TSVR, the kernel generating regression functions $f_1(x)=K(x^t,A^t)w_1+b_1$ and $f_2(x)=K(x^t,A^t)w_2+b_2$ are determined by the following QPPs as

$$\min\frac{1}{2}\|y-\varepsilon_1 e-(K(A,A^t)w_1+b_1 e)\|^2+C_1e^t\xi_1$$

subject to

$$y-(K(A,A^t)w_1+b_1 e)\ge\varepsilon_1 e-\xi_1,\ \xi_1\ge 0 \tag{10}$$

and

$$\min\frac{1}{2}\|y+\varepsilon_2 e-(K(A,A^t)w_2+b_2 e)\|^2+C_2e^t\xi_2$$

subject to

$$(K(A,A^t)w_2+b_2 e)-y\ge\varepsilon_2 e-\xi_2,\ \xi_2\ge 0 \tag{11}$$

Similarly as the linear TSVR, we get the dual QPP from the Eqs. (10) and (11)

$$\max -\frac{1}{2}\alpha_1^t T(T^tT)^{-1}T^t\alpha_1 + (y-\varepsilon_1 e)^t T(T^tT)^{-1}T^t\alpha_1-(y-\varepsilon_1 e)^t\alpha_1$$

subject to

$$0\le\alpha_1\le C_1 e \tag{12}$$

and

$$\max -\frac{1}{2}\alpha_2^t T(T^tT)^{-1}T^t\alpha_2+(y+\varepsilon_2 e)^t T(T^tT)^{-1}T^t\alpha_2 + (y+\varepsilon_2 e)^t\alpha_2$$

subject to

$$0\le\alpha_2\le C_2 e \tag{13}$$

where, $T=\begin{bmatrix}K(A,A^t) & e\end{bmatrix}$ is the augmented matrix; α_1 and α_2 are Lagrangian multipliers.

One can derive the values of w_1, w_2, b_1, b_2 from the Eqs. (12) and (13) as follows:

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (T^tT+\sigma I)^{-1}T^t(y-e\varepsilon_1-\alpha_1)$$

and

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (T^tT+\sigma I)^{-1}T^t(y+e\varepsilon_2+\alpha_2)$$

One can notice that σI is added as extra term in the matrix $(T^tT)^{-1}$ to make the matrix positive definite, where $\sigma>0$ is the small real positive value. Finally, end regression function is obtained from (9).

2.3 Twin support vector regression with Huber loss (HN-TSVR)

TSVR [22] is based on ε-insensitive loss function but fail to deal for data having Gaussian noise. Motivated by the work of [47, 48], TSVR with Huber loss function (HN-TSVR) [29] has been suggested in order to improve the generalization ability by suppress a variety of noise and outliers. Here, Huber loss function is given by

$$c(x) = \begin{cases} \frac{x^2}{2}, & if\ x \leq \varepsilon \\ \varepsilon|x| - \frac{\varepsilon^2}{2}, & otherwise \end{cases}.$$

The nonlinear HN-TSVR QPPs are as follows:

$$\min \frac{1}{2}\|y - e\varepsilon_1 - (K(A, A^t)w_1 + eb_1)\|^2 + C_1\left(\sum_{i \in U_1}\frac{1}{2}\xi_{1i}^2 + \varepsilon \sum_{i \in U_2}\left(\xi_{1i} - \frac{1}{2}\varepsilon\right)\right)$$

subject to

$$y - (K(A, A^t)w_1 + eb_1) \geq e\varepsilon_1 - \xi_1, \xi_1 \geq 0 \tag{14}$$

where, $U_1 = \{i|\ 0 \leq \xi_{1i} < \varepsilon\}$, $U_2 = \{i|\ \xi_{1i} \geq \varepsilon\}$.and

$$\min \frac{1}{2}\|y + e\varepsilon_2 - (K(A, A^t)w_2 + eb_2)\|^2 + C_2\left(\sum_{i \in V_1}\frac{1}{2}\xi_{2i}^2 + \varepsilon \sum_{i \in V_2}\left(\xi_{2i} - \frac{1}{2}\varepsilon\right)\right)$$

subject to

$$(K(A, A^t)w_2 + eb_2) - y \geq e\varepsilon_2 - \xi_2, \xi_2 \geq 0 \tag{15}$$

where, $V_1 = \{i|\ 0 \leq \xi_{2i} < \varepsilon\}$, $V_2 = \{i|\ \xi_{2i} \geq \varepsilon\}$;$\xi_1 = (\xi_{11}, ...\xi_{1m})^t$ and $\xi_2 = (\xi_{21}, ...\xi_{2m})^t$ are the slack variables; C_1, $C_2 > 0$, ε_1, $\varepsilon_2 > 0$ are parameters.

By applying the Lagrange's multipliers $\alpha_1 = (\alpha_{11}, ..., \alpha_{1m})^t$, $\alpha_2 = (\alpha_{21}, ..., \alpha_{2m})^t$, the dual formulation of problem (14) and (15) are determined as

$$\min \frac{1}{2}{\alpha_1}^t S(S^tS)^{-1}S^t\alpha_1 - (y - \varepsilon_1 e)^t S(S^tS)^{-1}S^t\alpha_1 + (y - \varepsilon_1 e)^t\alpha_1 + \frac{1}{2C_1}{\alpha_1}^t\alpha_1$$

subject to:

$$0 \leq \alpha_1 \leq C_1\varepsilon_1 e \tag{16}$$

and

$$\min \frac{1}{2}{\alpha_2}^t S(S^tS)^{-1}S^t\alpha_2 + (y + \varepsilon_2 e)^t S(S^tS)^{-1}S^t\alpha_2 - (y + \varepsilon_2 e)^t\alpha_2 + \frac{1}{2C_2}{\alpha_2}^t\alpha_2$$

subject to:

$$0 \leq \alpha_2 \leq C_2\varepsilon_2 e \tag{17}$$

where, $S = \begin{bmatrix} K(A, A^t) & e \end{bmatrix}$.

The value of corresponding w_1, w_2, b_1, b_2 are

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (S^tS)^{-1}S^t(y - e\varepsilon_1 - \alpha_1)$$

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (S^tS)^{-1}S^t(y + e\varepsilon_2 - \alpha_2)$$

Finally, the end regression function can be obtained as similar to (9).

2.4 Asymmetric ν-twin support vector regression (Asy-ν-TSVR)

Asymmetric ν-twin support vector regression with pinball loss function (Asy-ν-TSVR) [35] has been proposed in order to pursue the asymmetric tube where the different penalties are assigned to the above the up-bound and below the down-bound. Asy-ν-TSVR is highly influenced from Huang et al. [45] whereε-insensitive loss is replaced by pinball loss [40] where points having the different penalties based on their different position. Here, pinball loss function is defined as:

$$L_\varepsilon^p(x) = \begin{cases} \frac{1}{2p}(x - \varepsilon), & x \geq \varepsilon, \\ 0, & -\varepsilon < x < \varepsilon, \\ \frac{1}{2(1-p)}(-x - \varepsilon), & x \leq -\varepsilon, \end{cases} \tag{18}$$

where p is the asymmetric penalty parameter. One can be degraded into ε-insensitive loss by choosing the value $p = 0.5$..

In linear Asy-ν-TSVR case, two nonparallel ε_1-insensitive down-bound $f_1(x) = w_1^t x + b_1$ and up-bound $f_2(x) = w_2^t x + b_2$ functions are generated by solving the pair of QPPs in the following manner:

$$\min \frac{1}{2}\|y - (Aw_1 + b_1 e)\|^2 + C_1\nu_1\varepsilon_1 + \frac{1}{m}C_1 e^t\xi_1$$

subject to

$$y - (Aw_1 + b_1 e) \geq -\varepsilon_1 e - 2(1-p)\xi_1, \xi_1 \geq 0, \varepsilon_1 \geq 0 \tag{19}$$

and

$$\min \frac{1}{2}\|y - (Aw_2 + b_2 e)\|^2 + C_2\nu_2\varepsilon_2 + \frac{1}{m}C_2 e^t\xi_2$$

subject to

$$(Aw_2 + b_2 e) - y \geq -\varepsilon_2 e - 2p\xi_2, \xi_2 \geq 0, \varepsilon_2 \geq 0 \tag{20}$$

where, ξ_1, ξ_2 are the slack variables; C_1, $C_2 > 0$; ε_1, $\varepsilon_2 > 0$, ν_1, ν_2 are the input parameters.

Apply Lagrangian multipliers α_1, $\alpha_2 > 0 \in R^m$ and KKT conditions, we get the dual QPP of Asy-ν-TSVR from the Eqs. (19) and (20)

$$\min \frac{1}{2}\alpha_1^t S(S^tS)^{-1}S^t\alpha_1 - y^tS(S^tS)^{-1}S^t\alpha_1 + y^t\alpha_1$$

subject to

$$0 \le \alpha_1 \le \frac{C_1 e}{2(1-p)m}, e^t\alpha_1 \le C_1\nu_1 \tag{21}$$

and

$$\min \frac{1}{2}\alpha_2^t S(S^tS)^{-1}S^t\alpha_2 + y^tS(S^tS)^{-1}S^t\alpha_2 - y^t\alpha_2$$

subject to

$$0 \le \alpha_2 \le \frac{C_2 e}{2pm}, e^t\alpha_2 \le C_2\nu_2 \tag{22}$$

where, $S = [A \quad e]$.

After solving the Eqs. (21) and (22), one can compute the values of w_1, w_2, b_1, b_2 in the following manner:

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (S^tS)^{-1}S^t(y - \alpha_1)$$

and

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (S^tS)^{-1}S^t(y + \alpha_2)$$

Finally, the end regression function is obtained as similar to (9).

In the nonlinear case, kernel generating regression functions are $f_1(x) = K(x^t, A^t)w_1 + b_1$ and $f_2(x) = K(x^t, A^t)w_2 + b_2$ by solving the pair of QPPs in such a way:

$$\min \frac{1}{2}\left\| y - \left(K(A, A^t)w_1 + b_1 e\right)\right\|^2 + C_1\nu_1\varepsilon_1 + \frac{1}{m}C_1 e^t\xi_1$$

subject to

$$y - (K(A, A^t)w_1 + b_1 e) \ge -\varepsilon_1 e - 2(1-p)\xi_1, \xi_1 \ge 0, \varepsilon_1 \ge 0 \tag{23}$$

and

$$\min \frac{1}{2}\left\| y - \left(K(A, A^t)w_2 + b_2 e\right)\right\|^2 + C_2\nu_2\varepsilon_2 + \frac{1}{m}C_2 e^t\xi_2$$

subject to

$$(K(A, A^t)w_2 + b_2 e) - y \ge -\varepsilon_2 e - 2p\xi_2, \xi_2 \ge 0, \varepsilon_2 \ge 0 \tag{24}$$

Apply Lagrangian multipliers α_1, $\alpha_2$and KKT necessary conditions, the dual formation of (23) and (24) can be derived as follows:

$$\min \frac{1}{2}\alpha_1^t T(T^tT)^{-1}T^t\alpha_1 - y^tT(T^tT)^{-1}T^t\alpha_1 + y^t\alpha_1$$

subject to

$$0 \le \alpha_1 \le \frac{C_1 e}{2(1-p)m}, e^t\alpha_1 \le C_1\nu_1 \tag{25}$$

and

$$\min \frac{1}{2}\alpha_2^t T(T^tT)^{-1}T^t\alpha_2 + y^tT(T^tT)^{-1}T^t\alpha_2 - y^t\alpha_2$$

subject to

$$0 \le \alpha_2 \le \frac{C_2 e}{2pm}, e^t\alpha_2 \le C_2\nu_2 \tag{26}$$

where, $T = \begin{bmatrix} K(A, A^t) & e \end{bmatrix}$.

After solving the Eqs. (27) and (28) for α_1 and α_2, we can obtain the augmented vectors as

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (T^tT)^{-1}T^t(y - \alpha_1)$$

and

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (T^tT)^{-1}T^t(y + \alpha_2)$$

Finally, the end regression estimation function is given as similar to the linear case for any test sample$x \in R^n$.

2.5 A regularization on Lagrangian twin support vector regression (RLTSVR)

By considering the principle of structural risk minimization instead of usual empirical risk in ε-TSVR, recently Tanveer & Shubham [30] proposed a new algorithm termed as regularization on Lagrangian twin support vector regression (RLTSVR) whose solution is obtained by simple linearly convergent iterative approach. The two nonparallel functions $f_1(x) = K(x^t, A^t)w_1 + b_1$ and $f_2(x) = K(x^t, A^t)w_2 + b_2$ are determined by using the following constrained minimization problems:

$$\min \frac{C_3}{2}\left(w_1^tw_1 + b_1^2\right) + \frac{1}{2}\left\| y - \left(K\left(A, A^t\right)w_1 + eb_1\right)\right\|^2 + \frac{1}{2}C_1{\xi_1}^t\xi_1$$

subject to

$$y - (K(A, A^t)w_1 + eb_1) \ge e\varepsilon_1 - \xi_1 \tag{27}$$

and

$$\min \frac{C_4}{2}\left(w_2^tw_2 + b_2^2\right) + \frac{1}{2}\left\| y - \left(K\left(A, A^t\right)w_2 + eb_2\right)\right\|^2 + \frac{1}{2}C_2{\xi_2}^t\xi_2$$

Table 1 Different user define parameters used in numerical experiment

Parameters	Range	Models
ε	$\{0.1, 0.01\}$	SVR, RLTSVR
	$\{0.1, 0.3, 0.5, 0.7, 0.9\}$	TSVR, HN-TSVR
$\varepsilon_1, \varepsilon_2$	$\{0.1, 0.3, 0.5, 0.7, 0.9\}$	HN-TSVR
C	$\{10^{-5}, ..., 10^{5}\}$	SVR
$C_1 = C_2, C_3 = C_4$	$\{10^{-5}, 10^{-3}, 10^{-1}, 10^{1}, 10^{3}, 10^{5}\}$	TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR, Proposed LAsy-ν-TSVR
$\nu_1 = \nu_2$	$\{0.1, 0.3, 0.5, 0.7, 0.9\}$	Asy-ν-TSVR
	$\{0.01\}$	Proposed LAsy-ν-TSVR
p	$\{0.2, 0.4, 0.45, 0.5, 0.55, 0.6, 0.8\}$	Asy-ν-TSVR, Proposed LAsy-ν-TSVR
μ	$\{2^{-5}, ..., 2^{5}\}$	SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR Proposed LAsy-ν-TSVR

subject to

$$(K(A, A^t)w_2 + eb_2) - y \geq e\varepsilon_2 - \xi_2 \tag{28}$$

where, input parameters are C_1, C_2, C_3, $C_4 > 0$ and ε_1, $\varepsilon_2 > 0$; ξ_1, ξ_2 are slack variables.

Now introduce the Lagrangian multipliers $\alpha_1 = (\alpha_{11}, ..., \alpha_{1m})^t$ and $\alpha_2 = (\alpha_{21}, ..., \alpha_{2m})^t$, the dual form of the QPP of (27) and (28) can be written as:

$$\min_{\alpha_1 \geq 0} \frac{1}{2}\alpha_1^t \left(\frac{I}{C_1} + S(S^tS + C_3I)^{-1}S^t\right) \alpha_1 - \left(y^tS(S^tS + C_3I)^{-1}S^t - (y + e\varepsilon_1)^t\right)\alpha_1 \tag{29}$$

and

$$\min_{\alpha_2 \geq 0} \frac{1}{2}\alpha_2^t \left(\frac{I}{C_2} + S(S^tS + C_4I)^{-1}S^t\right) \alpha_2 - \left((y - e\varepsilon_2)^t - y^tS(S^tS + C_4I)^{-1}S^t\right)\alpha_2 \tag{30}$$

where, $S = \begin{bmatrix} K(A, A^t) & e \end{bmatrix}$ is the augmented matrix.

After solving the above pair of dual QPPs (29) and (30) for α_1 and α_2, one can derive the values as:

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (S^tS + C_3I)^{-1}S^t(y - \alpha_1)$$

and

$$\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = (S^tS + C_4I)^{-1}S^t(y + \alpha_2)$$

Finally, the end regression function is obtained as similar to (9). For more details, one can see [30].

3 Proposed improved regularization based Lagrangian asymmetric *v*-twin support vector regression using pinball loss (LAsy-*v*-TSVR)

Recently, Xu et al., [35] has suggested a novel approach termed as asymmetric ν-twin support vector regression using pinball loss function to handle the asymmetric noise and outliers in challenging real-world problems. In order to further improvement of generalization ability and reduction of computation cost, we propose another approach termed as improved regularization based Lagrangian asymmetric ν-twin support vector regression using pinball loss function (LAsy-ν-TSVR) whose solution is obtained by solving the linearly convergent iterative method in place of solving QPPs. To formulate our proposed LAsy-ν-TSVR formulation, we replace the 1-norm of vector of slack variables ξ_1 and ξ_2, by the square of the vector of slack variables in 2-norm which

Table 2 Functions used for generating artificial datasets

Function name	Function definition	Domain of definition	Noise type
Function 1	$f(x_1, x_2, x_3, x_4, x_5) = 0.79 + 1.27x_1x_2 + 1.56x_1x_4 + 3.42x_2x_5 + 2.06x_3x_4x_5 + \Omega$	$x_i \in [0, 1]$, $i \in \{1,2,3,4,5\}$	Type A: $\Omega \in U(-0.2, 0.2)$
Function 2			Type B: $\Omega \in N(0, 0.2^2)$
Function 3	$f(x_1, x_2) = 1.9[1.35 + e^{x_1}\sin\left(13(x_1 - 0.6)^2\right) + e^{3(x_2 - 0.5))}\sin\left(4\pi(x_2 - 0.9)^2\right)] + \Omega$	$x_1, x_2 \in U(0, 1)$	Type A: $\Omega \in U(-0.2, 0.2)$
Function 4			Type B: $\Omega \in N(0, 0.2^2)$
Function 5	$f(x) = \frac{\lvert x-1\rvert}{4} + \lvert\sin\left(\pi\left(1 + \frac{x-1}{4}\right)\right)\rvert + 1 + \Omega$	$x \in U(-10, 10)$	Type A: $\Omega \in U(-0.2, 0.2)$
Function 6			Type B: $\Omega \in N(0, 0.2^2)$
Function 7	$f(x) = \frac{\sin(x)}{x}$ such that $y_i = f(x_i) + \left(0.5 - \frac{\lvert x_i\rvert}{8\pi}\right)\Omega_i$	$x_i \in U(-4\pi, 4\pi)$, $i = 1, 2, \ldots, 200$	Type A: $\Omega \in U(-1, 1)$
Function 8			Type B: $\Omega \in N(0, 0.5^2)$

Table 3 Performance comparison of LAsy-ν-TSVR with SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR on artificial datasets using linear kernel

Dataset (Train size, Test size)	SVR (C,ε) Time	TSVR $(C_1=C_2,\varepsilon)$ Time	HN-TSVR $(C_1=C_2,\varepsilon_1,\varepsilon_2)$ Time	Asy-ν-TSVR $(C_1=C_2,\nu_1=\nu_2,p)$ Time	RLTSVR $(C_1=C_2,C_3=C_4,\varepsilon)$ Time	LAsy-ν-TSVR $(C_1=C_2,C_3=C_4,\nu_1=\nu_2,p)$ Time
Function 1 (200X6,500X6)	0.69482 (10^3, 0.1) 1.50567	0.69138 (10^-2, 0.9) 0.03879	0.69138 (10^-1, 0.9,0.1) 0.05623	0.7024 (10^3, 0.5,0.8) 0.05194	**0.69133** (10^-5, 10^-5,0.1) 0.00415	**0.69133** (10^-1, 10^-1,0.01,0.2) 0.00314
Function 2 (200X6,500X6)	0.68685 (10^3, 0.1) 1.21832	0.66616 (10^-2, 0.1) 0.03937	0.66616 (10^-1, 0.1,0.1) 0.14137	0.6665 (10^3, 0.1,0.45) 0.13428	**0.66584** (10^-4, 10^-1,0.1) 0.01022	**0.66584** (10^-5, 10^-5,0.01,0.2) 0.00258
Function 3 (200X3,500X3)	1.61958 (10^4, 0.1) 1.3133	1.61802 (10^1, 0.9) 0.13078	1.61818 (10^2, 0.9,0.1) 0.0853	**1.61179** (10^3, 0.1,0.2) 0.04752	1.6158 (10^0, 10^0,0.1) 0.03341	1.61413 (10^3, 10^3,0.01,0.2) 0.00345
Function 4 (200X3,500X3)	1.53898 (10^5, 0.1) 1.48626	1.56234 (10^3, 0.9) 0.05067	1.55185 (10^3, 0.1,0.5) 0.04576	1.55298 (10^5, 0.1,0.8) 0.05134	1.53524 (10^4, 10^-5,0.1) 0.03831	**1.5312** (10^3, 10^3,0.01,0.2) 0.00309
Function 5 (200X2,500X2)	0.79133 (10^-3, 0.1) 2.03451	0.78753 (10^1, 0.9) 0.04498	0.78687 (10^1, 0.9,0.9) 0.03896	0.78762 (10^3, 0.1,0.55) 0.04818	0.78633 (10^1, 10^-5,0.1) 0.02929	**0.78599** (10^-3, 10^-3,0.01,0.8) 0.00233
Function 6 (200X2,500X2)	0.79169 (10^-3, 0.1) 1.90186	0.77154 (10^-5, 0.1) 0.04453	0.77154 (10^-5, 0.1,0.1) 0.04055	**0.7697** (10^2, 0.1,0.4) 0.04887	0.77137 (10^0, 10^-5,0.1) 0.0364	0.77153 (10^-3, 10^-3,0.01,0.8) 0.00226
Function 7 (200X2,500X2)	0.31056 (10^2, 0.1) 2.02626	**0.30912** (10^-1, 0.1) 0.07148	0.31079 (10^-1, 0.9,0.5) 0.06311	0.31022 (10^1, 0.5,0.8) 0.04677	0.31038 (10^-4, 10^1,0.1) 0.00403	0.31038 (10^3, 10^3,0.01,0.2) 0.00448
Function 8 (200X2,500X2)	0.33939 (10^-5, 0.1) 2.03524	**0.33238** (10^-2, 0.1) 0.04947	0.33245 (10^-1, 0.9,0.5) 0.06093	0.33249 (10^2, 0.7,0.4) 0.04359	0.33247 (10^-3, 10^1,0.1) 0.03399	0.33248 (10^3, 10^3,0.01,0.2) 0.01928

The best result is shown as boldface

makes the problem strongly convex and yields the existence of global unique solution. In order to follow the SRM principle unlike in case of TSVR and Asy-ν-TSVR, the regularization terms $\frac{C_3}{2}\left(\|w_1\|^2+b_1^2\right)$ and $\frac{C_4}{2}\left(\|w_2\|^2+b_2^2\right)$ are also added in the objective functions of (19) and (20) respectively that improves the stability in the dual formulations as well as makes the model well-posed. In the formulations of linear proposed LAsy-ν-TSVR, the regression functions $f_1(x)=w_1^t x+b_1$ and $f_2(x)=w_2^t x+b_2$ are obtained by solving the modified QPPs as

$$\min \frac{C_3}{2}\left(\|w_1\|^2+b_1^2\right)+\frac{1}{2}\|y-(Aw_1+eb_1)\|^2+\frac{1}{m}C_1\xi_1^t\xi_1 + C_1\nu_1\varepsilon_1^2$$

subject to

$$y-(Aw_1+eb_1)\geq -e\varepsilon_1-2(1-p)\xi_1 \tag{31}$$

and

$$\min \frac{C_4}{2}\left(\|w_2\|^2+b_2^2\right)+\frac{1}{2}\|y-(Aw_2+eb_2)\|^2+\frac{1}{m}C_2\xi_2^t\xi_2 + C_2\nu_2\varepsilon_2^2$$

subject to

$$(Aw_2+eb_2)-y\geq -e\varepsilon_2-2p\xi_2 \tag{32}$$

where C_1, C_2, C_3, $C_4>0$, and ν_1, ν_2 are input parameters; $\xi_1=(\xi_{11},\ldots\xi_{1m})^t$, $\xi_2=(\xi_{21},\ldots\xi_{2m})^t$ are the slack variables. Here, the non-negative constraints of the slack variables are dropped in (31) and (32). The Lagrangian functions of (31) and (32) are obtained by using the Lagrangian multipliers α_1, $\alpha_2>0\in R^m$ as

$$L_1=\frac{C_3}{2}\left(\|w_1\|^2+b_1^2\right)+\frac{1}{2}\|y-(Aw_1+eb_1)\|^2 + \frac{1}{m}C_1\xi_1^t\xi_1 + C_1\nu_1\varepsilon_1^2-\alpha_1^t(y-(Aw_1+eb_1)+e\varepsilon_1+2(1-p)\xi_1) \tag{33}$$

and

$$L_2=\frac{C_4}{2}\left(\|w_2\|^2+b_2^2\right)+\frac{1}{2}\|y-(Aw_2+eb_2)\|^2 + \frac{1}{m}C_2\xi_2^t\xi_2 + C_2\nu_2\varepsilon_2^2-\alpha_2^t((Aw_2+eb_2)-y+e\varepsilon_2+2p\xi_2) \tag{34}$$

Further, apply the KKT conditions in (41), we get

$$\frac{\partial L_1}{\partial w_1}=C_3w_1-A^t(y-(Aw_1+eb_1))+A^t\alpha_1=0, \tag{35}$$

Table 4 Average ranks of SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on RMSE values for artificial datasets using linear kernel

Dataset	SVR	TSVR	HN-TSVR	Asy-ν-TSVR	RLTSVR	LAsy-ν-TSVR
Function 1	5	4	3	6	1.5	1.5
Function 2	6	3	4	5	1.5	1.5
Function 3	6	4	5	1	3	2
Function 4	3	6	4	5	2	1
Function 5	6	4	3	5	2	1
Function 6	6	4.5	4.5	1	2	3
Function 7	5	1	6	2	3.5	3.5
Function 8	6	1	2	5	3	4
Average rank	5.375	3.4375	3.9375	3.75	2.3125	**2.1875**

$$\frac{\partial L_1}{\partial b_1} = C_3 b_1 - e^t(y-(Aw_1+eb_1)) + e^t\alpha_1 = 0, \tag{36}$$

$$\frac{\partial L_1}{\partial \xi_1} = \frac{C_1}{m}\xi_1 - 2(1-p)\alpha_1 = 0, \tag{37}$$

$$\frac{\partial L_1}{\partial \varepsilon_1} = 2C_1\nu_1\varepsilon_1 - e^t\alpha_1 = 0. \tag{38}$$

By combining the Eqs. (35) and (36), we get

$$\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = (S^tS + C_3I)^{-1}S^t(y-\alpha_1) \tag{39}$$

where $S = [A \quad e]$ is an augmented matrix.

By using the Eqs. (33), (36), (37) and (38), the dual QPP of primal problem (33) is given as

$$\min \frac{1}{2}\alpha_1^t\left(S(S^tS + C_3I)^{-1}S^t + \frac{4m(1-p)^2}{C_1} + \frac{ee^t}{2C_1\nu_1}\right)\alpha_1 - \left(S(S^tS + C_3I)^{-1}S^ty - y\right)^t\alpha_1 \tag{40}$$

Similarly, we get the following dual QPP of the primal problem (34) as

$$\min \frac{1}{2}\alpha_2^t\left(S(S^tS + C_4I)^{-1}S^t + \frac{4mp^2}{C_2} + \frac{ee^t}{2C_2\nu_2}\right)\alpha_2 - \left(-S(S^tS + C_4I)^{-1}S^ty + y\right)^t\alpha_2 \tag{41}$$

The values of α_1 and α_2 are determined by solving the QPPs (40) and (41). The end regression function $f(.)$ is determined by taking the mean of $f_1(x)$ and $f_2(x)$ for any test sample $x \in R^n$:

$$f_1(x) = w_1^tx + b_1 = [x^t \quad 1]\left((S^tS + C_3I)^{-1}S^t(y-\alpha_1)\right) \tag{42}$$

and

$$f_2(x) = w_2^tx + b_2 = [x^t \quad 1]\left((S^tS + C_4I)^{-1}S^t(y+\alpha_2)\right). \tag{43}$$

In the formulation of non-linear LAsy-ν-TSVR, the kernel generated functions $f_1(x) = K(x^t, A^t)w_1 + b_1$ and $f_2(x) = K(x^t, A^t)w_2 + b_2$ are determined by the following QPPs as

$$\min \frac{C_3}{2}\left(\|w_1\|^2 + b_1^2\right) + \frac{1}{2}\left\|y-\left(K\left(A, A^t\right)w_1 + eb_1\right)\right\|^2 + \frac{1}{m}C_1\xi_1^t\xi_1 + C_1\nu_1\varepsilon_1^2$$

subject to

$$y-(K(A, A^t)w_1 + eb_1) \geq -e\varepsilon_1 - 2(1-p)\xi_1 \tag{44}$$

and

$$\min \frac{C_4}{2}\left(\|w_2\|^2 + b_2^2\right) + \frac{1}{2}\left\|y-\left(K\left(A, A^t\right)w_2 + eb_2\right)\right\|^2 + \frac{1}{m}C_2\xi_2^t\xi_2 + C_2\nu_2\varepsilon_2^2$$

subject to

$$(K(A, A^t)w_2 + eb_2) - y \geq -e\varepsilon_2 - 2p\xi_2 \tag{45}$$

respectively, where C_1, C_2, C_3, $C_4 > 0$; and ν_1, ν_2 are input parameters.

Using the Lagrangian multipliers α_1, $\alpha_2 > 0 \in R^m$, the Lagrangian functions of (44) and (45) are given by

$$L_1 = \frac{C_3}{2}\left(\|w_1\|^2 + b_1^2\right) + \frac{1}{2}\left\|y-\left(K\left(A, A^t\right)w_1 + eb_1\right)\right\|^2 + \frac{1}{m}C_1\xi_1^t\xi_1 + C_1\nu_1\varepsilon_1^2 - \alpha_1^t(y-(K(A, A^t)w_1 + eb_1) + e\varepsilon_1 + 2(1-p)\xi_1) \tag{46}$$

and

$$L_2 = \frac{C_4}{2}\left(\|w_2\|^2 + b_2^2\right) + \frac{1}{2}\left\|y-\left(K\left(A, A^t\right)w_2 + eb_2\right)\right\|^2 + \frac{1}{m}C_2\xi_2^t\xi_2 + C_2\nu_2\varepsilon_2^2 - \alpha_2^t((K(A, A^t)w_2 + eb_2) - y + e\varepsilon_2 + 2p\xi_2) \tag{47}$$

Further, apply the KKT conditions, the dual QPP of primal problems (46) & (47) are given as

Table 5 Performance comparison of LAsy-ν-TSVR with SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR on artificial datasets using Gaussian kernel

Dataset (Train size, Test size)	SVR (C, ε, μ)Time	TSVR $(C_1 = C_2, \varepsilon_1, \mu)$Time	HN-TSVR $(C_1 = C_2, \varepsilon_1, \varepsilon_2, \mu)$Time	Asy-ν-TSVR $(C_1 = C_2, \nu_1 = \nu_2, p, \mu)$ Time	RLTSVR $(C_1 = C_2, C_3 = C_4, \varepsilon, \mu)$Time	LAsy-ν-TSVR$(C_1 = C_2, C_3 = C_4, \nu_1 = \nu_2, p, \mu)$ Time
Function 1 (200X6,500X6)	0.01107 (10^4, 0.1,2^-5) 1.65705	0.01084 (10^1, 0.1,2^-2) 0.17169	0.01237 (10^5,0.1,0.9,2^ -2) 0.37267	0.0146 (10^3, 0.1,0.8,2^-2) 0.16657	0.01076 (10^1, 10^-4,0.1,2^-2) 0.13675	**0.00914** (10^-5, 10^-5,0.2,0.01,2^-3) 0.22534
Function 2 (200X6,500X6)	0.02812 (10^5, 0.1,2^-5) 1.56428	0.01231 (10^0, 0.1,2^0) 0.16375	0.01179 (10^1,0.9,0.1,2^ 0) 0.54477	0.0116 (10^2, 0.3,0.4,2^2) 0.16273	**0.00528** (10^1, 10^-5,0.1,2^-3) 0.13207	0.00531 (10^3, 10^3,0.8,0.01,2^-3) 4.48452
Function 3 (200X3,500X3)	0.14756 (10^5, 0.1,2^2) 1.49714	0.06686 (10^5, 0.9,2^5) 0.18529	0.06686 (10^5,0.9,0.3,2^ 5) 0.68926	0.06596 (10^2, 0.9,0.2,2^5) 0.13994	0.06189 (10^-1, 10^-5,0.1,2^5) 0.13198	**0.06179** (10^3, 10^3,0.2,0.01,2^5) 0.07248
Function 4 (200X3,500X3)	0.10716 (10^5, 0.1,2^2) 1.55264	0.1229 (10^0, 0.1,2^4) 0.48113	0.12157 (10^1,0.1,0.3,2^ 4) 0.16129	0.12748 (10^2, 0.3,0.4,2^4) 0.12761	0.1155 (10^1, 10^-5,0.1,2^4) 0.07862	**0.10709** (10^3, 10^3,0.8,0.01,2^5) 0.07674
Function 5 (200X2,500X2)	0.05235 (10^2, 0.1,2^-1) 1.74581	0.0294 (10^-2, 0.1,2^1) 0.25023	0.0298 (10^0,0.1,0.1,2^ 1) 0.22634	0.03127 (10^1, 0.3,0.8,2^1) 0.14995	0.03916 (10^0, 10^-4,0.1,2^1) 0.0585	**0.02939** (10^3, 10^3,0.8,0.01,2^1) 0.10211
Function 6 (200X2,500X2)	0.07822 (10^4, 0.1,2^-2) 2.22309	0.05721 (10^3, 0.1,2^-1) 0.16783	**0.05721** (10^5,0.1,0.9,2^ -1) 0.68878	0.06447 (10^2, 0.1,0.4,2^0) 0.09992	0.06204 (10^1, 10^-2,0.1,2^0) 0.12468	0.07023 (10^3, 10^3,0.8,0.01,2^-1) 0.09478
Function 7 (200X2,500X2)	0.03608 (10^2, 0.1,2^-3) 2.02376	**0.02116** (10^-5, 0.7,2^-3) 0.46059	0.0219 (10^0,0.9,0.1,2^ -3) 0.62534	0.02279 (10^1, 0.3,0.2,2^-3) 0.11131	0.02116 (10^5, 10^-3,0.1,2^-3) 0.0336	0.02116 (10^-3, 10^-3,0.8,0.01,2^-3) 0.26221
Function 8 (200X2,500X2)	0.03449 (10^2, 0.1,2^-5) 1.90055	0.01879 (10^3, 0.9,2^-5) 0.43558	0.01878 (10^5,0.9,0.7,2^ -5) 0.18905	0.01879 (10^4, 0.9,0.8,2^-5) 0.11251	0.02338 (10^1, 10^-2,0.1,2^-4) 0.12476	**0.01574** (10^-5, 10^-5,0.2,0.01,2^-5) 1.28936

The best result is shown as boldface

$$\min \frac{1}{2}\alpha_1^t\left(T(T^tT+C_3I)^{-1}T^t+\frac{4m(1-p)^2}{C_1}+\frac{ee^t}{2C_1\nu_1}\right)\alpha_1-\left(T(T^tT+C_3I)^{-1}T^ty-y\right)^t\alpha_1 \quad (48)$$

and

$$\min \frac{1}{2}\alpha_2^t\left(T(T^tT+C_4I)^{-1}T^t+\frac{4mp^2}{C_2}+\frac{ee^t}{2C_2\nu_2}\right)\alpha_2-\left(-T(T^tT+C_4I)^{-1}T^ty+y\right)^t\alpha_2 \quad (49)$$

where $T = \begin{bmatrix} K(A, A^t) & e \end{bmatrix}$ is an augmented matrix.

After computing the values of α_1 and α_2 from (48) and (49), the final estimation function $f(.)$ is determined for non-linear kernel by taking the mean of the following non-linear functions $f_1(x)$ and $f_2(x)$ as

$$f_1(x) = \begin{bmatrix} K(x^t, A^t) & 1 \end{bmatrix}\begin{bmatrix} w_1 \\ b_1 \end{bmatrix} = \begin{bmatrix} K(x^t, A^t) & 1 \end{bmatrix}\left((T^tT+C_3I)^{-1}T^t(y-\alpha_1)\right)$$

and

$$f_2(x) = \begin{bmatrix} K(x^t, A^t) & 1 \end{bmatrix}\begin{bmatrix} w_2 \\ b_2 \end{bmatrix} = \begin{bmatrix} K(x^t, A^t) & 1 \end{bmatrix}\left((T^tT+C_4I)^{-1}T^t(y+\alpha_2)\right)$$

One can rewrite the problems (48) and (49) in the following form:

$$\min_{0\le\alpha_1\in R^m} L_1(\alpha_1) = \frac{1}{2}\alpha_1^tD_1\alpha_1-r_1^t\alpha_1 \quad (50)$$

and

$$\min_{0\le\alpha_2\in R^m} L_2(\alpha_2) = \frac{1}{2}\alpha_2^tD_2\alpha_2-r_2^t\alpha_2 \quad (51)$$

respectively, where

$D_1 = \left(T(T^tT+C_3I)^{-1}T^t+\frac{4m(1-p)^2}{C_1}+\frac{ee^t}{2C_1\nu_1}\right)$,

$D_2 = \left(T(T^tT+C_4I)^{-1}T^t+\frac{4mp^2}{C_2}+\frac{ee^t}{2C_2\nu_2}\right)$, $r_1 = T(T^tT + C_3I)^{-1}T^ty - y$ and $r_2 = -T(T^tT + C_4I)^{-1}T^ty + y$.

The KKT optimality conditions [49] is applied on the QPPs (50) and (51) which lead to the following pair of classical complementary problems as

$$0\le(D_1\alpha_1-r_1)\perp\alpha_1\ge0 \quad (52)$$

and

$$0\le(D_2\alpha_2-r_2)\perp\alpha_2\ge0, \quad (53)$$

Table 6 Average ranks of SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on RMSE values for artificial datasets using Gaussian kernel

Dataset	SVR	TSVR	HN-TSVR	Asy-ν-TSVR	RLTSVR	LAsy-ν-TSVR
Function 1	4	3	5	6	2	1
Function 2	6	5	4	3	1	2
Function 3	6	5	4	3	2	1
Function 4	2	5	4	6	3	1
Function 5	6	2	3	4	5	1
Function 6	6	2	1	4	3	5
Function 7	6	1	4	5	2.5	2.5
Function 8	6	3	2	4	5	1
Average rank	5.25	3.25	3.375	4.375	2.9375	**1.8125**

respectively. By using the identity $0 \leq x \perp y \geq 0$ if and only if $x = (x - \psi y)_+$ for any vectors x, y and parameter $\psi > 0$, the equivalent pair of problems [50] of (52) and (53) are rewritten in the following fixed point theorems: for any ψ_1, $\psi_2 > 0$, the relations

$$(D_1\alpha_1 - r_1) = (D_1\alpha_1 - \psi_1\alpha_1 - r_1)_+ \tag{54}$$

and

$$(D_2\alpha_2 - r_2) = (D_2\alpha_2 - \psi_2\alpha_2 - r_2)_+. \tag{55}$$

To solve the above problems (50) and (51), one can propose the following simple iterative approach in the following form:

$$\alpha_1^{i+1} = D_1^{-1}\left((D_1\alpha_1^i - \psi_1\alpha_1^i - r_1)_+ + r_1\right) \tag{56}$$

and

$$\alpha_2^{i+1} = D_2^{-1}\left((D_2\alpha_2^i - \psi_2\alpha_2^i - r_2)_+ + r_2\right) \tag{57}$$

i.e.

$$\alpha_1^{i+1} = \left(T(T^tT + C_3I)^{-1}T^t + \frac{4m(1-p)^2}{C_1} + \frac{ee^t}{2C_1\nu_1}\right)^{-1}\left[\left(T(T^tT + C_3I)^{-1}T^t + \frac{4m(1-p)^2}{C_1} + \frac{ee^t}{2C_1\nu_1}\right)\alpha_1^i - \psi_1\alpha_1^i - \left(T(T^tT + C_3I)^{-1}T^ty - y\right)\right)_+ + T(T^tT + C_3I)^{-1}T^ty - y\right)\right] \tag{58}$$

and

$$\alpha_2^{i+1} = \left(T(T^tT + C_4I)^{-1}T^t + \frac{4mp^2}{C_2} + \frac{ee^t}{2C_2\nu_2}\right)^{-1}\left[\left(T(T^tT + C_4I)^{-1}T^t + \frac{4mp^2}{C_2} + \frac{ee^t}{2C_2\nu_2}\right)\alpha_2^i - \psi_2\alpha_2^i - \left(-T(T^tT + C_4I)^{-1}T^ty + y\right)\right)_+ + \left(-T(T^tT + C_4I)^{-1}T^ty + y\right)\right] \tag{59}$$

Remark 1 One may notice that we have to compute the inverse of the matrices $\left(T(T^tT + C_3I)^{-1}T^t + \frac{4m(1-p)^2}{C_1} + \frac{ee^t}{2C_1\nu_1}\right)$ and $\left(T(T^tT + C_4I)^{-1}T^t + \frac{4mp^2}{C_2} + \frac{ee^t}{2C_2\nu_2}\right)$ in the above iterative schemes (58) and (59) to find the solution of our proposed LAsy-ν-TSVR. Unlike the Asy-ν-TSVR and TSVR, these matrices are positive definite which can be compute at the very beginning of the algorithm.

Remark 2 Unlike the TSVR and Asy-ν-TSVR, there is not any required additions of extra term δI to make the matrix positive definite where δ is very small positive number and I is the identity matrix. Our proposed LAsy-ν-TSVR always gives unique global solution since $\left(T(T^tT + C_3I)^{-1}T^t + \frac{4m(1-p)^2}{C_1} + \frac{ee^t}{2C_1\nu_1}\right)$ and $\left(T(T^tT + C_4I)^{-1}T^t + \frac{4mp^2}{C_2} + \frac{ee^t}{2C_2\nu_2}\right)$ both are positive definite matrices.

Table 7 Performance comparison of LAsy-ν-TSVR with SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR using linear kernel on real-world datasets

Dataset (Train size, Test size)	SVR (C, ε) Time	TSVR ($C_1 = C_2$, ε) Time	HN-TSVR ($C_1 = C_2$, ε_1, ε_2)Time	Asy-ν-TSVR ($C_1 = C_2$, $\nu_1 = \nu_2$, p)Time	RLTSVR ($C_1 = C_2$, $C_3 = C_4$, ε) Time	LAsy-ν-TSVR ($C_1 = C_2$, $C_3 = C_4$, $\nu_1 = \nu_2$, p) Time
ConcreteCS (200X9,830X9)	0.14404 (10^0, 0.1) 1.01335	0.13828 (10^5, 0.9) 0.04292	0.13956 (10^3, 0.9, 0.5) 0.11974	0.1383 (10^5, 0.9, 0.6) 0.05649	0.14165 (10^1,10^ 0,0.01) 0.09619	**0.13395** (10^3,10^ 3,0.8,0.01) 0.13395
Boston (200X14,306X14)	**0.14021** (10^0, 0.1) 0.86142	0.29243 (10^-1, 0.1) 0.02771	0.32808 (10^0, 0.9, 0.1) 0.06508	0.31919 (10^1, 0.9, 0.8) 0.03808	0.14802 (10^-3,10^ -1, 0.01) 0.04847	0.14801 (10^1,10^ 1,0.2,0.01) 0.14801
Auto-MPG (100X8,292X8)	**0.17161** (10^0, 0.1) 0.23864	0.20651 (10^-1, 0.9) 0.01664	0.19958 (10^0, 0.1, 0.1) 0.01812	0.20004 (10^1, 0.3, 0.8) 0.0131	0.18885 (10^0,10^ 0, 0.01) 0.09525	0.19935 (10^1,10^ 1,0.8,0.01) 0.19935
Parkinsons (500X17,5375X17)	0.28368 (10^2, 0.1) 6.06146	0.27984 (10^-1, 0.1) 0.286	0.28231 (10^0, 0.1, 0.7) 0.63062	0.27993 (10^1, 0.9, 0.8) 0.29567	0.26227 (10^1,10^ -1, 0.01) 0.56417	**0.26012** (10^3,10^ 3,0.2,0.01) 0.26012
Winequality (500X12,4398X12)	0.13072 (10^2, 0.1) 6.11041	**0.12961** (10^0, 0.1) 0.28937	0.13127 (10^0, 0.9, 0.3) 0.55141	0.13177 (10^2, 0.3, 0.4) 0.26639	0.13029 (10^0,10^ 0, 0.01) 0.59904	0.13157 (10^1,10^ 1,0.2,0.01) 0.13157
Kin900 (250X33,650X33)	0.0971 (10^-1, 0.1) 1.64749	0.09541 (10^0, 0.1) 0.06896	0.09575 (10^1, 0.9, 0.9) 0.31254	0.09818 (10^3, 0.1, 0.8) 0.08089	0.09544 (10^1,10^ 0, 0.01) 0.20999	**0.09497** (10^3,10^ 3,0.2,0.01) 0.09497
Demo (500X5,1548X5)	0.11184 (10^3, 0.1) 5.97637	0.10226 (10^-1, 0.9) 0.35417	0.10187 (10^0, 0.9, 0.1) 0.4296	0.10205 (10^2, 0.7, 0.5) 0.30176	**0.10149** (10^4,10^ 0, 0.01) 0.54828	0.10171 (10^-3,10^ -3,0.2,0.01) 0.10171
Mg_{17} (500X6,995X6)	0.10893 (10^3, 0.1) 5.67111	0.1075 (10^-5, 0.1) 0.25074	0.1075 (10^-3, 0.9, 0.3) 0.37128	0.10751 (10^1, 0.1, 0.2) 0.30286	**0.1075** (10^-3,10^ -2, 0.01) 0.3159	0.1075 (10^-1,10^ -1,0.8,0.01) 0.1075
Google (200X6,550X6)	0.10317 (10^5, 0.1) 1.36584	**0.0291** (10^-2, 0.5) 0.04097	0.02961 (10^-1, 0.9, 0.1) 0.04389	0.02922 (10^1, 0.7, 0.5) 0.03942	0.02959 (10^-2,10^ -3, 0.01) 0.09673	0.0296 (10^-1,10^ -1,0.2,0.01) 0.0296
IBM (200X6,550X6)	0.26534 (10^5, 0.1) 1.23599	0.03054 (10^-2, 0.9) 0.10236	0.03054 (10^-1, 0.9, 0.9) 0.09743	0.03052 (10^1, 0.7, 0.6) 0.02931	**0.0305** (10^3,10^ -5, 0.01) 0.17269	**0.0305** (10^-5,10^ -5,0.8,0.01) 0.0305
Intel (200X6,550X6)	0.08187 (10^3, 0.1) 0.88873	0.03988 (10^0, 0.9) 0.05179	0.03587 (10^0, 0.9, 0.1) 0.33289	**0.03555** (10^1, 0.1, 0.45) 0.06317	0.03615 (10^-2,10^ -5, 0.01) 0.14209	0.03614 (10^-5,10^ -5,0.2,0.01) 0.03614
Microsoft (200X6,550X6)	0.1322 (10^4, 0.1) 1.14662	**0.03108** (10^-1, 0.1) 0.04568	0.0311 (10^0, 0.1, 0.1) 0.05827	0.03108 (10^1, 0.5, 0.45) 0.05256	0.03111 (10^4,10^ -3, 0.01) 0.11007	0.03111 (10^-5,10^ -5,0.8,0.01) 0.03111
RedHat (200X6,550X6)	0.04046 (10^4, 0.1) 1.33738	0.0236 (10^-1, 0.1) 0.04363	0.02361 (10^0, 0.9, 0.1) 0.05583	0.02378 (10^1, 0.3, 0.45) 0.05228	**0.02344** (10^5,10^ -5, 0.01) 0.00743	**0.02344** (10^-5,10^ -5,0.8,0.01) 0.02344
Pollution (30X16,30X16)	0.19385 (10^4, 0.1) 0.02086	0.25612 (10^-5, 0.9) 0.00606	0.25612 (10^-5, 0.1, 0.1) 0.01001	0.2561 (10^0, 0.3, 0.8) 0.00879	0.18527 (10^-1,10^ -1, 0.01) 0.00652	**0.18373** (10^3,10^ 3,0.2,0.01) 0.18373
Gas Furnace (150X7,143X7)	0.0712 (10^3, 0.1) 0.54059	0.02983 (10^-3, 0.7) 0.02535	0.02904 (10^1, 0.9, 0.3) 0.13019	0.02881 (10^1, 0.1, 0.2) 0.02141	0.02845 (10^0,10^ -5, 0.01) 0.01394	**0.0284** (10^-1,10^ -1,0.2,0.01) 0.0284
Flexible robot arm (500X10,519X10)	0.04236 (10^5, 0.1) 9.2244	0.01509 (10^0, 0.7) 0.47802	0.0149 (10^1, 0.1, 0.9) 0.44744	0.01494 (10^2, 0.3, 0.4) 0.29434	0.01479 (10^0,10^ -4, 0.01) 0.17969	**0.01479** (10^-1,10^ -1,0.2,0.01) 0.01479
S&P500 (200X6,550X6)	0.23351 (10^-5, 0.1) 1.23789	0.02547 (10^-2, 0.1) 0.0437	0.02552 (10^-1, 0.1, 0.5) 0.14388	**0.02545** (10^0, 0.5, 0.4) 0.04683	0.02565 (10^1,10^ -4, 0.01) 0.10934	0.02567 (10^-3,10^ -3,0.8,0.01) 0.02567
Space Ga (500X7,2607X7)	0.29919 (10^-1, 0.1) 8.79783	**0.2595** (10^5, 0.9) 0.51542	0.25952 (10^5, 0.9, 0.3) 0.75196	0.26528 (10^4, 0.3, 0.4) 0.36868	0.27444 (10^1,10^ 1, 0.01) 0.44424	0.28104 (10^3,10^ 3,0.8,0.01) 0.28104

The best result is shown as boldface

Table 8 Average ranks of SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on RMSE values for real-world datasets using linear kernel

Dataset	SVR	TSVR	HN-TSVR	Asy-ν-TSVR	RLTSVR	LAsy-ν-TSVR
ConcreteCS	6	2	4	3	5	1
Boston	1	4	6	5	3	2
Auto-MPG	1	6	4	5	2	3
Parkinsons	6	3	5	4	2	1
Winequality	3	1	4	6	2	5
Kin900	5	2	4	6	3	1
Demo	6	5	3	4	1	2
Mg_{17}	6	3	3	5	1	3
Google	6	1	5	2	3	4
IBM	6	4	5	3	1.5	1.5
Intel	6	5	2	1	4	3
Microsoft	6	1	3	2	4.5	4.5
RedHat	6	3	4	5	1.5	1.5
Pollution	3	6	5	4	2	1
Gas Furnace	6	5	4	3	2	1
Flexible robot arm	6	5	3	4	2	1
S&P500	6	2	3	1	4	5
Space Ga	6	1	2	3	4	5
Average rank	5.05556	3.27778	3.83333	3.66667	2.63889	**2.52778**

Remark 3 For any arbitrary vectors $\alpha_1^0 \in R^m$ and $\alpha_2^0 \in R^m$, the iterate $\alpha_1^i \in R^m$ and $\alpha_2^i \in R^m$ of iterative schemes (58) and (59) converge to the unique solution $\alpha_1^* \in R^m$ and $\alpha_2^* \in R^m$ respectively and also satisfying the following conditions as

$$\|D_1\alpha_1^{i+1} - D_1\alpha_1^*\| \leq \|I - \alpha_1 D_1^{-1}\| \, \|D_1\alpha_1^i - D_1\alpha_1^*\|$$

and

$$\|D_2\alpha_2^{i+1} - D_2\alpha_2^*\| \leq \|I - \alpha_2 D_2^{-1}\| \, \|D_2\alpha_2^i - D_2\alpha_2^*\|.$$

One can follow the proof of convergence of above from [50].

4 Numerical experiments

To measure the effectiveness of the proposed LAsy-ν-TSVR, several numerical experiments have been performed on standard benchmark real-world datasets for SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR. To conduct these numerical experiments, MATLAB software version2008b is used. In the formulations of SVR, TSVR, HN-TSVR, Asy-ν-TSVR, the QPPs are solved by using the external MOSEK optimization toolbox [51]. The number of interesting datasets are used in these numerical experiments such as Pollution, Space Ga [52]; Kin900, Demo [53]; the inverse dynamics of a Flexible robot arm [54]; S&P500, IBM, RedHat, Google, Intel, Microsoft [55]; Concrete CS, Boston, Auto-MPG, Parkinson, Gas furnace, Winequality from ([56]), Mg_{17} from [57]. In this paper, we consider both linear and non-linear case where Gaussian kernel function is taken in case of non-linear as

$$K(x_i, x_j) = \exp\left(-\mu \left\|x_i - x_j\right\|^2\right), \text{for} \, i, j = 1, \ldots, m$$

where, kernel parameter $\mu > 0$.

Here, the user input parameter values are described in Table 1. Finally, root mean square error (RMSE) is calculated based on optimal values for measuring the prediction accuracy by using the following formula:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(y_i - \tilde{y}_i\right)^2},$$

where, y_i are the observed values, $\tilde{y}_i$ are the predicted values respectively and N is the number of test data samples.

4.1 Artificial datasets

In this subsection, we have performed numerical experiments on 8 artificial generated datasets which are mentioned in Table 2 with their function definitions. In order to check the applicability of proposed LAsy-ν-TSVR for outliers and noise, we added two types of noise level in artificial datasets

Table 9 Performance comparison of LAsy-ν-TSVR with SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR using Gaussian kernel on real-world datasets

Dataset (Train size, Test size)	SVR (C, ε, μ) Time	TSVR ($C_1 = C_2, \varepsilon, \mu$) Time	HN-TSVR ($C_1 = C_2, \varepsilon_1, \varepsilon_2, \mu$) Time	Asy-ν-TSVR ($C_1 = C_2, \nu_1 = \nu_2, p, \mu$) Time	RLTSVR ($C_1 = C_2, C_3 = C_4, \varepsilon, \mu$) Time	LAsy-ν-TSVR ($C_1 = C_2, C_3 = C_4, \nu_1 = \nu_2, p, \mu$) Time
ConcreteCS (200X9,830X9)	0.21987 (10^5, 0.1,2^0) 1.07888	0.13593 (10^4, 0.9, 2^-1) 0.17614	0.13592 (10^5, 0.9, 0.5, 2^-1) 0.32181	0.13592 (10^5, 0.1, 0.2, 2^-1) 0.12466	0.19793 (10^0, 10^-1,0.01,2^4) 0.13272	**0.13562** (10^3, 10^3,0.8,0.01,2^1) 0.36546
Boston (200X14,306X14)	**0.13211** (10^1, 0.1,2^-5) 1.01044	0.39089 (10^0, 0.1, 2^-2) 0.09826	0.1645 (10^0, 0.9, 0.1, 2^-5) 0.15373	0.19912 (10^2, 0.9, 0.6, 2^-5) 0.10015	0.15282 (10^-1, 10^-3,0.01,2^-5) 0.20734	0.15085 (10^3, 10^3,0.2,0.01,2^-5) 0.50161
Auto-MPG (100X8,292X8)	0.33338 (10^0, 0.1,2^1) 0.2708	0.29912 (10^-2, 0.1, 2^0) 0.04727	0.29899 (10^-1, 0.1, 0.1, 2^0) 0.04945	0.29542 (10^1, 0.1, 0.4, 2^0) 0.04209	0.29897 (10^-2, 10^-4,0.01,2^-1) 0.07213	**0.2919** (10^3, 10^3,0.2,0.01,2^1) 0.05327
Parkinsons (500X17,5375X17)	0.32993 (10^1, 0.1,2^4) 6.51583	0.29921 (10^-5, 0.5, 2^4) 0.93348	0.31467 (10^-5, 0.1, 0.1, 2^3) 0.83478	0.29759(10^2, 0.1, 0.2, 2^4) 0.91714	**0.28555** (10^1, 10^-1,0.01,2^5) 1.44056	0.28583 (10^3, 10^3,0.2,0.01,2^5) 1.31966
Winequality (500X12,4398X12)	**0.13377** (10^2, 0.1,2^-3) 6.31838	0.13437 (10^-5, 0.7, 2^-1) 0.67405	0.135 (10^0, 0.1, 0.1, 2^-1) 0.61889	0.1349 (10^2, 0.3, 0.45, 2^-1) 0.68863	0.13457 (10^0, 10^-5,0.01,2^-3) 1.23389	0.13407 (10^-3, 10^-3,0.8,0.01,2^-3) 1.58545
Kin900 (250X33,650X33)	0.09794 (10^0, 0.1,2^-5) 1.44982	0.09729 (10^-5, 0.1, 2^-5) 0.19806	0.09729 (10^-5, 0.9, 0.1, 2^-5) 0.15782	0.09729 (10^-5, 0.1, 0.55, 2^-5) 0.18061	0.09476 (10^1, 10^-1,0.01,2^-5) 0.2887	**0.09475** (10^3, 10^3,0.8,0.01,2^-5) 0.23835
Demo (500X5,1548X5)	0.09388 (10^-1, 0.1,2^5) 5.91802	0.10106 (10^-1, 0.1, 2^3) 0.74691	0.10214 (10^0, 0.1, 0.1, 2^3) 0.76982	0.09926 (10^2, 0.1, 0.8, 2^3) 0.70036	0.0878 (10^-3, 10^0,0.01,2^5) 0.75285	**0.08708** (10^5, 10^5,0.8,0.01,2^5) 1.52279
Mg_{17} (500X6,995X6)	0.05985 (10^4, 0.1,2^-4) 6.65845	0.00346 (10^-1, 0.5, 2^3) 0.69266	0.00345 (10^3, 0.9, 0.9, 2^3) 1.10686	0.00343 (10^5, 0.9, 0.55, 2^3) 0.78205	0.00188 (10^-2, 10^-5,0.01,2^3) 1.01856	**0.00188** (10^-1, 10^-1,0.8,0.01,2^3) 2.38643
Google (200X6,550X6)	0.40066 (10^5, 0.1,2^2) 0.84436	0.17162 (10^-1, 0.9, 2^0) 0.16426	0.16014 (10^0, 0.9, 0.1, 2^0) 0.16433	0.17338 (10^1, 0.7, 0.5, 2^0) 0.09068	0.12165 (10^-2, 10^-4,0.01,2^-1) 0.17327	**0.08287** (10^-1, 10^-1,0.2,0.01,2^-1) 0.27634
IBM (200X6,550X6)	0.60049 (10^-1, 0.1,2^2) 0.82557	0.09633 (10^5, 0.1, 2^-1) 0.12129	0.09633 (10^5, 0.1, 0.1, 2^-1) 0.17919	0.09639 (10^5, 0.9, 0.6, 2^-1) 0.10195	0.0814 (10^1, 10^-5,0.01,2^-2) 0.17758	**0.0757** (10^3, 10^3,0.8,0.01,2^-3) 0.4903
Intel (200X6,550X6)	0.08068 (10^2, 0.1,2^-2) 0.82256	0.0459 (10^-1, 0.9, 2^-1) 0.10324	0.04921 (10^-1, 0.9, 0.1, 2^-1) 0.21832	0.04654 (10^2, 0.3, 0.2, 2^-2) 0.10064	0.04505 (10^0, 10^-4,0.01,2^-2) 0.13823	**0.0403** (10^3, 10^3,0.2,0.01,2^-3) 0.09551
Microsoft (200X6,550X6)	0.24695 (10^0, 0.1,2^0) 0.83487	0.11299 (10^-1, 0.1, 2^-1) 0.15409	0.10774 (10^0, 0.1, 0.1, 2^-1) 0.08638	0.10793 (10^1, 0.7, 0.4, 2^-1) 0.09278	**0.04689** (10^4, 10^-5,0.01,2^-3) 0.19988	**0.04689** (10^1, 10^1,0.8,0.01,2^-3) 0.11074
RedHat (200X6,550X6)	0.05603 (10^1, 0.1,2^-2) 0.82772	0.0582 (10^-1, 0.9, 2^-2) 0.10894	0.06807 (10^0, 0.1, 0.1, 2^-1) 0.27753	0.05021 (10^2, 0.1, 0.5, 2^-2) 0.09133	**0.02872** (10^3, 10^-5,0.01,2^-4) 0.20035	0.03858 (10^-1, 10^-1,0.8,0.01,2^-3) 0.09675
Pollution (30X16,30X16)	**0.13233** (10^0, 0.1,2^-2) 0.02132	0.13795 (10^-2, 0.7, 2^-4) 0.00709	0.16312 (10^-1, 0.1, 0.1, 2^-3) 0.01153	0.14392 (10^0, 0.5, 0.2, 2^-4) 0.00742	0.19262 (10^3, 10^-4,0.01,2^-4) 0.0103	0.16206 (10^1, 10^1,0.2,0.01,2^-3) 0.00209
Gas Furnace (150X7,143X7)	0.07154 (10^5, 0.1,2^-5) 0.47456	0.04394 (10^2, 0.1, 2^-4) 0.06976	0.04158 (10^1, 0.1, 0.1, 2^-3) 0.08725	0.04457 (10^2, 0.1, 0.8, 2^-4) 0.05331	**0.03897** (10^1, 10^-5,0.01,2^-5) 0.19557	**0.03897** (10^-1, 10^-1,0.2,0.01,2^-5) 0.08806
Flexible robot arm (500X10,519X10)	0.04337 (10^3, 0.1,2^-5) 9.5114	0.03595 (10^1, 0.1, 2^-1) 0.89527	0.03595 (10^3, 0.1, 0.9, 2^-1) 1.20655	0.03594 (10^3, 0.1, 0.4, 2^-1) 0.66941	**0.02117** (10^0, 10^-5,0.01,2^-1) 0.74859	0.02147 (10^1, 10^1,0.8,0.01,2^-1) 0.87327
S&P500 (200X6,550X6)	0.19605 (10^5, 0.1,2^-1) 1.41245	0.13739 (10^-1, 0.9, 2^1) 0.17418	0.13983 (10^1, 0.9, 0.1, 2^1) 0.33202	0.1449 (10^1, 0.3, 0.2, 2^1) 0.09276	0.1343 (10^0, 10^-5,0.01,2^-1) 0.07869	**0.12847** (10^3, 10^3,0.8,0.01,2^-1) 0.39596
Space Ga (500X7,2607X7)	0.30306 (10^-1, 0.1,2^-1) 9.72392	0.29628 (10^-3, 0.1, 2^5) 0.95483	0.29628 (10^-5, 0.1, 0.1, 2^5) 0.92673	0.29688 (10^1, 0.1, 0.2, 2^5) 0.63564	**0.29421** (10^0, 10^-1,0.01,2^5) 1.01221	0.29487 (10^3, 10^3,0.8,0.01,2^5) 0.68496

The best result is shown as boldface

Table 10 Average ranks of SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on RMSE values for real-world datasets using Gaussian kernel

Dataset	SVR	TSVR	HN-TSVR	Asy-ν-TSVR	RLTSVR	LAsy-ν-TSVR
ConcreteCS	6	4	3	2	5	1
Boston	1	6	4	5	3	2
Auto-MPG	6	5	4	2	3	1
Parkinsons	6	4	5	3	1	2
Winequality	1	3	6	5	4	2
Kin900	6	5	3.5	3.5	2	1
Demo	3	5	6	4	2	1
Mg_{17}	6	5	4	3	2	1
Google	6	4	3	5	2	1
IBM	6	4	3	5	2	1
Intel	6	3	5	4	2	1
Microsoft	6	5	3	4	1.5	1.5
RedHat	4	5	6	3	1	2
Pollution	1	2	5	3	6	4
Gas Furnace	6	4	3	5	1.5	1.5
Flexible robot arm	6	5	4	3	1	2
S&P500	6	3	4	5	2	1
Space Ga	6	3	4	5	1	2
Average rank	4.88889	4.16667	4.19444	3.86111	2.33333	**1.55556**

i.e. symmetric noise and asymmetric noise structure. Function 1 to Function 6 are having symmetric noise to generate artificial datasets in which variability of noise is proceeded from symmetric distribution and Function 7 and Function 8 are using the asymmetric noise such as heteroscedastic noise structure to generate the artificial dataset i.e. the noise is directly dependent on the value of input samples. Further, we use uniform probability distribution $\Omega \in U(a, b)$ with interval (a, b) for uniform noise and normal distribution $\Omega \in N(\mu, \sigma^2)$ where μ and σ^2 are the mean and variance respectively for Gaussian noise. Here, artificial dataset is generated by using 200 training samples with the additive noise and 500 testing samples without any addition of noise. To test the efficacy of proposed LAsy-ν-TSVR along with reported algorithms in this paper, a comparative analysis of their corresponding prediction errors for all artificial datasets are presented in Table 3 using linear kernel and Table 5 using Gaussian kernel. One can conclude from Tables 3 and 5 that our proposed LAsy-ν-TSVR performs better or comparable generalization performance in comparison to other methods. Further, Tables 4 and 6 are consisted the average ranks of SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR based on RMSE values for artificial datasets using linear and Gaussian kernel respectively. One can notice that proposed LAsy-ν-TSVR is having lowest rank among SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR in both linear and nonlinear case which shows the usability and effectiveness of proposed LAsy-ν-TSVR.

To check the performance for symmetric noise structure, the prediction values are plotted for SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR using Gaussian kernel of Function 5 in Fig. 1 with uniform noise. Similarly, for Function 6 with Gaussian noise, we depict the prediction plots in Fig. 2 respectively. It is easily noticeable that our proposed LAsy-ν-TSVR is having better agreement with final target values in comparison to SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR for symmetric noise structure having both uniform and Gaussian noise.

Further, to test the applicability of proposed LAsy-ν-TSVR on datasets having asymmetric noise structure i.e. heteroscedastic noise, the prediction plots are drawn in Fig. 3 for Function 7 using uniform noise. Similarly, for Function 8, we depict the prediction plots in Fig. 4 having Gaussian noise. One can observe from these results that LAsy-ν-TSVR is more effective to handle the asymmetric noise structure for both uniform and Gaussian noise.

4.2 Real world datasets

In this paper, we have shown comparative analysis of our proposed LAsy-ν-TSVR with SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR using real world datasets for linear and non-linear case that are tabulated in Tables 7 and 9 respectively. One can notice that the prediction accuracy of proposed LAsy-ν-TSVR is better or equal in 8 out of

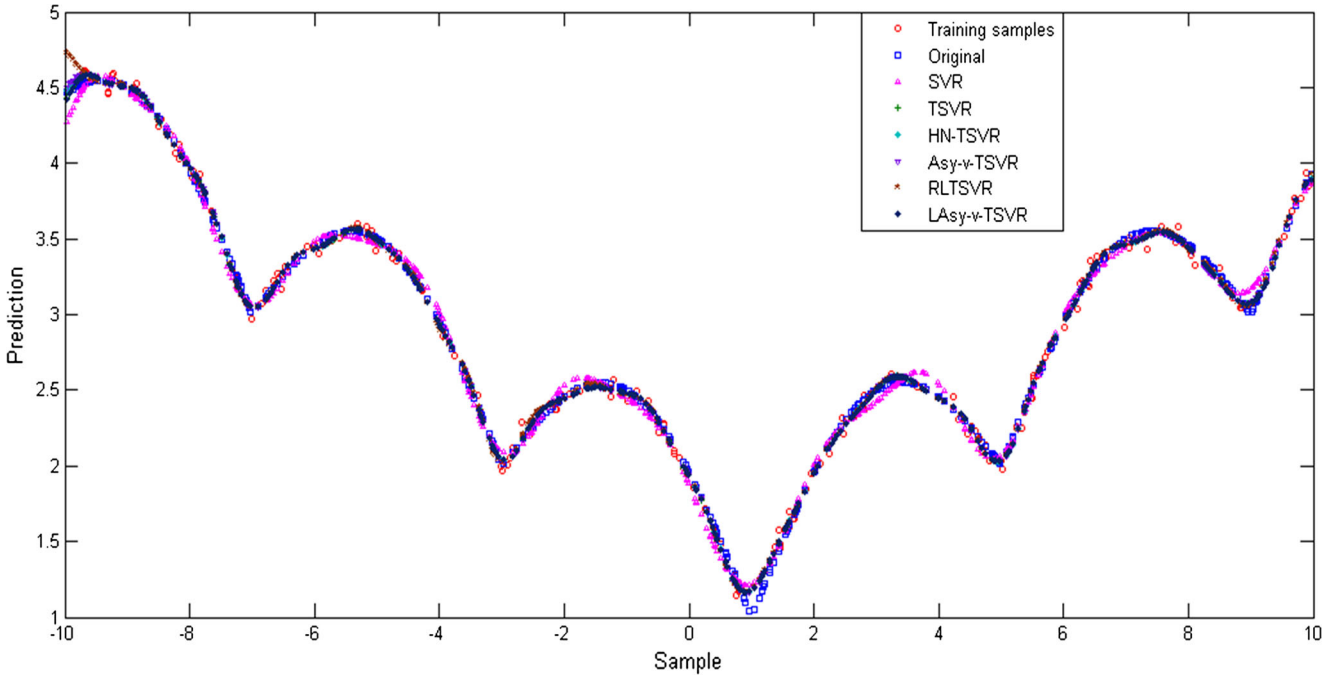

Fig. 1 Accuracy plot over the test set by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR using Gaussian kernel for Function 5 with uniform noise

18 standard benchmark real world datasets for linear kernel and 11 out of 18 standard real world datasets for Gaussian kernel which justified the applicability and usability. In order to show the performance graphically, we plot prediction values for Auto-MPG, Gas furnace and Intel datasets in Figs. 5, 7 and 9 respectively. Similarity, prediction error of Auto-MPG, Gas furnace and Intel are shown in Figs. 6, 8 and 10 respectively. One can conclude from these results that the prediction values of our proposed LAsy-ν-TSVR is very close to target values in comparison to SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR which justify the existence and usability of our approach. Further, to justify the performance statistically of our proposed LAsy-ν-TSVR, the average ranks are

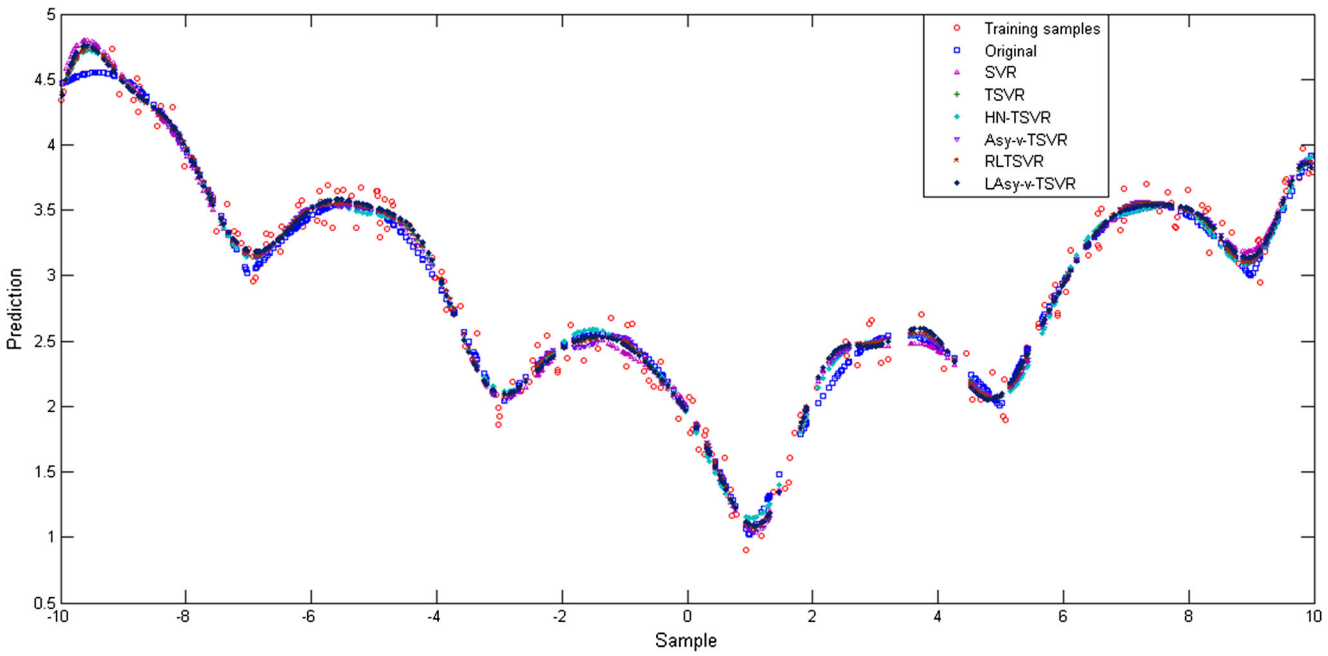

Fig. 2 Accuracy plot over the test set by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR using Gaussian kernel for Function 6 with Gaussian noise

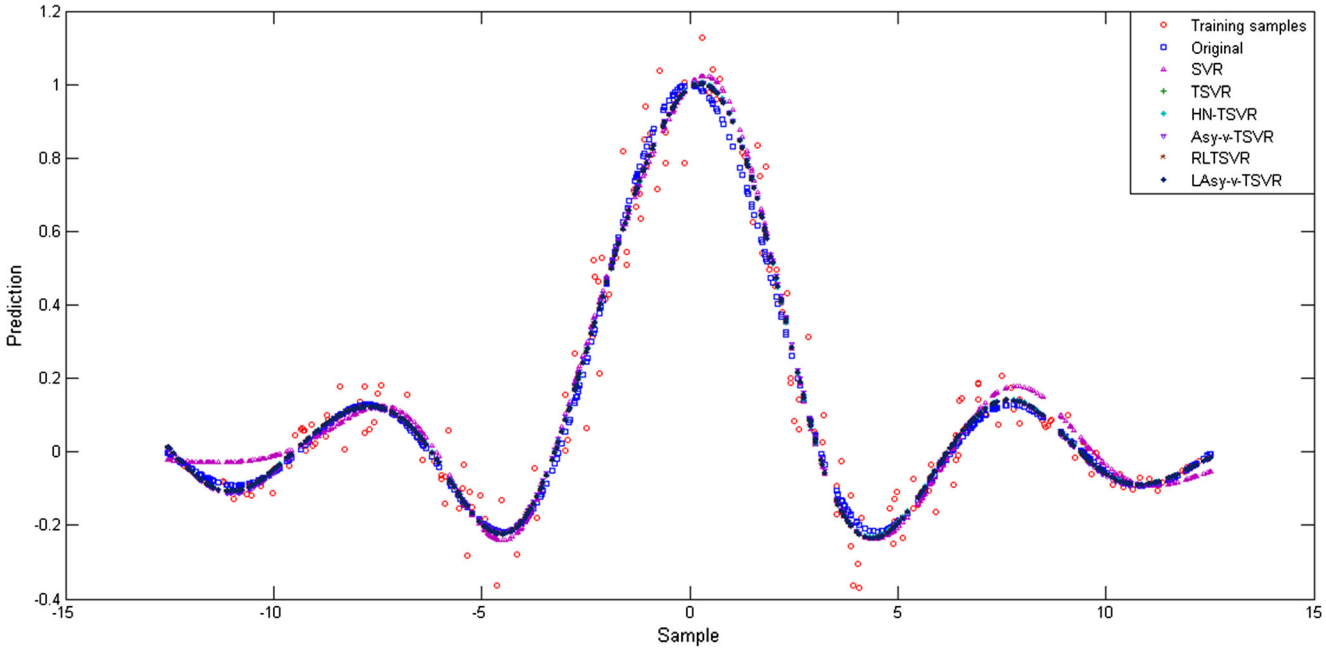

Fig. 3 Accuracy plot over the test set by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR using Gaussian kernel for Function 7 with uniform noise

depicted based on RMSE values in Tables 8 and 10 for all reported methods using both linear and nonlinear kernel respectively. It is clear form Tables 8 and 10 that proposed LAsy-ν-TSVR is having lowest rank among all in both cases.

Now, non-parametric Friedman test is conducted with the corresponding post hoc test [58] on 6 algorithms and 18 datasets in which it is used to detect differences in ranking of RMSE across multiple algorithms.

This test is mainly used for one-way repeated measures analysis of variance by ranks of different algorithms. Let us consider, all methods are equivalent under null hypothesis, the Friedman statistic is determined for linear cases from Table 8 as follows.

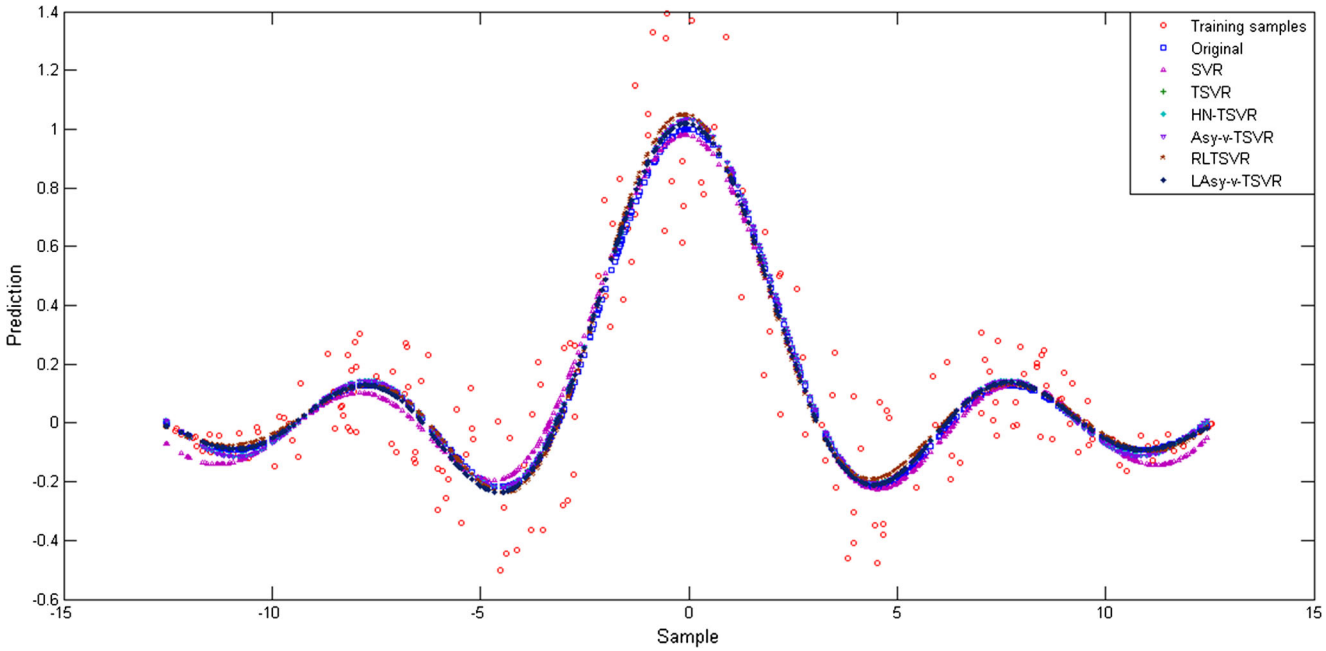

Fig. 4 Accuracy plot over the test set by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR using Gaussian kernel for Function 8 with Gaussian noise

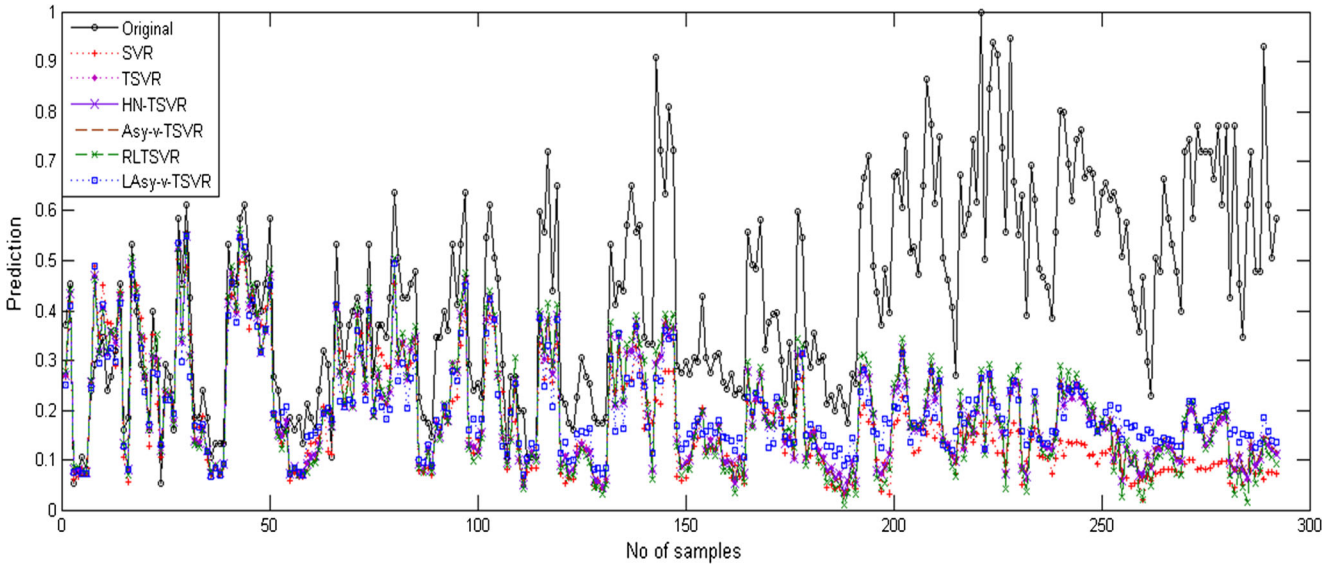

Fig. 5 Prediction over the testing dataset by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on the Auto-MPG dataset. Gaussian kernel was used

$$\chi_F^2 = \frac{12 \times 18}{6 \times 7}\left[(5.055556^2 + 3.27778^2 + 3.83333^2 + 3.66667^2 + 2.63889^2 + 2.52778^2) - \left(\frac{6 \times 7^2}{4}\right)\right] = 22.0873$$

$$F_F = \frac{17 \times 22.0873}{18 \times 5 - 22.0873} = 5.5289$$

According to Fisher–Snedecor F distribution, Friedman expression F_F is distributed with degree of freedom $(6-1, (6-1)*(18-1)) = (5, 85)$ degree of freedom. The critical value of $F(5, 85)$ is 2.321 for $\alpha = 0.05$. Since $F_F > 2.321$, we reject the null hypothesis i.e. all algorithms are not equivalent. Now, Nemenyi post hoc test is conducted for pair wise comparison of all methods. This test is applied after Friedman test if it rejects the null hypothesis, for comparison of pair wise performance. For this, we calculate the critical difference (CD) with $q_\alpha = 2.589$ as

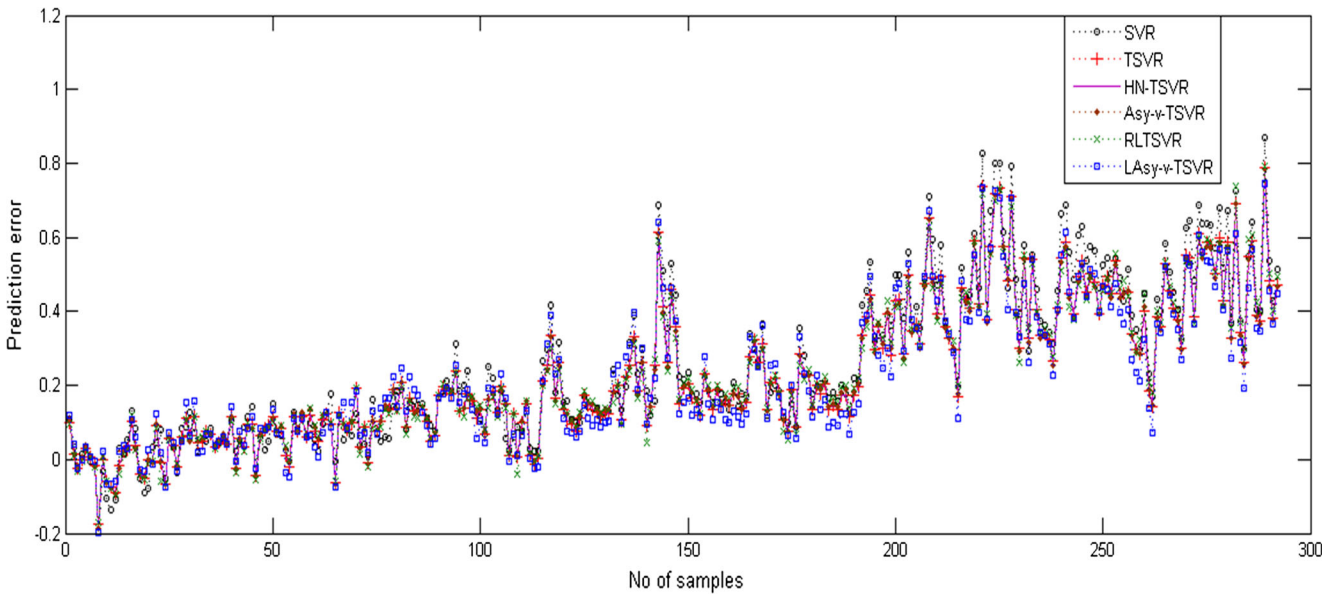

Fig. 6 Prediction error over the testing dataset by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on the Auto-MPG dataset. Gaussian kernel was used

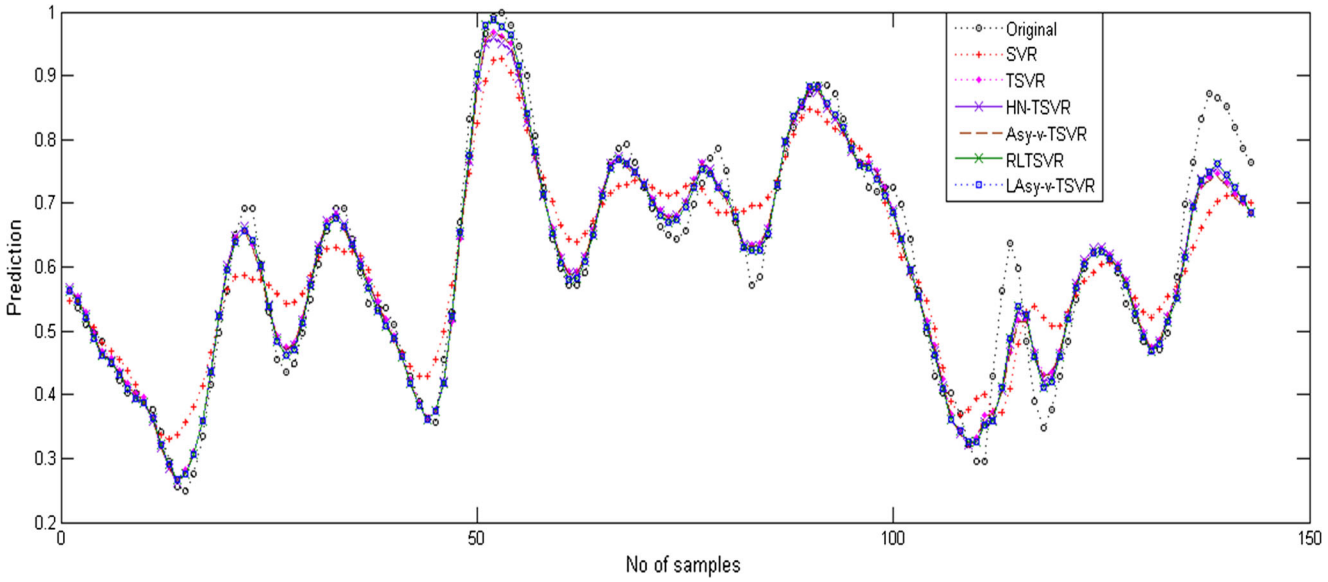

Fig. 7 Prediction over the testing dataset by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on the Gas furnace dataset. Gaussian kernel was used

$$\text{CD} = 2.589\sqrt{\frac{6\times 7}{6\times 18}} = 1.6145 \text{ for } \theta = 0.10.$$

where, the value of q_α is decided on the basis of number of concerned algorithms and the value of θ from Demsar,[58].

The difference of average rank between SVR and proposed LAsy-ν-TSVR (5.055556 − 2.527778 = 2.527778) which is greater than CD i.e. (1.6145). This result assures that the prediction performance of proposed algorithm LAsy-ν-TSVR is better than SVR. Further, the differences of average rank of proposed LAsy-ν-TSVR with TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR are not more than the CD, so there is not any significant differences among them.

Secondly, to apply Friedman test in non linear case for standard real world bench mark datasets on the average ranks of SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and proposed LAsy-ν-TSVR from Table 10 as follows:

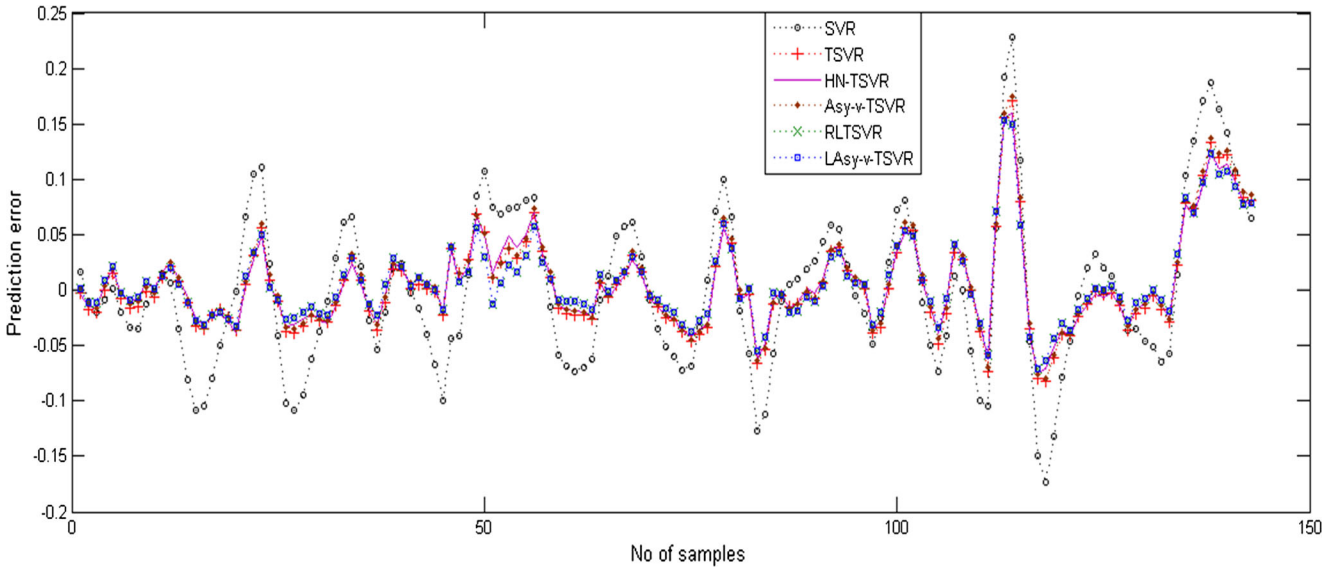

Fig. 8 Prediction error over the testing dataset by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on the Gas furnace dataset. Gaussian kernel was used

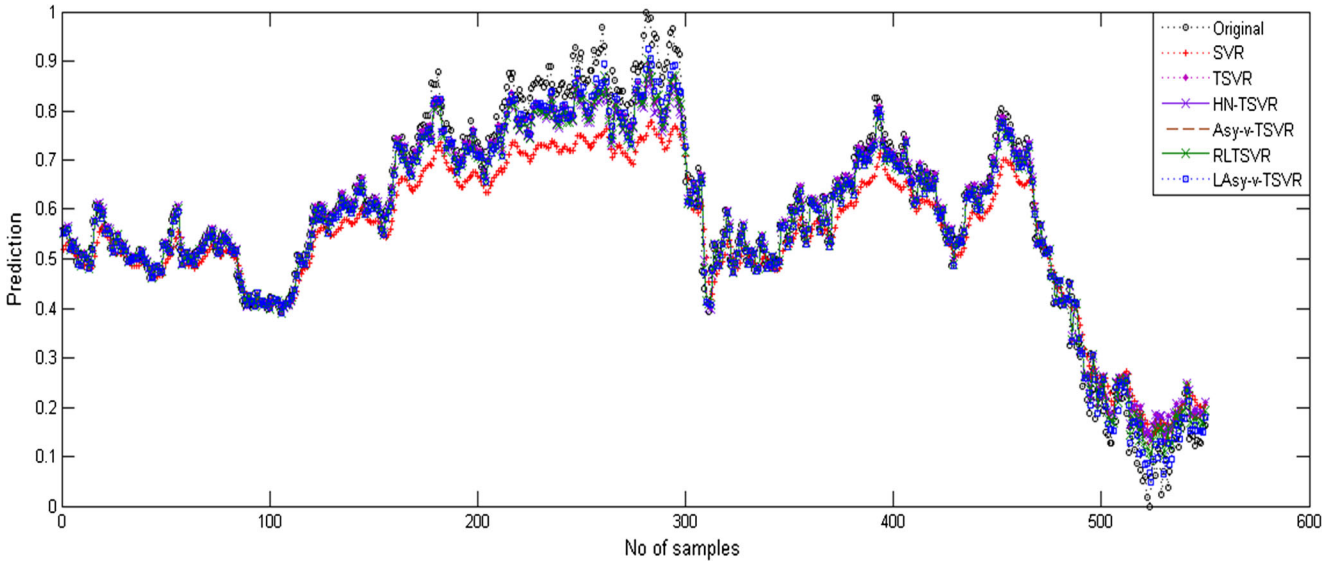

Fig. 9 Prediction over the testing dataset by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on the Intel dataset. Gaussian kernel was used

$$\chi_F^2 = \frac{12 \times 18}{6 \times 7}\left[(4.88889^2 + 4.16667^2 + 4.19444^2 + 3.86111^2 + 2.33333^2 + 1.55556^2) - \left(\frac{6 \times 7^2}{4}\right)\right] = 41.8016$$

$$F_F = \frac{17 \times 41.8016}{18 \times 5 - 41.8016} = 14.7438$$

The critical value of $F(5, 85)$ is 2.321 for$\alpha = 0.05$. Since $F_F > 2.321$, we reject the null hypothesis. Now, we perform the Nemenyi test to compare the methods pair-wise. Here, critical difference (CD) is 1.6145.

i. The differences between the average rank of SVR and proposed LAsy-ν-TSVR (4.888889 − 1.555556 = 3.333333) is greater than CD (1.6145) thus proposed LAsy-ν-TSVR is better than SVR.

ii. Further, check the dissimilarity between the proposed LAsy-ν-TSVR with TSVR, the difference between the average ranks i.e. (4.166667 − 1.555556 = 2.611111) is larger than(1.6145), thus the prediction performance of

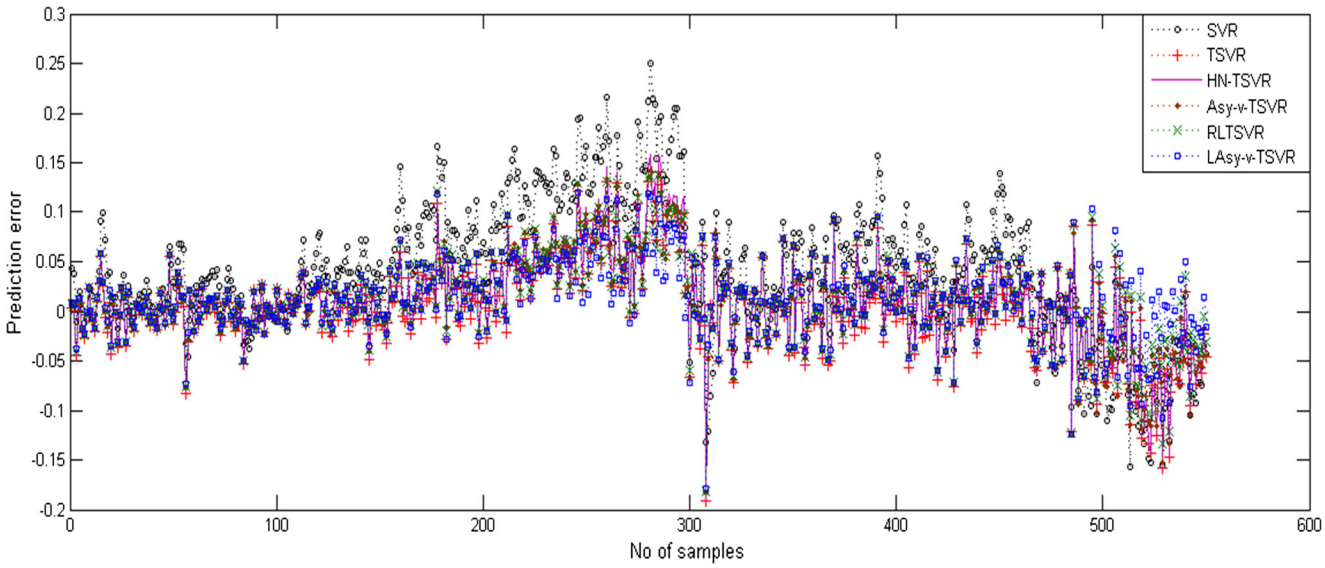

Fig. 10 Prediction error over the testing dataset by SVR, TSVR, HN-TSVR, Asy-ν-TSVR, RLTSVR and LAsy-ν-TSVR on the Intel dataset. Gaussian kernel was used

proposed LAsy-ν-TSVR is much effective in comparison to TSVR.

iii. The average rank difference between HN-TSVR and proposed LAsy-ν-TSVR is $(4.194444 - 1.555556 = 2.638889)$ which is greater than (1.6145), it implies that LAsy-ν-TSVR is better than HN-TSVR.

iv. Since the dissimilarity of average rank between Asy-ν-TSVR and proposed LAsy-ν-TSVR $(3.861111 - 1.555556 = 2.305556)$ is larger than (1.6145) which validates the existence and applicability of proposed algorithm LAsy-ν-TSVR in comparison to Asy-ν-TSVR.

5 Conclusions and future work

In this paper, we propose a new approach as improved regularization based Lagrangian asymmetric ν-twin support vector regression (LAsy-ν-TSVR) using pinball loss function that follows the gist of statistical learning theory i.e. SRM principle effectively. The solution of LAsy-ν-TSVR is determined by solving the linearly convergent iterative approach unlike solving the QPPs as used in SVR, TSVR, HN-TSVR and Asy-ν-TSVR. Thus, no external optimization toolbox is required in our case. Another advantage of proposed LAsy-ν-TSVR is that proposed LAsy-ν-TSVR is more effective and usable to handle both symmetric and asymmetric structure having two types uniform and Gaussian noise in comparison to SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR. In order to justify numerically, proposed LAsy-ν-TSVR is tested and validated on various artificial generated datasets having symmetric and heteroscedastic structure of uniform and Gaussian noise. One can conclude that proposed LAsy-ν-TSVR is much more effective to handle the noise in comparison to SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR. On the basis of experimental results for real world datasets, it can be stated that proposed LAsy-ν-TSVR are far better than SVR, TSVR, HN-TSVR, Asy-ν-TSVR and RLTSVR in terms of generalization ability as well as the faster learning ability clearly illustrate its efficacy and applicability. In future, one can apply the heuristic approach to select the optimum parameters and another, a sparse model can be proposed based on asymmetric pinball loss function.

References

1. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
2. Drucker H, Burges CJC, Kaufman L, Smola AJ, Vapnik V (1997) Support vector regression machines." In Advances in neural information processing systems, pp. 155–161
3. Trzciński T, Rokita P (2017) Predicting popularity of online videos using support vector regression. IEEE Trans Multimedia 19(11): 2561–2570
4. López-Martín C, Ulloa-Cazarez RL, García-Floriano A (2017) Support vector regression for predicting the productivity of higher education graduate students from individually developed software projects. IET Softw 11(5):265–270
5. Golkarnarenji G, Naebe M, Badii K, Milani AS, Jazar RN, Khayyam H (2018) Support vector regression modelling and optimization of energy consumption in carbon fiber production line. Comput Chem Eng 109:276–288
6. García-Floriano A, López-Martín C, Yáñez-Márquez C, Abran A (2018) Support vector regression for predicting software enhancement effort. Inf Softw Technol 97:99–109
7. Dong Y, Zhang Z, Hong W-C (2018) A hybrid seasonal mechanism with a chaotic cuckoo search algorithm with a support vector regression model for electric load forecasting. Energies 11(4):1009
8. Khosravi A, Koury RNN, Machado L, Pabon JJG (2018) Prediction of wind speed and wind direction using artificial neural network, support vector regression and adaptive neuro-fuzzy inference system. Sustainable Energy Technol Assess 25:146–160
9. Baydaroğlu Ö, Koçak K, Duran K (2018) River flow prediction using hybrid models of support vector regression with the wavelet transform, singular spectrum analysis and chaotic approach. Meteorog Atmos Phys 130(3):349–359
10. Xiao X, Zhang T, Zhong X, Shao W, Li X (2018) Support vector regression snow-depth retrieval algorithm using passive microwave remote sensing data. Remote Sens Environ 210:48–64
11. Fisher DM, Kelly RF, Patel DR, Gilmore M (2018) A support vector regression method for efficiently determining neutral profiles from laser induced fluorescence data. Rev Sci Instrum 89(10): 10C104
12. Zhang J, Teng Y-F, Chen W (2018) Support vector regression with modified firefly algorithm for stock price forecasting. Appl Intell:1–17
13. Schölkopf B, Smola AJ, Williamson RC, Bartlett PL (2000) New support vector algorithms. Neural Comput 12(5):1207–1245
14. Collobert R, Bengio S (2001) SVMTorch: support vector machines for large-scale regression problems. J Mach Learn Res 1:143–160
15. Law MHC, Kwok JT-Y (2001) Bayesian Support Vector Regression. AISTATS
16. Bi J, Bennett KP (2003) A geometric approach to support vector regression. Neurocomputing 55(1–2):79–108
17. Musicant DR, Feinberg A (2004) Active set support vector regression. IEEE Trans Neural Netw 15(2):268–275
18. Wang W, Xu Z (2004) A heuristic training for support vector regression. Neurocomputing 61:259–275
19. Lee Y-J, Hsieh W-F, Huang C-M (2005) ε-SSVR: a smooth support vector machine for ε-insensitive regression. IEEE Trans Knowl Data Eng 17(5):678–685
20. Chuang C-C (2007) Fuzzy weighted support vector regression with a fuzzy partition. IEEE Trans Syst Man Cybern B 37(3):630–640
21. Jayadeva, Khemchandani R, Chandra S (2007) Twin support vector machines for pattern classification. IEEE Trans Pattern Anal Mach Intell 29(5):905–910
22. Peng X (2010) TSVR: an efficient twin support vector machine for regression. Neural Netw 23(3):365–372
23. Singh M, Chadha J, Ahuja P, Chandra S (2011) Reduced twin support vector regression. Neurocomputing 74(9):1474–1477
24. Xu Y, Wang L (2012) A weighted twin support vector regression. Knowl-Based Syst 33:92–101
25. Zhao Y-P, Zhao J, Zhao M (2013) Twin least squares support vector regression. Neurocomputing 118:225–236
26. Suykens JAK, Vandewalle J (1999) Least squares support vector machine classifiers. Neural Process Lett 9(3):293–300

27. Balasundaram S, Tanveer M (2013) On Lagrangian twin support vector regression. Neural Comput & Applic 22(1):257–267
28. Balasundaram S, Gupta D (2014) Training Lagrangian twin support vector regression via unconstrained convex minimization. Knowl-Based Syst 59:85–96
29. Niu J, Chen J, Xu Y (2017) Twin support vector regression with Huber loss. J Intell Fuzzy Syst 32(6):4247–4258
30. Tanveer M, Shubham K (2017) A regularization on Lagrangian twin support vector regression. Int J Mach Learn Cybern 8(3): 807–821
31. Huang X, Shi L, Suykens JAK (2014a) Support vector machine classifier with pinball loss.". IEEE Trans Pattern Anal Mach Intell 36(5):984–997
32. Huang X, Shi L, Suykens JAK (2015) Sequential minimal optimization for SVM with pinball loss. Neurocomputing 149:1596–1603
33. Xu Y, Yang Z, Zhang Y, Pan X, Wang L (2016) A maximum margin and minimum volume hyper-spheres machine with pinball loss for imbalanced data classification. Knowl-Based Syst 95:75–85
34. Peng X, Xu D (2013) A twin-hypersphere support vector machine classifier and the fast learning algorithm. Inf Sci 221:12–27
35. Xu Y, Yang Z, Pan X (2017) A novel twin support-vector machine with pinball loss. IEEE Transactions on Neural Networks and Learning Systems 28(2):359–370
36. Nandan Sengupta R (2008) Use of asymmetric loss functions in sequential estimation problems for multiple linear regression. J Appl Stat 35(3):245–261
37. Reed C, Yu K (2009) A partially collapsed Gibbs sampler for Bayesian quantile regression
38. Le Masne Q, Pothier H, Birge NO, Urbina C, Esteve D (2009) Asymmetric noise probed with a Josephson junction. Phys Rev Lett 102(6):067002
39. Hao P-Y (2010) New support vector algorithms with parametric insensitive/margin model. Neural Netw 23(1):60–73
40. Steinwart I, Christmann A (2011) Estimating conditional quantiles with the help of the pinball loss. Bernoulli 17(1):211–225
41. Xu Y, Guo R (2014) An improved ν-twin support vector machine. Appl Intell 41(1):42–54
42. Rastogi R, Anand P, Chandra S (2017) A ν-twin support vector machine based regression with automatic accuracy control. Appl Intell 46(3):670–683
43. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
44. Xu Y, Li X, Pan X, Yang Z (2018) Asymmetric ν-twin support vector regression. Neural Comput & Applic 30(12):3799–3814
45. Huang X, Shi L, Pelckmans K, Suykens JAK (2014b) Asymmetric ν-tube support vector regression. Comput Stat Data Anal 77:371–382
46. Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge university press, Cambridge
47. Huber PJ (1964) Robust estimation of a location parameter. Ann Math Stat 35(1):73–101
48. Mangasarian OL, Musicant DR (2000) Robust linear and support vector regression. IEEE Trans Pattern Anal Mach Intell 22(9):950–955
49. Mangasarian OL (1994) Nonlinear programming. SIAM, Philadelphia
50. Mangasarian OL, Musicant DR (2001) Lagrangian support vector machines. J Mach Learn Res 1:161–177
51. Mosek.com (2018) 'MOSEK optimization software for solving QPPs.'[online]. Available: https://www.mosek.com
52. StatLib (2018) 'StatLib, Carnegie Mellon University.' [online]. Available: http://lib.stat.cmu.edu/datasets
53. DELVE (2018) 'DELVE, University of California.' [online]. Available: https://www.cs.toronto.edu/~delve/
54. DaISy (2018) 'DaISY: Database for the Identification of Systems, Department of Electrical Engineering, ESAT/STADIUS, KU Leuven, Belgium.' [online]. Available: http://homes.esat.kuleuven.be/~smc/daisydata.html
55. Yahoo Finance (2018) 'Yahoo Finance.' [online] Available: http://finance.yahoo.com/
56. Lichman M (2018) "UCI Machine Learning Repository. Irvine, University of California, Irvine, School of Information and Computer Sciences. (2013). 02–14. Available: https://archive.ics.uci.edu/ml/
57. Casdagli M (1989) Nonlinear prediction of chaotic time series. Physica D 35(3):335–356
58. Xu Y (2012) A rough margin-based linear ν support vector regression. Statistics & Probability Letters 82(3):528–534

Umesh Gupta Received B.Tech (Computer science & Engineering) in 2008 from Dr. APJ Abdul Kalam Technical University Lucknow, India and M. E. (Computer Science and Engineering) in 2013 from NITTTR, Chandigarh (National Institute of Technical Teacher's Training and Research), India and pursuing Ph.D in Machine Learning from National Institute of Technology, Arunachal Pradesh, India. He has worked almost six years at Dr. APJ Abdul Kalam Technical University, Lucknow as Assistant Professor. He also has one year industrial experience in telecom industry. He has published about 10 research papers in National/International conferences/journals. His research interest includes image and video processing, support vector machines and optimization.